PREFACE

Objective and Intended Readership of This Manual

The MB89160/160A/160L Series has been developed as a general-purpose version of the F\(^2\)MC*-8L family consisting of proprietary 8-bit, single-chip microcontrollers. The MB89160/160A/160L Series is applicable to a wide range of applications from welfare products to industrial equipment, including portable devices.


*: F\(^2\)MC stands for FUJITSU Flexible Microcontroller.

Configuration of This Manual

This manual consists of the following 16 chapters:

Chapter 1  Overview
- Provides an overview of the features and functions of the MB89160/160A/160L Series.

Chapter 2  Handling Devices
- Describes points to note when using the MB89620 series.

Chapter 3  CPU
- Describes the functions of the MB89160/160A/160L Series CPU.

Chapter 4  I/O Ports
- Describes the function and operation of the MB89160/160A/160L Series I/O ports.

Chapter 5  Timebase Timer
- Describes the function and operation of the MB89160/160A/160L Series timebase timer.

Chapter 6  Watchdog Timer
- Describes the functions and operation of the MB89160/160A/160L Series watchdog timer.

Chapter 7  8-bit PWM Timer (PWM Timer 1, PWM Timer 2)
- Describes the functions and operation of the MB89160/160A/160L Series 8-bit PWM timer.

Chapter 8  8/16-bit Timer/Counter
- Describes the functions and operation of the MB89160/160A/160L Series 8/16-bit timer/counter.

Chapter 9  8-bit Serial I/O Interface
- Describes the function and operation of the MB89160/160A/160L Series 8-bit serial I/O interface.

Chapter 10  Buzzer Output
- Describes the function and operation of the MB89160/160A/160L Series audible alarm output.
Chapter 11  External Interrupt Circuit 1 (Edge-triggered)
Describes the functions and operation of the MB89160/160A/160L Series external interrupt circuit 1 (edge-triggered interrupt).

Chapter 12  External Interrupt Circuit 2 (Level-triggered)
Describes the functions and operation of the MB89160/160A/160L Series external interrupt circuit 2 (level-triggered interrupt).

Chapter 13  A/D Converter
Describes the functions and operation of the MB89160/160A/160L Series A/D converter.

Chapter 14  Watch Prescaler
Describes the functions and operation of the MB89160/160A/160L Series watch prescaler.

Chapter 15  Remote Control Unit Transmit Frequency Generator (6-bit PPG)
Describes the functions and operation of the MB89160/160A/160L Series remote control transmission output.

Chapter 16  LCD Controller-Driver
Describes the functions and operation of the MB89160/160A/160L Series liquid crystal display (LCD) controller-driver circuit.

The appendices include the I/O map, mask options, instruction summary, instruction list, and instruction map.
1. The products described in this manual and the specifications thereof may be changed without prior notice. To obtain up-to-date information and/or specifications, contact your Fujitsu sales representative or Fujitsu authorized dealer.

2. Fujitsu will not be liable for infringement of copyright, industrial property right, or other rights of a third party caused by the use of information or drawings described in this manual.

3. The contents of this manual may not be transferred or copied without the express permission of Fujitsu.

4. The products contained in this document are not intended for use with equipments which require extremely high reliability such as aerospace equipments, undersea repeaters, nuclear control systems or medical equipments for life support.

5. Some of the products described in this manual may be strategic materials (or special technology) as defined by the Foreign Exchange and Foreign Trade Control Law. In such cases, the products or portions thereof must not be exported without permission as defined under the Law.

© 1996 FUJITSU LIMITED Printed in Japan
READING THIS MANUAL

■ Page Layout

In this manual, an entire section is presented on a single page or spread whenever possible. The reader can thus view a section without having to flip pages.

The content of each section is summarized immediately below the title. You can obtain a rough overview of this product by reading through these summaries.

Also, higher level section headings are given in lower sections so that you can know to which section the text you are currently reading belongs.

■ Finding Information

In addition to the standard table of contents and index, the following methods are available to find information in a particular section when required.

● Register index

Information can be looked up in the register index by register name, by bit name, and by their respective abbreviations.

● Subheading index

The sub-headings in each section (lines that start with ■) are collected together in the subheading index. The subheading index provides a means of looking up information at a finer level of detail than the table of contents.

■ Naming Conventions for Register Name and Pin Name

● Example for description of register name and bit name

By writing “1” to the sleep bit (STBC: SLP) in the standby control register...

Disable the interrupt request output (TBTC: TBIE = “0”) from the timebase timer.

Interrupt is accepted if the interrupt is enabled (CCR: I = “1”).

● Notations for shared pins

Pin P25/SCK

Many of the pins in the devices of this series are multi-function pins. (They can be switched between two or more functions under program control.) The multiple names of these pins (indicating their multiple functions) are separated by slant bars (/).
Facing Pages Organization and Notation Conventions

Subheading
Section summary
Section title
Higher level section
Table title

Check
Points requiring check and prohibited items.
Always read checks.

Note
Indicates an item or manual that should be referenced.

Reference
Provides useful information for reference.

Chapter title
Series title

Figure title

Subsection 3.2.4 Condition Code Register (CCR)

The condition code register (CCR) located in the lower 6 bits of the program status (PS) consists of the C, U, Z, S, and V bits indicating the results of arithmetic operations and the contents of transfer data, and the I, E1, and E2 bits to control whether or not the CPU accepts interrupt requests.

Structure of Condition Code Register (CCR)

Figure 3.2.4 Structure of Condition Code Register

In the case where the flag is set to 1 for the CPU and the CPU accepts interrupt requests, interrupts are processed when this flag is set to 1 for the CPU and the CPU does not accept interrupts. This is done after a reset of the CPU.

Pattern 1

Pattern 2

Figure 3.2.5 Pattern 1

Figure 3.2.6 Pattern 2

Interrupt Accuracy Control

The accuracy of the program status (PS) is set by the CPU. The accuracy is compared with the interrupt level setting register (PS) which has a setting for each interrupt level to ensure that the program status (PS) is set to 1 when the program status (PS) is set to 1.

Reference: Test Section 3.4. "Internal interrupt detection methods."

Chapter 2

Table 3.2.7
Development Tools and Other Resources Required for Development

The following items are required for developing using the MB89160/160A/160L Series. Contact FUJITSU sales staff for the required development tools and other resources.

- Manuals required for development
  - Checklist
    - FMC-8L MB89160/160A/160L Series Data Sheet
      (Provides electrical characteristics and various characteristic examples for the device.)
    - FMC-8L MB89600 Series Programming Manual
      (Describes the FMC-8L series instruction set.)
    - FMC-8L MB89600 Series C Compiler Manual
      (Only required when developing in C.)
      (Describes program development in C and how to run the compiler.)
    - FMC-8L MB89600 Series Assembler Manual
      (Describes program development in assembly language.)
    - FMC-8L MB89600 Series Support System Manual
      (Describes how to run the macro assembler, linker, and library manager.)
    - FMC-8L MB89600 Series Software Simulator Manual
      (Only required when performing evaluation using the simulator.)
      (Describes how to operate the software simulator.)

  Manuals marked with * are provided with the products. In addition, manuals for products such as development tools are provided with the product.

- Software required for development
  - Checklist
    - C compiler (Only required when developing in C.)
    - Assembler, linker, librarian
    - Software simulator (Only required when performing evaluation using the simulator.)
    - Emulator/debugger (Only required when performing evaluation using the MB2140A series.)

  The model number for each software package differs depending on the operating system. See the F-MC development tools catalog or product guide for details.

- Items required for evaluation using one-time PROM or EPROM microcontrollers (when performing your own PROM or EPROM programming)
  - Checklist
    - One of: MB89P165, MB89W165
    - ROM programmer (a programmer able to program an MBM27C256A)
      See the data sheet for details of recommended programmers.
    - Package conversion adaptor for writing (available from Sun Hayato Co., Ltd.)

<table>
<thead>
<tr>
<th>Package</th>
<th>Socket Model</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPT-80P-M05</td>
<td>ROM-80SQF-28DP-8L</td>
</tr>
<tr>
<td>FPT-80P-M11</td>
<td>ROM-80QF2-28DP-8L2</td>
</tr>
<tr>
<td>FPT-80P-M06</td>
<td>ROM-80QF-28DP-8L3</td>
</tr>
<tr>
<td>FPT-80C-A02</td>
<td>ROM-80QF-28DP-8L3</td>
</tr>
</tbody>
</table>

- Development tools
  - Checklist
    - MB89PV160 (Piggyback/evaluation device)
    - Evaluation tools
      (Main unit) (Pod) (Probe)
      MB2141A + MB2144-505+MB2144-202

- Reference material
  - FMC development tools catalog
  - Microcomputer product guide
<table>
<thead>
<tr>
<th>OVERVIEW</th>
<th>1</th>
</tr>
</thead>
<tbody>
<tr>
<td>HANDLING DEVICES</td>
<td>2</td>
</tr>
<tr>
<td>CPU</td>
<td>3</td>
</tr>
<tr>
<td>I/O PORTS</td>
<td>4</td>
</tr>
<tr>
<td>TIMEBASE TIMER</td>
<td>5</td>
</tr>
<tr>
<td>WATCHDOG TIMER</td>
<td>6</td>
</tr>
<tr>
<td>8-BIT PWM TIMER</td>
<td>7</td>
</tr>
<tr>
<td>8/16-BIT TIMER/COUNTER</td>
<td>8</td>
</tr>
<tr>
<td>8-BIT SERIAL I/O INTERFACE</td>
<td>9</td>
</tr>
<tr>
<td>BUZZER OUTPUT</td>
<td>10</td>
</tr>
<tr>
<td>EXTERNAL INTERRUPT CIRCUIT 1(EDGE)</td>
<td>11</td>
</tr>
<tr>
<td>EXTERNAL INTERRUPT CIRCUIT 2(EDGE)</td>
<td>12</td>
</tr>
<tr>
<td>A/D CONVERTER</td>
<td>13</td>
</tr>
<tr>
<td>TIMECLOCK PRESCALER</td>
<td>14</td>
</tr>
<tr>
<td>REMOTE CONTROL UNIT TRANSMIT</td>
<td>15</td>
</tr>
<tr>
<td>FREQUENCY GENERATOR (6-BIT PPG)</td>
<td></td>
</tr>
<tr>
<td>LIQUID CRYSTAL DISPLAY (LCD) CONTROLLER-DRIVER</td>
<td>16</td>
</tr>
<tr>
<td>APPENDIX</td>
<td>APPENDIX</td>
</tr>
<tr>
<td>INDEX</td>
<td>INDEX</td>
</tr>
</tbody>
</table>
CONTENTS

Chapter 1 OVERVIEW .......................................................................................................... 1
  1.1 MB89160/160A/160L Series Features ................................................................. 2
  1.2 MB89160/160A/160L Series Series ................................................................. 4
  1.3 Differences between Products ......................................................................... 6
  1.4 Block Diagram of MB89160/160A/160L Series .............................................. 7
  1.5 Pin Assignment ................................................................................................. 8
  1.6 Package Dimensions .......................................................................................10
  1.7 I/O Pins and Pin Functions .............................................................................15

Chapter 2 HANDLING DEVICES ......................................................................................... 21
  2.1 Notes on Handling Devices ........................................................................... 22

Chapter 3 CPU .................................................................................................................. 23
  3.1 Memory Space .................................................................................................. 24
    3.1.1 Special Areas ............................................................................................ 26
    3.1.2 Storing 16-bit Data in Memory .................................................................. 28
  3.2 Dedicated Registers ......................................................................................... 30
    3.2.1 Condition Code Register (CCR) .............................................................. 32
    3.2.2 Register Bank Pointer (RP) ..................................................................... 34
  3.3 General-purpose Registers .............................................................................. 36
  3.4 Interrupts ........................................................................................................... 38
    3.4.1 Interrupt Level Setting Registers (ILR1, ILR2, ILR3) .................................. 39
    3.4.2 Interrupt Processing .................................................................................. 40
    3.4.3 Multiple Interrupts ................................................................................... 42
    3.4.4 Interrupt Processing Time ......................................................................... 43
    3.4.5 Stack Operation during Interrupt Processing ......................................... 44
    3.4.6 Stack Area for Interrupt Processing ......................................................... 45
  3.5 Resets ............................................................................................................... 46
    3.5.1 External Reset Pin .................................................................................... 48
    3.5.2 Reset Operation ....................................................................................... 50
    3.5.3 Pin States during Reset ............................................................................ 52
  3.6 Clocks ............................................................................................................... 54
    3.6.1 Clock Generator ....................................................................................... 56
    3.6.2 Clock Controller ...................................................................................... 58
    3.6.3 System Clock Control Register (SYCC) ................................................... 60
    3.6.4 Clock Modes ........................................................................................... 62
    3.6.5 Oscillation Stabilization Delay Time ....................................................... 64
3.7 Standby Modes (Low-Power Consumption) .......................................................... 66
  3.7.1 Operating States in Standby Modes ................................................................. 67
  3.7.2 Sleep Mode .................................................................................................... 68
  3.7.3 Stop Mode ..................................................................................................... 69
  3.7.4 Watch Mode .................................................................................................. 70
  3.7.5 Standby Control Register (STBC) .................................................................. 72
  3.7.6 State Transition Diagram 1 (Options: Power-on Reset, Two Clocks) .......... 74
  3.7.7 State Transition Diagram 2 (Options: No Power-on Reset, Two Clocks) ...... 76
  3.7.8 State Transition Diagram 3 (One-clock Option) ........................................... 78
  3.7.9 Notes on Using Standby Modes ..................................................................... 80

3.8 Memory Access Mode .......................................................................................... 82

Chapter 4 I/O PORTS .......................................................................................................... 85
  4.1 Overview of I/O Ports ....................................................................................... 86
  4.2 Port 0 and Port 1 ............................................................................................... 88
    4.2.1 Port-0 and Port-1 Registers (PDR0, PDR1, DDR0, DDR1) ....................... 90
    4.2.2 Operation of Port 0 and Port 1 ................................................................. 92
  4.3 Port 2 ................................................................................................................. 94
    4.3.1 Port-2 Registers (PDR2, DDR2) ................................................................. 96
    4.3.2 Operation of Port 2 ................................................................................... 98
  4.4 Port 3 ................................................................................................................. 100
    4.4.1 Port-3 Register (PDR3) ............................................................................. 102
    4.4.2 Operation of Port 3 .................................................................................. 103
  4.5 Ports 4, 6 and 7 ................................................................................................. 104
    4.5.1 Port-4, Port-6, and Port-7 Registers (PDR4, PDR6, and PDR7) ............ 106
    4.5.2 Operation of Port 4, Port 6 and Port 7 ..................................................... 107
  4.6 Port 5 ................................................................................................................. 108
    4.6.1 Port-5 Register (PDR5) ............................................................................. 110
    4.6.2 Operation of Port 5 .................................................................................. 111
  4.7 Program Example for I/O Ports ....................................................................... 112

Chapter 5 TIMEBASE TIMER ........................................................................................... 113
  5.1 Overview of Timebase Timer ........................................................................... 114
  5.2 Block Diagram of Timebase Timer ................................................................. 115
  5.3 Timebase Timer Control Register (TBTC) ...................................................... 116
  5.4 Timebase Timer Interrupt ................................................................................ 118
  5.5 Operation of Timebase Timer ............................................................................ 120
  5.6 Notes on Using Timebase Timer ...................................................................... 122
Chapter 6  WATCHDOG TIMER ................................................................. 125
6.1 Overview of Watchdog Timer .............................................................. 126
6.2 Block Diagram of Watchdog Timer .......................................................... 128
6.3 Watchdog Timer Control Register (WDTC) ........................................ 130
6.4 Operation of Watchdog Timer ............................................................... 131
6.5 Notes on Using Watchdog Timer ........................................................... 132
6.6 Program Example for Watchdog Timer ................................................. 133

Chapter 7  8-BIT PWM TIMER ........................................................................... 135
7.1 Overview of 8-bit PWM Timer ............................................................... 136
7.2 Block Diagram of 8-bit PWM Timer ...................................................... 138
7.3 Structure of 8-bit PWM Timer 1 .............................................................. 140
7.3.1 PWM1 Control Register (CNTR1) ....................................................... 142
7.3.2 PWM 1 Compare Register (COMR1) .................................................. 144
7.4 Structure of 8-bit PWM Timer 2 .............................................................. 146
7.4.1 PWM 2 Control Register (CNTR2) ....................................................... 148
7.4.2 PWM 2 Compare Register (COMR2) .................................................. 150
7.5 8-bit PWM Timer Interrupts ................................................................. 151
7.6 Operation of Interval Timer Function ................................................... 152
7.7 Operation of PWM Timer Function ....................................................... 153
7.8 States in Each Mode during 8-bit PWM Timer Operation ..................... 154
7.9 Notes on Using 8-bit PWM Timer ........................................................... 156
7.10 Program Example for 8-bit PWM Timer .............................................. 158

Chapter 8  8/16-BIT TIMER/COUNTER ............................................................................. 161
8.1 Overview of 8/16-bit Timer/Counter ...................................................... 162
8.2 Block Diagram of 8/16-bit Timer/Counter ............................................. 164
8.3 Structure of 8/16-bit Timer/Counter ...................................................... 166
8.3.1 Timer 1 Control Register (T1CR) ....................................................... 168
8.3.2 Timer 2 Control Register (T2CR) ....................................................... 170
8.3.3 Timer 1 Data Register (T1DR) ........................................................... 172
8.3.4 Timer 2 Data Register (T2DR) ........................................................... 173
8.4 8/16-bit Timer/Counter Interrupt ......................................................... 174
8.5 Operation of Interval Timer Function .................................................... 176
8.6 Operation of Counter Function ............................................................. 178
FIGURES

Chapter 1 OVERVIEW ...................................................................................................... 1
   Figure 1.4a MB89160/160A/160L Series Overall Block Diagram ............. 7
   Figure 1.5a FPT-80P-M05 and FPT-80P-M11 Pin Assignment .................. 8
   Figure 1.5b FPT-80P-M06, FPT-80C-A02, and MQP-80C-P01 Pin Assignment . 9
   Figure 1.6a FPT-80P-M05 Package Dimensions ....................................... 10
   Figure 1.6b FPT-80P-M06 Package Dimensions ....................................... 11
   Figure 1.6c FPT-80P-M11 Package Dimensions ....................................... 12
   Figure 1.6d FPT-80C-A02 Package Dimensions ....................................... 13
   Figure 1.6e MQP-80C-P01 Package Dimensions ....................................... 14

Chapter 2 HANDLING DEVICES ................................................................................. 21

Chapter 3 CPU ................................................................................................................23
   Figure 3.1a Memory Map ........................................................................ 24
   Figure 3.1.2a Storing 16-bit Data in Memory ........................................... 28
   Figure 3.1.2b Byte Order of 16-bit Data in an Instruction ............................ 28
   Figure 3.2a Dedicated Register Configuration ......................................... 30
   Figure 3.2.1a Structure of Condition Code Register ................................. 32
   Figure 3.2.1b Change of Carry Flag by Shift Instruction ........................... 32
   Figure 3.2.2a Structure of Register Bank Pointer ..................................... 34
   Figure 3.2.2b Rule for Conversion of Actual Addresses of General-purpose Register Area 34
   Figure 3.3a Register Bank Structure ............................................................ 36
   Figure 3.4.1a Structure of Interrupt Level Setting Registers ....................... 39
   Figure 3.4.2a Interrupt Processing ............................................................... 40
   Figure 3.4.3a Example of Multiple Interrupts ............................................. 42
   Figure 3.4.4a Interrupt Processing Time ...................................................... 43
   Figure 3.4.5a Stack Operation at Start of Interrupt Processing ..................... 44
   Figure 3.4.6a Stack Area for Interrupt Processing ...................................... 45
   Figure 3.5.1a Block Diagram of External Reset Pin ........................................ 48
   Figure 3.5.2a Reset Operation Flow Diagram ............................................. 50
   Figure 3.6a Clock Supply Map .................................................................... 55
   Figure 3.6.1a Connection Example for a Crystal or Ceramic Resonator ........ 56
   Figure 3.6.1b Connection Example for CR ................................................... 56
   Figure 3.6.1c Connection Example for External Clock .............................. 57
   Figure 3.6.2a Block Diagram of Clock Controller ....................................... 58
   Figure 3.6.3a Structure of System Clock Control Register (SYCC) ............... 60
   Figure 3.6.5a Operation of Oscillator after Starting Oscillation .................. 64
   Figure 3.7.5a Standby Control Register (STBC) .......................................... 72
   Figure 3.7.6a State Transition Diagram 1 (Options: Power-on Reset, Two Clocks) 74
Figure 3.7.7a State Transition Diagram 2 (Options: Without Power-on Reset, Two Clocks) ................................................................. 76
Figure 3.7.8a State Transition Diagram 3 (Products with Power-on Reset) .......... 78
Figure 3.7.8b State Transition Diagram 3 (Products without Power-on Reset) .... 78
Figure 3.8a Mode Data Structure ....................................................................... 82
Figure 3.8b Memory Access Selection Operation ................................................ 83

Chapter 4  I/O PORTS ...................................................................................................... 85
Figure 4.2a Block Diagram of Port-0 and Port-1 Pin .......................................... 89
Figure 4.3a Block Diagram of Port-2 Pin .......................................................... 95
Figure 4.4a Block Diagram of Port-3 Pin (Pins P30 and P31) .......................... 101
Figure 4.4b Block Diagram of Port-3 Pin (Pins P32 and P33) ......................... 101
Figure 4.5a Block Diagram of Port-4, 6 and 7 Pin ............................................ 105
Figure 4.6a Block Diagram of Port-5 Pin .......................................................... 109
Figure 4.7a Connection Example for an Eight Segment LED ........................... 112

Chapter 5  TIMEBASE TIMER ....................................................................................... 113
Figure 5.2a Block Diagram of Timebase Timer ................................................ 115
Figure 5.3a Timebase Timer Control Register (TBTC) ..................................... 116
Figure 5.5a Interval Timer Function Settings .................................................... 120
Figure 5.5b Operation of Timebase Timer ....................................................... 121
Figure 5.6a Effect on Buzzer Output of Clearing Timebase Timer ................... 122

Chapter 6  WATCHDOG TIMER .................................................................................... 125
Figure 6.2a Block Diagram of Watchdog Timer ................................................ 128
Figure 6.3a Watchdog timer Control Register (WDTC) .................................... 130
Figure 6.4a Watchdog Timer Clear and Interval Time ...................................... 131

Chapter 7  8-BIT PWM TIMER ....................................................................................... 135
Figure 7.1a Example D/A Converter Configuration Using PWM Output and Low Pass Filter ......................................................................................... 137
Figure 7.2a Block Diagram of 8-bit PWM Timer .............................................. 138
Figure 7.3a Block Diagram of 8-bit PWM Timer Pin ....................................... 140
Figure 7.3b 8-bit PWM1 Timer Registers .......................................................... 141
Figure 7.3.1a PWM1 Control Register (CNTR 1) .............................................. 142
Figure 7.3.2a PWM1 Compare Register (COMR1) ............................................ 144
Figure 7.4a Block Diagram of 8-bit PWM Timer 2 Pin ................................... 146
Figure 7.4b 8-bit PWM Timer 2 Registers ....................................................... 147
Figure 7.4.1a PWM 2 Control Register (CNTR2) ............................................ 148
Figure 7.4.2a PWM2 Compare Register (CONTR2) ........................................ 150
Figure 7.6a Interval Timer Function Settings .................................................... 152
Figure 7.6b Operation of 8-bit PWM Timer ..................................................... 152
Figure 13.5a  A/D Conversion Function (Software Activation) Settings .......... 250
Figure 13.5b  A/D Conversion Function (Continuous Activation) Settings ........ 250
Figure 13.5c  Sense Function (Software Activation) Settings ..................... 251
Figure 13.5d  Sense Function (Continuous Activation) Settings .................. 251
Figure 13.6a  Analog Input Equivalent Circuit .......................................... 252

Chapter 14  WATCH PRESCALER ...................................................................... 257
Figure 14.2a  Block Diagram of Watch Prescaler ........................................... 259
Figure 14.3a  Watch Prescaler Control Register (WPCR) ............................... 260
Figure 14.5a  Interval Timer Function Settings ............................................... 264
Figure 14.5b  Operation of Watch Prescaler .................................................. 265
Figure 14.6a  Effect on Buzzer Output Due to Clearing of Watch Prescaler ...... 267

Chapter 15  REMOTE CONTROL TRANSMIT OUTPUT GENERATOR (6-BIT PPG) . 269
Figure 15.2  Block Diagram of Remote Control Transmit Output Generator ..... 274
Figure 15.3a  Block Diagram of P30/BZ/RCO Pin ......................................... 276
Figure 15.3b  Remote Control Transmit Generator Registers ....................... 276
Figure 15.3.1 Remote Control Register 1 (RCR1) ........................................... 277
Figure 15.3.1 Remote Control Register 2 (RCR2) .......................................... 278
Figure 15.4a  Remote Control Transmit Output Generator Settings ............... 279
Figure 15.4b  Operation of Remote Control Transmit Output Generator ........... 279
Figure 15.5a  Changing Settings during Operation (Remote Control) .............. 281
Figure 15.5b  Error during activating Operation (Remote Control) ................. 281

Chapter 16  LCD CONTROLLER/DRIVER ............................................................... 283
Figure 16.2a  Block Diagram of LCD Controller/Driver ............................... 286
Figure 16.2.1a Reference Voltage Generator/Voltage Booster
                  External Connections ................................................................. 289
Figure 16.2.2a  Internal Voltage Divider Equivalent Circuit .......................... 290
Figure 16.2.2b  Use of Internal Voltage Divider Resistors ............................ 291
Figure 16.2.2c  Use of Internal Voltage Divider Resistors with
                  Brightness Adjustment .............................................................. 291
Figure 16.2.3a  External Voltage Divider Resistor Connections .................... 292
Figure 16.2.3b  External Voltage Divider Resistor Connections .................... 293
Figure 16.3a  Block Diagram of LCD Controller/Driver Pin
                  (Dedicated Common/Segment Output Pins) .................................... 294
Figure 16.3b  Block Diagram of LCD Controller-Driver Pin
                  (Dual Function Common/Segment Output Pins) ............................. 295
Figure 16.3c  LCD Controller/Driver Register ............................................. 295
Figure 16.3.1a LCD Control Register 1 (LCR1) ............................................ 296
Figure 16.3.2a Segment/Common Output Pins and Corresponding Display RAM 298
# TABLES

## Chapter 1  OVERVIEW
- Table 1.2a MB89160/160A/160L Series Product Lineup ........................................ 4
- Table 1.2b MB89160/160A/160L Series Series CPU and Peripheral Functions ..... 5
- Table 1.3a Package and Corresponding Products ................................................... 6
- Table 1.7a Pin Description ...................................................................................... 15
- Table 1.7b Pin Description (Continued) ................................................................. 16
- Table 1.7c Pin Description (Continued) ................................................................. 17
- Table 1.7d Pin Description for External EPROM Pin (MB89PV160 only) .......... 18
- Table 1.7e I/O Circuit Type ..................................................................................... 19

## Chapter 2  HANDLING DEVICES ............................................................................. 21

## Chapter 3  CPU ........................................................................................................ 23
- Table 3.1.1a General-purpose Register Areas .......................................................... 26
- Table 3.1.1b Vector Table ...................................................................................... 27
- Table 3.2.1a Interrupt Level ....................................................................................... 33
- Table 3.4a Interrupt Request and Interrupt Vector ................................................. 38
- Table 3.4.1a Interrupt Level Setting Bit and Interrupt Level ....................................... 39
- Table 3.5a Reset Source ........................................................................................ 46
- Table 3.5b Reset Source and Oscillation Stabilization Delay Time ........................ 47
- Table 3.6.3a System Clock Control Register (SYCC) Bits ..................................... 61
- Table 3.6.4a Clock Mode Operating States ............................................................... 62
- Table 3.6.5a Main Clock Startup Conditions vs. Oscillation Stabilization Delay Time 65
- Table 3.7.1a Operating States of the CPU and Peripheral Functions in Standby Modes .................................................. 67
- Table 3.7.5a Standby Control Register (STBC) Bits ................................................. 73
- Table 3.7.6a Changing to/wake-up from Clock Modes ........................................... 75
- Table 3.7.7a Changing to/wake-up from Clock Modes .......................................... 77
- Table 3.7.8a Changing to Main Clock Mode Run State and Reset .......................... 79
- Table 3.7.9a Standby Control Register (STBC) Low-Power Consumption Mode Settings .......................................................... 81
- Table 3.8a Mode Pin Setting ................................................................................... 82
Table 3.8b Mode Pins and Mode Data ................................................................. 83

Chapter 4 I/O PORTS ........................................................................................................ 85
Table 4.1a Port Function ......................................................................................... 87
Table 4.1b Port Registers ....................................................................................... 87
Table 4.2a Port-0 and Port-1 Pins .......................................................................... 88
Table 4.2b Correspondence between Pin and Register for Port-0 and Port-1 ........ 89
Table 4.2.1a Port-0 and Port-1 Register Function .................................................. 91
Table 4.2.2a Port-0 and Port-1 Pin State ................................................................. 93
Table 4.3a Port-2 Pin ............................................................................................ 94
Table 4.3b Correspondence between Pin and Register for Port 2 ........................... 95
Table 4.3.1a Port-2 Register Function ................................................................. 97
Table 4.3.2a Port-2 Pin State .................................................................................. 99
Table 4.4a Port-3 Pin ........................................................................................... 100
Table 4.4b Correspondence between Pin and Register for Port 3 ........................... 101
Table 4.4.1a Port-3 Register Function ................................................................. 102
Table 4.4.2a Port-3 Pin State .................................................................................. 103
Table 4.5a Port-4, 6, and 7 Pin .......................................................................... 104
Table 4.5b Correspondence between Pin and Register for Port 4, 6, and 7 ......... 105
Table 4.5.1a Port-4, Port-6, and Port-7 Register Function ....................................... 106
Table 4.5.2a Port-4, 6, and 7 Pin State ................................................................. 107
Table 4.6a Port-5 Pin ........................................................................................... 108
Table 4.6b Correspondence between Pin and Register for Port 5 ........................... 109
Table 4.6.1a Port-5 Register Function ................................................................. 110
Table 4.6.2a Port-5 Pin State .................................................................................. 111

Chapter 5 TIMEBASE TIMER ...................................................................................... 113
Table 5.1a Timebase timer Interval time ............................................................... 114
Table 5.1b Clocks Supplied by Timebase Timer ................................................... 114
Table 5.3a Timebase Timer Control Register (TBTC) Bits .................................... 117
Table 5.4a Register and Vector Table for Timebase Timer Interrupt .................... 118

Chapter 6 WATCHDOG TIMER ................................................................................... 125
Table 6.1a Watchdog Timer Interval Time ............................................................ 126
Table 6.3a Watchdog Timer Control Register (WDTC) Bits ................................. 130

Chapter 7 8-BIT PWM TIMER ................................................................................... 135
Table 7.1a Interval Time and Square Wave Output Range .................................. 136
Table 7.1b Available PWM Wave Cycle for PWM Timer Function .................... 137
Table 7.3.1a PWM 1 Control Register (CNTR1) bit ............................................ 143
Table 7.4.1a PWM 2 Control Register (CNTR2) Bits ........................................... 149
Table 7.5a Registers and Vector Tables for 8-bit PWM Timer Interrupts .......... 151
Chapter 8 8/16-BIT TIMER/COUNTER ............................................................................. 161
Table 8.1a Timer 1 Interval Times and Square Wave Frequencies in 8-bit Mode 162
Table 8.1b Timer 2 Interval Times and Square Wave Frequencies in 8-bit Mode 162
Table 8.1c Interval Times and Square Wave Frequencies 16-bit mode 163
Table 8.3.1a Timer 1 Control Register (T1CR) Bits ................................................. 169
Table 8.3.2a Timer 2 Control Register (T2CR) Bits ................................................. 171
Table 8.4a 8/16-bit Timer/Counter Interrupt Control Bits and Interrupts ............... 174
Table 8.4b Registers and Vector Table for 8/16-bit Timer/Counter Interrupt ......... 175
Table 8.7a Square Wave Output Initial Setting Procedure (T1CR Register) 180
Table 8.8a Timer Stop and Restart ................................................................. 181

Chapter 9 8-BIT SERIAL I/O ............................................................................................. 189
Table 9.1a Shift Clock Cycle and Transfer Speed ................................................ 190
Table 9.3.1a Serial Mode Register (SMR) Bits .................................................... 197
Table 9.4a Register and Vector Table for 8-bit Serial I/O Interrupts ................. 199

Chapter 10 BUZZER OUTPUT .......................................................................................... 211
Table 10.1a Output Frequency ........................................................................ 212
Table 10.4a Buzzer Register (BZCR) Bits .......................................................... 215

Chapter 11 EXTERNAL INTERRUPT CIRCUIT 1 (EDGE) ................................................. 217
Table 11.3a External Interrupt Circuit 1 Pins .................................................... 220
Table 11.3.1a External Interrupt 1 Control Register (EIE1) Bits vs. Interrupts Pins 222
Table 11.3.1b External Interrupt 1 Control Register (EIE1) Bits 222
Table 11.3.2a External Interrupt 1 Flag Register (EIF1) Bits 223
Table 11.3.2b External Interrupt 1 Flag Register (EIF1) Bits 223
Table 11.4a Register and Vector Table for External Interrupt Circuit 1 Interrupts 224

Chapter 12 EXTERNAL INTERRUPT CIRCUIT 2 (LEVEL) ................................................. 227
Table 12.3a External Interrupt Circuit 2 Pins .................................................... 230
Table 12.3.1a External Interrupt 2 Control Register (EIE2) Bits vs. Pins 232
Table 12.3.1b External Interrupt 2 Control Register (EIE2) Bits 232
Table 12.3.2a External Interrupt 2 Flag Register (EIF2) Bits 233
Table 12.4a Registers and Vector Table for External Interrupt Circuit 2 Interrupts 234

Chapter 13 A/D CONVERTER .......................................................................................... 237
Table 13.3.1a A/D Control Register 1 (ADC1) Bits ............................................ 245
Table 13.3.2a A/D Control Register 2 (ADC2) Bits ............................................ 247
Table 13.3.3a Example of ADCD Register Setting for Sense Function .......... 248
Table 13.4a Register and Vector Table for A/D Converter Interrupt ............... 249

Chapter 14 WATCH PRESCALER ................................................................................... 257
Table 14.1a Watch Prescaler Interval Time ...................................................... 258
Table 14.1b Clocks Supplied by Watch Prescaler .............................................. 258
Table 14.3a Watch Prescaler Control Register (WPCR) Bits ............................... 261
Table 14.4a Register and Vector for Watch Prescaler Interrupt ........................... 262

Chapter 15 REMOTE CONTROL TRANSMIT OUTPUT GENERATOR (6-BIT PPG) ..... 269
Table 15.1a Output Cycles and "H" Pulse Width Ranges ..................................... 270
Table 15.1b 6-Bit PPG Resolution and Output Cycles (0.5 tinst count clock) ...... 272
Table 15.1c 6-Bit PPG Resolution and Output Cycles ......................................... 273
Table 15.3.1 Remote Control Register 1 (RCR1) Bits ........................................ 277
Table 15.3.1 Remote Control Register 2 (RCR2) Bits ........................................ 278

Chapter 16 LCD CONTROLLER/DRIVER ................................................................ 283
Table 16.1a Bias and Duty Ratio Combinations .................................................. 284
Table 16.2.3a LCD Drive Voltages and Biasing Modes ...................................... 292
Table 16.3.1a LCDC Control Register (LCR1) Bit Functions ............................... 297
Table 16.3.2a Segment Outputs, Display RAM Locations, and Sharing Port Pins 299
Table 16.3.2b Common Outputs and Display RAM Bits Used in Each Duty Ratio Mode ................................................................. 299
Table 16.4.1a Display RAM Contents Example .................................................. 302
Table 16.4.2a Display RAM Contents Example .................................................. 304
Table 16.4.3a Display RAM Contents Example .................................................. 306

APPENDIX ................................................................................................................. 309
Table Aa I/O Map ....................................................................................... 310
Table Ab I/O Map (Continued) ................................................................. 311
Table Ba Instruction List Symbols ................................................................. 312
Table Bb Instruction List Columns ................................................................. 313
Table B.3a Vector Table Address Correponding to “vct” ................................. 316
Table B.6a F2MC-8L Instruction Map .............................................................. 323
Table B.7a Bus Operation for Bit Manipulation Instructions ............................ 324
Table Ca Mask Options .................................................................................. 326
Table Cb Mask Options (Segment Options) .................................................. 327
Table Cc Versions ...................................................................................... 327
Table Cd Ordering Information ................................................................. 327
Table Da EPROM Programmer Socket Adaptor ............................................. 328
Table Db OTPROM Option Bit Map (MB89P165 and MB89W165) ................. 329
Table Ea Pin States in Each Mode ................................................................. 332
This chapter describes the main features and basic specifications of the MB89160/160A/160L Series.

1.1 MB89160/160A/160L Series Features ......................... 2
1.2 MB89160/160A/160L Series Series .......................... 4
1.3 Differences between Products ............................... 6
1.4 Block Diagram of MB89160/160A/160L Series .......... 7
1.5 Pin Assignment .................................................... 8
1.6 Package Dimensions .......................................... 10
1.7 I/O Pins and Pin Functions ................................. 15
1.1 MB89160/160A/160L Series Features

The MB89160/160A/160L Series is a line of the general-purpose, single-chip microcontrollers. In addition to a compact instruction set, the microcontrollers contain a variety of peripheral functions such as an LCD controller/driver, an A/D converter, timers, a serial interface, PWM timers, and external interrupts.

- Various package options
  - Two types of QFP packages (0.65-mm or 0.8-mm lead pitch)
  - SQFP package (0.5-mm lead pitch)

- High speed processing at low voltage
  Minimum execution time: 0.95μs/4.2 MHz

- F2MC-8L family CPU core
  Instruction set optimized for controllers
  - Multiplication and division instructions
  - 16-bit arithmetic operations
  - Test and branch instructions
  - Bit manipulation instructions, etc.

- Dual-clock control system
  - Main clock: max. 4.2 MHz (Four selectable speeds; oscillation stopped in subclock mode.)
  - Subclock: 32.768 kHz (Operating clock used in subclock mode)

- Six types of timers
  - 8-bit PWM timer 1 (also usable as an interval timer)
  - 8-bit PWM timer 2 (also usable as an interval timer)
  - 8/16-bit timer/counter (8 bits × 2 channels or 16 bits × 1 channel)
  - 21-bit timebase timer
  - Watch prescaler (15 bits)

- Serial interfaces (serial I/O)
  - Switchable transfer direction (MSB first or LSB first) allows communication with various equipment.

- A/D converter
  - Sense function enabling voltage comparison at 11.4 μs (at 4.2 MHz).
  - Activation by an 8/16-bit timer/counter output capable

- Remote control transmission output
  - Program-selectable pulse width and period.

- LCD controller/driver
  - 24 segments × 4 commons (max. 96 pixels)
  - LCD driving reference voltage generator and booster (MB89160A series only).
• **External interrupts (wake-up function)**
  - External Interrupt 1 (4 channels)
    Four channels are independent and capable of wake-up from low-power consumption modes (with an edge detection function).
  - External Interrupt 2 (8 inputs, 1 channel)
    Eight channels are independent and capable of wake-up from low-power consumption modes (with a “L” level detection function).

• **Standby modes (low power modes)**
  - Stop mode (If in sub mode, oscillation stops to minimize the current consumption.)
  - Sleep mode (The CPU stops to reduce the current consumption to approx. 1/3 of normal.)
  - Watch mode (Everything except the watch prescaler stops to reduce the power consumption to an extremely low level.)

• **I/O ports: max. 54 channels**
  - General-purpose I/O ports (N-ch open-drain): 8
  - Output-only ports (N-ch open-drain): 28
  - General-purpose I/O ports (CMOS): 16
  - Output-only ports (CMOS): 2
1.2 MB89160/160A/160L Series

The MB89160/160A/160L Series contains 9 types of products. Table 1.2a lists the product lineup and Table 1.2b lists the CPU and peripheral functions.

### MB89160/160A/160L Series Product Lineup

<table>
<thead>
<tr>
<th>Part number</th>
<th>MB89161</th>
<th>MB89163</th>
<th>MB89165</th>
<th>MB89163L</th>
<th>MB89165L</th>
<th>MB89P165*2</th>
<th>MB89W165*2</th>
<th>MB89PV160</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Classification</strong></td>
<td>Mass production products (mask ROM products)</td>
<td>One-time PROM products</td>
<td>EPROM product</td>
<td>Piggyback/evaluation product for evaluation and development</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>ROM size</strong></td>
<td>4K x 8 bits (Internal mask ROM)</td>
<td>8K x 8 bits (Internal mask ROM)</td>
<td>16K x 8 bits (Internal mask ROM)</td>
<td>8K x 8 bits (Internal mask ROM)</td>
<td>16K x 8 bits (Internal PROM, programming with general-purpose EPROM programmer)</td>
<td>32 K x 8 bits (External ROM)</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>RAM size</strong></td>
<td>128 x 8 bits</td>
<td>256 x 8 bits</td>
<td></td>
<td></td>
<td>512 x 8 bits</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Low-power consumption (stand by modes)</strong></td>
<td>Sleep mode, stop mode, and watch mode</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Process</strong></td>
<td>CMOS</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Operating voltage</strong></td>
<td>2.2 V to 6.0 V*3 (3.5 V to 6.0 V when using the A/D converter)*4</td>
<td>2.2 V to 3.6 V (2.7 V to 3.6 V when using the A/D converter)</td>
<td>2.7 V to 6.0 V*3</td>
<td>2.7 V to 6.0 V*3</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*1: Products with an internal booster.
*2: Products with an internal booster specified by version number. (See Appendix C, “Mask Options”).
*3: Varies with conditions such as operating frequencies. If dual-clock option is used, voltage range is 2.2 to 4.0 V. (Operation above 4.0 V is supported separately.)
*4: Operation below 3.5 V supported separately.
*5: Use MBM27C256A-20 as the external ROM (operating voltage: 4.5 V to 5.5 V).
Table 1.2b MB89160/160A/160L Series CPU and Peripheral Functions

<table>
<thead>
<tr>
<th>Parameter</th>
<th>MB89163/163A/163L</th>
<th>MB89161/161A</th>
<th>MB89165/165A</th>
<th>MB89165L</th>
<th>MB89P165</th>
<th>MB89W165</th>
<th>MB89PV160</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>CPU functions</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Number of instructions:</td>
<td>136</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instruction bit length:</td>
<td>8 bits</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Instruction length:</td>
<td>1 to 3 bytes</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Data bit length:</td>
<td>1, 8, 16 bits</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Minimum execution time:</td>
<td>0.95 µs to 15.2 µs/4.2 MHz, 61.0 µs/32.768 kHz</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Ports</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>General-purpose I/O ports (N-ch open-drain):</td>
<td>8 (6 ports also serve as peripherals, 3 ports are a heavy-current drive type.)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output-only ports (N-ch open-drain):</td>
<td>8 (10 ports also serve as peripherals)<em>, 16 ports serve as segment pins and 2 ports serve as common pins</em>)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>General-purpose I/O ports (CMOS):</td>
<td>16 (12 ports also serve as an external interrupt)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output-only ports (CMOS):</td>
<td>2 (Also serve as peripherals)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Total</td>
<td>54 (max.)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>21-bit timebase timer</strong></td>
<td>21 bits</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interrupt cycle:</td>
<td>1.95 ms, 7.80 ms, 62.41 ms, 996.64 ms/4.2 MHz for main clock</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Watchdog timer</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Reset generate cycle:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>min. 998.6 ms/4.2 MHz for main clock</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>min. 500 ms/32.768 kHz for subclock</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>8-bit PWM timer 1, PWM timer 2</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8-bit resolution PWM operation (conversion cycle: 243.8 µs to 63.9 s)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8/16-bit timer/counter for counter clock selectability</td>
<td></td>
<td></td>
<td></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>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Can be operated either as a 2-channel 8-bit timer/counter (Timer 1 and Timer 2, each with its own independent operating cycle), or as one 16-bit timer/counter (operating cycle: 1.90 µs to 487.6 µs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>In Timer 1 or 16-bit timer/counter operation, event counter operation (external clock-triggered) and square wave output capable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>8-bit serial I/O</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 bits</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LSB first/MSB first selectability</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>One clock selectable from four transfer clocks (one external shift clock, three internal shift clocks: 1.9 µs, 7.6 µs, 30.5 µs)</td>
<td></td>
<td></td>
<td></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>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Output frequency:</td>
<td>1025 Hz, 2051 Hz, 4102 Hz, 8203 Hz/4.2 MHz for main clock</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>1024 Hz, 2048 Hz, 4096 Hz/32.768 kHz for subclock</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>External interrupt 1 (wake-up function)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 independent channels (interrupt vector, request flag, request output enable)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Edge selectability (rising/falling)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Used also for wake-up from stop/sleep mode. (Low-level detection is also permitted in stop mode.)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>External interrupt 2 (wake-up function)</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8 inputs, one channel (&quot;L&quot; level interrupts, independent input enable)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Used also for wake-up from stop/sleep mode. (Low-level detection is also permitted in stop mode.)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>8-bit A/D converter</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8-bit resolution × 8 channels</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>A/D conversion function (conversion time: 41.9 µs for MB89160L series : conversion time is 49.5 µs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Sense function (comparison time: 11.4 µs)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Continuous activation by an 8/16-bit timer/counter output or a timebase timer output capable. Reference voltage input (AVR)</td>
<td></td>
<td></td>
<td></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>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15 bits</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Interrupt cycle:</td>
<td>31.25 ms, 0.25 s, 0.50 s, 1.00 s/32.768 kHz for subclock</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Remote control transmit output</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Internal 6 bit counter</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Pulse width (&quot;H&quot; level pulse width of 0 µs to 1920 µs) and cycle (0.95 µs to 1920 µs) are program selectable</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Can also be used as 6-bit PPG</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>LCD controller/driver</strong></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Common output:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Segment output:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4 (max.)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Without a booster for LCD driving, but has internal dividing resistors.</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LCD driving power (bias) pins:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>LCD display RAM size: 12 bytes (24 × 4 bits, max. 96 pixels)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Booster for LCD driving:</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Dividing resistor for LCD driving: Built-in (External dividing resistor connection capable in MB89160 series only)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Note:** Unless otherwise specified, values given for clock cycle, conversion times, etc. are for 4.2 MHz operation, with main clock maximum clock speed selected.

*1 In the MB89160A series, two of these pins are used to connect capacitors.

*2 The number of ports used for common/segment outputs is selected by mask option or version number.

*3 When booster is used, the bias is reduced by 1/3.
1.3 Differences between Products

This section describes the differences between the 9 products in the MB89160/160A/160L Series and lists points to note in product selection.

- Differences among Products and Points to Note for Product Selection

Table 1.3a Package and Corresponding Products

<table>
<thead>
<tr>
<th>Part number</th>
<th>MB89161</th>
<th>MB89161A</th>
<th>MB89163</th>
<th>MB89163A</th>
<th>MB89163L</th>
<th>MB89165</th>
<th>MB89P165</th>
<th>MB89W165</th>
<th>MB89PV160</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPT-80P-M05</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>FPT-80P-M06</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>FPT-80P-M11</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>○</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td></td>
</tr>
<tr>
<td>FPT-80C-A02</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>○</td>
<td>×</td>
<td>×</td>
<td>○</td>
</tr>
<tr>
<td>MQP-80C-P01</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>○</td>
<td>×</td>
<td>×</td>
<td>○</td>
</tr>
</tbody>
</table>

○: Available  ×: Not available

- Memory size

Before evaluating using the piggyback product, verify its differences from the product that will actually be used. Take particular care on the following points. (See section 3.1, “Memory Space.”):

- On the MB89161/161A, each general-purpose register area is limited to 0100H to 013FH (8 banks).
- On the MB89163/163A/163L, each general-purpose register area is limited to 0100H to 017FH (16 banks).
- The stack area, etc., is set at the upper limit of the RAM.

- Current consumption

- In the case of the MB89PV160 add the current consumed by the EPROM which is connected to the top socket.
- When operated at low speed, the product with a one-time PROM (OTPROM) or an EPROM will consume more current than the product with mask ROM. However the current consumption in sleep/stop modes, is the same.

Reference:
- For more information about the package, see Section 1.6, “Package Dimensions.”
- For more information about the current consumption, see the electrical characteristics in the Data Sheet.

- Mask options

Functions that can be selected as options and how to designate these options vary by the product. Before using, check Appendix C, “Mask Options.” Take particular care on the following points:

- A pull-up resistor cannot be set for P20 to P27, P40 to P47, or P60 to P67 on the MB89P165 and MB89W165.
- A pull-up resistor is not selectable for P40 to P47 and P60 to P67 if they are used as segment output pins.
- A pull-up resistor is not selectable for P50 to P57 of all products when an A/D converter is used.
- In the MB89P165, MB89W165, and MB89PV160, the number of common and segment outputs is specified by version number.
- Options are fixed on the MB89PV160. (See Appendix C, “Mask Options.”)

- Booster for LCD driving

- An internal booster is provided in the MB89160A series.
- In the MB89P165 and MB89W165, Products with an internal booster are specified by version number. (See Appendix C, “Mask Options.”)
- No internal booster is provided in the MB89PV160.
1.4  Block Diagram of MB89160/160A/160L Series

Figure 1.4a shows the block diagram of the MB89160/160A/160L Series.

- MB89160/160A/160L Series Series Block Diagram

*1: MB89160A series only. Internal dividing resistors in MB89160 series. (External dividing resistors connection capable).
*2: Used as connect capacitor pins (C0 and C1) in MB89160A series.
*3: Functions selected by mask option.
*4: Heavy-current drive type
*5: Can be used as a 16-bit timer/counter by connecting Timer 1 output to Timer 2 input.

Figure 1.4a MB89160/160A/160L Series Series Overall Block Diagram
1.5 Pin Assignment

Figure 1.5a and Figure 1.5b show the pin assignment diagrams for the MB89160/160A/160L Series.

- FPT-80P-M05 and FPT-80P-M11 Pin Assignment

*1: For products with a booster circuit (MB89160A series)
*2: For products without a booster circuit
*3: N-ch open-drain heavy-current drive type
*4: Selected by the mask option (in units of 4 pins)
*5: Selected by the mask option

Figure 1.5a FPT-80P-M05 and FPT-80P-M11 Pin Assignment
Figure 1.5b FPT-80P-M06, FPT-80C-A02, and MQP-80C-P01 Pin Assignment
1.6 Package Dimensions

Seven types of packages are available for MB89160/160A/160L Series. Figure 1.6a to Figure 1.6e show the package dimensions.

- FPT-80P-M05 Package Dimensions

<table>
<thead>
<tr>
<th>Dimension</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lead pitch</td>
<td>0.50 mm</td>
</tr>
<tr>
<td>Package width \times length</td>
<td>12 \times 12 mm</td>
</tr>
<tr>
<td>Lead shape</td>
<td>Gull-wing</td>
</tr>
<tr>
<td>Sealing method</td>
<td>Plastic mold</td>
</tr>
</tbody>
</table>

Figure 1.6a FPT-80P-M05 Package Dimensions
## FPT-80P-M06 Package Dimensions

<table>
<thead>
<tr>
<th>Details</th>
<th>Value</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Lead pitch</strong></td>
<td>0.80 mm</td>
</tr>
<tr>
<td><strong>Package width x length</strong></td>
<td>14 x 20 mm</td>
</tr>
<tr>
<td><strong>Lead shape</strong></td>
<td>Gull-wing</td>
</tr>
<tr>
<td><strong>Sealing method</strong></td>
<td>Plastic mold</td>
</tr>
<tr>
<td><strong>Length of flat section of pin</strong></td>
<td>0.80 mm</td>
</tr>
</tbody>
</table>

### Figure 1.6b FPT-80P-M06 Package Dimensions

- **80-pin Plastic QFP**
- **Lead pitch**: 0.80 mm
- **Package width x length**: 14 x 20 mm
- **Lead shape**: Gull-wing
- **Sealing method**: Plastic mold
- **Length of flat section of pin**: 0.80 mm

Dimensions in mm (inches): 23.90±0.40 (0.941±0.016)
### FPT-80P-M11 Package Dimensions

<table>
<thead>
<tr>
<th>Feature</th>
<th>Specification</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lead pitch</td>
<td>0.65 mm</td>
</tr>
<tr>
<td>Package width × length</td>
<td>14 × 14 mm</td>
</tr>
<tr>
<td>Lead shape</td>
<td>Gull-wing</td>
</tr>
<tr>
<td>Sealing method</td>
<td>Plastic mold</td>
</tr>
</tbody>
</table>

#### Figures

**Figure 1.6c FPT-80P-M11 Package Dimensions**
### FPT-80C-A02 Package Dimensions

<table>
<thead>
<tr>
<th>Lead pitch</th>
<th>0.8 mm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lead shape</td>
<td>Gull-wing</td>
</tr>
<tr>
<td>Sealing method</td>
<td>Metal seal</td>
</tr>
</tbody>
</table>

#### Dimensions in mm (inches)

![Diagram of FPT-80C-A02 Package Dimensions](image_url)

**Figure 1.6d FPT-80C-A02 Package Dimensions**

- Lead pitch: 0.8 mm
- Lead shape: Gull-wing
- Sealing method: Metal seal

Dimensions in mm (inches):

<table>
<thead>
<tr>
<th>Index Area</th>
<th>Dimensions</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.80±0.10</td>
<td>0.0315±.0040</td>
</tr>
<tr>
<td>0.25 ± 0.05</td>
<td>0.0100±.0020</td>
</tr>
</tbody>
</table>

**80-pin Ceramic QFP (FPT-80C-A02)**
### MQP-80C-P01 Package Dimensions

<table>
<thead>
<tr>
<th>Lead pitch</th>
<th>0.8 mm</th>
</tr>
</thead>
<tbody>
<tr>
<td>Lead shape</td>
<td>Straight</td>
</tr>
<tr>
<td>Motherboard material</td>
<td>Ceramic</td>
</tr>
<tr>
<td>Mounted socket material</td>
<td>Plastic</td>
</tr>
</tbody>
</table>

![80-pin Ceramic MQFP (MQP-80P-P01)](image)

© 1994 FUJITSU LIMITED MB9015C-4.2

Dimensions in mm (inches)

**Figure 1.6e MQP-80C-P01 Package Dimensions**
1.7 I/O Pins and Pin Functions

Table 1.7a and Table 1.7b list the MB89160/160A/160L Series I/O pins and their functions. Table 1.7e lists the I/O circuit types. The letter in the “I/O circuit type” column in Table 1.7a refers to the letter in the “Type” column Table 1.7c.

I/O Pins and Pin Functions

Table 1.7a Pin Description

<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>16</td>
<td>X0</td>
<td>A</td>
<td>Crystal or other resonator connector pins for the main clock. The external clock can be connected to X0. When this is done, be sure to leave X1 open. CR oscillation selectability in model with a mask ROM only.</td>
</tr>
<tr>
<td>15</td>
<td>X1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>MOD0</td>
<td>C</td>
<td>Memory access mode setting pins Connect directly to Vss.</td>
</tr>
<tr>
<td>17</td>
<td>MOD1</td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>RST</td>
<td>D</td>
<td>Reset I/O pin This pin is an N-ch open-drain output type with a pull-up resistor, and a hysteresis input type. “L” is output from this pin by an internal reset request (optional). The internal circuit is initialized by the input of “L”.</td>
</tr>
<tr>
<td>20 to 27</td>
<td>P00/INT20 to P07/INT27</td>
<td>E</td>
<td>General-purpose I/O ports Also serve as an external interrupt 2 input (wake-up function). External interrupt 2 input is hysteresis input.</td>
</tr>
<tr>
<td>28 to 31</td>
<td>P10/INT10 to P13/INT13</td>
<td>E</td>
<td>General-purpose I/O ports Also serve as input for external interrupt 1 input (wake-up function). External interrupt 1 input is hysteresis input.</td>
</tr>
<tr>
<td>32 to 35</td>
<td>P14 to P17</td>
<td>F</td>
<td>General-purpose I/O ports</td>
</tr>
<tr>
<td>36</td>
<td>P20/EC</td>
<td>H</td>
<td>N-ch open-drain general-purpose I/O port Also serve as the external clock input for the 8/16-bit timer/counter. The peripheral is a hysteresis input.</td>
</tr>
<tr>
<td>37</td>
<td>P21</td>
<td>I</td>
<td>N-ch open-drain general-purpose I/O port</td>
</tr>
<tr>
<td>38</td>
<td>P22/TO</td>
<td>I</td>
<td>N-ch open-drain general-purpose I/O port Also serves as an 8/16-bit timer/counter output.</td>
</tr>
<tr>
<td>39</td>
<td>P23/SI</td>
<td>H</td>
<td>N-ch open-drain general-purpose I/O port Also serves as the data input for the serial I/O. The peripheral is a hysteresis input type.</td>
</tr>
</tbody>
</table>

*1: FPT-80P-M05
*2: FPT-80P-M11
*3: MQP-80C-P01
*4: FPT-80P-M06, FPT-80C-A02
<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>40, 42</td>
<td>P24/SO</td>
<td>I</td>
<td>N-ch open-drain general-purpose I/O port. Also serves as the data output for the serial I/O.</td>
</tr>
<tr>
<td>41, 43</td>
<td>P25/SCK</td>
<td>H</td>
<td>N-ch open-drain general-purpose I/O port. Also serves as the clock I/O for the serial I/O. The peripheral is a hysteresis input type.</td>
</tr>
<tr>
<td>42, 44</td>
<td>P26</td>
<td>I</td>
<td>N-ch open-drain general-purpose I/O port.</td>
</tr>
<tr>
<td>43, 45</td>
<td>P27/PWM2</td>
<td>I</td>
<td>N-ch open-drain general-purpose I/O port. Also serves as the square wave or PWM wave output for the 8-bit PWM timer 2.</td>
</tr>
<tr>
<td>49, 51</td>
<td>P33</td>
<td>J</td>
<td>Functions as an N-ch open-drain general-purpose output port only in the products without a booster.</td>
</tr>
<tr>
<td></td>
<td>C0</td>
<td>—</td>
<td>Function as capacitor connection pin in the products with a booster (MB89160A series).</td>
</tr>
<tr>
<td>48, 50</td>
<td>P32</td>
<td>J</td>
<td>Functions as an N-ch open-drain general-purpose output port only in the products without a booster.</td>
</tr>
<tr>
<td></td>
<td>C1</td>
<td>—</td>
<td>Function as capacitor connection pin in the products with a booster (MB89160A series).</td>
</tr>
<tr>
<td>47, 49</td>
<td>P31/PWM1</td>
<td>G</td>
<td>General-purpose output-only port. Also serves as the square wave or PWM wave output for the 8-bit PWM timer 1.</td>
</tr>
<tr>
<td>46, 48</td>
<td>P30/BZ/ RCO</td>
<td>G</td>
<td>General-purpose output-only port. Also serves as buzzer output and a remote control transmit output.</td>
</tr>
<tr>
<td>6 to 12, 14</td>
<td>P50/AN0 to P57/AN7</td>
<td>L</td>
<td>N-ch open-drain general-purpose output ports. Also serve as the analog input for the A/D converter.</td>
</tr>
<tr>
<td>75 to 80, 1, 2</td>
<td>P40/SEG16 to P47/SEG23</td>
<td>J/K</td>
<td>N-ch open-drain general-purpose output ports. Also serve as an LCD controller/driver segment output. Switching between port and segment output is done by the mask option.</td>
</tr>
<tr>
<td>67 to 74</td>
<td>P60/SEG8 to P67/SEG15</td>
<td>J/K</td>
<td>N-ch open-drain general-purpose output ports. Also serve as an LCD controller/driver segment output. Switching between port and segment output is done by the mask option.</td>
</tr>
<tr>
<td>59 to 66</td>
<td>SEG0 to SEG7</td>
<td>K</td>
<td>LCD controller/driver segment output-only pins.</td>
</tr>
<tr>
<td>57, 58</td>
<td>P70/COM2, P71/COM3</td>
<td>J/K</td>
<td>N-ch open-drain general-purpose output ports. Also serve as an LCD controller/driver common output. Switching between port and common output is done by the mask option.</td>
</tr>
<tr>
<td>55, 56</td>
<td>COM0, COM1</td>
<td>K</td>
<td>LCD controller/driver common output-only pins.</td>
</tr>
</tbody>
</table>

*1: FPT-80P-M05
*2: FPT-80P-M11
*3: MQP-80C-P01
*4: FPT-80P-M06, FPT-80C-A02
**Table 1.7c Pin Description (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>50 to 52, 54</td>
<td>52 to 54, 56</td>
<td>V0 to V3</td>
<td>— LCD driving power supply pins.</td>
</tr>
<tr>
<td>44</td>
<td>46</td>
<td>X0A</td>
<td>B</td>
</tr>
<tr>
<td>45</td>
<td>47</td>
<td>X1A</td>
<td>—</td>
</tr>
<tr>
<td>53</td>
<td>55</td>
<td>Vcc</td>
<td>—</td>
</tr>
<tr>
<td>13</td>
<td>15</td>
<td>Vss</td>
<td>—</td>
</tr>
<tr>
<td>5</td>
<td>7</td>
<td>AVcc</td>
<td>—</td>
</tr>
<tr>
<td>4</td>
<td>6</td>
<td>AVR</td>
<td>—</td>
</tr>
<tr>
<td>3</td>
<td>5</td>
<td>AVss</td>
<td>—</td>
</tr>
</tbody>
</table>

*1: FPT-80P-M05  
*2: FPT-80P-M11  
*3: MQP-80C-P01  
*4: FPT-80P-M06, FPT-80C-A02
<table>
<thead>
<tr>
<th>Pin no.</th>
<th>Pin name</th>
<th>I/O</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>82</td>
<td>V_{PP}</td>
<td>O</td>
<td>“H” level output pin</td>
</tr>
<tr>
<td>83, 84, 85, 86, 87, 88, 89, 90, 91</td>
<td>A12, A7, A6, A5, A4, A3, A2, A1, A0</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>92, 93, 94, 95</td>
<td>O1, O2, O3</td>
<td>I</td>
<td>Data input pins</td>
</tr>
<tr>
<td>96</td>
<td>Vss</td>
<td>O</td>
<td>Power supply (GND) pin</td>
</tr>
<tr>
<td>98, 99, 100, 101, 102</td>
<td>O4, O5, O6, O7, O8</td>
<td>I</td>
<td>Data input pins</td>
</tr>
<tr>
<td>103</td>
<td>CE</td>
<td>O</td>
<td>ROM chip enable pin</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Outputs “H” during standby.</td>
</tr>
<tr>
<td>104</td>
<td>A10</td>
<td>O</td>
<td>Address output pin</td>
</tr>
<tr>
<td>105</td>
<td>CE</td>
<td>O</td>
<td>ROM output enable pin</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Outputs “L” at all times.</td>
</tr>
<tr>
<td>107, 108, 109</td>
<td>A11, A9, A8</td>
<td>O</td>
<td>Address output pins</td>
</tr>
<tr>
<td>110</td>
<td>A13</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td>111</td>
<td>A14</td>
<td>O</td>
<td></td>
</tr>
<tr>
<td>112</td>
<td>Vcc</td>
<td>O</td>
<td>EPROM power supply pin</td>
</tr>
<tr>
<td>81, 92, 97, 106</td>
<td>N.C.</td>
<td>—</td>
<td>Internally connected pins</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Be sure to leave them open.</td>
</tr>
</tbody>
</table>
### Table 1.7e I/O Circuit Type

<table>
<thead>
<tr>
<th>Type</th>
<th>Circuit</th>
<th>Remarks</th>
</tr>
</thead>
</table>
| A    | ![Circuit A](image) | Main clock (main clock crystal oscillator)  
• At an oscillation feedback resistor of approximately 1 MΩ  
• CR oscillation is selectable (MB8916X/16XA only) |
| B    | ![Circuit B](image) | Subclock (subclock crystal oscillator)  
• At an oscillation feedback resistor of approximately 4.5 MΩ |
| C    | ![Circuit C](image) |  
• Hysteresis input  
• The pull-down resistor (R) is approximately 50 kΩ for MB89165L only. |
| D    | ![Circuit D](image) |  
• At an output pull-up resistor (P-ch) of approximately 50 kΩ  
• Hysteresis input |
| E    | ![Circuit E](image) |  
• CMOS output  
• CMOS input  
• The peripheral is a hysteresis input type.  
• Pull-up resistor optional (except MB89PV160)  
Approximately 50 kΩ |
| F    | ![Circuit F](image) |  
• CMOS output  
• CMOS input  
• Pull-up resistor optional (except MB89PV160)  
Approximately 50 kΩ |
### Table 1.7e I/O Circuit Type (Continued)

<table>
<thead>
<tr>
<th>Type</th>
<th>Circuit</th>
<th>Remarks</th>
</tr>
</thead>
</table>
| G    | ![Circuit Diagram](image) | • CMOS output  
• P-ch output is a heavy-current drive type. |
| H    | ![Circuit Diagram](image) | • N-ch open-drain output  
• CMOS input  
• The peripheral is a hysteresis input type.  
• Pull-up resistor optional (except MB89P165, MB89W165, and MB89PV160)  
Approximately 50 kΩ |
| I    | ![Circuit Diagram](image) | • N-ch open-drain output  
• CMOS input  
• P21, P26, and P27 are a heavy-current drive type.  
• Pull-up resistor optional (except MB89P165, MB89W165, and MB89PV160)  
Approximately 50 kΩ |
| J    | ![Circuit Diagram](image) | • N-ch open-drain output  
• Pull-up resistor optional (other than P32 and P33, and except MB89P165, MB89W165, and MB89PV160)  
Approximately 50 kΩ |
| K    | ![Circuit Diagram](image) | • LCD controller/driver common/segment output |
| L    | ![Circuit Diagram](image) | • N-ch open-drain output  
• Analog input (A/D converter)  
• Pull-up resistor optional (except in MB89PV160. Pull-up resistors disable when used as an analog input).  
Approximately 50 kΩ |
CHAPTER 2
HANDLING DEVICES

This chapter describes points to note when using the general-purpose single-chip microcontroller.

2.1 Notes on Handling Devices ........................................ 22
2.1 Notes on Handling Devices

This section lists points to note regarding the power supply voltage, pins, and other device handling aspects.

- **Take great care not to exceed the maximum rated voltage (prevent latchup).**
  Latchup may occur on CMOS ICs if voltage higher than Vcc or lower than Vss is applied to input and output pins other than medium- to high-voltage pins, or if voltage higher than the ratings is applied between Vcc and Vss.
  When latchup occurs, power supply current increases rapidly and might thermally damage elements. When using, take great care not to exceed the absolute maximum ratings.
  Also, take care to prevent the analog power supply (AVcc and AVR) and analog input from exceeding the digital power supply (Vcc) when the analog system power supply is turned on and off.

- **Stabilizing supply voltage is important.**
  A rapid fluctuation of Vcc power supply voltage could cause malfunctions, even if it occurs within the operation assurance range of the voltage. As stabilization guidelines, it is recommended to control power so that Vcc ripple fluctuations (P-P value) will be less than 10% of the standard Vcc value at the commercial frequency (50 to 60 Hz) and the transient fluctuation rate will be less than 0.1 V/ms at the time of a momentary fluctuation such as when power is switched.

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

- **Treatment of N.C. pins**
  Be sure to leave (internally connected) N.C. pins open.

- **Treatment of power supply pins on microcontroller with A/D or D/A converters**
  Connect to be AVcc = DAVC = Vcc and AVss = AVR = Vss even if the A/D and D/A converters are not in use.

- **Precautions when using an external clock**
  Even when an external clock is used, oscillation stabilization delay time is required for power-on reset (optional) and wake-up from stop mode.
This chapter describes the functions and operation of the CPU.

3.1 Memory Space ............................................................. 24
3.2 Dedicated Registers ..................................................... 30
3.3 General-purpose Registers .......................................... 36
3.4 Interrupts ...................................................................... 38
3.5 Resets .......................................................................... 46
3.6 Clocks........................................................................... 54
3.7 Standby Modes (Low-Power Consumption) .................... 66
3.8 Memory Access Mode.................................................. 82
3.1 Memory Space

The microcontrollers of the MB89160/160A/160L series offer a memory space of 64 Kbytes. The memory space contains the I/O area, RAM area, ROM area, and external area. The memory space contains areas used for special purposes such as the general-purpose registers and vector table.

- Memory Space Structure
  - **I/O area (addresses: 0000H to 007FH)**
    - Control registers and data registers for the internal peripheral functions are located in this area.
    - As the I/O area is allocated within the memory space, I/O can be accessed in the same way as memory. High-speed access using direct addressing is available.
  - **RAM area**
    - Internal static RAM is provided as an internal data area.
    - The internal RAM size differs between products.
    - Addresses between 80H and FFH support high-speed access using direct addressing.
    - Addresses between 100H and 1FFH can be used as the general-purpose register area (restrictions apply for some products).
    - The contents of RAM is indeterminate after a reset.
  - **ROM area**
    - Internal ROM is provided as an internal program area.
    - The internal ROM size differs between products. Setting the memory access mode to external ROM mode enables internal ROM to be disconnected and set as an external area.
    - Addresses between FFC0H and FFFFH are used for the vector table, etc.

- Memory Map

![Figure 3.1a Memory Map](image-url)
Memo
3.1 Memory Space

3.1.1 Special Areas

In addition to the I/O area, the special purpose areas in the memory space include the general-purpose register area and the vector table area.

- **General-purpose Register Areas (Addresses: 0100H to 01FFH)**
  - Provides auxiliary registers for 8-bit arithmetic operation and transfer instructions.
  - Allocated to a region of the RAM area. Can also be used as normal RAM.
  - Using the area as general-purpose registers enables high-speed access by general-purpose register addressing using short instructions.

  Table 3.1.1a lists the areas in each device that can be used for general-purpose registers.

<table>
<thead>
<tr>
<th>CPU</th>
<th>MB89161</th>
<th>MB89161A</th>
<th>MB89163</th>
<th>MB89163A</th>
<th>MB89163L</th>
<th>MB89165</th>
<th>MB89165A</th>
<th>MB89P165</th>
<th>MB89W165</th>
<th>MB89PV160</th>
<th>MB89165L</th>
</tr>
</thead>
<tbody>
<tr>
<td>Number of Banks</td>
<td>8</td>
<td>16</td>
<td>32</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Address Range</td>
<td>0100H to 013FH</td>
<td>0100H to 017FH</td>
<td>0100H to 01FFH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Reference:** See section 3.2.2, “Register Bank Pointer (RP)” and section 3.3, “General-purpose Registers” for details.

- **Vector Table Area (Addresses: FFC0H to FFFFH)**
  - Used as the vector table for the vector call instruction, interrupts, and resets.
  - The vector table is allocated at the top of the ROM area. The start address of the corresponding processing routine is set as data at each vector table address.

  Table 3.1.1b lists the vector table addresses referenced by the vector call instruction, interrupts, and resets.

**Reference:** See Section 3.4, “Interrupts”, Section 3.5, “ Resets”, and “ (6) CALLV #vct” in Appendix B.2, “Special Instructions” for details.
### Table 3.1.1b Vector Table

<table>
<thead>
<tr>
<th>Vector call instruction</th>
<th>Vector table address</th>
<th>Interrupts</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td>CALLV #0</td>
<td>FFC0h</td>
<td>IRQB</td>
<td>FFE4h</td>
</tr>
<tr>
<td>CALLV #0</td>
<td>FFC1h</td>
<td>IRQA</td>
<td>FFE5h</td>
</tr>
<tr>
<td>CALLV #1</td>
<td>FFC2h</td>
<td>IRQ9</td>
<td>FFE6h</td>
</tr>
<tr>
<td>CALLV #2</td>
<td>FFC3h</td>
<td>IRQ8</td>
<td>FFE7h</td>
</tr>
<tr>
<td>CALLV #3</td>
<td>FFC4h</td>
<td>IRQ7</td>
<td>FFE8h</td>
</tr>
<tr>
<td>CALLV #4</td>
<td>FFC5h</td>
<td>IRQ6</td>
<td>FFE9h</td>
</tr>
<tr>
<td>CALLV #5</td>
<td>FFC6h</td>
<td>IRQ5</td>
<td>FFEAh</td>
</tr>
<tr>
<td>CALLV #6</td>
<td>FFC7h</td>
<td>IRQ4</td>
<td>FFEBh</td>
</tr>
<tr>
<td>CALLV #7</td>
<td>FFC8h</td>
<td>IRQ3</td>
<td>FFF0h</td>
</tr>
<tr>
<td></td>
<td>FFC9h</td>
<td>IRQ2</td>
<td>FFF1h</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IRQ1</td>
<td>FFF2h</td>
</tr>
<tr>
<td></td>
<td></td>
<td>IRQ0</td>
<td>FFF3h</td>
</tr>
<tr>
<td>Mode data</td>
<td>—*</td>
<td>Reset vector</td>
<td>FFF4h</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FFF5h</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FFF6h</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FFF7h</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FFF8h</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FFF9h</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FFFAh</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>FFFFh</td>
</tr>
</tbody>
</table>

* FFFCh is not available. (Set FFh.)
3.1 Memory Space

3.1.2 Storing 16-bit Data in Memory

For 16-bit data and the stack, store the upper data in the lower memory address value.

- **Storing 16-bit Data in RAM**
  
  When writing 16-bit data to memory, store the upper byte at the lower address and the lower byte at the next address. Handle reading of 16-bit data in the same way.
  
  Figure 3.1.2a shows how 16-bit data is stored in memory.

  ![Figure 3.1.2a Storing 16-bit Data in Memory](image)

- **Storing 16-bit Operands**
  
  The byte order applies when specifying a 16-bit operand in an instruction. Store the upper byte at the address following the operation code (instruction) and the lower byte at the next address.
  
  The byte ordering applies to both 16-bit immediate data and operands that specify a memory address.
  
  Figure 3.1.2b shows how 16-bit data is stored in an instruction.

  ![Figure 3.1.2b Byte Order of 16-bit Data in an Instruction](image)

- **Storing 16-bit Data on Stack**
  
  The same byte order applies when saving 16-bit register data on the stack during an interrupt or similar. The upper byte is stored in the lower address.
3.2 Dedicated Registers

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

- **Dedicated Register Configuration**

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

Figure 3.2a shows the structure of the dedicated registers.

<table>
<thead>
<tr>
<th>Initial value</th>
<th>16 bits</th>
<th>Program counter</th>
</tr>
</thead>
<tbody>
<tr>
<td>FFFDH</td>
<td>PC</td>
<td>A register for indicating the current instruction storage positions</td>
</tr>
<tr>
<td>Indeterminate</td>
<td>A</td>
<td>Temporary register for storing arithmetic operations or transfer instructions</td>
</tr>
<tr>
<td>Indeterminate</td>
<td>T</td>
<td>Temporary accumulator</td>
</tr>
<tr>
<td>Indeterminate</td>
<td>IX</td>
<td>A register which performs arithmetic operations with the accumulator</td>
</tr>
<tr>
<td>Indeterminate</td>
<td>EP</td>
<td>A register for indicating an index address</td>
</tr>
<tr>
<td>Indeterminate</td>
<td>SP</td>
<td>A pointer for indicating a memory address</td>
</tr>
<tr>
<td></td>
<td>RP CCR</td>
<td>A register for indicating the current stack location</td>
</tr>
<tr>
<td></td>
<td>PS</td>
<td>Program status</td>
</tr>
</tbody>
</table>

- **Program counter (PC)**

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

- **Accumulator (A)**

The accumulator is a 16-bit arithmetic operation register. The accumulator is used to perform arithmetic operations and data transfers with data in memory or in other registers such as the temporary accumulator (T). The content of the accumulator can be treated as either word (16-bit) or byte (8-bit) data. Only the lower 8 bits (AL) of the accumulator are used for byte arithmetic operations or transfers. In this case, the upper 8 bits (AH) remain unchanged. The content of the accumulator after a reset is indeterminate.
Temporary accumulator (T)
The temporary accumulator is an auxiliary 16-bit arithmetic operation register used to perform arithmetic operations with the data in the accumulator (A). The content of the temporary accumulator is treated as word data (16-bit) for word-length arithmetic operations with the accumulator and as byte data (8-bit) for byte-length arithmetic operations. For byte-length arithmetic operations, only the lower 8 bits of the temporary accumulator (TL) are used and the upper 8 bits (TH) are not used.

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

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

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

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

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

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

Reference: Refer to the FMC-8L MB89600 series Programming Manual for details on using the dedicated registers.
3.2 Dedicated Registers

3.2.1 Condition Code Register (CCR)

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

### Structure of Condition Code Register (CCR)

<table>
<thead>
<tr>
<th>Bit 15</th>
<th>Bit 14</th>
<th>Bit 13</th>
<th>Bit 12</th>
<th>Bit 11</th>
<th>Bit 10</th>
<th>Bit 9</th>
<th>Bit 8</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>R4</td>
<td>R3</td>
<td>R2</td>
<td>R1</td>
<td>R0</td>
<td>H</td>
<td>I</td>
<td>IL1</td>
<td>IL0</td>
<td>N</td>
<td>V</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Structure of Condition Code Register (CCR)**

<table>
<thead>
<tr>
<th>Bit 15</th>
<th>Bit 14</th>
<th>Bit 13</th>
<th>Bit 12</th>
<th>Bit 11</th>
<th>Bit 10</th>
<th>Bit 9</th>
<th>Bit 8</th>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
</tr>
</thead>
<tbody>
<tr>
<td>R4</td>
<td>R3</td>
<td>R2</td>
<td>R1</td>
<td>R0</td>
<td>H</td>
<td>I</td>
<td>IL1</td>
<td>IL0</td>
<td>N</td>
<td>V</td>
<td>C</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**CCR initial value**: X011XXXX

**Figure 3.2.1a Structure of Condition Code Register**

#### Arithmetic Operation Result Bits

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

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

- **Zero flag (Z)**
  
  Set when an arithmetic operation results in 0. Cleared otherwise.

- **Overflow flag (V)**
  
  Set if the complement on 2 overflows as a result of an arithmetic operation. Reset if the overflow does not occur.

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

**Figure 3.2.1b Change of Carry Flag by Shift Instruction**

- Left shift (ROLC)
  
  Bit 7
  
  Bit 0

- Right shift (RORC)
  
  Bit 7
  
  Bit 0
Check: The condition code register is part of the program status (PS) and cannot be accessed independently.

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

**Interrupt Acceptance Control Bit**

*Interrupt enable flag (I)*

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

The initial value after a reset is “0”.

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

*Interrupt level bits (IL1, IL0)*

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

Given that the interrupt enable flag is enabled (I = “1”), the CPU only performs interrupt processing for interrupt requests with an interrupt level value that is less than the value of these bits. Table 3.2.1a lists the interrupt level priorities. The initial value after a reset is “11”.

<table>
<thead>
<tr>
<th>IL1</th>
<th>IL0</th>
<th>Interrupt level</th>
<th>High-low</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>High</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>3</td>
<td>Low (no interrupt)</td>
</tr>
</tbody>
</table>

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

Reference: See Section 3.4, “Interrupts” for details on interrupts.
3.2 Dedicated Registers

3.2.2 Register Bank Pointer (RP)

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

■ Structure of Register Bank Pointer (RP)

Figure 3.2.2a shows the structure of the register bank pointer.

![Figure 3.2.2a Structure of Register Bank Pointer](image)

The register bank pointer indicates the address of the register bank currently in use. Figure 3.2.2b shows the relationship between the pointer contents and the actual address is based on the conversion rule.

![Figure 3.2.2b Rule for Conversion of Actual Addresses of General-purpose Register Area](image)

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

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

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

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

Structure of General-purpose Registers

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

Figure 3.3a shows the register bank structure.

![Figure 3.3a Register Bank Structure](image)

Reference: See Section 3.1.1, “Special Areas” for the general-purpose register area available for each product.
Features of General-purpose Registers

General-purpose registers have the following features:

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

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

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

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

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

The MB89160/160A/160L Series has 12 interrupt request input corresponding to peripheral functions. An interrupt level can be set independently. If an interrupt request output is enabled in the peripheral function, an interrupt request from a peripheral function is compared with the interrupt level in the interrupt controller. The CPU performs interrupt operation according to how the interrupt is accepted. The CPU wakes up from standby modes, and returns to the interrupt or normal operation.

Interrupt Requests from Peripheral Functions

Table 3.4a lists the interrupt requests corresponding to the peripheral functions. On acceptance of an interrupt, execution branches to the interrupt processing routine. The contents of interrupt the vector table address corresponding to the interrupt request specifies the branch destination address for the interrupt processing routine.

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

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

<table>
<thead>
<tr>
<th>Interrupt request</th>
<th>Vector table address</th>
<th>Bit names of the interrupt level setting register</th>
<th>Simultaneously-generated same-level IRQ priority</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRQ0 (External interrupt 1-1)</td>
<td>FFFAh</td>
<td>L01, L00</td>
<td>High</td>
</tr>
<tr>
<td>IRQ1 (External interrupt 1-2)</td>
<td>FFF8h</td>
<td>L11, L10</td>
<td></td>
</tr>
<tr>
<td>IRQ2 (External interrupt 1-3)</td>
<td>FFF6h</td>
<td>L21, L20</td>
<td></td>
</tr>
<tr>
<td>IRQ3 (External interrupt 1-4)</td>
<td>FFF4h</td>
<td>L31, L30</td>
<td></td>
</tr>
<tr>
<td>IRQ4 (External interrupt 2)</td>
<td>FFF2h</td>
<td>L41, L40</td>
<td></td>
</tr>
<tr>
<td>IRQ5 (8/16-bit timer/counter)</td>
<td>FFF0h</td>
<td>L51, L50</td>
<td></td>
</tr>
<tr>
<td>IRQ6 (8-bit serial I/O)</td>
<td>FFEeh</td>
<td>L61, L60</td>
<td></td>
</tr>
<tr>
<td>IRQ7 (Timebase timer)</td>
<td>FFECm</td>
<td>L71, L70</td>
<td></td>
</tr>
<tr>
<td>IRQ8 (Watch prescaler)</td>
<td>FFEAm</td>
<td>L81, L80</td>
<td></td>
</tr>
<tr>
<td>IRQ9 (PWM timer 1)</td>
<td>FFE8h</td>
<td>L91, L90</td>
<td></td>
</tr>
<tr>
<td>IRQA (PWM timer 2)</td>
<td>FFE6h</td>
<td>LA1,LA0</td>
<td></td>
</tr>
<tr>
<td>IRQB (A/D converter)</td>
<td>FFE4h</td>
<td>LB1, LB0</td>
<td></td>
</tr>
</tbody>
</table>
3.4 Interrupts

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

The interrupt level setting registers (ILR1, ILR2, ILR3) together contain 12 blocks of 2-bit data, with each data corresponding to an interrupt request from a peripheral function. The interrupt level for each interrupt is set in that interrupt’s corresponding 2-bit data (interrupt level setting bits).

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

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

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

The CPU does not accept interrupt requests set to interrupt level 3.

Table 3.4.1a shows the relationship between the interrupt level setting bits and the interrupt levels.

---

<table>
<thead>
<tr>
<th>Register</th>
<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>ILR1</td>
<td>007Ch</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>11111111b</td>
</tr>
<tr>
<td>ILR2</td>
<td>007Dh</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>11111111b</td>
</tr>
<tr>
<td>ILR3</td>
<td>007Eh</td>
<td>LB1</td>
<td>LB0</td>
<td>LA1</td>
<td>LA0</td>
<td>L91</td>
<td>L90</td>
<td>L81</td>
<td>L80</td>
<td>11111111b</td>
</tr>
</tbody>
</table>

W: Write-only

---

Figure 3.4.1a Structure of Interrupt Level Setting Registers

Table 3.4.1a shows the relationship between the interrupt level setting bits and the interrupt levels.

<table>
<thead>
<tr>
<th>L01 to LB1</th>
<th>L00 to LB0</th>
<th>Request interrupt level</th>
<th>High-low</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>High</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>2</td>
<td>Low (no interrupt)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>3</td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

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

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

3.4.2 Interrupt Processing

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

**Interrupt Processing**

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

Figure 3.4.2a shows the interrupt processing.
(1) After a reset, all interrupt requests are disabled.

Initialize the peripheral functions that are to generate interrupts in the peripheral function
initialization program, set the interrupt levels in the appropriate interrupt level setting
registers (ILR1, ILR2, ILR3), and start peripheral function.

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

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

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

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

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

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

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

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

Reference: An interrupt wakes up the CPU from standby mode (low-power consumption). See Section 3.7,
“Standby Modes (Low-power Consumption)” for details.

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

3.4.3 Multiple Interrupts

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

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

- **Example of multiple interrupts**
  
  As an example of multiple interrupt processing, assume that an external interrupt has a higher priority than the timer interrupt. The timer interrupt is set to level 2 and the external interrupt is set to level 1. Figure 3.4.3a shows the processing when the external interrupt occurs during execution of timer interrupt processing.

![Figure 3.4.3a Example of Multiple Interrupts](image)

- During execution of timer interrupt processing, the interrupt level bits in the condition code register (CCR:IL1, IL0) are automatically set to the same value as the interrupt level setting register (ILR1, ILR2, ILR3) corresponding to the timer interrupt (level 2 in this example). If the interrupt request set to higher interrupt level (level 1 in this example) occurs at this time, the interrupt processing has priority.

- To temporarily disable multiple interrupts during the timer interrupt, the interrupt enable flag in the condition code register is set to “interrupts disabled” (CCR: I = “0”) or the interrupt level bits (IL1, IL0) set to “00”.

- On execution of the interrupt return instruction (RETI) at the completion of interrupt processing, the CPU restores the program counter (PC) and program status (PS) values saved on the stack and resumes execution of the interrupted program.

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

3.4.4 Interrupt Processing Time

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

### Interrupt Processing Time

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

#### Interrupt request sampling time

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

#### Interrupt handling time

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

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

Figure 3.4.4a shows the interrupt processing time.

![Figure 3.4.4a Interrupt Processing Time](image)

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

**Reference:** The time of one instruction cycle changes with the clock mode and the main clock frequency as selected by the “speed-shift” (gear) function. See Section 3.6, “Clocks” for details.
3.4 Interrupts

3.4.5 Stack Operation during Interrupt Processing

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

- Stack Operation at Start of Interrupt Processing

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

Figure 3.4.5a shows the stack operation at the start of interrupt processing.

- Stack Operation at Interrupt Return

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

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

3.4.6 Stack Area for Interrupt Processing

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

Stack Area for Interrupt Processing

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

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

Figure 3.4.6a shows the example of stack area setting.

---

**Figure 3.4.6a Stack Area for Interrupt Processing**

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

The MB89160/160A/160L Series supports the following four types of reset source:

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

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

### Table 3.5a Reset Source

<table>
<thead>
<tr>
<th>Reset source</th>
<th>Reset conditions</th>
</tr>
</thead>
<tbody>
<tr>
<td>External reset</td>
<td>Set the external reset pin to the “L” level.</td>
</tr>
<tr>
<td>Software reset</td>
<td>Write “0” to the software reset bit in the standby control register (STBC: RST).</td>
</tr>
<tr>
<td>Watchdog reset</td>
<td>Watchdog timer overflow</td>
</tr>
<tr>
<td>Power-on reset</td>
<td>Power is turned on (only on products with a power-on reset).</td>
</tr>
</tbody>
</table>

- **External reset**

  Inputting an “L” level to the external reset pin (RST) generates an external reset. Returning the reset pin to the “H” level wakes up the CPU from the external reset.

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

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

- **Software reset**

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

- **Watchdog reset**

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

- **Power-on reset**

  Products can be set to with or without power-on reset (optional). On products with power-on reset, turning on the power generates a reset. The reset operation is performed after the oscillation stabilization delay time has passed.

  On products with power-on reset, an external reset circuit is required to generate a reset when the power is turned on.
Main Clock Oscillation Stabilization Delay Time and the Reset Source

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

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

In software or watchdog reset, if the reset occurs while the device is in main clock mode, no stabilization time is provided. If it occurs in the subclock mode, however, a stabilization time is provided since the main clock oscillation is stopped.

Table 3.5a shows the relationships between the reset sources and the main clock oscillation stabilization delay time, and reset mode (mode fetch) operations.

Table 3.5b  Reset Source and Oscillation Stabilization Delay Time

<table>
<thead>
<tr>
<th>Reset source</th>
<th>Operating state</th>
<th>Reset operation and main clock oscillation stabilization delay time</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>With power-on reset</td>
</tr>
<tr>
<td></td>
<td></td>
<td>Without power-on reset</td>
</tr>
<tr>
<td>External reset*1</td>
<td>At power on, during stop mode, or subclock mode</td>
<td>After the main clock oscillation stabilization delay time, if the external reset is waked up, reset is operated.</td>
</tr>
<tr>
<td>Software and watchdog reset</td>
<td>Main clock mode</td>
<td>After 4-instruction-cycle reset occurs, reset is operated.</td>
</tr>
<tr>
<td></td>
<td>Subclock mode</td>
<td>Reset is operated after the main clock oscillation stabilization delay time.</td>
</tr>
<tr>
<td>Power-on reset</td>
<td>Device enters main clock oscillation stabilization delay time at power on. Reset is operated after delay time ends.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>An external circuit must be provided to hold external reset asserted at power on until main clock has had time to stabilize.</td>
<td></td>
</tr>
</tbody>
</table>

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

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

*3: If the reset output option is selected, “L” level is output at RST pin during 4-instruction-cycle.
3.5 Resets

3.5.1 External Reset Pin

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

- Block Diagram of External Reset Pin

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

The external reset pin on products without a reset output option is only for the reset input.

Figure 3.5.1a shows the block diagram of the external reset pin.

- External Reset Pin Functions

Inputting an “L” level to the external reset pin (RST) generates an internal reset signal. On products with the reset output, the pin outputs an “L” level depending on internal reset sources or during the oscillation stabilization delay time due to an external reset. Software reset, watchdog reset, and power-on reset are classed as internal reset sources.

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

3.5.2 Reset Operation

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

Overview of Reset Operation

Figure 3.5.2a  Reset Operation Flow Diagram

---

Figure 3.5.2a  Reset Operation Flow Diagram
■ Mode Pins

The MB89160/160A/160L Series devices are single-chip mode devices. The mode pins (MOD1 and MOD0) must be tied to VSS. The mode pin settings determine whether the mode data and reset vector are read from internal ROM.

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

■ Mode Fetch

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

- **Mode data (address: FFFDh)**
  Always set the mode to “00h” (single-chip mode).

- **Reset vector (address: FFFEh (upper), FFFFh (lower))**
  Contains the address where execution is to start after completion of the reset. The CPU starts executing instructions from the address contained in the reset vector.

■ Oscillation Stabilization Delay Reset State

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

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

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

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

■ Effect of Reset on RAM Contents

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

3.5.3  Pin States during Reset

Reset initialized the pin states.

■ Pin States during Reset

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

■ Pin States after Reading Mode Data

With a few exceptions, the I/O pins remain in the high-impedance state immediately after reading the mode data. (Pins with a pull-up resistor (optional) go to the “H” level.)

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

**Reference:** See Appendix E, “MB89160/160A/160L Series Pin States” for pin states at times other than a reset.
3.6 Clocks

The clock generator is provided with two oscillators. By connecting with external resonators, the two circuits generate the high speed main clock and low speed subclock source oscillators. Alternatively, externally generated clock inputs can be used. Clock controller controls the speed and supply of the dual-clock signals according to the clock and standby modes. As an option, a one-clock system can also be selected.

Clock Supply Map

Oscillation of a clock and its supply to the CPU and peripheral circuit (peripheral functions) are controlled by the clock controller. As shown in the map, operating clocks fed to the CPU and peripheral circuits are affected by main clock/subclock switching (clock mode), main clock speed switching (speed-shift function), and standby modes (sleep/stop/watch). Divide-by-n output derived from the free-run counter clocked by the peripheral circuit clock is supplied to the peripheral functions. Divide-by-n outputs from the timebase timer and watch prescaler are also supplied to the peripheral functions. These clocks, however, are not affected by the speed-shift function, etc. The timebase timer is clocked by the output of the main clock source oscillator after it is fed through a divide-by-n circuit, and the watch prescaler is clocked directly by the subclock oscillator.
Figure 3.6a shows the clock supply map.

*1: Not affected by clock mode, speed-shift function, etc.
*2: Operating speed, etc., affected by clock mode or speed-shift function.
*3: Stops operating if its clock source (main or subclock oscillator) stops.
*4: Timebase timer output can be selected in continuous A/D conversion operating mode. In other modes, clock speed is affected by clock mode and speed-shift function.
3.6 Clocks

3.6.1 Clock Generator

Enable and stop of the main clock and subclock oscillations are controlled by clock and stop modes.

- Clock Generator
  - Crystal or ceramic resonator
    Connect as shown in Figure 3.6.1a.

![Figure 3.6.1a Connection Example for a Crystal or Ceramic Resonator](image)

Note: A piezoelectric resonator (FAR series) that contains the external capacitors can also be used. See Data Sheet for details.

- CR (main clock only) (Not applicable in MB89160L series)
  Connect as shown in Figure 3.6.1b. External resistors and capacitors can only be used on devices with mask ROM.

![Figure 3.6.1b Connection Example for CR](image)
● **External clock**

Connect the external clock to the X0 pin and leave X1 pin open, as shown in Figure 3.6.1c. To use an external subclock source, connect the external clock to the X0A pin and leave the X1A pin open.

**Check:**
- The one-clock option cannot be selected in devices that have internal voltage booster (MB89160A series).
- In the MB89160 series, you can select a single clock system as an option. If only the main clock were to be used without the single clock option, there would be no way to recover once the system goes into subclock mode. Therefore, the single clock option must be selected in order to operate with one clock.
- To use CR instead of a main clock oscillator, the device you are using must have mask ROM. For information on selecting mask ROM, See the Appendix C, "Mask Options".

---

**Figure 3.6.1c Connection Example for External Clock**

The diagram shows the connection options for dual-clock and one-clock systems. The dual-clock option includes both main and subclock oscillators, while the one-clock option includes only the main clock oscillator and leaves the subclock oscillator open. The 32.768 kHz oscillator is shown connected to the dual-clock option.
3.6 Clocks

3.6.2 Clock Controller

The clock controller contains the following seven blocks:

- Main clock oscillator
- Subclock oscillator
- System clock selector
- Clock controller
- Oscillation stabilization delay time selector
- System clock control register (SYCC)
- Standby control register (STBC)

---

Figure 3.6.2a shows the block diagram of the clock controller.

![Block Diagram of Clock Controller](image)
Main clock oscillator

The main clock oscillator is stopped in main-stop and subclock modes.

Subclock oscillator

The subclock oscillator is normally running except in sub-stop mode. It does not operate in "one-clock" option devices.

System clock selector

The system clock selector selects one of five clocks: the subclock, or one of four divided clocks derived from the main clock master clock oscillator.

Clock controller

This circuit controls the supply of operating clocks to the CPU and peripheral circuits, selecting the clock based on the active mode: normal (RUN), or standby (sleep/stop/watch) mode.

Supply of the clock to the CPU is stopped until the clock supply stop signal in the oscillation stabilization delay time selector is released.

Oscillation stabilization delay time selector

This register selector selects a delay time from among four main clock oscillation stabilization times timed by the timebase timer and a subclock oscillation stabilization time timed by the watch prescaler, and outputs the time as the clock supply stop signal to the CPU based on the clock mode, standby mode and reset.

SYCC register

The SYCC register is used to select the clock mode, the speed of the main clock, and the main clock oscillation stabilization delay time, and to check the status of these selections.

STBC register

This register controls from normal operation (RUN) to the standby modes, sets the pin states in the stop or watch mode, and initiates software reset.
3.6 Clocks

3.6.3 System Clock Control Register (SYCC)

The system clock control register (SYCC) controls main clock/subclock switching, main clock speed selection, and oscillation stabilization delay time selection.

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

<table>
<thead>
<tr>
<th>Address Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SCM</td>
<td>—</td>
<td>—</td>
<td>WT1</td>
<td>WT0</td>
<td>SCS</td>
<td>CS1</td>
<td>CS0</td>
<td>X—MM100s</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>CS1</th>
<th>CS0</th>
<th>Main clock speed select bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>64/FCH (15.2µs)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>16/FCH (3.81 µs)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>8/FCH (1.90 µs)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>4/FCH (0.95µs)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SCS</th>
<th>System clock select bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Selects subclock (32 kHz) mode</td>
</tr>
<tr>
<td>1</td>
<td>Selects main clock mode</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>WT1</th>
<th>WT0</th>
<th>Oscillation stabilization delay time select bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Approx. 2¹/FCH (Approx. 3.8 µs)</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Approx. 2⁰/FCH (Approx. 1.0 ms)</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Approx. 2¹/FCH (Approx. 15.6 ms)</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Approx. 2⁰/FCH (Approx. 62.4 ms)</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>SCM</th>
<th>System clock monitor bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Subclock (main clock stopped or in oscillation stabilization delay time)</td>
</tr>
<tr>
<td>1</td>
<td>Main clock</td>
</tr>
</tbody>
</table>

FCH: Main clock oscillation
Instruction Cycle (t\text{inst})

Instruction cycle (minimum execution time) can be selected as 1/4, 1/8, 1/16, or 1/64 of the main clock, or 1/2 of the subclock (32.768 kHz clock) period. The selection is made by the system clock select bit (SCS) and main clock speed select bits (CS1 and CS0) of the SYCC register.

With main clock mode, and the highest clock speed selected (SYCC: SCS = 1, CS1 = 11b, CS0 = 11b), and with a main clock source oscillation (F\text{CH}) of 4.2 MHz, the instruction cycle is 4/F\text{CH} = approximately 0.95 μs.

With subclock mode selected (SCS = 0), and with a subclock source oscillation (F\text{CL}) of 32.768 kHz, the instruction cycle is 2/F\text{CL} = approximately 61.0 μs.

<table>
<thead>
<tr>
<th>Table 3.6.3a System Clock Control Register (SYCC) Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit</td>
</tr>
<tr>
<td>-----</td>
</tr>
<tr>
<td>Bit 7</td>
</tr>
<tr>
<td>• Indicates the current clock mode (operating clock).</td>
</tr>
<tr>
<td>• “0” indicates subclock mode (main clock is stopped or in the oscillation stabilization delay time to go to main clock mode).</td>
</tr>
<tr>
<td>• “1” indicates main clock mode.</td>
</tr>
<tr>
<td>Note:</td>
</tr>
<tr>
<td>Bit 6</td>
</tr>
<tr>
<td>• The read value is indeterminate.</td>
</tr>
<tr>
<td>• Writing to these bits has no effect on operation.</td>
</tr>
<tr>
<td>Bit 4, Bit 3</td>
</tr>
<tr>
<td>• Select main clock oscillation stabilization delay time.</td>
</tr>
<tr>
<td>• Selected wait time applies when going from subclock to main clock mode, or if external interrupt causes “wakeup” from main-stop mode (transition to normal (run) mode).</td>
</tr>
<tr>
<td>• Initial value of these bits is an option selection. Accordingly, when an oscillation stabilization delay time is provided at reset, the delay time will be as selected by the option.</td>
</tr>
<tr>
<td>Note:</td>
</tr>
<tr>
<td>Bit 2</td>
</tr>
<tr>
<td>• Specifies the clock mode.</td>
</tr>
<tr>
<td>• Writing “0” to this bit sets the CPU changing from main clock to subclock mode.</td>
</tr>
<tr>
<td>• Writing “1” to this bit causes the device to go from subclock to main clock mode after the oscillation stabilization delay time set by WT1 and WT0 bits.</td>
</tr>
<tr>
<td>Note:</td>
</tr>
<tr>
<td>Bit 1, Bit 0</td>
</tr>
<tr>
<td>These bits select the clock speed for the main clock mode. Four different speeds can be set for CPU and peripheral function operating clocks (speed-shift function). The clocks that clock the timebase timer and watch prescaler are not affected by these bits.</td>
</tr>
</tbody>
</table>
3.6 Clocks

3.6.4 Clock Modes

The clock modes consists of main clock mode and subclock mode.

In the main clock mode, the primary operating clock is provided by the main clock oscillator. The speed of the operating clock is selected by switching between one of four clocks obtained by dividing the output of the main clock oscillator (speed-shift function).

In the subclock mode, the main clock oscillator is stopped, and operating clocks are provided solely by the subclock.

Clock Mode Operating States

<table>
<thead>
<tr>
<th>Clock mode</th>
<th>Main clock speed SYCC register (SYCC: CS1, CS0)</th>
<th>Standby mode</th>
<th>Clock Oscillator</th>
<th>Operating clocks for various sections</th>
<th>Non-reset event triggering exit from standby</th>
</tr>
</thead>
<tbody>
<tr>
<td>(1.1)</td>
<td>Fast</td>
<td>RUN Sleep</td>
<td>Oscillate</td>
<td>FCH/4 Stop FCH/4</td>
<td>FCL IRQ External interrupt</td>
</tr>
<tr>
<td>(1.0)</td>
<td>Stop</td>
<td>RUN Sleep</td>
<td>Oscillate</td>
<td>FCH/8 Stop FCH/8</td>
<td>FCL IRQ External interrupt</td>
</tr>
<tr>
<td>(0.1)</td>
<td>Slow</td>
<td>RUN Sleep</td>
<td>Oscillate</td>
<td>FCH/16 Stop FCH/16</td>
<td>FCL IRQ External interrupt</td>
</tr>
<tr>
<td>(0.0)</td>
<td>Slow</td>
<td>RUN Sleep</td>
<td>Oscillate</td>
<td>FCH/64 Stop FCH/64</td>
<td>FCL IRQ External interrupt</td>
</tr>
</tbody>
</table>

FCH: Main clock source oscillation
FCL: Subclock source oscillation
*: Since the timebase timer is derived from the main clock, it stops in subclock mode.

Reference: See Section 3.7, "Standby Modes" for a description of the standby modes.

Speed-Shift (Main Clock Speed-switching) Function

One of four main clock frequencies can be selected by writing the appropriate values between “00b” and “11b” to main clock speed select bits of the system clock control register (SYCC: CS1, CS0).

The switch-selected clock signal provides the operating clock for the CPU and peripheral circuits. The timebase timer and watch prescaler, however, are not affected by the speed-shift (gear) function.

A slower main clock speed reduces power consumption.
Operation of Main Clock Mode

The main clock and the subclock oscillators both run in the “main-run” mode (the normal main clock operating mode). The watch prescaler runs on the subclock, but the CPU, timebase timer, and other peripheral circuits all use the main clock.

When operating in main clock mode, the speed-shift function can be used to select a main clock speed. This selection affects all circuits that are clocked by the main clock except for the timebase timer. By specifying a standby mode, you can also go to “main-sleep,” or “main-stop” mode.

When the device is reset, the system always starts out in “main-run” mode regardless of how the reset was initiated. (Each operating mode exited by reset.)

Changing from main clock mode to subclock mode

Writing “0” to the system clock select bit in the system clock control register (SYCC:SCS) changes the CPU from the main clock to subclock mode. You can determine which clock is currently being used by checking the system clock monitor bit of the same register (SYCC: SCM).

Check: If you go to subclock mode immediately after power on, write the software so as to provide a longer subclock oscillation stabilization delay time than that defined by the watch prescaler.

Operation of Subclock Mode

In the normal subclock operating mode (“sub-run” mode), the system runs on the subclock only. The main clock oscillator is stopped. Using the low speed subclock reduces power consumption.

Other than the timebase timer, all functions operate the same in subclock mode as they do in main clock mode. If standby mode is specified while operating in subclock mode, the device goes to “sub-sleep,” “sub-stop,” or “watch” mode.

Returning to main clock mode from subclock mode

Writing “1” to the system clock select bit in the system clock control register (SYCC:SCS) returns to main clock mode from subclock mode.

Operation from the main clock, however, will not start until after the main clock oscillation stabilization delay time has passed. One of four wait times can be selected by setting the oscillator stabilization delay time select bits of the system clock control register (SYCC: WT1, WT0).

Check: Do not change the oscillation stabilization delay time select bits (SYCC: WT1, WT0) at the same time you switch from subclock to main clock mode (SYCC: SCS = 1), or during the oscillator stabilization delay time. Always check the system clock monitor bit to verify that the main clock is the active operating clock (SYCC: SCM = 1) before changing these bits.

If the device has the power-on reset option it always enters the oscillator stabilization delay time when the system is returned from subclock mode to main clock mode by reset. If the device does not have power-on reset option, there will be no delay time unless the reset was a software or watchdog reset.
3.6 Clocks

3.6.5 Oscillation Stabilization Delay Time

When the system transitions to main-run mode from a state in which the main clock is stopped (such as at power-on, and in main-stop and subclock modes, etc.), a delay time is required for oscillation to stabilize before operation starts. Similarly, a subclock oscillation stabilization wait time is required when exiting the sub-stop mode, because the subclock oscillator is stopped in that mode.

- Oscillation Stabilization Delay Time

  After starting, ceramic, crystal, and other resonators typically require the time between several milliseconds and several tens of milliseconds to stabilize at their fixed oscillation frequency.

  Therefore, operation of the CPU and other functions is disabled when oscillation first starts and no clock signal is supplied to the CPU and peripheral functions until the oscillation stabilization delay time has passed and the oscillation has sufficiently stabilized.

  The time required for oscillation to stabilize depends on the resonator type (crystal, ceramic, etc.) connected to the clock generator. Consequently, it is necessary to select an oscillation stabilization delay time that matches the type of oscillator being used.

Figure 3.6.5a shows the operation of an oscillator after starting oscillation.

![Figure 3.6.5a Operation of Oscillator after Starting Oscillation](image-url)
Main Clock Oscillation Stabilization Delay Time

When first starting operation in main clock mode after a state in which the main clock oscillator is stopped, a delay time is required for oscillation to stabilize. This delay time starts when the timebase timer starts counting up from its cleared state, and ends when the count overflows at the specified bit.

Oscillation stabilization delay time during operation

A time length must be selected for the oscillation stabilization delay time when an external interrupt takes the system from main-stop mode back to main-run mode, or when going from subclock to main clock mode. One of four possible delay times can be selected, using the oscillator stabilization delay time select bits of the system clock control register (SYCC: WT1, WT0).

Oscillation stabilization delay time at reset

The oscillation stabilization delay time at reset (the initial values of WT1 and WT0) is selected as an option setting.

Products with power-on reset require an oscillation stabilization delay time when exit from stop mode is triggered by resets in subclock mode (multiple), power-on reset, or external reset. Products without power-on reset only require an oscillation stabilization delay time for watchdog reset or software reset during subclock mode.

Table 3.6.5a shows the relationships between the conditions in which main clock mode operation is started and oscillation stabilization delay time.

Table 3.6.5a Main Clock Startup Conditions vs. Oscillation Stabilization Delay Time

<table>
<thead>
<tr>
<th>Main clock mode startup conditions</th>
<th>At power-on</th>
<th>During subclock mode</th>
<th>Exit from main-stop</th>
<th>Transition from subclock to main clock mode (SYCC: SCS = 1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Oscillation stabilization delay time selection</td>
<td></td>
<td>External reset</td>
<td>Software reset and watchdog reset</td>
<td>External reset</td>
</tr>
<tr>
<td>With power-on reset</td>
<td></td>
<td>O</td>
<td>O</td>
<td>O</td>
</tr>
<tr>
<td>No power-on reset</td>
<td>X</td>
<td>X</td>
<td>O</td>
<td>X</td>
</tr>
</tbody>
</table>

O: Oscillation stabilization delay time provided
X: Oscillation stabilization delay time not provided
*1: System clock select bit of system clock control register
*2: Oscillation stabilization delay time select bits of system clock control register

Subclock Oscillation Stabilization Delay Time

When an external interrupt returns the system from sub-stop (subclock oscillator stopped) to sub-run mode (thus starting the subclock oscillator), a set subclock oscillation stabilization delay time is provided. (This set delay time is equal to 2^15/FCL, where FCL is the subclock oscillator frequency.)

The subclock oscillation stabilization delay time is also entered at power-on. Therefore, if you go to subclock mode after power-on, you should insert a software delay, to provide a longer delay time before starting this transition than the subclock oscillation stabilization delay time alone.

The subclock oscillation stabilization delay time starts when the watch prescaler starts counting up from the cleared state, and ends when it overflows.
3.7 Standby Modes (Low-Power Consumption)

The standby modes consist of sleep mode, stop mode, and watch mode. From both main and subclock clock modes, standby modes are changed to sleep mode, stop mode, or watch mode by setting the standby control register (STBC). From main clock mode, you can go only to sleep or stop mode, but from subclock mode you can go any of the three standby modes. Standby mode reduces the power consumption by stopping the operation of the CPU and peripheral functions. This section describes the relationship between standby mode and clock mode, and the operation of various sections during standby.

### Standby Modes

- **Main-sleep Mode**
  - Main-sleep mode stops the CPU and watchdog timer, but operate the peripheral functions except watch prescaler by the main clock. (Certain functions can still run on the subclock.)

- **Sub-sleep Mode**
  - Sub-sleep mode stops the main clock oscillator, CPU, watchdog timer, and timebase timer, but operate the peripheral functions on the subclock.

- **Main-stop Mode**
  - Main-stop mode stops the CPU and peripheral functions. The main clock oscillator is stopped, but the subclock oscillator keeps running. Everything is shut down except external interrupt servicing, watch prescaler counter operation, and some functions that operate from the subclock.

- **Sub-stop Mode**
  - Sub-stop mode stops all chip functions except the external interrupt, and stops the main clock and subclock oscillations.

- **Watch Mode**
  - You can only go to watch mode from the subclock clock mode. All functions are shut down except the watch prescaler (watch interrupt), external interrupts, and some functions that operate from the subclock.

**Reference:** Even when the main clock is stopped, as it is in the main-stop and watch modes, as long as the subclock oscillation is still operating, some buzzer outputs and LCD controller/drivers will still operate. See Chapter 10, “Output”, Chapter 16, “LCD Controller/Driver”, and the clock supply map provided in Section 3.6, “Clocks” for details.
3.7 Standby Modes (Low-Power Consumption)

3.7.1 Operating States in Standby Modes

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

- Operating States during Standby Modes

Table 3.7.1a Operating States of the CPU and Peripheral Functions in Standby Modes

<table>
<thead>
<tr>
<th>Function</th>
<th>Main clock mode</th>
<th>Subclock mode</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>RUN</td>
<td>Sleep</td>
</tr>
<tr>
<td>Main clock</td>
<td>Operating</td>
<td>Operating</td>
</tr>
<tr>
<td>Subclock</td>
<td>Operating</td>
<td>Operating</td>
</tr>
<tr>
<td>CPU</td>
<td>Operating</td>
<td>Stop</td>
</tr>
<tr>
<td>Instructions</td>
<td>Operating</td>
<td>Operating</td>
</tr>
<tr>
<td>ROM</td>
<td>Operating</td>
<td>Hold</td>
</tr>
<tr>
<td>RAM</td>
<td>Operating</td>
<td>Hold</td>
</tr>
<tr>
<td>I/O ports</td>
<td>Operating</td>
<td>Hold</td>
</tr>
<tr>
<td>Watch prescaler</td>
<td>Operating</td>
<td>Operating</td>
</tr>
<tr>
<td>Timebase timer</td>
<td>Operating</td>
<td>Operating</td>
</tr>
<tr>
<td>8/16-bit timer/counter</td>
<td>Operating</td>
<td>Operating</td>
</tr>
<tr>
<td>8-bit serial I/O</td>
<td>Operating</td>
<td>Operating</td>
</tr>
<tr>
<td>Peripheral functions</td>
<td>Operating</td>
<td>Operating</td>
</tr>
<tr>
<td>LCD controller-driver</td>
<td>Operating</td>
<td>Operating</td>
</tr>
<tr>
<td>8-bit PWM timers</td>
<td>Operating</td>
<td>Operating</td>
</tr>
<tr>
<td>A/D converter</td>
<td>Operating</td>
<td>Operating</td>
</tr>
<tr>
<td>External Interrupts 1 and 2</td>
<td>Operating</td>
<td>Operating</td>
</tr>
<tr>
<td>Buzzer output</td>
<td>Operating</td>
<td>Operating</td>
</tr>
<tr>
<td>Watchdog timer</td>
<td>Operating</td>
<td>Stop</td>
</tr>
</tbody>
</table>

*1: Watch prescaler counts but does not generate watch interrupts.
*2: If the subclock is selected as the operating clock, in watch mode, LCD controller/driver operation must be enabled.
*3: In devices with on-chip voltage boosters (MB89160A series), the voltage booster stops operating.
*4: Can be operated if watch prescaler output is selected as its operating clock.

- Pin States in Standby Mode

Almost all I/O pins will either keep the state they were placed in by the pin state control bit of the standby control register (STBC: SPL) just prior to going to the stop or watch mode, or will go to the high impedance state. This is true regardless of the clock mode.

3.7 Standby Modes (Low-Power Consumption)

3.7.2 Sleep Mode

This section describes the operations of sleep mode.

Operation of Sleep Mode

- Changing to sleep mode
  
  Sleep mode stops the CPU operating clock. The CPU stops while maintaining all register contents, RAM contents, and pin states at their values immediately prior to entering sleep mode. However, peripheral functions except the watchdog timer continue to operate.

  If the system is in subclock mode, however, the main clock oscillation is stopped; and because the timebase timer operates on a divide-by-two version of the main clock source oscillation, it also stops operating.

  Writing “1” to the sleep bit in the standby control register (STBC: SLP) changes the CPU to sleep mode. If an interrupt request is generated when “1” is written to the SLP bit, the write to the bit is ignored, and the CPU continues the instruction execution without change to sleep mode. (The CPU does not change to sleep mode even after completion of the interrupt processing.)

- Wake-up from sleep mode
  
  A reset or an interrupt from a peripheral function wakes up the CPU from sleep mode.

  If a reset occurs during sub-sleep mode on a product with power-on reset, the reset operation starts after the main clock oscillation stabilization delay time.

  In products without power-on reset, or if the reset occurs in main-sleep mode, there is no oscillation stabilization delay period.

  The reset operation also initializes the pin states.

  If an interrupt request with an interrupt level higher than “11” occurs from a peripheral function or an external interrupt circuit during sleep mode, the CPU wakes up from sleep mode, regardless of the interrupt enable flag (CCR: I) and interrupt level bits (CCR: IL1 and IL0) in the CPU.

  The normal interrupt operation is performed after wake-up from sleep mode. If the interrupt request is accepted, the CPU executes interrupt processing. If the interrupt request is not accepted, the CPU continues execution from the subsequent instruction following the instruction executed immediately before changing to sleep mode.
3.7 Standby Modes (Low-Power Consumption)

3.7.3 Stop Mode

This section describes the operations of stop mode.

- **Operation of Stop Mode**
  
  - **Changing to stop mode**
    
    Stop mode the source oscillation. Almost functions stop while maintaining all register and RAM contents at their value immediately before changing to stop mode.

    - If the system is in main clock mode, the main clock oscillation stops, but the subclock oscillation continues to run. This means that the watch prescaler can still count and some functions that run on the subclock can still function. Except the external interrupt circuit, however, the CPU and other peripheral functions stop operating.

    - If the system is in subclock mode, both the main and subclock oscillations are stopped. All chip functions other than external interrupt circuits stop. Accordingly, data can be held with minimum power consumption.

    Writing “1” to the stop bit in the standby control register (STBC: STP) changes the CPU to stop mode. At this time, external pin states are held if the pin state specification bit (STBC: SPL) is “0”. If SPL is “1”, external pins go to the high-impedance state. (Pins with the pull-up resistor (optional) go to the “H” level.)

    - If an interrupt request is generated when “1” is written to the STP bit, the write to the bit is ignored, and the CPU continues the instruction execution without change to stop mode. (The CPU does not assume stop mode even after completion of the interrupt processing.)

    Prohibit interrupt request out from the timebase timer (TBTC: TBIE = “0”) before changing to stop mode in main clock mode as necessary. Similarly, prohibit timeclock interrupt request output from the watch prescaler (WPCR: WIE = 0) before changing to stop mode in subclock mode.

  - **Wake-up from stop mode**
    
    A reset or an external interrupt wakes up the CPU from stop mode.

    - If reset occurs during stop mode on a product with power-on reset, the reset operation starts after the main clock oscillation stabilization delay time. Products without power-on reset do not require for the oscillation stabilization delay time after a reset in stop mode. The reset initializes pin states.

    - If an interrupt request with an interrupt level higher than “11” occurs from an external interrupt circuit during stop mode, the CPU wakes up from stop mode, regardless of the interrupt enable flag (CCR: I) and interrupt level bits (CCR: IL1, IL0) in the CPU. Only external interrupt requests can occur during stop mode because peripheral functions are stopped. In main-stop mode, the watch prescaler operates, but it does not generate watch interrupts.

    After wake-up from stop mode, the normal interrupt operation is performed after the oscillation stabilization delay time has passed. If the interrupt request is accepted, the CPU executes interrupt processing. If the interrupt request is not accepted, the CPU continues execution from the subsequent instruction following the instruction executed immediately before entering stop mode.

    Some peripheral functions restart from mid-operation when the CPU wakes up from stop mode by an external interrupt. The first interval time from the interval timer function, for example, is indeterminate. Therefore, initialize all peripheral functions after wake-up from stop mode.

  **Check:** Only interrupt requests from external interrupt circuits can be used to wake up from stop mode by an interrupt.
3.7 Standby Modes (Low-Power Consumption)

3.7.4 Watch Mode

This section describes the operations of watch mode.

- **Operation of watch Mode**
  
  **Changing to watch mode**
  
  Watch mode stops the clocks that clock the CPU and the main peripheral functions. You can go to watch mode only from subclock mode (in which the main clock oscillation is stopped).

  Prior to going to watch mode, registers are saved and the contents of RAM are held. All chip functions other than watch prescaler (timeclock interrupt), external interrupt circuit, and certain functions that run off of the subclock stop. Accordingly, data can be held with extremely small power consumption.

  Writing “1” to the timeclock bit in the standby control register (STBC: TMD) changes the CPU to watch mode.

  This can be done, however, only when the system clock select bit of the system clock control register (SYCC: SCS) is “0” (subclock mode active).

  When you go to watch mode, external pin states are held if the pin state specification bit in the standby control register (STBC: SPL) is “0”. If SPL is “1”, external pins go the high-impedance state. (Pins with a pull-up resistor (optional) go to the “H’ level)

  If an interrupt request is generated when “1” is written to the TMD bit, the write to the bit is ignored, and the CPU continues the instruction execution without change to watch mode. (The CPU does not assume watch mode even after completion of the interrupt processing.)

  **Wake-up from watch mode**

  A reset, a timeclock interrupt or an external interrupt wakes up CPU from watch mode.

  If a reset occurs during watch mode on a product with power-on reset, the reset operation starts after the main clock oscillation stabilization delay time.

  Products without power-on reset do not require for the oscillation stabilization delay time after a reset in watch mode.

  The reset initializes pin states.

  If an interrupt request with an interrupt level higher than 11” occurs from a watch prescaler or an external interrupt circuit during watch mode, the CPU wakes up from watch mode, regardless of the interrupt enable flag (CCR:1) and interrupt level bits ((CCR: IL1, IL0) in the CPU. Only timeclock or external interrupt requests can occur during watch mode because most of the peripheral functions except watch prescaler are stopped.

  After wake-up from stop mode, the normal interrupt operation is performed. If the interrupt request is accepted, the CPU executes interrupt processing. If the interrupt request is not accepted, the CPU continues execution from the subsequent instruction following the instruction executed immediately before entering watch mode.

  Some peripheral functions restart from mid-operation when the CPU wakes up from watch mode. The first interval time from the interval timer function, for example, is indeterminate. Therefore, initialize all peripheral functions after wake-up from watch mode.
3.7 Standby Modes (Low-Power Consumption)

3.7.5 Standby Control Register (STBC)

The standby control register (STBC) controls the changing to sleep mode, stop mode, or watch mode, sets the pin states in stop mode and watch mode, and initiates software resets.

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>STP</td>
<td>SLP</td>
<td>SPL</td>
<td>RST</td>
<td>TMD</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>00010XXXs</td>
</tr>
</tbody>
</table>

- **STP**: Stop bit
- **SLP**: Sleep bit
- **SPL**: Pin state specification bit
- **TMD**: Watch bit
- **RST**: Software reset bit

**Watch bit**

- **TMD**: Valid only in subclock mode (SYCC: SCS = 0)
  - **Read**: Changing to watch mode
  - **Write**: Changing to watch mode

**Software reset bit**

- **RST**: Generates a reset signal for four instruction cycles.
  - **Read**: Changing to stop mode
  - **Write**: Changing to stop mode

**Pin state specification bit**

- **SPL**: External pins hold their states prior to entering stop mode or watch mode.
  - **Read**: External pins go to high-impedance state on entering stop mode or watch mode.

- **SLP**: External pins go to high-impedance state on entering stop mode or watch mode.
  - **Read**: External pins go to high-impedance state on entering stop mode or watch mode.

**Stop bit**

- **STP**: Change to stop mode.
  - **Read**: Changing to stop mode
  - **Write**: Changing to stop mode

![Figure 3.7.5a Standby Control Register (STBC)](image-url)
### Table 3.7.5a Standby Control Register (STBC) Bits

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
</table>
| Bit 7 | **STP:** Stop bit  
• Sets the CPU changing to stop mode.  
• Writing “1” to this bit sets the CPU changing to stop mode.  
• Writing “0” to this bit has no effect on operation.  
• Reading this bit always returns “0”. |
| Bit 6 | **SLP:** Sleep bit  
• Sets the CPU changing to sleep mode.  
• Writing “1” to this bit sets the CPU changing to sleep mode.  
• Writing “0” to this bit has no effect on operation.  
• Reading this bit always returns “0”. |
| Bit 5 | **SPL:** Pin state specification bit  
• Specifies the states of the external pins during stop mode and watch mode.  
• Writing “0” to this bit specifies that external pin hold their states (levels) on changing to stop mode or watch mode.  
• Writing “1” to this bit specifies that external pins to go to high impedance state on entering stop mode or watch mode. (Pin with a pull-up resistor (optional) go to “H” level.)  
• Initialized to “0” by a reset. |
| Bit 4 | **RST:** Software reset bit  
• Specifies a software reset.  
• Writing “0” to this bit generates an internal reset source for four instruction cycles.  
• Writing “1” to this bit has no effect on operation.  
• Reading this bit always returns “1”.  
**Note:** When the software reset is applied in subclock mode, operation will start up in main clock mode after an oscillation stabilization delay time. For this reason, if the reset output option is selected, the RST signal will be output during the oscillation stabilization delay time. |
| Bit 3 | **TMD:** Watch bit  
• Sets the CPU changing to watch mode.  
• A write to this bit is valid only in subclock mode (SYCC: SCS = 0).  
• Writing “1” to this bit sets the CPU changing to watch mode.  
• Writing “0” to this bit has no effect on operation.  
• Reading this bit always returns “0”. |
| Bit 2, Bit 1, Bit 0 | **Unused bits**  
• The read value is indeterminate.  
• Writing to these bits has no effect on operation. |
3.7 Standby Modes (Low-Power Consumption)

3.7.6 State Transition Diagram 1 (Options: Power-on Reset, Two Clocks)

This section shows the state transition diagram for products with power-on reset and dualclock options.

Figure 3.7.6a State Transition Diagram 1 (Options: Power-on Reset, Two Clocks)
### Changing to/wake-up from clock modes (non-standby modes)

**Table 3.7.6a Changing to/wake-up from Clock Modes (Options: Power-on Reset, Two Clocks)**

<table>
<thead>
<tr>
<th>State transition</th>
<th>Conditions/events required to transition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Changing from main-RUN state to sub-RUN state</td>
<td>[4] SYCC: SCS=0*</td>
</tr>
<tr>
<td>Reset in sub-RUN state</td>
<td>[8] Have external, software, or watchdog reset.</td>
</tr>
</tbody>
</table>

SYCC: System clock control register  
*: Changing to sub-RUN state at power-on occurs after the subclock oscillation stabilization delay time complete.

### Changing to/wake-up from standby modes

**Table 3.7.6b Changing to/wake-up from Standby Modes (Options: Power-on Reset, Two Clocks)**

<table>
<thead>
<tr>
<th>State transition</th>
<th>Conditions/events required to transition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Changing to sleep mode</td>
<td>[1] STBC: SLP = 1</td>
</tr>
<tr>
<td>Changing to stop mode</td>
<td>[4] STBC: STP = 1</td>
</tr>
<tr>
<td>Changing to watch mode</td>
<td>–</td>
</tr>
<tr>
<td>Wake-up from watch mode</td>
<td>–</td>
</tr>
</tbody>
</table>

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

**Note:** Neither software nor watchdog resets can occur during standby because the CPU and watchdog timer are both stopped.
3.7 Standby Modes (Low-Power Consumption)

3.7.7 State Transition Diagram 2 (Options: No Power-on Reset, Two Clocks)

This section shows the state transition diagram for products without power-on reset and dualclock options.

Figure 3.7.7a State Transition Diagram 2 (Options: Without Power-on Reset, Two Clocks)
Changing to/wake-up from clock modes (non-standby modes)

Table 3.7.7a Changing to/wake-up from Clock Modes (Options: Without Power-on Reset, Two Clocks)

<table>
<thead>
<tr>
<th>State transition</th>
<th>Conditions/events required to transition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Changing to main-RUN state (normal main clock mode)</td>
<td>[1] External reset input must be held asserted until main clock oscillation</td>
</tr>
<tr>
<td></td>
<td>has had time to stabilize.</td>
</tr>
<tr>
<td>Changing from main-RUN state to sub-RUN state</td>
<td>[4] SYCC: SCS = 0*</td>
</tr>
<tr>
<td>Changing from sub-RUN state back to main-RUN state</td>
<td>[5] SYCC: SCS = 1</td>
</tr>
<tr>
<td></td>
<td>[6] Main clock oscillation stabilization delay time complete. (Can be</td>
</tr>
<tr>
<td></td>
<td>checked by looking at SYCC: SCM)</td>
</tr>
<tr>
<td></td>
<td>[7] Have software or watchdog reset.</td>
</tr>
<tr>
<td>Reset in sub-RUN state</td>
<td>[9] Have software or watchdog reset.</td>
</tr>
</tbody>
</table>

SYCC: System clock control register
*: Changing to sub-RUN state at power-on occurs after the subclock oscillation stabilization delay time complete.

Changing to/wake-up from standby modes

Table 3.7.7b Changing to/Wake-up from Standby Modes (Options: Without Power-on Reset, Two Clocks)

<table>
<thead>
<tr>
<th>State transition</th>
<th>Conditions/events required to transition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Changing to sleep mode</td>
<td>[1] STBC: SLP = 1</td>
</tr>
<tr>
<td></td>
<td>&lt;1&gt; STBC: SLP = 1</td>
</tr>
<tr>
<td>Wake-up from sleep mode</td>
<td>[2] Interrupt (any)</td>
</tr>
<tr>
<td></td>
<td>[3] External reset</td>
</tr>
<tr>
<td></td>
<td>&lt;2&gt; Interrupt (any)</td>
</tr>
<tr>
<td></td>
<td>&lt;3&gt; External reset</td>
</tr>
<tr>
<td>Changing to stop mode</td>
<td>[4] STBC: STP = 1</td>
</tr>
<tr>
<td></td>
<td>&lt;4&gt; STBC: STP = 1</td>
</tr>
<tr>
<td>Wake-up from stop mode</td>
<td>[5] External interrupt</td>
</tr>
<tr>
<td></td>
<td>[6] Main clock oscillation stabilization delay time complete. (Timebase timer output.)</td>
</tr>
<tr>
<td></td>
<td>[7] External reset</td>
</tr>
<tr>
<td></td>
<td>[8] External reset (during oscillation stabilization delay time)</td>
</tr>
<tr>
<td></td>
<td>&lt;5&gt; External interrupt</td>
</tr>
<tr>
<td></td>
<td>&lt;6&gt; Subclock oscillation stabilization delay time complete. (watch prescaler output.)</td>
</tr>
<tr>
<td></td>
<td>&lt;7&gt; External reset</td>
</tr>
<tr>
<td></td>
<td>&lt;8&gt; External reset (during oscillation stabilization delay time)</td>
</tr>
<tr>
<td>Changing to watch mode</td>
<td>[9] STBC: TMD = 1</td>
</tr>
<tr>
<td></td>
<td>&lt;9&gt; STBC: TMD = 1</td>
</tr>
<tr>
<td>Wake-up from watch mode</td>
<td>--</td>
</tr>
<tr>
<td></td>
<td>&lt;10&gt; External or watch interrupt</td>
</tr>
<tr>
<td></td>
<td>&lt;11&gt; External reset</td>
</tr>
</tbody>
</table>

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

Check: In all states except the main clock mode normal run (main-RUN) and sleep states, the external reset input must be held asserted until main clock oscillation has had time to stabilize.
### 3.7 Standby Modes (Low-Power Consumption)

#### 3.7.8 State Transition Diagram 3 (One-clock Option)

This section shows two state transition diagrams for one-clock option products: one diagram for “with power-on reset” option products and one for “without power-on reset” products. There are no subclock or watch modes when one clock is used.

---

**State Transition Diagram 3 (One-clock Option)**

1. **Power on**
2. **Oscillation stabilization delay reset state**
3. **Reset state**
4. **Main-stop mode**
5. **Main-run state**
6. **Main-sleep mode**
7. **Main clock mode**
8. **External reset**

---

**Figure 3.7.8a State Transition Diagram 3 (Products with Power-on Reset)**

**Figure 3.7.8b State Transition Diagram 3 (Products without Power-on Reset)**
- **Changing to normal state (RUN) and reset**

  **Table 3.7.8a Changing to Main Clock Mode Run State and Reset (One-Clock Option)**

<table>
<thead>
<tr>
<th>State transition</th>
<th>Conditions/events required to transition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Changing to normal state (RUN) after power-on</td>
<td>Products with power-on reset (Figure 3.7.8a)</td>
</tr>
<tr>
<td></td>
<td>[1] Main clock oscillation stabilization delay time complete. (Timebase timer output.)</td>
</tr>
<tr>
<td>Reset in RUN state</td>
<td>Products without power-on reset (Figure 3.7.8b)</td>
</tr>
<tr>
<td></td>
<td>[1] External reset input must be held asserted until main clock oscillation has had time to stabilize.</td>
</tr>
</tbody>
</table>

- **Changing to/wake-up from standby mode**

  **Table 3.7.8b Changing to/Wake-up from Standby Mode (One-Clock Option)**

<table>
<thead>
<tr>
<th>State transition</th>
<th>Conditions/events required to transition</th>
</tr>
</thead>
<tbody>
<tr>
<td>Changing to sleep mode</td>
<td>Products with power-on reset (Figure 3.7.8a)</td>
</tr>
<tr>
<td></td>
<td>[1] STBC: SLP=1</td>
</tr>
<tr>
<td></td>
<td>[2] Interrupt</td>
</tr>
<tr>
<td></td>
<td>[3] External reset</td>
</tr>
<tr>
<td>Wake-up from sleep mode</td>
<td>Products without power-on reset (Figure 3.7.8b)</td>
</tr>
<tr>
<td></td>
<td>[1] STBC: SLP=1</td>
</tr>
<tr>
<td></td>
<td>[2] Interrupt</td>
</tr>
<tr>
<td></td>
<td>[3] External reset</td>
</tr>
<tr>
<td>Changing to stop mode</td>
<td>Products with power-on reset (Figure 3.7.8a)</td>
</tr>
<tr>
<td></td>
<td>[4] STBC: STP=1</td>
</tr>
<tr>
<td>Wake-up from stop mode</td>
<td>Products without power-on reset (Figure 3.7.8b)</td>
</tr>
<tr>
<td></td>
<td>[5] External interrupt</td>
</tr>
<tr>
<td></td>
<td>[6] Main clock oscillation stabilization delay time complete. (Timebase timer output.)</td>
</tr>
<tr>
<td></td>
<td>[7] External reset</td>
</tr>
<tr>
<td></td>
<td>[8] External reset (during oscillation stabilization delay time)</td>
</tr>
<tr>
<td></td>
<td>[5] External interrupt</td>
</tr>
<tr>
<td></td>
<td>[6] Main clock oscillation stabilization delay time complete. (Timebase timer output.)</td>
</tr>
<tr>
<td></td>
<td>[7] External reset</td>
</tr>
<tr>
<td></td>
<td>[8] External reset (during oscillation stabilization delay time)</td>
</tr>
</tbody>
</table>

STBC: Standby control register
3.7 Standby Modes (Low-Power Consumption)

3.7.9 Notes on Using Standby Modes

The CPU does not change to a standby mode if an interrupt request occurs from a peripheral function when a standby mode is set in the standby control register (STBC). Also, if an interrupt is used to wake up from a standby mode to the normal operating state, the operation after wake-up differs depending on whether or not the interrupt request is accepted.

■ Changing to a Standby Mode and Interrupts

If an interrupt request with an interrupt level higher than “11” occurs from a peripheral function to the CPU, writing “1” to the stop bit (STP), sleep bit (SLP), or watch bit (TMD) in the standby control register (STBC) is ignored. Therefore, the CPU does not change to a standby mode. The CPU also does not change to the standby mode after completing interrupt processing.

This does not depend on whether or not the CPU accepts the interrupt.

Even if the CPU is currently performing interrupt processing, the interrupt request flag bit is cleared and, if no other interrupt request is present, the device can change to the standby mode.

■ Wake-up from Standby Mode by Interrupt

If an interrupt request with an interrupt level higher than “11” occurs from a peripheral function or others during sleep or stop mode, the CPU wakes up from a standby mode. This does not depend on whether or not the CPU accepts the interrupt.

After wake-up from a standby mode, the CPU performs the normal interrupt operations. If the level set in the interrupt level setting register (ILR1 to ILR3) corresponding to the interrupt request is higher than the interrupt level bits in the condition code register (CCR: IL1, IL0), and if the interrupt enable flag is enabled (CCR: I = “1”), the CPU branches to the interrupt processing routine. If the interrupt is not accepted, operation restarts from the instruction following the instruction that activated a standby mode.

To prevent control from branching to an interrupt processing routine after wake-up, take measures such as disabling interrupts before setting a standby mode.
Notes on Setting Standby Mode

When setting the standby control register (STBC) to go to a standby mode, make the settings in accordance with Table 3.7.9a. The order of precedence as to which mode will be activated if more than one bit is set to “1” is “stop” mode, “watch” mode, and “sleep” mode. Other factors being equal, it is best to set “1” for just one bit.

Also avoid going to stop, sleep, or watch mode immediately after switching from subclock to main clock mode (SYCC: SCS=0 → 1). First verify that the clock monitor bit (SYCC: SCM) of the system control register is “1,” then make the standby mode change.

Note that you cannot go to the watch standby mode when operating in main clock mode. (A write to the TMD bit will be ignored.)

Table 3.7.9a Standby Control Register (STBC) Low-Power Consumption Mode Settings

<table>
<thead>
<tr>
<th>STBC register</th>
<th>Mode</th>
</tr>
</thead>
<tbody>
<tr>
<td>STP (Bit 7)</td>
<td>SLP (Bit 6)</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>
</tbody>
</table>

Oscillation Stabilization Delay Time

As the oscillator that provides the source oscillation is stopped during stop mode in both main clock mode and subclock mode, a delay time is required for oscillation to stabilize after the oscillator restarts operation.

In main clock mode, the main clock oscillation stabilization delay time is selected from one of four possible delay times defined by the timebase timer. In subclock mode, the subclock oscillation stabilization delay time is defined by the watch prescaler.

In main clock mode, if the interval time set for the timebase timer is less than the oscillation stabilization delay time, the timebase timer generates an interval timer interrupt request before the end of the oscillation stabilization delay time. To prevent this, disable the interval timer interrupt request for the timebase timer (TBTC: TBIE ="0") before changing to stop mode in main clock mode as necessary.

Selection of a watch prescaler interrupt interval shorter than the oscillation stabilization delay time will similarly cause the watch interrupt request to be generated during the oscillation stabilization delay time. To prevent this, disable the watch interrupt request output for the watch prescaler (WPCR: WIE =0) before changing to stop mode in subclock mode as necessary.
3.8 Memory Access Mode

In the MB89160/160A/160L Series, the only memory access mode is the single-chip mode.

- Single-chip Mode
  In single-chip mode, the device uses internal RAM and ROM only. Therefore, the CPU can access no areas other than the internal I/O area, RAM area, and ROM area (internal access).

- Mode Pins (MOD1, MOD0)
  Always set the mode pins, MOD1 and MOD0, for Vss.
  At reset, reads the mode data and reset vector from internal ROM.
  Do not change the mode pin settings, even after completion of the reset (i.e. during normal operation).
  Table 3.8a lists the mode pin settings.

<table>
<thead>
<tr>
<th>Pin state</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Vss Vss</td>
<td>Reads the mode data and reset vector from internal ROM.</td>
</tr>
<tr>
<td>Vss Vcc</td>
<td>Prohibited settings</td>
</tr>
<tr>
<td>Vcc Vss</td>
<td></td>
</tr>
<tr>
<td>Vcc Vcc</td>
<td></td>
</tr>
</tbody>
</table>

- Mode Data
  Always set the mode data in internal ROM to “00H” to select single-chip mode.

Figure 3.8a Mode Data Structure

<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>FFFDH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Data</th>
<th>Operation</th>
</tr>
</thead>
<tbody>
<tr>
<td>00H</td>
<td>Selects single-chip mode.</td>
</tr>
<tr>
<td>Other than 00H</td>
<td>Reserved. Do not set this value.</td>
</tr>
</tbody>
</table>
Memory Access Mode Selection Operation

Only the single-chip mode can be selected.

Table 3.8b lists the mode pin and mode data options.

<table>
<thead>
<tr>
<th>Memory access mode</th>
<th>Mode pins (MOD1, MOD0)</th>
<th>Mode data</th>
</tr>
</thead>
<tbody>
<tr>
<td>Single-chip mode</td>
<td>VSS, Vss</td>
<td>00H</td>
</tr>
<tr>
<td>Other modes</td>
<td>Prohibited settings</td>
<td>Prohibited settings</td>
</tr>
</tbody>
</table>

Figure 3.8b shows the operation for memory access mode selection.

Figure 3.8b Memory Access Selection Operation
This chapter describes the functions and operation of the I/O ports.

4.1 Overview of I/O Ports ....................................................86
4.2 Port 0 and Port 1 ...........................................................88
4.3 Port 2.............................................................................94
4.4 Port 3...........................................................................100
4.5 Ports 4, 6 and 7...........................................................104
4.6 Port 5...........................................................................108
4.7 Program Example for I/O Ports .................................112
4.1 Overview of I/O Ports

The I/O ports consist of eight ports (54 pins) including output-only and general-purpose I/O ports (parallel I/O ports).

The ports also serve as peripherals (I/O pins of peripheral functions).

I/O Port Functions

The functions of the I/O ports are to output data from the CPU via the I/O pins and to fetch signals input to the I/O pins into the CPU. Input and output are performed via the port data registers (PDR). Also, for certain ports the direction of each I/O pin can be individually set to either input or output for each bit by the port data direction register (DDR).

The following lists the functions of each port and the peripheral with which the ports also serve as:

- Port 0: General-purpose I/O port. Also serves as peripherals (external interrupt 2 pins).
- Port 1: General-purpose I/O port. Also serves as peripherals (external interrupt 1 pins).
- Port 2: General-purpose I/O port. Also serves as peripherals (timer, serial I/O, PWM 2 pins).
- Port 3: Output-only port. Also serves as peripherals (buzzer/remote control, PWM 1, external capacitance connection pins).
- Port 4: Output-only port. Also serves as peripherals LCDC segment outputs.
- Port 5: Output-only port. Also serves as peripherals (analog input pins).
- Port 6: Output-only port. Also serves as peripherals (LCDC segment output pins).
- Port 7: output-only port. Also serves as peripherals (LCDC common output pins).

Table 4.1a lists the functions of each port and Table 4.1b lists the registers for each port.
### Table 4.1a Port Function

<table>
<thead>
<tr>
<th>Port</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>Port0</td>
<td>P00/INT20 to P07/INT27</td>
<td>CMOS (resource: hysteresis)</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>External interrupts 2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>INT27</td>
<td>INT26</td>
<td>INT25</td>
<td>INT24</td>
</tr>
<tr>
<td>Port1</td>
<td>P10/INT10 to P14/INT13 P14 to P17</td>
<td></td>
<td>General-purpose I/O port</td>
<td>P17 P16 P15 P14 P13 P12 P11 P10</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>INT13</td>
<td>INT12</td>
<td>INT11</td>
<td>INT10</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Peripherals</td>
<td>PWL2 SCK SO SI TO EC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port3</td>
<td>P30/B2/RCO to P33/C0</td>
<td>Analog channel selector</td>
<td>N-ch open-drain</td>
<td>Output-only port</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>P33</td>
<td>P32</td>
<td>P31</td>
<td>P30</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Peripherals</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>C0</td>
<td>C1</td>
<td>PWM1</td>
<td>B2/RCO</td>
</tr>
<tr>
<td>Port4</td>
<td>P40/SEG16 to P47/SEG23</td>
<td></td>
<td>N-ch open-drain</td>
<td>Output-only port</td>
<td>P47</td>
<td>P46</td>
<td>P45</td>
<td>P44</td>
<td>P43</td>
<td>P42</td>
<td>P41</td>
<td>P40</td>
</tr>
<tr>
<td>Port5</td>
<td>P50/AN0 to P57/AN7</td>
<td>Analog channel selector</td>
<td>N-ch open-drain</td>
<td>Output-only 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>Analog input</td>
<td></td>
<td>AN7</td>
<td>AN6</td>
<td>AN5</td>
<td>AN4</td>
<td>AN3</td>
<td>AN2</td>
<td>AN1</td>
<td>AN0</td>
</tr>
<tr>
<td>Port6</td>
<td>P60/SEG8 to P67/SEG15</td>
<td></td>
<td>N-ch open-drain</td>
<td>Output-only 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>LCDC segment output</td>
<td></td>
<td>SEG15</td>
<td>SEG14</td>
<td>SEG13</td>
<td>SEG12</td>
<td>SEG11</td>
<td>SEG10</td>
<td>SEG9</td>
<td>SEG8</td>
</tr>
<tr>
<td>Port7</td>
<td>P70/COM2 to P71/COM3</td>
<td></td>
<td>N-ch open-drain</td>
<td>Output-only port</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>P71</td>
<td>P70</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>LCDC common output</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>COM3</td>
<td>COM2</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

1: Ports 4, 6, and 7 are output ports only when the ports are selected as a mask option.
2: In devices that have on-chip voltage boosters (MB89160A series), P32 and P33 of Port 3 are used to connect external capacitors, and cannot be used as output ports. P30 and P31 are CMOS outputs and P32 and P33 are N-ch open drain outputs.
3: Pins P21, P26, and P27 have high current drive-type output circuits.

### Table 4.1b Port Registers

<table>
<thead>
<tr>
<th>Register</th>
<th>Read/Write</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>XXXXX0000x</td>
</tr>
<tr>
<td>Port 0 data direction register (DDR0)</td>
<td>W*</td>
<td>0011H</td>
<td>00000000x</td>
</tr>
<tr>
<td>Port 1 data register (PDR1)</td>
<td>R/W</td>
<td>0002H</td>
<td>XXXXX0000x</td>
</tr>
<tr>
<td>Port 1 data direction register (DDR1)</td>
<td>W*</td>
<td>0003H</td>
<td>00000000s</td>
</tr>
<tr>
<td>Port 2 data register (PDR2)</td>
<td>R/W</td>
<td>0004H</td>
<td>XXXXX0000x</td>
</tr>
<tr>
<td>Port 2 data direction register (DDR2)</td>
<td>W*</td>
<td>0005H</td>
<td>00000000s</td>
</tr>
<tr>
<td>Port 3 data register (PDR3)</td>
<td>R/W</td>
<td>000CH</td>
<td>XXXX11111s</td>
</tr>
<tr>
<td>Port 4 data register (PDR4)</td>
<td>R/W</td>
<td>000EH</td>
<td>111111111s</td>
</tr>
<tr>
<td>Port 5 data register (PDR5)</td>
<td>R/W</td>
<td>000FH</td>
<td>111111111s</td>
</tr>
<tr>
<td>Port 6 data register (PDR6)</td>
<td>R/W</td>
<td>0012H</td>
<td>111111111s</td>
</tr>
<tr>
<td>Port 7 data register (PDR7)</td>
<td>R/W</td>
<td>0013H</td>
<td>XXXX11111s</td>
</tr>
</tbody>
</table>

* Bit manipulation instructions cannot be used on DDR0, DDR1, and DDR2. R/W: Readable and writable

R : Read-only
W : Write-only
X : Indeterminate
4.2 Port 0 and Port 1

Port 0 and port 1 are general-purpose I/O ports that also serve as input pins for external interrupts.

This section principally describes the port functions when operating as general-purpose I/O ports.

The section describes the port structure and pins, the pin block diagram, and the registers for port 0 and port 1.

■ Structure of Port 0 and Port 1

Port 0 and port 1 consist of three components respectively.

- **Port 0**
  - General-purpose I/O pins/external interrupt 2 input pins (P00/INT20 to P07/INT27)
  - Port 0 data register (PDR0)
  - Port 0 data direction register (DDR0)

- **Port 1**
  - General-purpose I/O pins/external interrupt 1 input pins (P10/INT10 to P13/INT13) and general-purpose I/O pins (P14 to P17)
  - Port 1 data register (PDR1)
  - Port 1 data direction register (DDR1)

■ Port-0 and Port-1 Pins

Port 0 and port 1 both consist of eight I/O pins of a CMOS input and CMOS output type respectively.

When P00/INT20 to P07/INT27 (Port 0) and P10/INT10 to P13/INT13 (Port 1) are used as input pins, they can also be used as external interrupt input pins.

* External interrupt inputs are hysteresis inputs.

Reference: See Section 1.7, "I/O Pins and Pin Functions" for a description of the circuit type.

<table>
<thead>
<tr>
<th>Port</th>
<th>Pin name</th>
<th>Function</th>
<th>Shared peripheral</th>
<th>I/O type</th>
<th>Circuit type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Input</td>
<td>Output</td>
</tr>
<tr>
<td>Port0</td>
<td>P00/INT20 to P07/INT27</td>
<td>General-purpose I/O</td>
<td>External interrupt 2</td>
<td>CMOS*</td>
<td>E</td>
</tr>
<tr>
<td>Port1</td>
<td>P10/INT10 to P13/INT13</td>
<td>General-purpose I/O</td>
<td>External interrupt 1</td>
<td>CMOS</td>
<td>F</td>
</tr>
<tr>
<td></td>
<td>P14 to P17</td>
<td>General-purpose I/O</td>
<td>—</td>
<td>CMOS</td>
<td></td>
</tr>
</tbody>
</table>

* External interrupt inputs are hysteresis inputs.

Reference: See Section 1.7, "I/O Pins and Pin Functions" for a description of the circuit type.
Block Diagram of Port-0 and Port-1 Pin

![Diagram of Port-0 and Port-1 Pin](image)

**Figure 4.2a Block Diagram of Port-0 and Port-1 Pin**

Check: When a port is used as a normal input port, external interrupt circuit operation that use the same pins must be disabled, see Chapter 11, “External Interrupt Circuit 1” and Chapter 12, “External Interrupt Circuit 2” for details.

Port-0 and Port-1 Registers

The port-0 registers consist of PDR0 and DDR0. The port-1 registers consist of PDR1 and DDR1. Each bit in these registers has a one-to-one relationship with port-0 and port-1 pin respectively. Table 4.2b shows the correspondence between pins and registers for port-0 and port-1.

**Table 4.2b Correspondence between Pin and Register for Port-0 and Port-1**

<table>
<thead>
<tr>
<th>Port</th>
<th>Correspondence between register bit and pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port0</td>
<td>PDR0,DDR0 Bit 7 Bit 6 Bit 5 Bit 4 Bit 3 Bit 2 Bit 1 Bit 0</td>
</tr>
<tr>
<td></td>
<td>Corresponding pin P07 P06 P05 P04 P03 P02 P01 P00</td>
</tr>
<tr>
<td>Port1</td>
<td>PDR1,DDR1 Bit 7 Bit 6 Bit 5 Bit 4 Bit 3 Bit 2 Bit 1 Bit 0</td>
</tr>
<tr>
<td></td>
<td>Corresponding pin P17 P16 P15 P14 P13 P12 P11 P10</td>
</tr>
</tbody>
</table>
4.2 Port 0 and Port 1

4.2.1 Port-0 and Port-1 Registers (PDR0, PDR1, DDR0, DDR1)

This section describes the port-0 and port-1 registers.

- **Port-0 and Port-1 Register Functions**
  
  - **Port 0, 1 data registers (PDR0, PDR1)**
    
    The PDR0 and PDR1 registers hold the pin states. Therefore, a bit corresponding to a pin set as an output port can be read as the same state ("0" or "1") as the output latch, but when it is an input port, it cannot be read as the output latch state.

    **Note:** For SETB and CLRB bit operation instructions, since the state of output latch (not the pin) is read, the output latch states of bits other than those being operated on are not changed.

  - **Port 0, 1 data direction registers (DDR0, DDR1)**
    
    The DDR0 and DDR1 registers set the direction (input or output) for each pin (bit).

    Setting "1" to the bit corresponding to a port (pin) sets the pin as an output port. Setting "0" sets the pin as an input port.

    **Check:** As the DDR0 and DDR1 registers are write-only, the bit manipulation instructions (SETB and CLRB) cannot be used.

  - **Settings when pins are used as external interrupt inputs**
    
    When port pins are used as external interrupt input pins, in addition to enabling the interrupt circuit (external interrupt 1 or 2), the corresponding pins must also be set as inputs. (The corresponding output latch data has no significance in this case.)
Table 4.2.1a lists the functions of the port-0 and port-1 registers.

#### Table 4.2.1a Port-0 and Port-1 Register Function

<table>
<thead>
<tr>
<th>Register</th>
<th>Data</th>
<th>Read</th>
<th>Write</th>
<th>Read/Write</th>
<th>Address</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 0 data register (PDR0)</td>
<td>0</td>
<td>Pin state is the “L” level.</td>
<td>R/W 0000h</td>
<td>XXXXXXXX1a</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Pin state is the “H” level.</td>
<td>R/W 0001h</td>
<td>000000001a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 0 data direction register (DDR0)</td>
<td>0</td>
<td>Reading is not permitted (write-only).</td>
<td>W 0011h</td>
<td>000000001a</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Enables output transistor and sets the pin as output pin.</td>
<td>R/W 0002h</td>
<td>XXXXXXXX1a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 1 data register (PDR1)</td>
<td>0</td>
<td>Pin state is the “L” level.</td>
<td>R/W 0000h</td>
<td>XXXXXXXX1a</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Pin state is the “H” level.</td>
<td>R/W 0003h</td>
<td>000000001a</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 1 data direction register (DDR1)</td>
<td>0</td>
<td>Reading is not permitted. (write-only.)</td>
<td>W 0011h</td>
<td>000000001a</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Enables output transistor and sets the pin as output pin.</td>
<td>R/W 0002h</td>
<td>XXXXXXXX1a</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Readable and writable
W: Write-only
X: Indeterminate
4.2 Port 0 and Port 1

4.2.2 Operation of Port 0 and Port 1

This section describes the operations of the port 0 and port 1.

- **Operation of Port 0 and Port 1**
  - **Operation as an output port**
    - Setting the corresponding DDR0 or DDR1 register bit to “1” sets a pin as an output port.
    - When a pin is set as an output port, its output transistor is enabled and the pin outputs the data stored in the output latch.
    - Writing data to the PDR0 and PDR1 registers stores the data in the output latch and outputs the data directly to the pin.
    - Reading the PDR0 or PDR1 register returns the pin value.
  - **Operation as an input port**
    - Setting the corresponding DDR0 or DDR1 register bit to “0” sets a pin as an input port.
    - When a pin is set as an input port, the output transistor is “OFF” and the pin goes to the high-impedance state.
    - Writing data to the PDR0 and PDR1 registers stores the data in the output latch but does not output the data to the pin.
    - Reading the PDR0 or PDR1 register returns the pin value.
  - **Operation as an external interrupt input**
    - When a port is an external interrupt input, the port is made an input by setting the corresponding DDR0 or DDR1 register bits to “0”.
    - Reading the PDRO or PDR1 register returns the pin value, regardless of whether external interrupt inputs or interrupt request outputs are enabled/disabled.
  - **Operation at reset**
    - Resetting the CPU initializes the DDR0 and DDR1 register values to “0”. This sets the output transistors “OFF” (all pins become input ports) and sets the pins to the high-impedance state.
    - The PDR0 and PDR1 registers are not initialized by a reset. Therefore, to use as output ports, the output data must be set in the PDR0 and PDR1 registers before setting the corresponding DDR0 or DDR1 register bits to output mode.
  - **Operation in stop and watch modes**
    - The pins go to the high-impedance state if the pin state specification bit in the standby control register (STBC: SPL) is “1” when the device changes to stop or watch mode. This is achieved by forcibly setting the output transistor “OFF” regardless of the DDR0 and DDR1 register values.

Table 4.2.2a lists the port-0 and port-1 pin states.
Table 4.2.2a Port-0 and Port-1 Pin State

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Normal operation</th>
<th>Main-stop mode (SPL = 1)</th>
<th>Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>main-sleep mode</td>
<td>sub-stop mode (SPL = 0)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>main-stop mode</td>
<td>sub-stop mode (SPL = 0)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>watch mode</td>
<td>watch mode (SPL = 0)</td>
<td></td>
</tr>
<tr>
<td>P00/INT20 to P07/INT27</td>
<td>General-purpose I/O ports/external interrupt input</td>
<td>Hi-z (external interrupt input)</td>
<td>Hi-z</td>
</tr>
<tr>
<td>P10/INT10 to P13/INT13 P14 to P17</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Note: Pins with a pull-up resistor (optional) go to the "H" level (pull-up state) rather than to the high-impedance state when the output transistors are all "OFF."

SPL: Pin state specification bit in the standby control register (STBC)
Hi-z: High impedance
4.3 Port 2

Port 2 is a general-purpose I/O port that also serves as the resource signal I/O pins. Individual pins can be switched in units of a bit between the port and resource functions. This section principally describes the port functions when operating as a general-purpose I/O port.

The section describes the port structure and pins, the pin block diagram, and the port registers for port 2.

Structure of Port 2

Port 2 consists of the following three components:

- General-purpose I/O port/resource I/O pins (P20/EC to P27/PWM2)
- Port 2 data register (PDR2)
- Port 2 data direction register (DDR2)

Port-2 Pins

Port 2 consists of eight I/O pins of a CMOS input, an N-ch open-drain output types. Six of these pins are also used as signal I/O pins for various resources. While they are being used by the resource, these pins cannot be used as the general-purpose I/O port. Some pin outputs are high current drive-type outputs.

Table 4.3a lists the port-2 pins.

<table>
<thead>
<tr>
<th>Port 2</th>
<th>Pin name</th>
<th>Function</th>
<th>Shared peripheral</th>
<th>I/O type</th>
<th>Circuit type</th>
</tr>
</thead>
<tbody>
<tr>
<td>P20/EC</td>
<td>P20 General-purpose I/O</td>
<td>EC 8/16-bit timer, pulse input</td>
<td>CMOS$^{1}$</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td>P21</td>
<td>P21 General-purpose I/O</td>
<td>–</td>
<td>CMOS</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>P22/TO</td>
<td>P22 General-purpose I/O</td>
<td>TO 8/16-bit timer, timer output</td>
<td>CMOS$^{1}$</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td>P23/SI</td>
<td>P23 General-purpose I/O</td>
<td>SI 8-bit serial I/O, data input</td>
<td>CMOS$^{1}$</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>P24/SO</td>
<td>P24 General-purpose I/O</td>
<td>SO 8-bit serial I/O, data output</td>
<td>CMOS</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>P25/SCK</td>
<td>P25 General-purpose I/O</td>
<td>SCK 8-bit serial I/O, clock I/O</td>
<td>CMOS$^{1}$</td>
<td>H</td>
<td></td>
</tr>
<tr>
<td>P26</td>
<td>P26 General-purpose I/O</td>
<td>–</td>
<td>CMOS</td>
<td>I</td>
<td></td>
</tr>
<tr>
<td>P27/PWM2</td>
<td>P27 General-purpose I/O</td>
<td>PWM 2 8-bit PWM timer 2, timer output</td>
<td>CMOS</td>
<td>H</td>
<td></td>
</tr>
</tbody>
</table>

$^{1}$ Peripheral inputs are hysteresis inputs.

$^{2}$ P21, P26, and P27 have high current drive-type outputs.

Reference: See Section 1.7, “I/O Pins and Pin Functions” for a description of the circuit type.
■ Block Diagram of Port-2 Pin

![Block Diagram of Port-2 Pin](image)

**Figure 4.3a Block Diagram of Port-2 Pin**

*Note*: Peripheral inputs continuously are input the pin value (except during stop and watch modes).

*Check*: The products that can be set with a pull-up resistor (optional) are the MB89161, MB89163, and MB89165.

■ Port-2 Registers

The port-2 registers consist of PDR2 and DDR2.

Each bit in these registers has a one-to-one relationship with a port 2 bit and port 2 pin.

Table 4.3b shows the correspondence between pins and registers for port 2.

**Table 4.3b Correspondence between Pin and Register for Port 2**

<table>
<thead>
<tr>
<th>Port</th>
<th>Correspondence between register bit and pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 2</td>
<td></td>
</tr>
<tr>
<td>PDR2,DDR2</td>
<td>Bit 7</td>
</tr>
<tr>
<td>P27, P26, P25, P24, P23, P22, P21, P20</td>
<td></td>
</tr>
</tbody>
</table>
4.3 Port 2

4.3.1 Port-2 Registers (PDR2, DDR2)

This section describes the port-2 registers.

- **Port-2 Register Functions**
  - **Port 2 data register (PDR2)**
    The PDR2 register holds the pin states. Therefore, when used as an output port that is not a peripheral output, it reads out as the same state ("0" or "1") as that of the output data latch; and when it is an input port, the output latch state cannot be read out.
    
    **Note:** As the bit manipulation instructions (SETB and CLRB) read the output latch data rather than the pin level, the instructions do not change the output latch values for bits other than the bit being set or cleared.
  
  - **Port 2 data direction register (DDR2)**
    The DDR2 register sets the direction (input or output) for each pin (bit).
    Setting "1" to the bit corresponding to a port (pin) sets the pin as an output port. Setting "0" sets the pin as an input port.
    
    **Check:** As the DDR2 register is write-only, the bit manipulation instructions (SETB and CLRB) cannot be used.
  
  - **Settings as a peripheral output**
    To use a peripheral that has an output pin, set the peripheral output enable bit for that pin to the "enable" state. As can be seen in the block diagram, the peripheral has precedence over the general-purpose port for use of the output pin. Once the peripheral output is enabled, the states set in the PDR2 and DDR2 registers are no longer valid, and do not affect the data output by the peripheral, or the enabling of the output.
  
  - **Settings as a peripheral input**
    To use a peripheral that has a Port 2 pin as an input pin, set that pin as an input port. The output latch data for that pin will no longer be valid.
Table 4.3.1a lists the functions of the port-2 registers.

Table 4.3.1a Port-2 Register Function

<table>
<thead>
<tr>
<th>Register</th>
<th>Data</th>
<th>Read</th>
<th>Write</th>
<th>Read/Write</th>
<th>Address</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 2 data register (PDR2)</td>
<td></td>
<td></td>
<td>Pin state is the “L” level. Outputs an “L” level to the pin if the</td>
<td>R/W</td>
<td>0004H</td>
<td>XXXXXXXXa</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>pin functions as an output port. (Sets “0” to the output latch and</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>turns the output transistor “ON”.)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td></td>
<td>Pin state is the “H” level. Sets the pin to the high-impedance</td>
<td>R/W</td>
<td>0005H</td>
<td>000000000a</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>state if the pin functions as an output port”. (Sets “1” to the</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>output latch and</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>turns the output transistor “OFF”;)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 2 data direction register (DDR2)</td>
<td>0</td>
<td>Reading</td>
<td>Disables the output transistor and sets the pin as an input pin.</td>
<td>W</td>
<td>0005H</td>
<td>000000000a</td>
</tr>
<tr>
<td></td>
<td></td>
<td>not</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>permitted</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>(write-only)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>Enables the output transistor and sets the pin as an output pin.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Readable and writable
W: Write-only
X: Indeterminate
* : Pins with a pull-up resistor (optional), go to the pull-up state.
4.3 Port 2

4.3.2 Operation of Port 2

This section describes the operations of the port 2.

### Operation of Port 2

- **Operation as an output port**
  - Setting the corresponding DDR2 register bit to “1” sets a pin as an output port.
  - When a pin is as an output port, the output transistor is enabled. When the output latch value is “0,” the output transistor turns “ON” and an “L” level is output from the pin. When the output latch value is “1,” the transistor turns “OFF” and the pin goes to the high-impedance state. If a pull-up is set to the output pin, the pin goes to the pull-up state when the output latch value is “1”.
  - Writing data to the PDR2 register stores the data in the output latch and outputs the data to the pin.
  - Reading the PDR2 register returns the pin value.

- **Operation as an input port**
  - Setting the corresponding DDR2 register bit to “0” sets a pin as an input port.
  - When a pin is set as an input port, the output transistor is “OFF” and the pin goes to the high-impedance state.
  - Writing data to the PDP2 register stores the data in the output latch but does not output the data to the pin.
  - Reading the PDR2 register returns the pin value.

- **Operation as a peripheral output**
  - If a peripheral output enable bit is set to “enable,” the corresponding pin becomes a peripheral output.
  - As the pin value can be read even if the peripheral output is enabled, the peripheral output value can be read via the PDR2 register.

- **Operation as a peripheral input**
  - A port pin is set as a peripheral input by setting the corresponding DDR2 register bit to “0”.
  - Reading the PDR2 register returns the pin value, regardless of whether or not the peripheral is using the input pin.

- **Operation at reset**
  - Resetting the CPU initializes the DDR2 register value to “0”. This sets output transistors “OFF” (pins become input ports) and sets the pins to the high-impedance state.
  - The PDR2 register is not initialized by a reset. Therefore, to use as output ports, the output data must be set in the PDR2 register before setting the corresponding DDR2 register bit to output mode.
  - P27 pin state is undetermined until the internal clock starts operation.
**Operation in stop and watch modes**

The pins go to the high-impedance state, if the pin state specification bit in the standby control register (STBC: SPL) is “1” when the device changes to stop or watch mode. This is achieved by forcibly setting the output transistor “OFF” regardless of the DDR2 register value.

Table 4.3.2a lists the port-2 pin states.

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Normal operation main-sleep mode</th>
<th>Main-stop mode (SPL = 1)</th>
<th>Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>sub-sleep mode (SPL = 0)</td>
<td>sub-stop mode (SPL = 0)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>watch mode (SPL = 0)</td>
<td>watch mode (SPL = 1)</td>
<td></td>
</tr>
<tr>
<td>P20/EC to P27/PWM2</td>
<td>General-purpose I/O port/peripheral I/O</td>
<td>Hi-z</td>
<td>Hi-z*1</td>
</tr>
</tbody>
</table>

*1: P27 pin state is undetermined until the internal clock starts operation.
SPL: Pin state specification bit in the standby control register (STBC)
Hi-z: High impedance

**Note:** Pins with a pull-up resistor (optional) go to the “H” level (pull-up state) rather than to the high-impedance state when the output transistor is turned “OFF”.

Table 4.3.2a Port-2 Pin State
4.4 Port 3

Port 3 is an output-only port that also serves as peripheral output. Each pin can be switched between peripheral and port operation in bit units. This section principally describes the port functions when operating as an output-only port. The section describes the port structure and pins, the pin block diagram, and the port registers for port 3.

Structure of Port 3

Port 3 consist of the following two components:
- Output-only pins/peripheral output pins (P30/BZ/RCO to P33/C0)
- Port 3 data register (PDR3)

Port-3 Pins

Port 3 has four output-only pins: two CMOS outputs and two N-ch open-drain outputs. In devices that have on-chip voltage boosters (MB89160A series) pins P32/C1 and P33/C0 are used only as external capacitor connections C0 and C1, and should not be used as P32 and P33 output-only port pins.

Table 4.4a lists the port-3 pins.

Table 4.4a Port-3 Pin

<table>
<thead>
<tr>
<th>Port</th>
<th>Pin name</th>
<th>Function</th>
<th>Shared peripheral</th>
<th>I/O type</th>
<th>Circuit type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port3</td>
<td>P30/BZ/RCO</td>
<td>P30 Output-only</td>
<td>BZ/RCO Buzzer output/remote control output</td>
<td>CMOS</td>
<td>G</td>
</tr>
<tr>
<td></td>
<td>P31/PWM1</td>
<td>P31 Output-only</td>
<td>PWM 1 output for the 8-bit PWM timer/counter</td>
<td>–</td>
<td>J^3</td>
</tr>
<tr>
<td></td>
<td>P32/C1</td>
<td>P32 Output-only(^1)</td>
<td>C1 External capacitor connection pin(^4)</td>
<td>–</td>
<td>–</td>
</tr>
<tr>
<td></td>
<td>P33/C0</td>
<td>P33 Output-only(^1)</td>
<td>C0 External capacitor connection pin(^4)</td>
<td>–</td>
<td>–</td>
</tr>
</tbody>
</table>

\(^1\), \(^2\), \(^3\): Functions as output port only in devices that do not have on-chip voltage boosters. (MB89160 & 160L series)

\(^4\): Function as capacitor connection pins only in devices that have voltage boosters. (MB89160A series)

Reference: See Section 1.7, “I/O Pins and Pin Functions” for a description of the circuit type.
Block Diagram of Port-3 Pin

Figure 4.4a Block Diagram of Port-3 Pin (Pins P30 and P31)

Figure 4.4b Block Diagram of Port-3 Pin (Pins P32 and P33)

Port-3 Registers

The port-3 registers consist of PDR3.

Each bit in PDR3 registers has a one-to-one relationship with a port-3 pin. Table 4.4b shows the correspondence between pins and registers for port 3.

Table 4.4b Correspondence between Pin and Register for Port 3

<table>
<thead>
<tr>
<th>Port</th>
<th>Correspondence between register bit and pin</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port3</td>
<td></td>
</tr>
<tr>
<td>PDR3</td>
<td>-  -  -  -  Bit 3  Bit 2  Bit 1  Bit 0</td>
</tr>
<tr>
<td></td>
<td></td>
</tr>
<tr>
<td>Corresponding Pin</td>
<td>-  -  -  -  P33  P32  P31  P30</td>
</tr>
</tbody>
</table>
4.4 Port 3

4.4.1 Port-3 Register (PDR3)

This section describes the Port-3 register.

### Port-3 Register Functions

- **Port 3 data register (PDR3)**
  
  The PDR3 register holds the output latch states. Therefore, it does not read out as the pin states or peripheral output data.

- **Settings as a peripheral output**
  
  When using peripherals that have output pins, set the peripheral’s output enable bit for that pin to the “enable” state.

  Since the peripheral has precedence over the port for use of the output pin, once the peripheral output is enabled, the data in the PDR3 register has no significance, and has no affect on the data output by the peripheral, or the enabling of the output.

Table 4.4.1a lists the functions of the port-3 registers.

<table>
<thead>
<tr>
<th>Register</th>
<th>Data</th>
<th>Read</th>
<th>Write</th>
<th>Read/Write</th>
<th>Address</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 3 data register (PDR3)</td>
<td>0</td>
<td>Output latch value is “0”.</td>
<td>Sets “0” to the output latch.</td>
<td>R/W</td>
<td>000CH</td>
<td>XXXX1111B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Output latch value is “1”.</td>
<td>Sets “1” to the output latch.</td>
<td>R/W</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Readable and writable
X: Indeterminate
* A “1” in the register turns the output transistor off. Since P32 and P33 are N-ch open-drain outputs, this puts the pin in the high-impedance state.
4.4 Port 3

4.4.2 Operation of Port 3

This section describes the operations of the port 3.

- Operation of Port 3

  - Operation as an output port
    - Writing data to the PDR3 register stores the data in the output latch. The pin outputs the data stored in the output latch at P30 and P31. In P32 and P33, when the output latch value is “0,” the output transistor turns “ON” and an “L” level is output from the pin. When the output latch value is “1”, the transistor turns “OFF” and the pin goes to the high-impedance state. If a pull-up is set to the output pin, the pin goes to the output pin, the pin goes to the pull-up state when the output latch value is “1”.
    - Reading the PDR3 register always returns the output latch value.
    - Check: P32 and P33 cannot be used as an output port in devices that have on-chip voltage boosters (MB89160A series). Set the PDR3-register bits 2 and 3 corresponding to the peripheral input pins to “1” to turn the output transistor “OFF”.

  - Operation as a peripheral output (P30/BZ/RCO and P31/PWM1 only)
    - Setting the output enable bit of the peripheral to “enable” makes the corresponding pin a peripheral output.
    - You cannot read the peripheral output value by reading PDR3. (PDR3 contains the output latch value.)

  - Operation at reset
    Resetting the CPU initializes the PDR3 register values to “1”. This outputs “H” level at P30 and P31, and turns “OFF” the P32 and P33 output transistors, which sets the pins to the high-impedance state.
    - P31 pin state is undetermined until the internal clock starts operation.

  - Operation in stop and watch modes
    The output transistors are forcibly turned “OFF” and the pins go to the high-impedance state if the pin state specification bit in the standby control register (STBC:SPL) is “1” when the device changes to stop or watch mode.
    - Table 4.4.2a lists the port-3 pin states.

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Normal operation</th>
<th>Main-sleep mode</th>
<th>Main-stop mode (SPL = 0)</th>
<th>Sub-sleep mode</th>
<th>Sub-stop mode (SPL = 0)</th>
<th>Watch mode (SPL = 0)</th>
<th>Main-stop mode (SPL = 1)</th>
<th>Sub-stop mode (SPL = 1)</th>
<th>Watch mode (SPL = 1)</th>
<th>Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>P30/BZ/RCO</td>
<td>Output-only port/peripheral output</td>
<td>Hi-z</td>
<td>Output “H”</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P31/PWM1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P32/C1</td>
<td>Output-only port (capacitor connection pins in A series devices)</td>
<td>Hi-z</td>
<td>Hi-z*1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P33/C0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Hi-z*1</td>
<td></td>
</tr>
</tbody>
</table>

*1: P31 pin state is undetermined until the internal clock starts operation.
SPL: Pin state specification bit in the standby control register (STBC)
Hi-z: High impedance
4.5 Ports 4, 6 and 7

Ports 4, 6, and 7 are output-only ports that also serve as the LCDC common and segment outputs. The output ports and LCDC segment (and common) outputs are selected as mask options. This section principally describes the port functions when operating as an output-only port. The section describes the port structure and pins, the pin block diagram, and the port register for port 4, 6, and 7.

■ Structure of Port 4, 6, and 7

Ports 4, 6 and 7 are each made up of two elements.

Port 4:
• Output-only pins/LCDC segment output pins (P40/SEG16 to P47/SEG23)
• Port 4 data register (PDR4)

Port 6:
• Output-only pins/LCDC segment output pins (P60/SEG8 to P67/SEG15)
• Port 6 data register (PDR6)

Port 7:
• Output-only pins/LCDC common output pins (P70/COM2 to P71/COM3)
• Port 7 data register (PDR7)

■ Port-4, 6, and 7 Pins

Ports 4 and 6 both consist of eight output-only pins of an N-ch open-drain output type, and Port 7 consists of two.

The pin functions are set as mask options. When the LCDC common and segment outputs are selected Do not use these pins as output-only ports. Ports 4 and 6 options can be selected in four-bit groups.

Table 4.5a lists the port-4, 6, and 7 pins.

### Table 4.5a Port-4, 6, and 7 Pin

<table>
<thead>
<tr>
<th>Port</th>
<th>Pin name</th>
<th>Function</th>
<th>Shared peripheral</th>
<th>I/O type</th>
<th>Circuit type</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Input</td>
<td>Output</td>
</tr>
<tr>
<td>Port 4</td>
<td>P40/SEG16 to P47/SEG23</td>
<td>P40 to P47</td>
<td>SEG16 to SEG23 LCDC segment output</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 6</td>
<td>P60/SEG8 to P67/SEG15</td>
<td>P60 to P67</td>
<td>SEG8 to SEG15 LCDC segment output</td>
<td>—</td>
<td>N-ch open-drain</td>
</tr>
<tr>
<td>Port 7</td>
<td>P70/COM2, P71/COM3</td>
<td>P70, P71</td>
<td>COM2, COM3 LCDC common output</td>
<td></td>
<td>J*</td>
</tr>
</tbody>
</table>

*: The circuit type is “K” when LCDC segment output pin is selected as a mask option.

Reference: See Section 1.7, “I/O Pins and Pin Functions” for a description of the circuit type.

See Chapter 16, “LCD Controller-Drive” for details of pin operation when used as an LCDC common and segment outputs.
Block Diagram of Port-4, 6 and 7 Pin

Check:
- If you use Port 4, 6 and 7 pins as the LCD common and segment output pins, do not set a pull-up resistor (optional) for those pins, and do not use the pins as output ports.
- The products that can be set with a pull-up resistor (optional) are the MB89161, MB89163, and MB89165. Pull-up resistors cannot be selected for Port 7 pins. (Select the "no pull-up" option.)

Port-4, 6, and 7 Registers

Port 4, 6, and 7 registers consist of PDR4, PDR6, and PDR7. Each bit in these registers has a one-to-one relationship with a port 4, 6, and 7 bit and pin respectively.

Table 4.5b shows the correspondence between the pins and register for port-4, 6, and 7.

| Table 4.5b  Correspondence between Pin and Register for Port 4, 6, and 7 |
|-------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|-----------------|
| Port 4      | PDR4            | Bit 7           | Bit 6           | Bit 5           | Bit 4           | Bit 3           | Bit 2           | Bit 1           | Bit 0           |
| Corresponding pin | P47   | P46             | P45             | P44             | P43             | P42             | P41             | P40             |
| Port 6      | PDR6            | Bit 7           | Bit 6           | Bit 5           | Bit 4           | Bit 3           | Bit 2           | Bit 1           | Bit 0           |
| Corresponding pin | P67   | P66             | P65             | P64             | P63             | P62             | P61             | P60             |
| Port 7      | PDR7            | —               | —               | —               | —               | —               | —               | —               | —               |
| Corresponding pin | —     | —               | —               | —               | —               | —               | —               | —               | —               |

SPL: Pin state specification bit in the standby control register (STBC)

Figure 4.5a Block Diagram of Port-4, 6 and 7 Pin
4.5 Ports 4, 6 and 7

4.5.1 Port-4, Port-6, and Port-7 Registers (PDR4, PDR6, and PDR7)

This section describes the port-4, port-6, and port-7 registers.

- Port-4, port-6, and port-7 Register Functions

  - Port 4, port 6, and port 7 data registers (PDR4, PDR6, and PDR7)
    The PDR4, 6, and 7 registers hold the states of their respective output data latches. Therefore, you cannot determine the pin states or LCDC common and segment output states by reading the registers.

  - Settings as an LCDC common and segment output
    To use pins as LCDC common and segment outputs, that function must be set as a mask option. The register bits for pins used for this purpose should be set to turn the output transistor “OFF”, to prevent it from interfering with the LCDC common and segment output.

Table 4.5.1a lists the functions of Port-4, port-6, and port-7 register.

Table 4.5.1a Port-4, Port-6, and Port-7 Register Function

<table>
<thead>
<tr>
<th>Register</th>
<th>Data</th>
<th>Read</th>
<th>Write</th>
<th>Read/Write</th>
<th>Address</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 4 data register (PDR4)</td>
<td>0</td>
<td>Output latch value is “0”.</td>
<td>Outputs an “L” level to the pin.¹</td>
<td>R/W</td>
<td>000EH</td>
<td>11111111B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Output latch value is “1”.</td>
<td>Sets the pin to the high-impedance state.²</td>
<td>R/W</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 6 Data Register (PDR6)</td>
<td>0</td>
<td>Output latch value is “0”.</td>
<td>Outputs an “L” level to the pin.¹</td>
<td>R/W</td>
<td>0012H</td>
<td>11111111B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Output latch value is “1”.</td>
<td>Sets the pin to the high-impedance state.²</td>
<td>R/W</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Port 7 Data Register (PDR7)</td>
<td>0</td>
<td>Output latch value is “0”.</td>
<td>Outputs an “L” level to the pin.¹</td>
<td>R/W</td>
<td>0013H</td>
<td>XXXXXX11B</td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Output latch value is “1”.</td>
<td>Sets the pin to the high-impedance state.²</td>
<td>R/W</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Readable and writable
X: Indeterminate
¹: Sets “0” to the output latch and turn the output transistor “ON”.
²: Sets “1” to the output latch and turn the output transistor “OFF”. Pins with a pull-up resistor (optional) go to the pull-up state.
4.5 Ports 4, 6 and 7

4.5.2 Operation of Port 4, Port 6 and Port 7

This section describes the operations of the port 4, port 6 and port 7.

- **Operation of Port 4, 6, and 7**
  
  **Operation as an output port**
  
  - When used as an output-only port (mask option), the pins cannot be used for LCDC common and segment outputs.
  
  - Writing data to the PDR4, 6, and 7 register stores the data in the output latches. When the output latch value is “0”, the output transistor turns “ON” and an “L” level is output from the pin. When the output latch value is “1”, the transistor turns “OFF” and the pin goes to the high-impedance state. If a pull-up is set to the output pin, the pin goes to the pull-up state when the output latch value is “1”.
  
  - Reading the PDR4, 6, and 7 register always returns the output latch data.

  **Operation as an LCDC common and segment output**
  
  - When the LCDC output option is selected, set the PDR4, 6, and 7-register bits corresponding to the LCDC common and segment output pins to “1” to turn the output transistor “OFF”.
  
  - You cannot read the LCDC output data by reading PDR4, 6 or 7. (If you read the PDR registers you will get the output latch data, not the LCDC output data.)

  **Operation at reset**
  
  Resetting the CPU initializes the PDR4, 6, and 7 register values to “1”. This turns “OFF” the output transistor for all pins and sets the pins to the high-impedance state.

  **Operation in stop and watch modes**
  
  The output transistors are forcibly turned “OFF” and the pins go to the high-impedance state if the pin state specification bit in the standby control register (STBC: SPL) is “1” when the device changes to stop or watch mode.

  Table 4.5.2a lists the Port-4, 6, and 7 pin states.

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Normal operation</th>
<th>Main-stop mode (SPL = 1)</th>
<th>Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>main-sleep mode</td>
<td>sub-stop mode (SPL = 0)</td>
<td></td>
</tr>
<tr>
<td></td>
<td>main-stop mode</td>
<td>watch mode (SPL = 0)</td>
<td></td>
</tr>
<tr>
<td>P40/SEG16 to P47/SEG23</td>
<td>Output-only port/LCDC segment output</td>
<td></td>
<td>Hi-z¹</td>
</tr>
<tr>
<td>P60/SEG8 to P67/SEG15</td>
<td>Output-only port/LCDC segment output</td>
<td></td>
<td>Hi-z²</td>
</tr>
<tr>
<td>P70/COM2, P71/COM3</td>
<td>Output-only port/LCDC common output</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*¹ : Does not go to the high-impedance state when the pin is used as an LCDC common and segment output.
*² : Goes “L” when the pin is used as an LCDC common and segment output.
SPL : Pin state specification bit in the standby control register (STBC)
Hi-z : High impedance

**Note**: Pins with a pull-up resistor (optional) go to the “H” level (pull-up state) rather than to the high-impedance state when the output transistor is turned “OFF”.

Table 4.5.2a Port-4, 6, and 7 Pin State
4.6 Port 5

Port 5 is an output-only port that also serves as an analog input. Each pin can be switched between analog input and port operation in bit units. This section principally describes the port functions when operating as an output-only port. The section describes the port structure and pins, the pin block diagram, and the port register for port 5.

Structure of Port 5

Port 5 consists of the following two components:
- Output-only pins/analog input pins (P50/AN0 to P57/AN7)
- Port-5 data register (PDR5)

Port-5 Pins

Port 5 consists of eight output pins of an N-ch open-drain output type. Do not use these pins as output-only ports when the pins are used as the analog input for the A/D converter.

Table 4.6a lists the port-5 pins.

<table>
<thead>
<tr>
<th>Port</th>
<th>Pin name</th>
<th>Function</th>
<th>Shared peripheral</th>
<th>I/O type</th>
<th>Circuit type</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 5</td>
<td>P50/AN0</td>
<td>P50 Output-only</td>
<td>AN0 Analog input 0</td>
<td>Input</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P51/AN1</td>
<td>P51 Output-only</td>
<td>AN1 Analog input 1</td>
<td>Output</td>
<td></td>
</tr>
<tr>
<td></td>
<td>P52/AN2</td>
<td>P52 Output-only</td>
<td>AN2 Analog input 2</td>
<td>Analog</td>
<td>L</td>
</tr>
<tr>
<td></td>
<td>P53/AN3</td>
<td>P53 Output-only</td>
<td>AN3 Analog input 3</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P54/AN4</td>
<td>P54 Output-only</td>
<td>AN4 Analog input 4</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P55/AN5</td>
<td>P55 Output-only</td>
<td>AN5 Analog input 5</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P56/AN6</td>
<td>P56 Output-only</td>
<td>AN6 Analog input 6</td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>P57/AN7</td>
<td>P57 Output-only</td>
<td>AN7 Analog input 7</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table 4.6a Port-5 Pin

Reference:
- See Section 1.7, “I/O Pins and Pin Functions” for a description of the circuit type.
- See Chapter 13, “A/D Converter” for details of pin operation when used as an analog input.
## Block Diagram of Port-5 Pin

![Block Diagram of Port-5 Pin](image)

- **PDR (Port data register)**
- **Internal data bus**
- **PDR read**
- **PDR write**
- **Output latch**
- **A/D converter channel selector**
- **Pull-up resistor (optional)** Approx. 50 kΩ / 5.0 V
- **P50 to P57**

**SPL**: Pin state specification bit in the standby control register (STBC)

### Figure 4.6a Block Diagram of Port-5 Pin

**Check**: • If using the A/D converter, do not set a pull-up resistor (optional) for any of P57/AN7 to P50/AN0.
• Do not use the pins as output ports if using as an analog input.

## Port-5 Registers

The port-5 register consists of PDR5. Each bit in the register has a one-to-one relationship with a port 5 pin. Table 4.6b shows the correspondence between the pins and register for port-5.

### Table 4.6b Correspondence between Pin and Register for Port 5

<table>
<thead>
<tr>
<th>Port</th>
<th>Correspondence between register bit and pin</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Port5</strong></td>
<td>Bit 7</td>
</tr>
<tr>
<td>PDR5</td>
<td></td>
</tr>
<tr>
<td>Corresponding pin</td>
<td>P57</td>
</tr>
</tbody>
</table>
4.6 Port 5

4.6.1 Port-5 Register (PDR5)

This section describes the port-5 register.

- Port-5 Register Functions
  - Port 5 data register (PDR5)
    The PDR5 register holds the output latch states. Therefore, pin states cannot be checked by reading this register.
  - Settings as an analog input
    When Port 5 pins are used as analog signal inputs, write “1” to the corresponding bits of PDR5 to turn the output transistors “OFF”, Setting the pins to the high-impedance state.

Table 4.6.1a lists the functions of the port-5 register.

<table>
<thead>
<tr>
<th>Register</th>
<th>Data</th>
<th>Read</th>
<th>Write</th>
<th>Read/Write</th>
<th>Address</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>Port 5 data</td>
<td>0</td>
<td>Output latch</td>
<td>Outputs an “L” level to the pin.</td>
<td>R/W</td>
<td>000Fh</td>
<td>11111111b</td>
</tr>
<tr>
<td>register (PDR5)</td>
<td></td>
<td>value is “0”.</td>
<td>(Sets “0” to the output latch and turn the output transistor “ON”.)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td>1</td>
<td>Output latch</td>
<td>Sets the pin to the high-impedance state.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>value is “1”.</td>
<td>(Sets “1” to the output latch and turn the output transistor “OFF”.)</td>
<td>R/W</td>
<td>000Fh</td>
<td>11111111b</td>
</tr>
</tbody>
</table>

R/W: Readable and writable
*: Pins with a pull-up resistor (optional) go to the pull-up state.
4.6.2 Operation of Port 5

This section describes the operations of the port 5.

- **Operation of Port 5**
  - **Operation as an output port**
    - Writing data to the PDR5 register stores the data in the output latch. When the output latch value is "0", the output transistor turns "ON" and an "L" level is output from the pin. When the output latch value is "1", the transistor turns "OFF" and the pin goes to the high-impedance state. If a pull-up is provided the output pin the pin goes to the pull-up state when the output latch value is "1".
    - Reading the PDR5 register always returns the output latch value.
  - **Operation as an analog input**
    - Set the PDR5 bit that corresponds to the analog input pin to "1" to turn the output transistor "OFF".
    - Reading the PDR5 register always returns the output latch value.
  - **Operation at reset**
    Resetting the CPU initializes the PDR5 register value to "1". This turns all the output transistors "OFF" and sets the pins to the high-impedance state.
  - **Operation in stop and watch modes**
    The output transistors are forcibly turned "OFF" and the pins go to the high-impedance state if the pin state specification bit in the standby control register (STBC: SPL) is "1" when the device changes to stop or watch mode.

Table 4.6.2a lists the port-5 pin states.

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Normal operation (main-sleep mode)</th>
<th>Normal operation (main-stop mode (SPL = 0))</th>
<th>Normal operation (sub-sleep mode)</th>
<th>Normal operation (sub-stop mode (SPL = 0))</th>
<th>Normal operation (watch mode (SPL = 0))</th>
<th>Main-stop mode (SPL = 1)</th>
<th>Sub-stop mode (SPL = 1)</th>
<th>Watch mode (SPL = 1)</th>
<th>Reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>P50/AN0 to P57/AN7</td>
<td>Output-only ports/analog input</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>Hi-z</td>
</tr>
</tbody>
</table>

SPL: Pin state specification bit in the standby control register (STBC)
Hi-z: High impedance

**Note**: Pins with a pull-up resistor (optional) go to the "H" level (pull-up state) rather than to the high-impedance state when the output transistor is turned "OFF".
4.7 Program Example for I/O Ports

This section gives an example program using the I/O ports.

- **Program Example for I/O Ports**
  - **Processing description**
    - Port 0 and port 1 are used to illuminate all elements of seven segment LED (eight segments if the decimal point is included).
    - The P00 pin is used for the anode common pin of the LED and the P10 to P17 pins operate as the segment pins.
  
  Figure 4.7a shows the connection example for an eight segment LED.

- **Coding example**
  ```assembly
  PDR0 EQU 0000H ; Address of the Port 0 data register
  DDR0 EQU 0001H ; Address of the Port 0 data direction register
  PDR1 EQU 0002H ; Address of the Port 1 data register
  DDR1 EQU 0003H ; Address of the Port 1 data direction register
  
  CODE SEGMENT
  
  CLRB PDR0:0; Set P00 to the “L” level.
  MOV PDR1, #11111111B; Set all port-1 pins to the “H” level.
  MOV DDR0, #11111111B; Set P00 as an output (#xxxxxxx1B).
  MOV DDR1, #11111111B; Set all port-1 pins as outputs.
  
  ENDS
  
  END
  ```

Figure 4.7a Connection Example for an Eight Segment LED
CHAPTER 5
TIMEBASE TIMER

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

5.1 Overview of Timebase Timer ........................................ 114
5.2 Block Diagram of Timebase Timer .............................. 115
5.3 Timebase Timer Control Register (TBTC)............... 116
5.4 Timebase Timer Interrupt ........................................... 118
5.5 Operation of Timebase Timer................................. 120
5.6 Notes on Using Timebase Timer.............................. 122
5.7 Program Example for Timebase Timer ...................... 123
5.1 Overview of Timebase Timer

The timebase timer provides interval timer functions. Four different interval times can be selected. The timebase timer uses a 21-bit free-run counter which counts-up in sync with the internal count clock (divide-by-two the main clock source oscillation). The timebase timer also provides the timer output for the oscillation stabilization delay time and the operating clock for the watchdog and timers.

The timebase timer stops operating in modes in which the main clock master oscillator is stopped.

- **Interval Timer Function**
  
  The interval timer function generates repeated interrupts at fixed time intervals.
  
  - The timer generates an interrupt each time the interval timer bit overflows on the timebase timer counter.
  
  - The interval timer bit (interval time) can be selected from four different settings.

  Table 5.1a lists the available interval time for the timebase timer.

<table>
<thead>
<tr>
<th>Internal count clock cycle</th>
<th>Interval time</th>
</tr>
</thead>
<tbody>
<tr>
<td>2/FCH (0.48 μs)</td>
<td>2^15/FCH (approx. 1.95 ms)</td>
</tr>
<tr>
<td></td>
<td>2^16/FCH (approx. 7.80 ms)</td>
</tr>
<tr>
<td></td>
<td>2^18/FCH (approx. 62.4 ms)</td>
</tr>
<tr>
<td></td>
<td>2^22/FCH (approx. 998.6 ms)</td>
</tr>
</tbody>
</table>

  FCH: Main clock source oscillation
  
  The values enclosed in parentheses ( ) are for a 4.2 MHz main clock source oscillation.

- **Clock Supply Function**

  The clock supply function provides the timer output used for the main clock oscillation stabilization delay time (four values), and operation clock for some peripheral functions.

  Table 5.1b lists the cycles of the clocks that the timebase timer supplies to various peripherals.

<table>
<thead>
<tr>
<th>Clock destination</th>
<th>Clock cycle</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Main clock oscillation stabilization delay time</td>
<td>2^15/FCH (approx. 1.95 ms)</td>
<td>Selected by the oscillator stabilization wait time select bit of the system clock control register (SYCC: WT1, WT0), which is in the clock control section.</td>
</tr>
<tr>
<td>Watchdog timer</td>
<td>2^16/FCH (approx. 7.80 ms)</td>
<td>Count-up clock for the watchdog timer</td>
</tr>
<tr>
<td>Buzzer output</td>
<td>2^18/FCH (approx. 62.4 ms)</td>
<td>See Chapter 10, &quot;Buzzer Output.&quot;</td>
</tr>
<tr>
<td>A/D converter</td>
<td>2^16/FCH (approx. 61.0 μs)</td>
<td>Clock for continuous activation</td>
</tr>
<tr>
<td>LCD controller/driver</td>
<td>2^16/FCH (approx. 30.5 μs)</td>
<td>Frame cycle clock</td>
</tr>
</tbody>
</table>

  FCH: Main clock source oscillation
  
  The values enclosed in parentheses ( ) are for a 4.2 MHz main clock source oscillation.

  **Note:** The oscillation stabilization delay time should be used as a guide line since the oscillation cycle is unstable immediately after oscillation starts.
5.2 Block Diagram of Timebase Timer

The timebase timer consists of the following four blocks:

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

![Block Diagram of Timebase Timer](image)

- **Timebase timer counter**
  A 21-bit up-counter that uses the divide-by-two main clock source oscillation as a count clock. The counter stops when the main clock oscillator is stopped.

- **Counter clear circuit**
  In addition to being cleared by setting the TBTC register (TBR="0"), the counter is cleared when device changes to main stop (STBC: STP ="1") and subclock (SYCC: SCS="0") mode and by power-on reset (optional).

- **Interval timer selector**
  Selects one of four operating timebase timer counter bits as the interval timer bit. An overflow on the selected bit triggers an interrupt.

- **TBTC register**
  The TBTC register is used to select the interval timer bit, clear the counter, control interrupts, and check the state of the timebase timer.
5.3 Timebase Timer Control Register (TBTC)

The timebase timer control register (TBTC) is used to select the interval times bit, clear the counter, control interrupts, and check the state of the timebase timer.

**Timebase Timer Control Register (TBTC)**

```
<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>000Ah</td>
<td>TBOF</td>
<td>TBIE</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>TBC1</td>
<td>TBC0</td>
<td>TBR</td>
<td>00XX000B</td>
</tr>
</tbody>
</table>

R/W: Readable and writable
W: Write-only
—: Unused
X: Indeterminate

**TBR**

Timebase timer initialization bit

<table>
<thead>
<tr>
<th>Read</th>
<th>Write</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>—</td>
</tr>
<tr>
<td>1</td>
<td>—</td>
</tr>
</tbody>
</table>

- 0: Clears timebase timer counter
- 1: Reading always returns "1." No effect. The bit does not change.

**TBC1 TBC0**

Interval time selection bits

| 0 | 0 | 2^(FCH) |
| 0 | 1 | 2^(FCH) |
| 1 | 0 | 2^(FCH) |
| 1 | 1 | 2^(FCH) |

FCH: Main clock source oscillation

**TBIE**

Interrupt request enable bit

| 0 | Enables interrupt output. |
| 1 | Enables interrupt output. |

**TBOF**

Overflow interrupt request flag bit

| 0 | No overflow on specified bit |
| 1 | Overflow on specified bit |

- 0: Clear this bit.
- 1: No effect. The bit does not change.
### Table 5.3a  Timebase Timer Control Register (TBTC) Bits

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7</td>
<td>TBOF: Overflow interrupt request flag bit</td>
</tr>
<tr>
<td></td>
<td>- This bit is set to “1” when counter overflow occurs on the specified bit of the timebase timer counter.</td>
</tr>
<tr>
<td></td>
<td>- An interrupt request is output when both this bit and the interrupt request enable bit (TBIE) are “1”.</td>
</tr>
<tr>
<td></td>
<td>- Writing “0” clears this bit. Writing “1” has no effect and does not change the bit value.</td>
</tr>
<tr>
<td>Bit 6</td>
<td>TBIE: interrupt request</td>
</tr>
<tr>
<td></td>
<td>- This bit enables or disables an interrupt request output to the CPU. An interrupt request is output when both this bit and the overflow interrupt request flag bit (TBOF) are “1”.</td>
</tr>
<tr>
<td>Bit 5</td>
<td>Unused bits</td>
</tr>
<tr>
<td>Bit 4</td>
<td>Unused bits</td>
</tr>
<tr>
<td>Bit 3</td>
<td>Unused bits</td>
</tr>
<tr>
<td>Bit 2</td>
<td>TBC1, TBC0: Interval time selection bits</td>
</tr>
<tr>
<td>Bit 1</td>
<td>- These bits select the cycle of the interval timer.</td>
</tr>
<tr>
<td></td>
<td>- These bits select which bit of the timebase timer counter to use as the interval timer bit.</td>
</tr>
<tr>
<td></td>
<td>- Four different interval times can be selected.</td>
</tr>
<tr>
<td>Bit 0</td>
<td>TBR: Timebase timer initialization bit</td>
</tr>
<tr>
<td></td>
<td>- This bit clears the timebase timer counter.</td>
</tr>
<tr>
<td></td>
<td>- Writing “0” to this bit clears the counter to “00000000H”. Writing “1” has no effect and does not change the bit value.</td>
</tr>
<tr>
<td></td>
<td>Note: The read value is always “1”.</td>
</tr>
</tbody>
</table>
5.4 Timebase Timer Interrupt

The timebase timer can generate an interrupt request when an overflow occurs on the specified bit of the timebase counter (for the interval timer function).

- **Interruption for Interval Timer Function**
  
  The counter counts-up on the internal count clock. When an overflow occurs on the selected interval timer bit, the overflow interrupt request flag bit (TBTC: TBOF) is set to “1”. At this time, an interrupt request (IRQ7) to the CPU is generated if the interrupt request enable bit is enabled (TBTC: TBIE = “1”). Write “0” to the TBOF bit in the interrupt processing routine to clear the interrupt request. The TBOF bit is set when at the specified counter bit overflows, regardless of the TBIE bit value.

  **Check:** When enabling an interrupt request output (TBIE = “1”) after wake-up from a reset, always clear the TBOF bit (TBOF = “0”) at the same time.

  **Notes:**
  - An interrupt request is generated immediately if the TBOF bit is “1” when the TBIE bit is changed from disabled to enabled (“0” → “1”).
  - The TBOF bit is not set if the counter is cleared (TBTC: TBR = “0”) at the same time as an overflow on the specified bit occurs.

- **Oscillation Stabilization Delay Time and Timebase Timer Interrupt**

  If the interval time is set shorter than the main clock oscillation stabilization delay time, an interval interrupt request from the timebase timer (TBTC: TBOF = “1”) is generated at the time when the main clock mode starts operation. In this case, disable the timebase timer interrupt when changing to a mode in which the main clock oscillation is stopped (main stop mode and subclock mode).

- **Register and Vector Table for Timebase Timer Interrupts**

  **Table 5.4a** Register and Vector Table for Timebase Timer Interrupt

<table>
<thead>
<tr>
<th>Interrupt</th>
<th>Interrupt level settings register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRQ7</td>
<td>ILR2 (007DH)</td>
<td>FFEC H</td>
</tr>
<tr>
<td></td>
<td>L71 (Bit 7)</td>
<td>FFED H</td>
</tr>
<tr>
<td></td>
<td>L70 (Bit 6)</td>
<td></td>
</tr>
</tbody>
</table>

  **Reference:** See Section 3.4.2, “Interrupt Processing” for details on the operation of interrupt.
5.5 Operation of Timebase Timer

The timebase timer has the interval timer function and the clock supply function for some peripherals.

- **Operation of Interval Timer Function (Timebase Timer)**

  Figure 5.5a shows the settings required to operate the interval timer function.

<table>
<thead>
<tr>
<th>TBTC</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>TBOF</td>
<td>TBIE</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>TBC1</td>
<td>TBC0</td>
<td>TBR</td>
<td>—</td>
</tr>
<tr>
<td>0</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0</td>
</tr>
</tbody>
</table>

  **Figure 5.5a Interval Timer Function Settings**

  Provided the main clock is oscillating, the timebase timer counter continues to count-up in sync with the internal count clock (divide-by-two main clock source oscillation).

  After being cleared (TBR = "0"), the counter restarts counting-up from zero. The timebase timer sets the overflow interrupt request flag bit (TBOF) to "1" when an overflow occurs on the interval timer bit. Consequently, the timebase timer generates interrupt requests at fixed intervals (the selected interval time), based on the time that the counter is cleared.

- **Operation of Clock Supply Function**

  The timebase timer is also used as a timer to generate the main clock oscillation stabilization delay time. The time from when the timebase timer counter is cleared and starts counting-up until an overflow occurs on the oscillation stabilization delay time bit is the oscillation stabilization delay time. One of four possible delay times is selected by the oscillation stabilization delay time bits of the system clock control register (SYCC: WT1, WT0).

  The timebase timer also provides the clock for the watchdog timer and buzzer output, A/D converter, and LCD controller/driver. Clearing the timebase timer counter affects the operation of the continuous activation cycle of the A/D converter, the frame cycle of the LCD controller/driver, and the buzzer output.

  If the timebase timer output is selected by the watchdog timer counter (WDTC:CS = 0), clearing the timebase timer counter by switching to main stop (STBC:STP = 1) of subclock (SYCC:SCS=0) mode will also clear the watchdog timer at the same time.
Operation of Timebase Timer

The state of following operations are shown in Figure 5.5b.

- A power-on reset occurs.
- Changes to sleep mode during operation of the interval timer function in the main clock mode.
- Changes to main stop mode.
- A counter clear request occurs.

The timebase timer is cleared by changing to subclock and main stop modes, and stops operation. The timebase timer counts the oscillation stabilization delay time after wake-up from subclock and main stop modes.

Counter value
1FFFFFFH

Oscillation stabilization delay overflow
0000H

Power-on reset (optional)

CPU operation start

Interval cycle
(TBTC: TBC1, TBC0 = “11”)

Cleared by the interrupt processing routine.

Cleared by changing to main stop mode.

Counter clear
(TBTC: TBR = “0”)

TBOF bit

TBIE bit

SLP bit
(STBC register)

STP bit
(STBC register)

Sleep mode

Wake-up from Sleep mode by IRQA

Stop mode

Wake-up from Stop mode by an external interrupt

For the case when the interval time selection bits in the timebase timer control register (TBTC: TBC1, TBC0) are set to “11” (2^21/Fc).

: Indicates the oscillation stabilization delay time.

Figure 5.5b Operation of Timebase Timer
5.6 Notes on Using Timebase Timer

This section lists points to note when using the timebase timer.

- **Notes on Using Timebase Timer**
  - **Notes on setting bits by program**
    The system cannot recover from interrupt processing if the overflow interrupt request flag bit (TBTC: TBOC) is “1” and the interrupt request enable bit is enabled (TBTC: TBIE = “1”). Always clear the TBOF bit.
  - **Clearing timebase timer**
    In addition to being cleared by the timebase timer initialization bit (TBTC: TBR = “0”), the timer is cleared whenever the main clock oscillation stabilization delay time is required. When the timebase timer is selected as a count clock of the watchdog timer, clearing the timebase timer also clears the watchdog timer.
  - **Using as timer for oscillation stabilization delay time**
    As the main clock source oscillation is stopped when the power is turned on during main-stop mode, and during subclock mode, the timebase timer provides the oscillation stabilization delay time after the oscillator starts.
    An appropriate oscillation stabilization delay time must be selected for the type of resonator connected to the main clock oscillator (clock generator).

  **Reference:** See Section 3.6.1, “Clock Generator Section”.

- **Notes on peripheral functions that provided a clock supply from timebase timer**
  In modes in which the main clock source oscillation is stopped, the timebase timer also stops, and the counter is cleared.
  As the clock derived from the timebase timer restarts output from its initial state when the timebase timer counter is cleared, the “H” level may be shorter or the “L” level longer by a maximum of half cycle. The clock of the watchdog timer also restarts output from its initial state.
  Figure 5.6a shows the effect on the buzzer output of clearing the timebase timer.

![Counter value diagram](image)

For the case when the buzzer selection bits in the buzzer register (BZCR: BZ1, BZ0) are set to “01”. (Divide-by-8192 source oscillation, approximately 1.221 kHz output at 10 MHz operation)

**Figure 5.6a Effect on Buzzer Output of Clearing Timebase Timer**
5.7 Program Example for Timebase Timer

This section gives a program example for the timebase timer.

- **Program Example for Timebase Timer**

  - **Processing description**
    - Generates repeated interval timer interrupts at $2^{18}/F_{CH}$ ($F_{CH}$: the main clock source oscillation) intervals. At this time, the interval time is approximately 52.4 ms (at 4.2 MHz operation).

  - **Coding example**

    ```assembly
    TBTC EQU 0000AH ; Address of the timebase timer control register
    TBOF EQU TBTC:3 ; Define the interrupt request flag bit.
    ILR2 EQU 007DH ; Address of the interrupt level setting register 2
    INT_V DSEG ABS ; [DATA SEGMENT]
    ORG 0FFECH
    IQ7 DW WARI ; Set interrupt vector.
    INT_V ENDS
    ;-----Main program------------------------------------------------------------------------------------------------------------- --------------------------
    CSEG ; [CODE SEGMENT]
    ; Stack pointer (SP) etc. are already initialized.
    CLR1 ; Disable interrupts.
    MOV ILR2,#01111111B ; Set interrupt level (level 1).
    MOV TBTC,#01000100B ; Clear interrupt request flag enable interrupt request output, select 2^{18}/FC, and clear timebase timer.
    SETI ; Enable interrupts.
    ;-----Interrupt program-------------------------------------------------------------------------------------------------------- --------------------------
    WARI CLRB TBOF ; Clear 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 operation of the watchdog timer.

6.1 Overview of Watchdog Timer ........................................... 126
6.2 Block Diagram of Watchdog Timer ................................. 128
6.3 Watchdog Timer Control Register (WDTC) ..................... 130
6.4 Operation of Watchdog Timer ......................................... 131
6.5 Notes on Using Watchdog Timer ..................................... 132
6.6 Program Example for Watchdog Timer ......................... 133
6.1 Overview of Watchdog Timer

The watchdog timer is a 1-bit counter that uses, as its count clock source, either the
timebase timer derived from the main clock, or the watch prescaler derived from the
subclock. The watchdog timer resets the CPU if not cleared within a fixed time after
activation.

- Watchdog Timer Function

The watchdog timer is a counter provided to guard against program runaway. Once activated,
the counter must be repeatedly cleared within a fixed time interval. If the program becomes
trapped in an endless loop or similar and does not clear the counter within the fixed time, the
watchdog timer generates a four-instruction cycle watchdog reset to the CPU.

Either the timebase timer output or the watch prescaler output can be selected as the watchdog
timer count clock.

Table 6.1a lists the watchdog timer interval times. If not cleared, the watchdog timer generates a
watchdog reset at a time between the minimum and maximum times listed. Clear the counter
within the minimum time given in the table.

Table 6.1a Watchdog Timer Interval Time

<table>
<thead>
<tr>
<th></th>
<th>Count clock</th>
</tr>
</thead>
<tbody>
<tr>
<td>Minimum time</td>
<td>Timebase timer output (main clock oscillator frequency at 4.2 MHz) Approx. 998.6 ms¹ 500 ms²</td>
</tr>
<tr>
<td>Maximum time</td>
<td>Approx. 1997.3 ms                                1000 ms</td>
</tr>
</tbody>
</table>

¹: Divide-by-two the main clock source oscillation (F_{CH}) x timebase timer count value (2^{21}).
²: The time of a clock cycle at the subclock oscillator frequency (F_{CL}) x watch prescaler count (2^{14}).

Reference: See Section 6.4, “Watchdog Timer Operation” for the details on the minimum and maximum
time of the watchdog timer interval times.

Note: The watchdog timer counter is cleared whenever the device changes to sleep or stop watch mode. Operation halts until the device returns to normal operation (RUN state).
### 6.2 Block Diagram of Watchdog Timer

The watchdog timer consists of the following six blocks:

- Count clock selector
- Watchdog timer counter
- Reset controller
- Watchdog timer clear selector
- Counter clear controller
- Watchdog timer control register (WDTC)

![Block Diagram of Watchdog Timer](image)

**Figure 6.2a Block Diagram of Watchdog Timer**
- **Count clock selector**
  The count clock selector selects the count clock for the watchdog timer counter. Either the timebase timer counter output or the watch prescaler output can be selected as the count clock.

- **Watchdog timer counter (1-bit counter)**
  A 1-bit counter that uses the timebase timer output or the watch prescaler output as a count clock.

- **Reset controller**
  Generates a reset signal to the CPU when an overflow occurs on the watchdog timer counter.

- **Watchdog timer clear selector**
  The clear selector selects a watchdog timer clear signal from either the timebase timer or watch prescaler at the same time as the count clock selector selects a clock. (It selects the clear signal from the selected clock source.)

- **Counter clear controller**
  Controls clearing and halting the operation of watchdog timer counter.

- **WDTC register**
  The WDTC register is used to select the count clock, and to activate or clear the watchdog timer counter. As the register is write-only, the bit manipulation instructions cannot be used.
6.3 Watchdog Timer Control Register (WDTC)

The watchdog timer control register (WDTC) is used to activate or clear the watchdog timer.

Watchdog Timer Control Register (WDTC)

Table 6.3a Watchdog Timer Control Register (WDTC) Bits

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>CS: Count clock select bit</th>
<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>W</td>
<td>WTE3 WTE2 WTE1 WTE0</td>
<td>0009H</td>
<td>0009H</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>0XXXXXXXXXa</td>
</tr>
</tbody>
</table>

- **CS**: Count clock select bit
  - At watchdog timer startup, selects the watchdog timer count clock. Selects either the timebase timer output or the watch prescaler output as the count clock.
  - **Note**: When using the subclock mode, always select the watch prescaler output. Make the count clock selection when the watchdog timer is started. Once the timer is started, do not change the count clock. Bit operation instructions cannot be used.
  - **Other than the above**: No operation
  - **W**: Write-only
  - **-**: Unused
  - **X**: Indeterminate
  - **a**: Initial value

<table>
<thead>
<tr>
<th>CS</th>
<th>Watchdog timer control bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Cycle time of timebase timer output ( \left( \frac{2^{22}}{F_{CH}} \right) )</td>
</tr>
<tr>
<td>1</td>
<td>Cycle time of watch prescaler output ( \left( \frac{2^{14}}{F_{CL}} \right) )</td>
</tr>
</tbody>
</table>

- **Note**: As the bit 3 ~ bit 0 are write-only, the bit manipulation instructions for these bits cannot be used.

- **R/W**: Readable and Writable
- **W**: Write-only
- **-**: Unused
- **X**: Indeterminate
- **: Initial value

Figure 6.3a Watchdog timer Control Register (WDTC)
6.4 Operation of Watchdog Timer

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

- **Operation of Watchdog Timer**
  - **Activating watchdog timer**
    - The watchdog timer is activated by writing “0101B” to the watchdog control bits in the watchdog control register (WDTC: WTE3 to WTE0) for the first time after a reset. The count clock select bit (WDTC: CS) is written to the desired state in the same write operation.
    - Once activated, the watchdog timer cannot be stopped other than by a reset.
  - **Clearing watchdog timer**
    - The watchdog timer counter is cleared by writing “0101B” to the watchdog control bits in the watchdog control register (WDTC: WTE3 to WTE0) for the second or subsequent times after a reset.
    - If the counter is not cleared within the interval time of the watchdog timer, the counter overflows and the watchdog timer generates an internal reset signal for four-instruction cycles.

- **Interval time of watchdog timer**

  The interval time changes depending on when the watchdog timer is cleared.

  Figure 6.4a shows the relationship between the watchdog timer clear timing and the interval time.

  The indicated times apply if the timebase timer output is selected as the count clock, and the main clock source oscillation is 4.2 MHz.

![Figure 6.4a Watchdog Timer Clear and Interval Time](image-url)
6.5 Notes on Using Watchdog Timer

This section lists points to note when using the watchdog timer.

- **Stopping watchdog timer**

  Once activated, the watchdog timer cannot stop until a reset generates.

- **Count clock selection**

  The count clock select bit (WDTC: CS) can only be changed during activating or clearing the watchdog timer. You can change it by writing the desired state to the count clock select bit (WDTC: CS) at the same time as you write “0101B” to the watchdog control bits (WDTC: WTE3 to WTE0) to activate or clear the watchdog timer. Therefore, the CS bit cannot be changed by a bit operation instruction. Do not change the CS bit after activating the timer.

  In the subclock mode, the main clock source oscillation is stopped, which means that the timebase timer also stops.

  For the watchdog timer to operate in subclock mode, then, the watch prescaler must have been selected in advance as the count clock (WDTC: CS = 1).

- **Clearing watchdog timer**

  - Clearing the counter being used as a count clock of the watchdog timer (timebase timer or watch prescaler) also simultaneously clears the watchdog timer counter.
  - The watchdog timer counter is cleared on changing to sleep, stop or watch mode.

- **Notes on programming**

  When writing a program in which the watchdog timer is repeatedly cleared in the main loop, including interrupt processing, is less than the minimum watchdog timer interval time.

- **Operation in subclock mode**

  If the watchdog reset signal is generated in subclock mode, operation will start in main clock mode after an oscillation stabilization delay time. Therefore, if the device has the reset signal output option, a reset signal will be output during the oscillation stabilization delay time.
6.6 Program Example for Watchdog Timer

This section gives a program example for the watchdog timer.

- **Program Example for Watchdog Timer**
  - **Processing description**
    - Selects the watch prescaler as the count clock and activates the watchdog timer immediately after the program.
    - Clears the watchdog timer in each loop of the main program.
    - The processing time for the main loop, including interrupt processing, must be less than the minimum interval time of the watchdog timer (approximately 209.7 ms at 10 MHz operation).
  - **Coding example**

    ```assembly
    WDTC EQU 00009H ; Address of the watchdog timer control register
    WDT_CLR EQU 10000101B

    VECT DSEG ABS ; [DATA SEGMENT]
    ORG 0FFFFH
    RST_V DW PROG ; Set reset vector.
    VECT ENDS

    ---Main program-------------------------------------------------------------------------------------------
    PROG ; [CODE SEGMENT]
    CSEG ; Initialization routine after a reset
    MOVW SP,#0280H ; Set initial value of (for interrupt processing).
    INIT MOV WDTC,#WDT_CLR ; Activate the watchdog timer.
    : ; Initialization of peripheral functions (interrupts), etc.
    : ; Select the watch prescaler as the count clock.
    MAIN MOV WDTC,#WDT_CLR ; Clear the watchdog timer.
    : User processing (interrupt processing may occur during this cycle)
    : ; The loop must be executed in less than the minimum interval time of the watchdog timer.
    JMP MAIN
    ENDS
    ---Main program-------------------------------------------------------------------------------------------
    END
    ```
This chapter describes the functions and operation of the 8-bit PWM timer.

7.1 Overview of 8-bit PWM Timer................................. 136
7.2 Block Diagram of 8-bit PWM Timer ......................... 138
7.3 Structure of 8-bit PWM Timer 1 ................................. 140
7.4 Structure of 8-bit PWM Timer 2 ................................. 146
7.5 8-bit PWM Timer Interrupts ................................. 151
7.6 Operation of Interval Timer Function .......................... 152
7.7 Operation of PWM Timer Function ........................... 153
7.8 States in Each Mode during 8-bit PWM Timer Operation 154
7.9 Notes on Using 8-bit PWM Timer .............................. 156
7.10 Program Example for 8-bit PWM Timer .................... 158
7.1 Overview of 8-bit PWM Timer

The 8-bit PWM timer can be selected to function as either an interval timer or PWM timer with 8-bit resolution. The interval timer function count-up in sync with either the 8/16-bit timer/counter (timer 1, timer 2) or one of three internal count clocks. Therefore, an 8-bit interval timer time can be set and the output can be used to generate variable frequency square waves. Also, the 8-bit PWM timer can be used as a D/A converter by connecting the PWM output to low pass filter.

There are two 8-bit PWM timer “channels,” that perform the same function: 8-Bit PWM Timer 1 and 8-Bit PWM Timer 2.

- Interval Timer Function (Square Wave Output Function)

  The interval timer function generates repeated interrupts at variable time intervals.
  Also, as the 8-bit PWM timer can invert the output level of the pin (PWM1, PWM2) each time an interrupt is generated, the 8-bit PWM timer can output a variable frequency square waves.
  Each 8-bit PWM timer (1 and 2) can operate independently of the other.
  - The interval timer can operate with a cycle among 1 and 2⁸ times the count clock cycle.
  - The count clock can be selected from four different clocks.

Table 7.1a lists the range for the interval time and square wave output.

<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 clock</td>
<td>1 tinst to 2⁴ tinst</td>
<td>1/(2 tinst) to 1/(2⁹ tinst)</td>
</tr>
<tr>
<td>16 tinst</td>
<td>2⁴ tinst to 2⁵ tinst</td>
<td>1/(2⁰ tinst) to 1/(2¹⁰ tinst)</td>
</tr>
<tr>
<td>64 tinst</td>
<td>2⁵ tinst to 2⁶ tinst</td>
<td>1/(2⁶ tinst) to 1/(2¹⁶ tinst)</td>
</tr>
<tr>
<td>8-bit timer output cycle</td>
<td>2⁶ tinst to 2⁷ tinst</td>
<td>1/(2⁷ tinst) to 1/(2¹⁷ tinst)</td>
</tr>
<tr>
<td>2⁷ tinst to 2⁸ tinst</td>
<td>1/(2⁸ tinst) to 1/(2¹⁸ tinst)</td>
<td></td>
</tr>
<tr>
<td>1 tinst to 2⁹ tinst</td>
<td>1/(2⁹ tinst) to 1/(2¹⁹ tinst)</td>
<td></td>
</tr>
<tr>
<td>2⁹ tinst to 2¹⁰ tinst</td>
<td>1/(2¹⁰ tinst) to 1/(2²⁰ tinst)</td>
<td></td>
</tr>
<tr>
<td>2¹⁰ tinst to 2¹¹ tinst</td>
<td>1/(2¹¹ tinst) to 1/(2²¹ tinst)</td>
<td></td>
</tr>
<tr>
<td>1 tinst to 2¹¹ tinst</td>
<td>1/(2¹¹ tinst) to 1/(2²¹ tinst)</td>
<td></td>
</tr>
</tbody>
</table>

Note: Calculation example for the interval time and square wave frequency

In this example, the main clock source oscillation (FCH) is 4.2 MHz, the PWM compare register (COMR) value is set to “DDH (221)”, and the count clock cycle is set to 1 tinst. In this case, the interval time and the frequency of the square wave output from the PWM1 or PWM2 pin (where the PWM timer operates continuously and the value of the COMR register is constant) are calculated as follows.

Assume that the main clock mode (SCS = 1) and its highest clock speed has been selected via the system clock control register (STCC: SCS = 1, CS = 11B, CS0 = 11B, 1 instruction cycle = 4/FCH).

Interval time = \((1 \times 4/FCH) \times (COMR\text{ register value} + 1)\)
= \((4/4.2\text{ MHz}) \times (221 + 1)\)
= 211 μs

Output frequency = FCH/(1 \times 8 \times (COMR\text{ register value} +1))
= 4.2 MHz/(8 \times (221 +1))
= 2.4 kHz
PWM Timer Function

The PWM timer function has 8-bit resolution and can control the “H” and “L” widths of one cycle. 8-Bit PWM Timers 1 and 2 can be operated independently of each other.

- As the resolution is 1/256, pulses can be output with duty ratios of between 0 and 99.6%.
- The cycle of the PWM wave can be selected from four types.
- The PWM timer can be used as a D/A converter by connecting the output to a low pass filter.

Table 7.1b lists the available PWM wave cycles for the PWM timer function. Figure 7.1a shows an example D/A converter configuration.

**Table 7.1b  Available PWM Wave Cycle for PWM Timer Function**

<table>
<thead>
<tr>
<th>Count clock cycle</th>
<th>1 tinst</th>
<th>16 tinst</th>
<th>64 tinst</th>
<th>256 tinst</th>
<th>1 tlext</th>
<th>16 tlext</th>
</tr>
</thead>
<tbody>
<tr>
<td>PWM wave cycle</td>
<td>2 tinst</td>
<td>2 tinst</td>
<td>2 tinst</td>
<td>4 tinst</td>
<td>2 tlext</td>
<td>4 tlext</td>
</tr>
</tbody>
</table>

**Note:**
Interrupt requests are not generated during operation of the PWM function.

Figure 7.1a  Example D/A Converter Configuration Using PWM Output and Low Pass Filter

The relationship between the analog output voltage and PWM output waveform is:

\[
\frac{V_a}{V_{CC}} = \frac{TH}{T} 
\]

\*Tr is the time taken for the output to stabilize.

Figure 7.1a  Example D/A Converter Configuration Using PWM Output and Low Pass Filter

Note: Interrupt requests are not generated during operation of the PWM function.
7.2 Block Diagram of 8-bit PWM Timer

The 8-bit PWM timer consists of the following six blocks:

- Count clock selector
- 8-bit counter
- Comparator circuit
- PWM generator and output controller
- PWM compare register (COMR)
- PWM control register (CNTR)

8-bit PWM Timer-1 and 8-bit PWM Timer-2 have the same functions.

---

Figure 7.2a Block Diagram of 8-bit PWM Timer

Note: The register and pin names listed first are for 8-bit PWM timer-1. The names enclosed in < > are for 8-bit PWM timer-2.

- Instruction cycle
- 8-bit timer output: Timer 1 or Timer 2 output (TO1 or TO2) of 8/16-bit timer/counter in 8-bit mode
● **Count clock selector**

Selects a count-up clock for the 8-bit counter from the three internal count clocks and the 8-bit timer output cycle of the 8/16-bit timer/counter.

● **8-bit counter**

The 8-bit counter counts-up on the count clock selected by the count clock selector.

● **Comparator circuit**

The comparator circuit has a latch to hold the COMR register value. The circuit latches the COMR register value when the 8-bit counter value is “00H”. The comparator circuit compares the 8-bit counter value with the latched COMR register value, and detects when a match occurs.

● **PWM generator and output controller**

When a match is detected during interval timer operation, an interrupt request is generated and, if the output pin control bit (CNTR: OE) is “1”, the output controller inverts the output level of the PWM1(PWM2) pin. At the same time, the 8-bit counter is cleared.

When a match is detected during PWM timer operation, the PWM generator changes the output level of the PWM1(PWM2) pin from “H” to “L”. The pin is set back to the “H” level when the next overflow occurs on the 8-bit counter.

● **COMR register**

The COMR register is used to set the value that is compared with the value of the 8-bit counter.

● **CNTR register**

The CNTR register is used to select the operating mode, enable or disable operation, set the count clock, control interrupts, and check the PWM status.

Setting the operation to PWM timer mode (P/TX = “0”) disables clearing of the 8-bit counter and generation of interrupt requests (IRQ9 or IRQA) when the comparator circuit detects a match.
7.3 Structure of 8-bit PWM Timer 1

This section describes the pin, pin block diagram, register source, and interrupts of the 8-bit PWM timer.

- **8-bit PWM Timer 1 Pin**

  The 8-bit PWM timer uses the P31/PWM 1 pin. This pin can function either as a output-only port (P31) or as the interval timer or PWM timer output (PWM1).

  PWM1: When the interval timer function is selected, the square waves are output to this pin.

  When the PWM timer function is selected, the pin outputs the PWM wave.

  Setting the output pin control bit (CNTR1: OE) to “1” makes Pin P31/PWM1 the output-only pin for 8-Bit PWM Timer 1. Once this has been done, the pin performs its PWM1 function regardless of the state of the port data register output latch data (PDR3: Bit 1).

- **Block Diagram of 8-bit PWM Timer Pin**

  ![Block Diagram of 8-bit PWM Timer Pin](image)

  SPL: Pin state specification bit in the standby control register (STBC)

  Figure 7.3a Block Diagram of 8-bit PWM Timer Pin
8-bit PWM Timer 1 Registers

**CNTR 1 (PWM1 control register)**

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>001ExH</td>
<td>P/TX</td>
<td>—</td>
<td>P1</td>
<td>P0</td>
<td>TPE</td>
<td>TIR</td>
<td>OE</td>
<td>TIE</td>
<td>0X0000000e</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>

**COMR 1 (PWM1 compare register)**

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>001Fh</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>XXXXXXXXXs</td>
</tr>
</tbody>
</table>

R/W: Readable and writable
W: Write-only
—: Unused
X: Indeterminate

**Figure 7.3b 8-bit PWM1 Timer Registers**

Check: As the PWM1 compare register (COMR1) is write-only, the bit manipulation instructions can be used.

8-bit PWM Timer 1 Interrupt Source

IRQ9: For the interval timer function, the 8-bit PWM timer generates an interrupt request if interrupt request output is enabled (CNTR: TIE = “1”) when the counter value matches the value set in the COMR register. (No interrupt requests are generated when the PWM function is operating.)
7.3 Structure of 8-bit PWM Timer 1

7.3.1 PWM1 Control Register (CNTR1)

The PWM1 control register (CNTR1) is used to select the operating mode of the 8-bit PWM timer (interval timer operation or PWM timer operation), enable or disable operation, select the count clock, control interrupts, and check the state of the 8-bit PWM timer.

**PWM1 Control Register (CNTR1)**

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>001EH</td>
<td>P/TX</td>
<td>—</td>
<td>P1</td>
<td>P0</td>
<td>TPE</td>
<td>TIR</td>
<td>OE</td>
<td>TIE</td>
<td>0X00000000</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>

- **TIE**: Interrupt request enable bit
  - 0: Disables interrupt request output.
  - 1: Enables interrupt request output.

- **OE**: Output pin control bit
  - 0: Functions as a general-purpose port (P31).
  - 1: Functions as the interval timer/PWM timer output pin (PWM1).

- **TIR**: Interrupt request flag bit
  - 0: Counter value and set value do not match. No change
  - 1: Counter value and set value match. No effect. The bit does not change.

- **P1 P0**: Clock selection bits
  - 0 0: Internal
  - 1 0: 16 tinst
  - 1 1: 64 tinst
  - 1 1: Timer 1 output

- **P/TX**: Operating mode selection bit
  - 0: Operates as an interval timer.
  - 1: Operates as a PWM timer.

---

*1: tinst: Instruction cycle
*2: The “Timer 1 output” referred to here is the timer 1 output cycle when the 8/16-bit timer/counter is used in the 8-bit mode
R/W: Readable and writable
—: Unused
X: Indeterminate
0X: Initial value

Figure 7.3.1a PWM1 Control Register (CNTR 1)
Table 7.3.1a PWM 1 Control Register (CNTR1) bit

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7</td>
<td><strong>P/TX: Operating mode selection bit</strong>&lt;br&gt;- This bit switches between the interval timer function (P/TX = “0”) and PWM timer function (P/TX = “1”).&lt;br&gt;- <strong>Check:</strong> Write to this bit when the counter operation is stopped (TPE = “0”), interrupts are disabled (TIE = “0”), and the interrupt request flag bit is cleared (TIR = “0”).</td>
</tr>
<tr>
<td>Bit 6</td>
<td><strong>Unused bit</strong>&lt;br&gt;- The read value is indeterminate.&lt;br&gt;- Writing to this bit has no effect on the operation.</td>
</tr>
<tr>
<td>Bit 5</td>
<td><strong>P1, P0: Clock selection bit</strong>&lt;br&gt;- These bits select the count clock for the interval timer function and PWM timer function.&lt;br&gt;- These bits can select the count clock from three internal count clocks or the output cycle of the timer 1.&lt;br&gt;- If the timer 1 output is selected, operate the 8/16-bit timer/counter in 8-bit mode.&lt;br&gt;- <strong>Check:</strong> Do not change P1 and P0 when the counter is operating (TPE = “1”).</td>
</tr>
<tr>
<td>Bit 4</td>
<td><strong>TPE: Counter operation enable bit</strong>&lt;br&gt;- This bit activates or stops operation of the PWM timer function and interval timer function.&lt;br&gt;- Writing “1” to this bit starts the count operation. Writing “0” to this bit stops the count and clears the counter to “00H”.</td>
</tr>
<tr>
<td>Bit 3</td>
<td><strong>TIR: Interrupt request flag bit</strong>&lt;br&gt;- For the interval timer function: This bit is set to “1” when the counter and PWM 1 compare register (COMR1) values match.&lt;br&gt;- An interrupt request is output to the CPU when both this bit and the interrupt request enable bit (TIE) are “1”.&lt;br&gt;- For the PWM timer function: Interrupt requests are not generated.&lt;br&gt;- Writing “0” clears this bit. Writing “1” has no effect and does not change the bit value.</td>
</tr>
<tr>
<td>Bit 2</td>
<td><strong>OE: Output pin control bit</strong>&lt;br&gt;- The P31/PWM1 pin functions as a general-purpose port (P31) when this bit is set to “0”, and a dedicated pin (PWM1) when this bit is set to “1”.&lt;br&gt;- The PWM1 pin outputs a square wave when the interval timer function is selected and a PWM waveform when the PWM timer function is selected.</td>
</tr>
<tr>
<td>Bit 1</td>
<td><strong>TIE: Interrupt request enable bit</strong>&lt;br&gt;- This bit enables or disables an interrupt request output to the CPU. An interrupt request is output when both this bit and the interrupt request flag bit (TIR) are “1”.</td>
</tr>
</tbody>
</table>
7.3 Structure of 8-bit PWM Timer 1

7.3.2 PWM 1 Compare Register (COMR1)

The PWM 1 compare register (COMR1) sets the interval time for the interval timer function. The register value sets the “H” width of the pulse for the PWM timer function.

| PWM 1 Compare Register (COMR1) |

Figure 7.3.2a shows the bit structure of the PWM1 compare register.

As the register is write-only, bit manipulation instructions cannot be used.

Figure 7.3.2a  PWM 1 Compare Register (COMR1)

- **Interval timer operation**

This register is used to set the value to be compared with the counter value. The register specifies the interval time.

The counter is cleared when the counter value matches the value set in this register, and the interrupt request flag bit is set to “1” (CNTR 1: TIR = “1”).

If data is written to the COMR1 register during counter operation, the new value applies from the next cycle (after the next match is detected).

**Note:** The COMR1 setting for interval timer operation can be calculated using the following formula.

The instruction cycle time is affected by the clock mode, and the speed-shift selection. COMR 1 register value = interval time/(count clock cycle × instruction cycle) – 1

- **PWM timer operation**

This register is used to set the value to be compared with the counter value. The register therefore sets the “H” width of the pulse.

The PWM1 pin outputs an “H” level until the counter value matches the value set in this register. From the match until the counter value overflows, the PWM1 pin outputs an “L” level.

If data is written to the COMR 1 register during counter operation, the new value applies from the next cycle (after the next overflow).

**Note:** In PWM timer operation, the COMR1 setting and the PWM cycle time can be calculated using the following formulas. (The instruction cycle time is affected by the clock mode, and the speed-shift selection.)

COMR1 register value = duty ratio (%) × 256

PWM wave cycle = count clock cycle × instruction cycle × 256
Memo
7.4 Structure of 8-bit PWM Timer 2

This section describes the pin, pin block diagram, register source, and interrupts of the 8-bit PWM timer.

- 8-bit PWM Timer 2 Pin

The 8-bit PWM timer 2 uses the P27/PWM2 pin. This pin can function either as a general-purpose I/O port (P27) or as the interval timer or PWM timer output (N-ch open-drain output)(PWM2).

PWM2: When the interval timer function is selected, the square waves are output to this pin.

When the PWM timer function is selected, the pin outputs the PWM wave.

Setting the P27/PWM2 pin as a dedicated pin in the output pin control bit (CNTR 2: OE = “1”) automatically sets the pin as an output pin, regardless of the port data direction register (DDR2: bit 7) value, and sets the pin to function as the PWM2 pin.

- Block Diagram of 8-bit PWM Timer 2 Pin

![Block Diagram of 8-bit PWM Timer 2 Pin](image)

**Figure 7.4a Block Diagram of 8-bit PWM Timer 2 Pin**

**Note:** Pins with a pull-up resistor (optional) go to the “H” level during a reset or in stop and watch modes.
### 8-bit PWM Timer 2 Registers

<table>
<thead>
<tr>
<th>CNTR 2 (PWM2 control register)</th>
<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0020h</td>
<td>P/TX</td>
<td>—</td>
<td>P1</td>
<td>P0</td>
<td>TPE</td>
<td>TIR</td>
<td>OE</td>
<td>TIE</td>
<td>0X00000000</td>
</tr>
<tr>
<td></td>
<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>

<table>
<thead>
<tr>
<th>CONTR2 (PWM2 compare register)</th>
<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>0021h</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>XXXXXX0X00</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>

R/W: Readable and writable  
W: Write-only  
—: Unused  
X: Indeterminate

**Figure 7.4b 8-bit PWM Timer 2 Registers**

Check: As the PWM2 compare register (COMR2) is write-only, the bit manipulation instructions cannot be used.

### 8-bit PWM Timer 2 Interrupt Source

IRQA: For the interval timer function, the 8-bit PWM timer generates an interrupt request if interrupt request output is enabled (CNTR 2: TIE = “1”) when the counter value matches the value set in the COMR2 register. (No interrupt requests are generated when the PWM function is operating.)
7.4 Structure of 8-bit PWM Timer 2

7.4.1 PWM 2 Control Register (CNTR2)

The PWM 2 control register (CNTR2) is used to select the operating mode of the 8-bit PWM timer (interval timer operation or PWM timer operation), enable or disable operation, select the count clock, control interrupts, and check the state of the 8-bit PWM timer.

- **PWM 2 Control Register (CNTR2)**

![Figure 7.4.1a PWM 2 Control Register (CNTR2)](image)

- **Address** 0020H
- **Bit 7** 00 P/TX — P1 P0 TPE TIR OE TIE
- **Initial value** 0X00000000
- **R/W** R/W R/W R/W R/W R/W R/W

- **TIE** Interrupt request enable bit
  - 0: Disables interrupt request output.
  - 1: Enables interrupt request output.

- **OE** Output pin control bit
  - 0: Function as a general-purpose port (P27).
  - 1: Function as the interval timer/PWM timer output pin (PWM2).

- **TIR** Interrupt request flag bit
  - Read: Interval timer function
    - 0: Counter value and set value do not match. No change
    - 1: Counter value and set value match.
  - Write: PWM timer function
    - 0: Counter value and set value do not match. Cleared this bit.
    - 1: Counter value and set value match. No effect. The bit does not change.

- **TPE** Counter operation enable bit
  - 0: Stops count operation.
  - 1: Starts count operation.

- **P1 P0** Clock selection bits
  - 00: Internal count clock
  - 01: 16 timer output
  - 10: 64 timer output
  - 11: Timer 2 output

- **P/TX** Operating mode selection bit
  - 0: Operates as an interval timer.
  - 1: Operates as a PWM timer.

*1: instr : Instruction cycle
*2: The "Timer 2 output" referred to here is the timer 1 output cycle when the 8/16-bit timer/counter is used in the 8-bit mode

R/W : Readable and writable
— : Unused
X : Indeterminate
- : Initial value
Table 7.4.1a PWM 2 Control Register (CNTR2) Bits

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7</td>
<td>P/TX: Operating mode selection bit</td>
</tr>
<tr>
<td>Bit 6</td>
<td>Unused bit</td>
</tr>
<tr>
<td>Bit 5</td>
<td>P1, P0: Clock selection bits</td>
</tr>
<tr>
<td>Bit 4</td>
<td>TPE: Counter operation enable bit</td>
</tr>
<tr>
<td>Bit 3</td>
<td>TIR: Interrupt request flag bit</td>
</tr>
<tr>
<td>Bit 2</td>
<td>OE: Output pin control bit</td>
</tr>
<tr>
<td>Bit 1</td>
<td>TIE: Interrupt request enable bit</td>
</tr>
</tbody>
</table>

**Bit 7**
- **P/TX: Operating mode selection bit**
  - This bit switches between the interval timer function (P/TX = '0') and PWM timer function (P/TX = '1').
  - **Check:** Write to this bit when the counter operation is stopped (TPE = '0'), interrupts are disabled (TIE = '0'), and the interrupt request flag bit is cleared (TIR = '0').

**Bit 6**
- **Unused bit**
  - The read value is indeterminate.
  - Writing to this bit has no effect on the operation.

**Bit 5**
- **P1, P0: Clock selection bits**
  - These bits select the count clock for the interval timer function and PWM timer function.
  - Selects either one of three internal count clock, or the Timer 2 output cycle.
  - When you selects the Timer 2 output, operate the 8/16-bit timer/counter in its 8-bit mode.
  - **Check:** Do not change P1 and P0 when the counter is operating (TPE = '1').

**Bit 4**
- **TPE: Counter operation enable bit**
  - This bit activates or stops operation of the PWM timer function and interval timer function.
  - Writing '1' to this bit starts the count operation. Writing '0' to this bit stops the count and clears the counter to '000H'.

**Bit 3**
- **TIR: Interrupt request flag bit**
  - For the interval timer function:
    - This bit is set to '1' when the counter and PWM 2 compare register (CONTR2) values match.
    - An interrupt request is output to the CPU when both this bit and the interrupt request enable bit (TIE) are '1'.
  - For the PWM timer function:
    - Interrupt requests are not generated.
    - Writing '0' clears this bit. Writing '1' has no effect and does not change the bit value.

**Bit 2**
- **OE: Output pin control bit**
  - The P27/PWM2 pin functions as a general-purpose port (P27) when this bit is set to '0', and a dedicated pin (PWM2) when this bit is set to '1'.
  - The PWM2 pin outputs a square wave when the interval timer function is selected and a PWM waveform when the PWM timer function is selected.

**Bit 1**
- **TIE: Interrupt request enable bit**
  - This bit enables or disables an interrupt request output to the CPU. An interrupt request is output when both this bit and the interrupt request flag bit (TIR) are '1'.
7.4 Structure of 8-bit PWM Timer 2

7.4.2 PWM 2 Compare Register (COMR2)

The PWM 2 compare register (COMR2) sets the interval time for the interval timer function. The register value sets the “H” width of the pulse for the PWM timer function.

**PWM 2 Compare Register (COMR2)**

Figure 7.4.2a shows the bit structure of the PWM 2 compare register.

As the register is write-only, bit manipulation instructions cannot be used.

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0021H</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>XXXXXXXXa</td>
</tr>
</tbody>
</table>

* W: Write-only
* X: Indeterminate

**Figure 7.4.2a PWM2 Compare Register (CONTR2)**

- **Interval timer operation**

  This register is used to set the value to be compared with the counter value. The register specifies the interval time.
  
  The counter is cleared when the counter value matches the value set in this register, and the interrupt request flag bit is set to “1” (CNTR2: TIR = “1”).
  
  If data is written to the COMR2 register during counter operation, the new value applies from the next cycle (after the next match is detected).
  
  **Note:** The COMR2 setting for interval timer operation can be calculated using the following formula.
  
  \[
  \text{COMR2 register value} = \frac{\text{interval time}}{\text{(count clock cycle} \times \text{instruction cycle})} - 1
  \]

- **PWM timer operation**

  This register is used to set the value to be compared with the counter value. The register therefore sets the “H” width of the pulse.
  
  The PWM2 pin outputs an “H” level until the counter value matches the value set in this register. From the match until the counter value overflows, the PWM2 pin outputs an “L” level.
  
  If data is written to the COMR2 register during counter operation, the new value applies from the next cycle (after the next overflow).
  
  **Note:** In PWM timer operation, the COMR2 setting and the PWM cycle time can be calculated using the following formulas. (The instruction cycle time is affected by the clock mode, and the speed-shift selection.)
  
  \[
  \text{COMR2 register value} = \text{duty ratio} \times 256
  \]
  
  \[
  \text{PWM wave cycle} = \text{count clock cycle} \times \text{instruction cycle} \times 256
  \]
## 7.5 8-bit PWM Timer Interrupts

The 8-bit PWM timer can generate an interrupt request when a match is detected between the counter value and PWM compare register value for the interval timer function. Interrupt requests are not generated for the PWM timer function.

8-bit PWM timer-1 generates the IRQ9 as an interrupt request and 8-bit PWM timer-2 generates the IRQA as an interrupt request.

### Intervals for Interval Timer Function

- The counter value is counted-up from “00H” on the selected count clock. When the counter value matches the PWM compare register (COMR) value, the interrupt request flag bit (CNTR: TIR) is set to “1”.

- At this time, an interrupt request to the CPU is generated if the interrupt request enable bit is enabled (CNTR: TIE = “1”). Write “0” to 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.

- **Note**: The TIR bit is not set if the counter is stopped (CNTR: TPE = “0”) at the same time as the counter value matches the COMR register value.

- An interrupt request is generated immediately if the TIR bit is “1” when the TIE bit is changed from disabled to enabled (“0” → “1”).

### Registers and Vector Tables for 8-bit PWM Timer Interrupts

<table>
<thead>
<tr>
<th>Interrupt</th>
<th>Interrupt level setting register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td>8-bit PWM Timer 1</td>
<td>IRQ9</td>
<td>ILR3 (007EH) L91 (Bit3) L90 (Bit2)</td>
</tr>
<tr>
<td>8-bit PWM Timer 2</td>
<td>IRQA</td>
<td>ILR3 (007EH) LA1 (Bit5) LA0 (Bit4)</td>
</tr>
</tbody>
</table>

**Reference**: See Section 3.4.2, “Interrupt Processing” for details on the interrupt operation.
7.6 Operation of Interval Timer Function

This section describes the operation of the timer interval timer function of the 8-bit PWM timer.

### Operation of Interval Timer Function

Figure 7.6a shows the settings required to operate as an interval timer function.

![Figure 7.6a Interval Timer Function Settings](image)

On activation, the counter starts counting-up from "00H" on the rising edge of the selected count clock. When the counter value matches the value set in the COMR register (compare value), the PWM timer inverts the level of the output pin (PWM1, PWM2) on the next rising edge of the count clock, clears the counter, sets the interrupt request flag bit (CNTR: TIR = "1"), and restarts counting from "00H".

Figure 7.6b shows the operation of the 8-bit PWM timer.

![Figure 7.6b Operation of 8-bit PWM Timer](image)

**Check:** Do not change the count clock cycle (CNTR: P1, P0) during operation of the interval timer function (CNTR: TPE = "1").

**Notes:**
- Setting the COMR register value to "00H" causes the PWM pin output to be inverted with the cycle of the selected count clock.
- When the counter is stopped (CNTR: TPE = "0") while the interval timer function is selected, the PWM pin outputs an "L" level.
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 PWM Timer Function

Figure 7.7a shows the settings required to operate as the PWM timer function.

<table>
<thead>
<tr>
<th>CNTR</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></td>
<td>P/TX</td>
<td>—</td>
<td>P1</td>
<td>P0</td>
<td>TPE</td>
<td>TIR</td>
<td>OE</td>
<td>TIE</td>
</tr>
<tr>
<td>1</td>
<td>Q</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>x</td>
<td>1</td>
<td>x</td>
<td></td>
</tr>
</tbody>
</table>

Figure 7.7a PWM Timer Function Settings

On activation, the counter starts counting-up from “00H” on the rising edge of the selected count clock. The PWM pin (PWM1, PWM2) outputs (PWM waveform) an “H” level until the counter value matches the value set in the COMR register. From the match until the counter value overflows (FFH → 00H), the PWM pin outputs an “L” level.

Figure 7.7b shows the PWM waveforms output from the PTO pin.

For COMR register value of “00H” (duty ratio = 0%)

Counter value: 00H → FFH
PWM waveform: H → L

For COMR register value of “80H” (duty ratio = 50%)

Counter value: 00H → 80H
PWM waveform: H → L

For COMR register value of “FFH” (duty ratio = 99.6%)

Counter value: 00H → FFH
PWM waveform: H → L

Figure 7.7b Example of PWM Waveform Output (PTO Pin)

Check: Do not change the count clock cycle (CNTR: P1, P0) during operation of the PWM timer function (CNTR: TPE = “1”).

Note: When the PWM timer function is selected, the PWM pin maintains its existing level when the counter is stopped (CNTR: TPE = “0”).
7.8 States in Each Mode during 8-bit PWM Timer Operation

This section describes the operation of the 8-bit PWM timer when the device changes to sleep or stop mode or an operation halt request occurs during operation.

- Operation during Standby Mode or Operation Halt

Figures 7.8a and 7.8b show the counter value states when the device changes to sleep or stop mode, or an operation halt request occurs, during operation of the interval timer function or PWM timer function.

The counter halts and maintains its current value when the device changes to stop mode. Operation starts again from the stored counter value after wake-up from stop mode by an external interrupt. Therefore, the first interval time or PWM wave cycle does not match the set value. Always initialize the 8-bit PWM timer after wake-up from stop mode.

- For interval timer function

![Figure 7.8a Counter Operation during Standby Modes or Operation Halt (For Interval Timer Function)](image)

* : The PWM pin (PWM1, PWM2) goes to the high-impedance state during stop mode if the pin state specification bit in the standby control register (STBC: SPL) is “1” and the PWM pin is not set to with a pull-up resistor (unless the pin is pulled up, a pull-up option selectable for the PWM2 pin only).
When the SPL bit is “0”, the pin maintains its value prior to changing to stop mode.
For PWM timer function

* The PWM pin (PWM1, PWM2) goes to the high-impedance state during stop mode if the pin state specification bit in the standby control register (STBC: SPL) is “1” and the PWM pin is not set to with a pull-up resistor (unless the pin is pulled, a pull-up option selectable for the PWM2 pin only). When the SPL bit is “0”, the pin maintains its value prior to changing to stop mode.

Figure 7.8b Operation during Standby Modes or Operation Halt (For PWM Timer Function)
7.9 Notes on Using 8-bit PWM Timer

This section lists points to note when using the 8-bit PWM timer.

- **Notes on Using 8-bit PWM Timer**
  - **Error**
    Activating the counter by program is not synchronized with the start of counting-up using the selected count clock. Therefore, the time from activating the counter until a match with the PWM compare register (COMR) is detected may be shorter than the theoretical time by a maximum of one cycle of the count clock. Figure 7.9a shows the error that occurs on starting counter operation.

  ![Figure 7.9a Error on Starting Counter Operation](image)

- **Notes on setting by program**
  - Do not change the count clock cycle (CNTR: P1, P0) when the interval timer function or PWM timer function is operating (CNTR: TPE = "1").
  - Stop the counter (CNTR: TPE = "0"), disable interrupts (TIE = "0"), and clear the interrupt request flag (TIR = "0") before switching between the interval timer function and PWM timer function (CNTR: P/TX).
  - Interrupt processing cannot return if the interrupt request flag bit (CNTR: TIR) is “1” and the interrupt request enable bit is enabled (CNTR: TIE = "1"). Always clear the TIR bit.
  - The TIR bit is not set if the counter is disabled (TPE = "0") at the same time as the counter and COMR register values match.
7.10 Program Example for 8-bit PWM Timer

This section gives program examples for the 8-bit PWM timer.

Program Example for Interval Timer Function

- **Processing description**
  - Generates repeated interval timer interrupts at 5 ms intervals.
  - Outputs a square wave to the PWM1 pin that inverts after each interval time.
  - With a main clock master oscillation \( F_{CH} \) of 4.2 MHz, and the highest speed clock selected by the speed-shift function (1 instruction cycle time = \( 4/F_{CH} \)), the COMR register is set for an interval time of approximately 5 ms. (An internal clock period of 64 \( t_{inst} \) is selected as the count clock.) The COMR register setting is calculated as follows:

\[
\text{COMR register value} = \frac{5 \text{ ms}}{64 \cdot \frac{4}{4.2 \text{ MHz}}} - 1 = 81.0 \text{ (051H)}
\]

- **Coding example**

```
CNTR1 EQU 001EH ; Address of the PWM 1 control register
COMR1 EQU 001FH ; Address of the PWM 1 compare register
TPE EQU CNTR1:3 ; Define the counter operation enable bit.
TIR EQU CNTR1:2 ; Define the interrupt request flag bit.
ILR3 EQU 007EH ; Address of the interrupt level setting register 3
INT_V DSEG ABS ; [DATA SEGMENT]
ORG 0FFE8H
IRQ9 DW WARI ; Set interrupt vector.
INT_V ENDS

;-----Main program------------------------------------------------------------------------------------------------------------- --------------------------
CSEG ; [CODE SEGMENT]
; Stack pointer (SP) etc. are already initialized.
CLRI ; Disable interrupts.
CLRB TPE ; Stop counter operation.
MOV ILR3,#11110111B ; Set interrupt level (level 1).
MOV COMR1,#051H ; Value compared with the counter value (interval time)
MOV CNTR1,#00101011B ; Operate interval timer, select 64 \( t_{inst} \),
; start counter operation, clear interrupt request flag, enable PTO pin output, enable interrupt request output.
SETI ; Enable interrupts.

;-----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
INT_V ENDS

END
```
Program Example for PWM Timer Function

Processing description

• Generates a PWM wave with a duty ratio of 50%. Then, changes the duty ratio to 25%.
• Does not generate interrupts.
• For a 4.2 MHz source oscillation, \((F_{CH})\), and the highest speed clock selected by the speed-shift function \(1\) instruction cycle time = \(4/F_{CH}\), selecting the interval \(16\) \(t_{inst}\) count clock gives a PWM wave cycle of \(16 \times 4/4.2 \text{ MHz} \times 256 \approx 3.901\) ms.
• The following shows the COMR register value required for a duty ratio of 50%:

\[
\text{COMR register value} = \frac{50}{100} \times 256 = 128 \text{ (080 H)}
\]

Coding example

```assembly
CSEG ; [CODE SEGMENT] 

CLRB TPE ; Stop counter operation.
MOV COMR1,#80H ; Set "H" width of pulse. Duty ratio = 50%
MOV CNTR1,#10011010B ; Operate PWM timer, select 16 \(t_{inst}\), start counter operation, clear interrupt request flag, enable PTO pin output, and disable interrupt request output.

MOV COMR1,#40H ; Change the duty ratio to 25% (effective from the next PWM wave cycle).

ENDS 

END 
```
This chapter describes the functions and operation of the 8/16-bit Timer/Counter.
8.1 Overview of 8/16-bit Timer/Counter

The 8/16-bit timer/counter is made up of two 8-bit timers (Timer 1 and Timer 2) that can be used separately (8-bit mode) or connected in cascade to form one counter (16-bit mode).

Timer 1 can be selected to function as either an interval timer or a counter. The interval timer function counts up in sync with one of three interval count clocks. The counter function counts up by a clock input to the external pin. The output can be used to generate variable frequency square wave output.

Timer 2 functions as an interval timer clocked by one of three internal count clocks. In the 16-bit mode, it is connected in series with Timer 1.

- Interval Timer Function

The interval timer function generates repeated interrupts at variable intervals. Also, as the 8/16-bit timer/counter can invert the output level of the pin (TO pin) each time an interval time is generated, the 8/16-bit timer/counter can output variable frequency square waves (Timer 1 in 8-bit mode, or 16-bit mode).

- In 8-bit mode, Timer 1 and Timer 2 operate as two independent interval timers, each of which can count time intervals ranging from the clock period (the time of one clock cycle) to \(2^n\) times the clock period.

- In 16-bit mode, the two counters form a single 16-bit timer, with Timer 1 containing the LSBs and Timer 2 the MSBs. The interval timer can operate with a cycle among 1 and \(2^16\) times the internal count clock cycle.

- The count clock can be selected from three different internal clocks. (An external clock can be selected for Timer 1, but it will then function as a counter).

- Timer 1 and Timer 2 outputs can be used as the count clocks for 8-Bit PWM Timer 1 and 8-Bit PWM Timer 2, respectively.

- The Timer 1 output can be used as the clock for the A/D converter in continuous operation mode.

Tables 8.1a to 8.1c list the interval time and square wave output ranges for the various modes.

### Table 8.1a Timer 1 Interval Times and Square Wave Frequencies 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 tinst</td>
<td>2 tinst to 2^2 tinst</td>
</tr>
<tr>
<td></td>
<td>32 tinst</td>
<td>2^5 tinst to 2^13 tinst</td>
</tr>
<tr>
<td></td>
<td>512 tinst</td>
<td>2^9 tinst to 2^17 tinst</td>
</tr>
<tr>
<td>External clock</td>
<td>1 ttext</td>
<td>1 ttext to 2 ttext</td>
</tr>
</tbody>
</table>

### Table 8.1b Timer 2 Interval Times and Square Wave Frequencies 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 tinst</td>
<td>2 tinst to 2^2 tinst</td>
</tr>
<tr>
<td></td>
<td>32 tinst</td>
<td>2^5 tinst to 2^13 tinst</td>
</tr>
<tr>
<td></td>
<td>512 tinst</td>
<td>2^9 tinst to 2^17 tinst</td>
</tr>
</tbody>
</table>
**Table 8.1c  Interval Times and Square Wave Frequencies 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 tinst to 2^17 tinst</td>
<td>1/(2^17 tinst) to 1/(2^18 tinst)</td>
</tr>
<tr>
<td>External clock</td>
<td>1 text to 2^11 text</td>
<td>1/(2 text) to 1/(2^12 text)</td>
</tr>
</tbody>
</table>

**Note:** Calculation example for the interval time and square wave frequency:

In this example, the main clock source oscillation (FCH) is 4.2 MHz, the timer 1 data register (T1DR) value is set to "DDH(221)", and the count clock cycle is set to the 8-bit mode operation at 2 tinst. In this case, the timer 1 interval time and frequency of square wave output from the TO pin (where the PWM timer operates continuously and the value of the T1DR register is constant) are calculated as follows.

Assume that the main clock mode (SCS = 1) and the highest clock speed (CS1/CS0 = 11s) has been selected via the system clock control register (SYCC: SCS = 1, CS1 = 11s, CS0 = 11s) (1 instruction cycle = 4/FCH).

**Interval time**

\[
\text{Interval time} = (2 \times 4/FCH) \times (\text{T1DR register value} + 1)
\]
\[
= (8/4.2 \text{ MHz}) \times (221 + 1)
\]
\[
\approx 422.9 \mu\text{s}
\]

**Output frequency**

\[
\text{Output frequency} = FCH/(2 \times 8 \times (\text{T1DR register value} + 1))
\]
\[
= 4.2 \text{ MHz} / (16 \times (221 + 1))
\]
\[
\approx 1.18 \text{ kHz}
\]

### Counter Function

The counter function counts rising edges of an external count clock applied to the external pin (EC pin). Since the external clock can be selected only for Timer 1, the counter function operates in either the 8-bit Timer 1 or 16 bit mode.

- The counter counts up, clocked by external clocks. When the count equals the set value, it generates an interrupt request and inverts the level being output at the TO pin.

- In the 8 bit mode, Timer 1 can count as high as 2^8.

- In the 16 bit mode, the function counts as high as 2^16.

- By injecting an external clock having a set period, the counter function can be used the same way as the interval timer function.
8.2 Block Diagram of 8/16-bit Timer/Counter

The 8/16-bit timer/counter consists of the following five blocks:
- Count clock selectors 1 and 2
- Counter circuits 1 and 2
- Square wave output controller
- Timer data registers (T1DR and T2DR)
- Timer control registers (T1CR and T2CR)

![Block diagram of 8/16-bit Timer/Counter](image)

Figure 8.2a Block Diagram 8/16-bit Timer/Counter
This circuit selects an input clock. In the 8-bit timer 1 and 16 bit modes, count clock selector 1 selects one of four clocks: three internal clocks, and an external clock. In the 8-bit mode, count clock Selector 2 selects one of three internal clocks only.

- **Counter circuit 1, 2**
  
  Counter circuit 1 and counter circuit 2 are each made up of an 8-bit counter, a comparator, a comparison data latch, and a data register (T1DR or T2DR).
  In each counter circuit, the 8-bit counter is an up-counter clocked by the selected count clock. The comparator compares the count in the counter with the value in the comparison data latch. When it detects a match, it clears the counter, and loads the contents of the data register into the comparison data latch.
  In the 8 bit-mode, the two counter circuits operate independently as timer 1 and timer 2. In the 16-bit mode, the two circuits are connected in series to form a single 16-bit counter with counter circuit 1 forming the low (8 LSBs) end of the counter, and counter circuit 2 at the high (8 MSBs) end.

- **Square wave output control circuit**
  
  An interrupt request is generated when the comparator detects a match in the 8-bit timer 1 mode or the 16-bit mode. At this time, if the square wave output is enabled, the output control circuit inverts the level output at the TO pin.
  The circuit can also initialize the output level to have the output square wave start out in a specific state ("H" or "L").

- **T1DR and T2DR registers**
  
  The value to be compared with the count in the counter is set by writing the desired value into these registers. They can be read to determine the current counter values.

- **T1CR and T2CR registers**
  
  The T1CR and T2CR registers are used to select the function, to enable or disable operation, control interrupts, and check the timer/counter status.
8.3 Structure of 8/16-bit Timer/Counter

This section describes the pins, pin block diagram, registers, and interrupt source of the 8/16-bit Timer/Counter.

■ 8/16-bit Timer/Counter Pins

The 8/16-bit timer/counter uses the P20/EC and P22/TO pins. The P20/EC pin can function either as a general-purpose I/O port (P20), as the external clock input pin of timer (EC). The P22/TO pin can function either as general-purpose I/O port (P22), as the square wave output pin of time (TO).

EC:

In the 8-bit timer 1 or 16-bit mode, if external clock input (counter function) is selected (T1CR: T1CS1, T1CS0 = 11B), the counter counts the external clocks applied to this pin. The P20/EC pin sets the pin as an input port in the port data direction register DDR2: bit 0 = “0”) when using as the EC pin.

TO:

In the 8-bit timer 1 or 16-bit mode, a square wave is output at this pin. Enabling square wave output (T1CR: T1OS1, T1OS0 = other than 00B) automatically sets the P22/TO pin as an output pin, regardless of the port data direction register (DDR2: bit 2) value, and sets the pin to function as the TO pin.

■ Block Diagram of 8/16-bit Timer Counter Pins

![Block Diagram of 8/16-Bit Timer/Counter Pins](image)

Note: Pins with a pull-up resistor (optional) go to the "H" level during a reset or in stop mode and watch mode (SPL = "1").
8/16-bit Timer/Counter Registers

**T1CR (Timer 1 control register)**

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0019H</td>
<td>T1IF</td>
<td>T1IE</td>
<td>T1OS1</td>
<td>T1OS0</td>
<td>T1CS1</td>
<td>T1CS0</td>
<td>T1STP</td>
<td>T1STR</td>
<td>X000XXX0s</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>

**T2CR (Timer 2 control register)**

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0018H</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>X000XXX0s</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>

**T1DR (Timer 1 data register)**

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>001BH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X000XXX0s</td>
<td></td>
</tr>
<tr>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**T2DR (Timer 2 data register)**

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>001AH</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X000XXX0s</td>
<td></td>
</tr>
<tr>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

R/W: Readable and writable
X: Indeterminate

**Figure 8.3b 8/16-bit Timer/Counter Registers**

- **8/16-bit Timer/Counter Interrupt Source**

IRQ5:
In the interval timer and counter functions, if the interrupt request output is enabled, an IRQ5 interrupt request will be generated when the count in the counter equals the value set in the data register. Interrupt request outputs are enabled by setting the proper timer control register bit (T1CR: T1IE = 1 in the 8-bit Timer 1 or 16-bit mode; or T2CR: T2IE = 1 in the 8-bit Timer 2 mode).
8.3 Structure of 8/16-bit Timer/Counter

8.3.1 Timer 1 Control Register (T1CR)

In the 8-bit Timer 1 and in the 16-bit mode, the Timer 1 Control Register (T1CR) is used to select functions, to enable/disable operation, to control interrupts, and to check status. In the 8-bit mode, the Timer 2 control register (T2CR) must still be initialized, even when only Timer 1 is used.

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0019h</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>X000XXX0b</td>
</tr>
</tbody>
</table>

- **T1STR** Timer activation bit
  - 0: Stops counter
  - 1: Clears counter, then starts it

- **T1STP** Timer stop bits
  - 0: Causes counter to resume counting without clearing
  - 1: Temporarily stops counter

- **T1CS1** T1CS0 Count clock selection Bits
  - 00: 2 t inst
  - 01: 32 t inst
  - 10: 512 t inst
  - 11: External clock

- **T1OS1** T1OS0 Square wave output control bits
  - 00: Use pin for general-purpose port (P22)
  - 01: Set data to output square wave “L” state
  - 10: Set data to output square wave “H” state
  - 11: Output set level at square wave output pin (TO).*

- **T1IE** interrupt request enable bit
  - 0: Disable interrupt request output
  - 1: Enable interrupt request output

- **T1IF** interrupt request Flag Bit
  - 0: No counter match clears this bit
  - 1: Have counter match No effect: The bit does not change.

---

* The square wave output pin will go to the level corresponding to data set when T1STR is “0.”

---

**Figure 8.3.1a PWC Pulse Width Control Register 1 (PCR1)**
Check: Before using 8/16-bit timer/counter Timer 1 only in 8 bit mode, first set the timer count clock selection bits of the Timer 2 control register (T2CR: T2CS1, T2CS0) to some state other than “11e.” Operating in this mode without making this register setting could result in faulty operation.

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>T1IF: Interrupt request flag bit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• 8 bit-mode:</td>
</tr>
<tr>
<td></td>
<td>• Set to “1” when the count in the timer 1 counter matches the value set in the T1DR, the timer 1 data register (comparison data latch).</td>
</tr>
<tr>
<td></td>
<td>• 16-bit mode:</td>
</tr>
<tr>
<td></td>
<td>• Set to “1” when the counts in the timer 1 and timer 2 counters match the values set in the T1DR and T2DR registers, respectively.</td>
</tr>
</tbody>
</table>
|       | • An interrupt request is output when both this bit and the interrupt request enable bit (T1IE) are “1”.
|       | • Writing “0” clears this bit. Writing “1” has no effect and does not change the bit value. |

<table>
<thead>
<tr>
<th>Bit 6</th>
<th>T1IE: Interrupt request enable bit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• This bit enables or disables an interrupt request output to the CPU.</td>
</tr>
<tr>
<td></td>
<td>• An interrupt request is output when both this bit and the interrupt request flag bit (T1IF) are “1”.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 5</th>
<th>T1OS1 and T1OS0: square wave output control bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 4</td>
<td></td>
</tr>
<tr>
<td></td>
<td>• P22/TO is a general-purpose I/O port pin (P22) if both of these bits are “00B.” If either bit is “1,” it is the square wave output pin (TO).</td>
</tr>
<tr>
<td></td>
<td>• If written to “01B,” level “L” will be output to TO pin. If written to “10B,” level “H” will be output to TO pin.</td>
</tr>
<tr>
<td></td>
<td>• If both bits are “11B,” and the function is in the stop timer state (T1STR = 0), the TO pin is set to a level corresponding to the initialize data.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 3</th>
<th>Bit 2</th>
<th>T1CS1 and T1CS0: Clock source selection bits</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>• Selects the count clock to be supplied to the counter.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• Selects one of three internal clocks, or an external clock.</td>
</tr>
<tr>
<td></td>
<td></td>
<td>• When both bits are “11B,” Timer 1 operates as a counter with the external clock is selected as the count clock.</td>
</tr>
<tr>
<td></td>
<td></td>
<td><strong>Check:</strong> If external clock input is selected (T1CS1, T1CS0 = 11B), P20/EC must be set as an input port.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 1</th>
<th>T1STP: Timer stop bit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• This bit is used to temporarily stop the counter.</td>
</tr>
<tr>
<td></td>
<td>• Writing this bit to “1” temporarily stops the counter. Writing it to “0” when the timer in startup state (T1STR = 1), restarts the counter where it left off.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 0</th>
<th>T1STR: Timer activation bit</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>• Starts and stops timer.</td>
</tr>
<tr>
<td></td>
<td>• Changing this bit from “0” to “1” clears the counter. At this time, if the timer is in the continuous operation mode (T1STP = 0), the counter starts (counts up, clocked by the selected count clock). Writing this bit to “0” stops the counter.</td>
</tr>
<tr>
<td></td>
<td>• In the 16 bit mode, both Timer 1 and Timer 2 are cleared at timer start (T1STP = 0 → 1).</td>
</tr>
</tbody>
</table>

Table 8.3.1a Timer 1 Control Register (T1CR) Bits
8.3 Structure of 8/16-bit Timer/Counter

8.3.2 Timer 2 Control Register (T2CR)

In the 8 bit mode, the Timer 2 Control Register (T2CR) is used to select functions, to enable/disable operation, to control interrupts, and to check states. In the 16 bit mode, although the function is controlled by the Timer 1 control register (T1CR), the Timer 2 control register (T2CR) must still be set.

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0018H</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>X000XXX0B</td>
</tr>
</tbody>
</table>

- **T2STR** Timer activation bit
  - 0: Stops counter
  - 1: Clears counter, then starts it

- **T2STP** Timer stop bit
  - 0: Causes counter to resume counting without clearing
  - 1: Temporarily stops counter

- **T2CS1** Count clock selection bits
  - 00: 2 t inst
  - 01: 32 t inst
  - 10: 512 t inst
  - 11: 16-bit mode

- **T2OS1** Square wave output control bits
  - 00: Always write to "00."
  - 01: Prohibited setting
  - 10: Prohibited setting
  - 11: Prohibited setting

- **T2IE** Interrupt request enable bit
  - 0: Disables interrupt request output
  - 1: Enables interrupt request output

- **T2IF** Interrupt request flag bit
  - 0: No counter match
  - 1: Have counter match

*Figure 8.3.2a Timer 2 Control Register (T2CR)*
### Table 8.3.2a Timer 2 Control Register (T2CR) Bits

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
</table>
| Bit 7 | T2IF: **Interrupt request flag bit**  
  - Set to “1” when the count in the timer 2 counter matches the value set in the T2DR, the Timer 2 data register (comparison data latch).  
  - An interrupt request is output when both this bit and the interrupt request enable bit (T2IE) are “1”.  
  - Writing “0” clears this bit. Writing “1” has no effect and does not change the bit value.  
  - **Check:** In the 16-bit mode, the T1IF bit is the valid interrupt request flag, and the T2IF bit has no effect. |
| Bit 6 | T2IE: **Interrupt request enable bit**  
  - This bit enables or disables an interrupt request output to the CPU.  
  - An interrupt request is output when both this bit and the interrupt request flag bit (T2IF) are “1”.  
  - **Check:** In the 16-bit mode, the T1IE bit is the valid interrupt request enable bit, and the T2IE bit has no effect. |
| Bit 5 and Bit 4 | T2OS1 and T2OS0: **Unused bits**  
  These bits are not used in timer 2. They should always be written to “00.” |
| Bit 3 and Bit 2 | T2CS1 and T2CS0: **Clock Source Selection Bits**  
  - Selects the count clock to be supplied to the counter.  
  - Selects one of three internal clocks.  
  - Setting to “11B” selects the 16-bit mode.  
  - **Check:** In 16-bit mode, T1CS1 and T1CS0 select the clock. T2CS1 and T2CS0 serve only to select the 16-bit mode. |
| Bit 1 | T2STP: **Timer stop Bit**  
  - This bit is used to temporarily stop the counter.  
  - Writing this bit to “1” temporarily stops the counter. Writing it to “0” when the timer start bit (T2STR) is “1” restarts the counter where it left off.  
  - **Check:** In 16-bit mode, T1STP is the stop bit, and T2STP has no effect. |
| Bit 0 | T2STR: **Timer activation bit**  
  - Starts and stops timer.  
  - Changing this bit from “0” to “1” clears the counter. At this time, if the T2STR bit is “0,” the counter starts (counts up, clocked by selected count clock). Writing this bit to “0” stops the counter.  
  - **Check:** In 16-bit mode, T1STR is the start bit, and T2STR has no effect. |

**Check:** When using timer 2 in the 16-bit mode, set T2CS1 and T2CS0 to “11B,” then use the T1CR register to control the circuit.
8.3 Structure of 8/16-bit Timer/Counter

8.3.3 Timer 1 Data Register (T1DR)

The Timer 1 data register (T1DR) is used to set all or part of the interval time or counter value, and to read out all or part of the counter value, depending on the mode and function being used. In 8-bit mode, it sets the Timer 1 interval time (interval timer function) or counter value (counter function), and reads out the counter value. In 16-bit mode, it sets the 8 LSBs of the 16-bit timer interval (interval timer function) or counter value (counter function), and reads out the counter value.

- Timer 1 Data Register (T1DR)

The value set into this register is compared with the counter value (count). If you read the register, you get the current counter value. The register setting cannot be read out.

Figure 8.3.3a shows the bit structure of the Timer 1 data register.

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>001FH</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>XXXXXXXXH</td>
</tr>
</tbody>
</table>

R/W: Readable and writable

- 8-bit mode (Timer 1)

The value in this register is compared with the count in the timer 1 counter. For the interval timer function it sets the interval time, and for the counter function, it sets the count to be detected. When count operation enabled (T1CR: T1STR = 0 → 1, T1STP = 0), the value in the T1DR register is loaded into the comparison data latch, and the counter starts counting up.

When the counter counts up to where it matches the value in the comparison data latch, the value in the T1DR register is re-loaded into the comparison data latch, and the counter is cleared and continues to count.

Since the comparison data latch is reloaded when a match is detected, if a new value is loaded into the T1DR register while the counter is counting, the new value will not take effect until the next count cycle (after a match is detected in the current cycle).

Note: The T1DR setting for interval timer operation can be calculated using the following formula. (The instruction cycle time is affected by the clock mode, and the speed-shift selection.)

\[ T1DR \text{ register value} = \frac{\text{interval time}}{\text{count clock cycle} \times \text{instruction cycle}} - 1 \]

- 16-bit mode

The value in this register is compared with the counter value for the lower 8 bits (LSBs) of the 16-bit timer. In the interval timer function, this sets the lower 8 bits of the interval time setting, and in the counter function, the lower 8 bits of the count to be detected. The contents of the T1DR register are loaded into the lower 8 bits of the comparison data latch when the counter first starts operating and when a match is detected in the 16-bit count. Therefore, if a new value is loaded into the T1DR register while the 16-bit counter is counting, the new value will not take effect until after the next match is detected.

Reference: For information on T1DR settings in the interval timer mode, refer to Section 8.3.4, "Timer 2 Data Register (T2DR)."
8.3 Structure of 8/16-bit Timer/Counter

8.3.4 Timer 2 Data Register (T2DR)

The Timer 2 data register (T2DR) is used to set all or part of the interval time or counter value, and to read out all or part of the counter value, depending on the mode and function being used. In 8-bit mode, it sets the Timer 2 interval time (interval timer function) or counter value (counter function), and reads out the counter value. In 16-bit mode, it sets the 8 MSBs of the 16-bit timer interval (interval timer function) or counter value (counter function), and reads out the counter value.

- Timer 2 Data Register (T2DR)

The value set into this register is compared with the counter value (count). If you read the register, you get the current counter value. The register setting cannot be read out. Figure 8.3.4a shows the bit structure of the Timer 2 data register.

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>001Ah</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>XXXXXXXXs</td>
</tr>
</tbody>
</table>

R/W: Readable and writable

- 8-bit mode (Timer 2)

The value in this register is compared with the count in the timer 2 counter. For the interval timer function, it sets the interval time, and for the counter function, it sets the count to be detected. The value in the T2DR register is reloaded into the comparison data latch when counter operation starts, and when a match is detected. If a new value is loaded into the T2DR register while the counter is counting, the new value will not take effect until the next count cycle (after a match is detected in the current cycle).

**Note:** The T2DR setting for interval timer operation can be calculated using the following formula. (The instruction cycle time is affected by the clock mode, and the speed-shift selection.)

\[
\text{T2DR register value} = \frac{\text{interval time}}{\text{count clock cycle} \times \text{instruction cycle time}} - 1
\]

- 16-bit mode

The value in this register is compared with upper 8 bits (MSBs) of the 16-bit timer. In the interval timer function, this sets the upper 8 bits of the interval time setting, and in the counter function, the upper 8 bits of the count to be detected. The contents of the T2DR register are loaded into the upper 8 bits of the comparison data latch when the counter first starts operating and when a match is detected in the 16-bit count. Therefore, if a new value is loaded into the T2DR register while the 16-bit counter is counting, the new value will not take effect until after the next match is detected. In the 16 bit mode, the operation of the counter is controlled by the Timer 1 control register (T1CR).

**Note:** In the interval timer function, the T1DR and T2DR register settings can be calculated from the following formula. (The instruction cycle time is affected by the clock mode, and the speed-shift selection.)

\[
\text{16-bit data value} = \frac{\text{interval time}}{\text{count clock cycle} \times \text{instruction cycle}} - 1
\]

The 8 MSBs of the 16-bit data value are the T2DR setting, and the 8 LSBs are the T1DR setting.
8.4 8/16-bit Timer/Counter Interrupt

In the 8/16-bit timer/counter, interrupt conditions are satisfied (if interrupts are enabled) when the counter matches the data register. This is true for both the interval timer and counter functions.

8/16-bit Timer/Counter Interrupt

Table 8.4a lists the 8/16-bit timer/counter interrupts, interrupt request flags and IRQ output enable bits.

Table 8.4a 8/16-bit Timer/Counter Interrupt Control Bits and Interrupts

<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:T1IE</td>
<td>T1CR:T1IE</td>
</tr>
</tbody>
</table>

In 8-bit mode, 8/16-bit timer/counter interrupt requests are generated independently for Timer 1 and Timer 2. In 16-bit mode, the interrupt request is generated only for Timer 1, but basic operation is the same. Interrupt operation will therefore be described only for Timer 1 in 8-bit mode.

8-bit mode timer 1 interrupt operation

The counter counts up from "00H", clocked by the selected count clock. When the count in the counter matches the value in the comparison data latch (corresponding to the value in timer data register T1DR), the interrupt request flag bit is set to "1" (T1CR:T1IF).

At this time, an interrupt request (IRQ5) to the CPU is generated if the interrupt request enable bit is enabled (T1CR: T1IE="1"). Write "0" to 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 T1IE bit.

In 8-bit mode, although Timer 1 and Timer 2 operate independently of each other, they both generate IRQ5. When processing IRQ5, then, the software may have to check the interrupt request flag bits to determine which timer generated the interrupt.

Note:
• The T1IF bit is not set if the counter is stopped (T1CR: T1STR = '0') at the same time as the counter value matches the T1DR register.
• An interrupt request is generated immediately if the T1IF bit is "1" when the T1IE bit is changed from disabled to enabled ("0"→"1").
## Registers and Vector Table for 8/16-bit Timer/Counter Interrupt

### Table 8.4b  Registers and Vector Table for 8/16-bit Timer/Counter Interrupt

<table>
<thead>
<tr>
<th>Interrupt</th>
<th>Interrupt level settings register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRQ5</td>
<td>ILR2 (007DH)</td>
<td>Upper: FFF0H, Lower: FFF1H</td>
</tr>
</tbody>
</table>

Reference: See section 3.4.2, "Interrupt Processing" for details on the interrupt operation.
8.5 Operation of Interval Timer Function

This section describes the operation of the interval timer function of the 8/16-bit timer/counter.

- **Operation of Interval Timer Function**
  - **8-bit mode**
    - Figure 8.5a shows the settings required to operate Timer 1 as the interval timer function in the 8-bit mode.
      - **Figure 8.5a Interval Timer Function (Timer 1) Settings**
        - Bit 7 Bit 6 Bit 5 Bit 4 Bit 3 Bit 2 Bit 1 Bit 0
          - T1CR: T1IF T1IE T1OS1 T1OS0 T1CS1 T1CS0 T1STP T1STR
            - ⊗ ⊗ ⊗ ⊗ ⊗ ⊗ ⊗ ⊗
          - T1DR: Sets the interval time (compare value).
          - T2CR: T2IF T2IE T2OS1 T2OS0 T2CS1 T2CS0 T2STP T2STR
            - × × 0 0 ⊗ ⊗ ⊗ ⊗
          - T2DR: Sets the interval time (compare value).

        - Figure 8.5b shows the settings required to operate Timer 2 as the interval timer function in the 8-bit mode.
    - On activation in 8-bit mode, the counter starts counting-up from “00H”, on the rising edge of the selected count clock. Eventually, the count in the counter will match the value set in the data register (comparison data latch). When this occurs, the timer control register interrupt request flag bit (T1CR: T1IF) is set to “1”, and the counter starts counting-up again from “00H”. If using timer 1, the output of the square wave output control circuit is inverted when a match is detected; and if square wave output is enabled (T1CR: T1OS1, T1OS0 = values other than “00”), a square wave is output at the TO pin.
Figure 8.5c shows the interval timer function operation in the 8-bit mode.

### Figure 8.5c Operation of Interval Timer (Timer 1)

- **16-bit mode**

Figure 8.5d shows the settings required to operate the interval timer function in the 16-bit mode.

<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>T1CR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T1IF</td>
<td>T1IE</td>
<td>T1OS1</td>
<td>T1OS0</td>
<td>T1CS1</td>
<td>T1CS0</td>
<td>T1STP</td>
<td>T1STR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T2CR</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>T2IF</td>
<td>T2IE</td>
<td>T2OS1</td>
<td>T2OS0</td>
<td>T2CS1</td>
<td>T2CS0</td>
<td>T2STP</td>
<td>T2STR</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- **Counter count**
- **Compare value (E0H)**
- **Compare value (FFH)**
- **Time**

- **T1DR value (E0H)**
- **Cleared by the program**
- **T1DR value modified (E0H \rightarrow FFH)**

- **T1IF bit**
- **Activate**
- **Match**
- **Counter Clear**

*1 If a new value is written to the data register during counter operation, the new value is used from the next cycle.
*2 At activation, and each time a match is detected, the counter is cleared and the data register setting is loaded into the comparison data latch.

**Figure 8.5d Interval Timer Function Settings (16-bit Mode)**

In 16-bit mode, the timer 1 control register (T1CR) controls the timer. The timer 2 control register (T2CR) must, however, still be initialized. The data to be compared with the 16-bit counter is set in both data registers: the upper 8 bits in T2DR and the lower 8 bits in T1DR. All 16 bits of the counter are cleared simultaneously. All other operation in the 16-bit mode is the same as timer 1 operation in 8-bit mode.
8.6 Operation of Counter Function

This section describes the operations of the counter function of the 8/16-bit timer/counter.

- 8-bit mode

Figure 8.6a shows the settings required to operate the timer 1 as the counter function in the 8-bit mode.

![Figure 8.6a Counter Function Settings (8-Bit Mode)]

Counter operation in the 8-bit mode is the same as interval timer operation of timer 1 in 8-bit mode, except that an external clock is used in lieu of the internal clock.

- 16-bit mode

Figure 8.6b shows the settings required to operate the counter function in the 16-bit mode.

![Figure 8.6b Counter Function Settings (16-Bit Mode)]

Counter operation in the 16-bit mode is the same as interval timer operation in 16-bit mode, except that an external clock is used in lieu of the internal clock.
Figure 8.6c shows the counter function operation in the 16-bit mode.

Check: When the counter value during operation is read out in 16-bit mode, always read it twice, and verify that a proper value is got before using it.
8.7 **Operation of the Square Wave Output Initial Setting Function**

The square wave output can be set to the desired initial value using the timer 1 control register (T1CR).

### Operation of Square Wave Output Initial Setting Function

The square wave output can be set to the desired initial value by the program, but this can be done only when the timer operation is stopped (T1CR: T1STR = 0).

Figure 8.7a shows an equivalent circuit for the square wave output control circuit initial setting. To perform the initial setting, follow the procedure in Table 8.7a. The operation of the square wave output when this is done is as shown in Figure 8.7b.

![Figure 8.7a Square Wave Output Initial Setting Equivalent Circuit](image)

**Table 8.7a Square Wave Output Initial Setting Procedure (T1CR Register)**

<table>
<thead>
<tr>
<th>Step</th>
<th>Settings and Operation</th>
</tr>
</thead>
</table>
| (1)   | To set the square wave output pin (TO) “L”, set the square wave output control bits (T1CR: T1OS1, T1OS0) first to “01B”, then to “11B”. To set the TO pin “H”, set the bits to “10B”, then “11B.”  
**Note:** Until the bits are written to “11B”, the circuit simply holds the latched value, and the TO pin level remains in its current or previous state. |
| (2)   | If the square wave output control bits (T1OS1, T1OS0) are written to “11B” and the timer operation stopped (T1STR = 0), the TO pin will output the level corresponding to the level latch value (initial value). This can also be accomplished by setting T1OS1, T1OS0, and T1STR simultaneously. If the timer activation bit is set (T1STR= 1), the counter will start. |
| (3)   | The square wave output is inverted each time the counter value matches the data register settings. |

![Figure 8.7b Square Wave Output Initial Setting Operation](image)

*1: When the T1OS1 and T1OS0 bits of the T1CR register are both “00B”, the P22/TO pin is a general-purpose port pin (P22).  
*2: If either T1OS1 or T1OS2 bit is “1”, the P22/TO pin is a square wave output pin (TO).
8.8 Operation of 8/16-bit Timer/Counter Stop and Restart

This section describes the operation of stop and restart operation functions of the 8/16-bit timer/counter.

- Timer Stop and Restart

Operation is described for timer 1 only. Timer 2, however, operates in the same way.

Timer 1 is stopped and restarted using the timer 1 control register stop and start bits (T1CR: T1STP and T1STR).
- To start the counter after clearing it, with “0” in T1STR bit
  
Set T1STP, T1STR bit to “01”. On the T1STR bit rising edge, the counter will be cleared and start counting.
- To temporarily stop the counter and then resume counting (without clearing the counter)
  
First stop the counter by setting T1STP, T1STR to “11”, then set T1STP, T1STR to “01” to resume counting where you left off.

Table 8.8a lists the timer states for each T1STP, T1STR bit, and operation when the timer is activated from that state (T1STP, T1STR = 01).

<table>
<thead>
<tr>
<th>T1STP (T2STP)</th>
<th>T1STR (T2STR)</th>
<th>Timer State</th>
<th>Timer operation when counter is activated (T1STP, T1STR=01B) from the state shown at the left</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Counter stopped</td>
<td>Counter cleared and starts counting</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>Counter operating</td>
<td>Counter keeps on operating as-is.</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>Counter stopped</td>
<td>Counter cleared and starts counting</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>Counter temporarily stopped</td>
<td>Counter resumes counting without being cleared</td>
</tr>
</tbody>
</table>
8.9 States in Each Mode during 8/16-bit Timer/Counter Operation

This section describes the operation of the 8/16-bit timer/counter when the device changes to sleep or stop mode or an operation halt request occurs during operation.

- **Operation during Subclock Mode, Standby Mode, or Operation Halt**

  Figure 8.9a shows the counter value state when the device changes to sleep or stop mode, or an operation halt request occurs, during operation of the interval timer function or counter function (for timer 1).

  The counter halts and maintains its current value when the device changes to stop mode. Operation starts again from the stored counter value after wake-up from stop mode by an external interrupt; therefore, the first interval time or external clock count is not correct value. Always initialize the 8/16-bit timer/counter after wake-up from stop mode.

  Operation when entering or exiting watch mode (STBC: TMD = 1) is the same as when entering or exiting stop mode. Watch mode is cleared by a watch prescaler interrupt or external interrupt. When the counter is stopped temporarily (T1STP = 1), it holds the count it had when it was stopped. When it is restarted (T1STP= 0), it resumes counting from the count at which it was stopped.

  *: The TO pin goes to the high-impedance state during stop mode if the pin state specification bit in the standby control register (STBC: SPL) is "1" and the TO pin is not set to with a pull-up resistor (optional). When the SPL bit is "0", the pin maintains its value prior to changing to stop mode.
8.10 Notes on Using 8/16-bit Timer/Counter

This section lists points to note when using the 8/16-bit timer/counter.

- Notes on Using 8/16-bit Timer/Counter
  - Notes when counter is stopped
    
    This information is described for timer 1, but the same information applies to timer 2. As shown in Figure 8.10a, if the clock is “L” when T1STP temporarily stops the timer, the count will be incremented by 1. This may also occur if the input clock is “L” after a temporary stop, and the T1STP and T1STR bits are both written to “00” simultaneously. When using the T1STP bit to temporarily stop the counter, first read out the counter value; then write T1STP bit to “0”.

    Figure 8.10a Operation when Timer Stop Bit is Used

- Error

  Activating of by program 8/16-bit timer/counter is not synchronized with the start of counting-up using the selected count clock. Therefore, the time from activating the counter to match the register setting may be shorter than the theoretical time by a maximum of one cycle of the count clock. Figure 8.10b shows the error that occurs on standing counter operation.

    Figure 8.10b Error on Starting Counter Operation
● Using one 8-bit channel

When 8/16-bit timer/counter timer 1 only is used in the 8-bit mode, before doing so, first set the timer count clock select bits of the timer 2 control register (T2CR: T2CS1, T2CS0) to some state other than "11". Failure to do so may result in faulty operation.

● Notes on setting by program

• When the 8/16-bit timer/counter Timer 1 only is used in the 16-bit mode, the timer 2 control register count clock select bits (T2CR: T2CS1, T2CS0) should always be set to "11", and bits 5 and 4, the unused bits (T2CR: TSOS1, TSOS0) to "00".

• In 16-bit mode, when the counter value is read out during operation, always read it twice and verify that a proper value is got before using it.

• While the timer is operating (T1CR: T1STR = 1), performing the initial state setting will not immediately cause the square wave output level to change. The output state will be initialized when the timer stops.

• Interrupt processing cannot return if the interrupt request flag bit (T1CR: T1IF, T2CR: T2IF) is "1" and the interrupt request enable bit is enabled (T1CR: T1IE= "1", T2CR: T2IE= "1"). Always clear the interrupt request flag bit.

• The interrupt request flag bit (T1CR: T1IF or T2CR: T2IF) is not set if the counter is disabled by the timer start bit (T1CR: T1STR=0 or T2CR: T2CR =0) at the same time as an interrupt source is generated.
8.11 Program Examples for 8/16-bit Timer/Counter

This section gives a program examples for 8/16-bit timer/counter.

Program Example for Interval Timer Function

- **Processing description**
  - Using timer 1 only, in 8-bit mode, generates repeated interval timer interrupts at 20 ms intervals.
  - Outputs a square wave to the TO pin that inverts after each interval time.
  - With a main clock master oscillation $F_{CH}$ of 4.2 MHz, and the highest speed main clock selected by the speed-shift function (1 instruction cycle time = $4/F_{CH}$), and with an internal clock period of 512 $t_{inst}$ selected as the count clock, the T1DR setting for an interval of approximately 20 ms is calculated as follows:

$$T1DR \text{ register value} = \frac{20 \text{ ms}}{(512 \times 4/4.2 \text{ MHz})} - 1 = 40.0 \ (28 \text{H})$$

- **Coding example**

```assembly
T2CREQU 0018H; Address of the Timer 2 control register
T1CREQU 0019H; Address of the Timer 1 control register
T2DREQU 001AH; Address of the Timer 2 data register
T1DREQU 001BH; Address of the Timer 1 data register
T1IFEQU T1CR:7; Define the timer 1 interrupt request flag bit.
ILR2EQU 007DH; Address of the interrupt level setting register 2

INT_VDSEGABS
ORG 0FFF0H
IRQ5DW WARI; Set interrupt vector.
ENDS;

;----------Main program--------------------------------------------------------------------------------------------------------
CSEG ;
[CODE SEGMENT]
; Stack pointer (SP) etc. are already initialized.
CLRI ; Disable interrupts.
MOV ILR2,#11111011B; Set interrupt priority to level 2.
MOV T2CR,#00000010B; Clear timer 2 interrupt request flag, disable interrupt request output, set other
; than 16-bit mode, stop operation.
MOV T1CR,#00011000B; Clear timer 1 interrupt request flag, initialize square wave output
; "L", select 512 $t_{inst}$, and stop operation.
MOV T1DR,#28H; Set value compared with the counter value (interval time).
MOV T1CR,#00111000B; Output "L" at square wave output pin (TO).
MOV T1CR,#11111001B; timer 1 interrupt request, clear counter,
; and start timer.
SETI ; Enable CPU interrupts.

;----------Interrupt Program------------------------------------------------------------------------------------------------------
WARICLRBT1IF; Clear interrupt request flag.
PUSHWA
XCHW A,T
PUSHWA
; User processing

: POPW A
XCHW A,T
POPW A
RETI
ENDS
;-------------------------------------------------------------------------------------------------------------------------------

-END
```

```
Program Example for Pulse Counter Function

- Processing description
  - Using timer 1 and timer 2 in 16-bit mode, count external clocks input to the EC pin, and generate an interrupt once for each 5000 clocks (1388H).
  - Shows a sample program (READ16) for reading out the count in the 16-bit counter, while the counter is counting.

- Coding example
  
  DDR2EQU 0000DH; Address of the Port 2 data direction register
  T2CREQU 0018H; Address of the timer 2 control register
  T1CREQU 0019H; Address of the timer 1 control register
  T2DREQU 001AH; Address of the timer 2 data register
  T1DREQU 001BH; Address of the timer 1 data register
  T1IFEQU T1CR:7; Define the timer 1 interrupt request flag bit.
  ILR2EQU 007DH; Address of the interrupt level setting register 2
  NT_VDSEGABS
  ORG 0FFF0H
  IRQ5DW WARI; Set interrupt vector.

  ENDS

  ;----------Main program-------------------------------------------------------------------------------------------------------- -------------------
  CSEG
  [CODE SEGMENT]

  ; Stack pointer (SP) etc. are already initialized.
  MOV DDR2,#00000000B; Set P20/EC pin as an input.
  CLR1 ; Disable interrupts.
  MOV ILR2,#11111011B; Set interrupt level 2.
  MOV T1DR,#088H; Set lower 8 bits of counter comparison value.
  MOV T2DR,#013H; Set upper 8 bits of counter comparison 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 P22/TO as general-purpose port (P22), select external clock, clear counter, and start operation.
  SETI ; Enable CPU interrupts.

  ;----------Data read subroutine----------------------------------------------------------------------------------------------------------
  READ16MOVWA T2DR; 16-bit read, T1DR + T2DR.
  MOVWA T2DR; 16-bit read, T1DR + T2DR, save old value in T register.
  CMPWA ; Check first and second reads, compare A and T registers.
  BEQ RET16; If match, return.
  XCHW A,T
  INCWA ; Old value + 1
  CMPWA
  BNE READ16; If mismatch, read again.
  RET16RET

  ;----------Interrupt program-------------------------------------------------------------------------------------------------------- -------------------
  WARI CLRBT1IF; Clear interrupt request flag.
  PUSHWA
  XCHW A,T
  PUSHWA

  ; User process
  POPWA
  XCHW A,T
  POPWA
  RETI
  ENDS

END

---
CHAPTER 9
8-BIT SERIAL I/O

This chapter describes the functions and operation of the 8-bit serial I/O.

9.1 Overview of 8-bit Serial I/O ........................................ 190
9.2 Block Diagram of 8-bit Serial I/O .............................. 192
9.3 Structure of 8-bit Serial I/O ..................................... 194
9.4 8-bit Serial I/O Interrupts ........................................ 199
9.5 Operation of Serial Output ....................................... 200
9.6 Operation of Serial Input ......................................... 202
9.7 States in Each Mode during 8-bit Serial I/O Operation... 204
9.8 Notes on Using 8-bit Serial I/O ................................. 206
9.9 Connection Example for 8-bit Serial I/O .................... 207
9.10 Program Example for 8-bit Serial I/O ...................... 208
9.1 Overview of 8-bit Serial I/O

The 8-bit serial I/O function is the serial transfer of 8-bit data, synchronized with the shift clock. The shift clock can be selected among one external and three internal clocks. The data shift direction can be selected as either LSB first or MSB first.

---

## Serial I/O Function

The 8-bit serial I/O function is the serial input and output of 8-bit data, synchronized with the shift clock.

- The serial I/O converts 8-bit parallel data to serial and outputs the serial data. Similarly, the serial I/O converts input serial data to parallel and stores the data.
- One shift clock can be selected from one external and three internal clocks.
- The serial I/O can control input and output of the shift clock and can output the internal shift clock.
- The data shift direction (transfer direction) can be selected as either LSB first or MSB first.

### Table 9.1a Shift Clock Cycle and Transfer Speed

<table>
<thead>
<tr>
<th>Shift clock</th>
<th>Clock (cycle)</th>
<th>Frequency (Hz)</th>
<th>Transfer speed (FCH=4.2 MHz; at max. speed selected*)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Internal shift clock (output)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2 t inst</td>
<td>1/(2 t inst)</td>
<td>1/(2 t inst)</td>
<td>525 kbps</td>
</tr>
<tr>
<td>8 t inst</td>
<td>1/(8 t inst)</td>
<td>1/(8 t inst)</td>
<td>131.5 kbps</td>
</tr>
<tr>
<td>32 t inst</td>
<td>1/(32 t inst)</td>
<td>1/(32 t inst)</td>
<td>32.8 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 525 kbps</td>
</tr>
</tbody>
</table>

F<sub>CH</sub>: Main clock source oscillation

F<sub>inst</sub>: Instruction cycle (affected by clock mode, etc.)

*: Main clock and highest clock speed selected via the system clock control register (SYCC: SCS,CS1,CS0:111b) (1 instruction cycle = 4/F<sub>CH</sub>).

---
9.2 Block Diagram of 8-bit Serial I/O

Each Channel of the 8-bit serial I/O consists of the following four blocks:
- Shift clock controller
- Shift clock counter
- Serial data register (SDR)
- Serial mode register (SMR)

![Figure 9.2a Block Diagram of 8-Bit Serial I/O](image-url)
- **Shift clock control circuit**
  Selects the shift clock from one external and three internal clocks.

  If an internal shift clock is selected, the shift clock can be output to the SCK pin. If external shift clock is selected, the clock input from the SCK pin is used as the shift clock. The SDR register shifts in sync with the shift clock and the shifted-out value is output to the SO pin. Similarly, the serial input is obtained by shifting-in the SI pin input to the SDR register.

- **Shift clock counter**
  The shift clock counter counts the number of SDR register shifts generated by the shift clock and overflows after eight shifts.

  The overflow clears the serial I/O transfer start bit in the SMR register (SST = “0”) and sets the interrupt request flag (SIOF = “1”). The shift clock counter stops counting when serial transfer halts (SST = “0”). The shift clock counter is cleared when serial transfer restarts (SST = “1”).

- **SDR register**
  The SDR register is used to store the transfer data. Data written to this register is converted to serial and output. Serial input is converted to parallel data and stored in this register.

- **SMR register**
  The SMR register is used to enable or disable serial I/O operation, select the shift clock, set the transfer (shift) direction, control interrupts, and check the serial I/O status.
9.3 Structure of 8-bit Serial I/O

This section describes the pins, pin block diagram, registers, and interrupt source of 8-bit serial I/O-1.

- 8-bit Serial I/O Pins


  - P23/SI pin
    The P23/SI pin can function either as a general-purpose I/O port (P23) or as the serial data input (hysteresis input) for 8-bit serial I/O (SI).
    Set the P23/SI pin as an input port in the port data direction register (DDR2 bit 3 = “0”) when using as the SI pin.

  - P24/SO pin
    The P24/SO pin can function either as a general-purpose I/O port (P24) or as the serial data output (an N-ch open-drain output) for 8-bit serial I/O (SO).
    Enabling serial data output (SMR : SOE = “1”) automatically sets the P24/SO pin as an output pin, regardless of the port data direction register (DDR2 : bit 4) value, and sets the pin to function as the SO pin.

  - P25/SCK pin
    The P25/SCK pin can function either as a general-purpose I/O port (P25) or as the shift clock I/O for 8-bit I/O (SCK) (hysteresis input and N-ch open-drain output).
    - When using as the shift clock input pin
      When using SCK as an input pin, set the pin as an input port in the port data direction register (DDR2 : bit 5 = “0”) and disable shift clock output (SMR : SCKE = “0”). In this case, always select external shift clock operation (SMR : CKS1, CKS0 = “11”).
    - When using as the shift clock output pin
      Enabling shift clock output (SMR : SCKE = “1”) automatically sets the P25/SCK pin as an output pin, regardless of the port data direction register (DDR2 : bit 5) value, and sets the pin to function as the SCK output pin. In this case, always select internal shift clock operation (SMR : CKS1, CKS0 = other than “11”).
**Block Diagram of 8-bit Serial I/O Pins**

- **PDR (Port data register)**
  - To input (SI and SCK pins only)
  - Pull-up resistor (optional): approx. 50 kΩ, 5.0 V

- **DDR (Port data direction register)**
  - Stop, watch modes (SPL = 1)

- **Output latch**

**Figure 9.3a Block Diagram of 8-bit Serial I/O Pin**

**Note:** Pins with a pull-up resistor (optional) go to the “H” level during a reset or in stop and watch modes (SPL = “1”).

**8-bit Serial I/O Registers**

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>001Ch</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>00000000b</td>
</tr>
</tbody>
</table>

**SMR (Serial mode register)**

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>001Dh</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>XX...XX...a</td>
</tr>
</tbody>
</table>

**SDR (Serial data register)**

**8-bit Serial I/O Interrupt Source**

IRQ6: 8-bit serial I/O generates an interrupt request (IRQ6) if interrupt request output is enabled (SMR : SIOE = “1”) when the I/O function completes input or output of 8 bits of serial data.
9.3 Structure of 8-bit Serial

9.3.1 Serial Mode Register (SMR)

The serial mode register (SMR) is used to enable or disable operation, select the shift clock, set the transfer direction, control interrupts, and check the state of 8-bit serial I/O.

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>001Ch</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>00000000B</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>

- **SST** Serial I/O transfer start bit
  - Read
  - 0: Serial transfer stopped. Stop disable serial transfer.
  - 1: Serial transfer operating. Starts/enables serial transfer.

- **BDS** Transfer direction selection bit
  - 0: LSB first (starts transfer from the least significant bit)
  - 1: MSB first (starts transfer from the most significant bit)

- **CKS1** and **CKS0** Shift clock selection bits SCK pin
  - 00: Internal shift clock. 2 tinst Output
  - 01: Internal shift clock. 8 tinst Output
  - 10: 32 tinst Output
  - 11: External shift clock Input
  - tinst: Instruction cycle

- **SOE** Serial data output enable bit
  - 0: Functions P24/S0 as the general-purpose port.
  - 1: Functions P24/S0 as the serial data output pin.

- **SCKE** Shift clock output enable bit
  - 0: Functions P25/SCK as a general-purpose port or shift clock input pin.
  - 1: Functions P25/SCK as the shift clock output pin.

- **SIOE** Interrupt request enable bit
  - 0: Enables Interrupt request output.
  - 1: Enables Interrupt request output.

- **SIOF** Interrupt request flag bit
  - Read
  - 0: Transfer has not complete. Clears this bit.
  - 1: Transfer has completed. No effect. The bit does not change.

Figure 9.3.1a Serial Mode Register (SMR)
### Table 9.3.1a Serial Mode Register (SMR) Bits

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Bit 7</strong></td>
<td>SIOF: Interrupt request flag bit</td>
</tr>
<tr>
<td>Bit 7</td>
<td>This bit is set to “1” when the serial output operation has output 8 bits of serial data or the serial input operation has input 8 bits serial data. An interrupt request is output when both this bit and the interrupt request enable bit (SIOE) are “1”.</td>
</tr>
<tr>
<td>Bit 7</td>
<td>Writing “0,” clears this bit. Writing “1” has not effect and does not change the bit value.</td>
</tr>
<tr>
<td><strong>Bit 6</strong></td>
<td>SIOE: Interrupt request Enable Bit</td>
</tr>
<tr>
<td>Bit 6</td>
<td>This bit enables or disables an interrupt request output to the CPU. An Interrupt request is output when both this bit and the Interrupt request flag bit (SIOF) are “1.”</td>
</tr>
<tr>
<td><strong>Bit 5</strong></td>
<td>SCKE: Shift clock output enable bit</td>
</tr>
<tr>
<td>Bit 5</td>
<td>• This bit controls shift clock input and output.</td>
</tr>
<tr>
<td>Bit 5</td>
<td>• The P25/SCK pin function as the shift clock input pin when this bit is set to “0” and as the shift clock output pin when set to “1.”</td>
</tr>
<tr>
<td>Bit 5</td>
<td><strong>Check:</strong></td>
</tr>
<tr>
<td>Bit 5</td>
<td>• Set the P25/SCK pin as an input port when using this pin as the shift clock input. Also, selects external shift clock operation in the shift clock selection bits (CKS1, CKS0 = “11B”).</td>
</tr>
<tr>
<td>Bit 5</td>
<td>• When using this pin as shift clock output (SCK = “1”), select internal shift clock operation (CKS1, CKS0 = values other than 00B).</td>
</tr>
<tr>
<td><strong>Bit 5</strong></td>
<td>Notes:</td>
</tr>
<tr>
<td>Bit 5</td>
<td>• The pin functions as the SCK output pin when shift clock is enabled (SCKE = “1”) regardless of the state of the general-purpose port (P25).</td>
</tr>
<tr>
<td>Bit 5</td>
<td>• Set to shift clock input operation (SCKE = “0”) when using this pin as a general-purpose port (P25).</td>
</tr>
<tr>
<td><strong>Bit 4</strong></td>
<td>SOE: Serial data output enable bit</td>
</tr>
<tr>
<td>Bit 4</td>
<td>The P24/SO pin functions as a general-purpose port (P24) when this bit is set to “0” and as the serial data output pin (SO) when set to “1.”</td>
</tr>
<tr>
<td><strong>Bit 4</strong></td>
<td><strong>Note:</strong></td>
</tr>
<tr>
<td>Bit 4</td>
<td>• The pin functions as the (SO) pin when serial data output is enabled (SOE = “1”), regardless of the state of the general-purpose port (P24).</td>
</tr>
<tr>
<td><strong>Bit 3</strong></td>
<td>CKS1, CKS0: Shift clock selection bits</td>
</tr>
<tr>
<td>Bit 3</td>
<td>• These bits select the shift clock from one external and three internal shift clocks.</td>
</tr>
<tr>
<td>Bit 2</td>
<td>• Setting these bits to other than “11B” selects an internal shift clock. In this case, the shift clock is output from the SCK pin when shift clock output enable bit (SCKE) is “1.”</td>
</tr>
<tr>
<td>Bit 0</td>
<td>• Setting these bits to “11B” selects the external shift clock. This inputs the shift clock from the SCK pin if shift clock input is enabled (SCKE = “0” and DDR2: bit 5 = “0”).</td>
</tr>
<tr>
<td><strong>Bit 1</strong></td>
<td>BDS: Transfer direction selection bit</td>
</tr>
<tr>
<td>Bit 1</td>
<td>This bit selects whether serial data is transferred with the least significant bit first (LSB first, BDS = “0”) or the most significant bit first (MSB first, BDS = “1”).</td>
</tr>
<tr>
<td><strong>Bit 1</strong></td>
<td><strong>Check:</strong></td>
</tr>
<tr>
<td>Bit 1</td>
<td>As bits are set in the appropriate order when writing to or reading from the serial data register (SDR), modifying this bit does not apply to any data already set in the SDR register.</td>
</tr>
<tr>
<td><strong>Bit 0</strong></td>
<td>SST: Serial I/O transfer start bit</td>
</tr>
<tr>
<td>Bit 0</td>
<td>• This bit controls serial I/O transfer start and transfer enable. This bit can also be used to determine whether transfer has completed.</td>
</tr>
<tr>
<td>Bit 0</td>
<td>• Writing “1” to this bit when an internal shift clock is selected (CKS1, CKS0 = other than “11B”) clears the shift clock counter and starts data transfer.</td>
</tr>
<tr>
<td>Bit 0</td>
<td>• Writing “1” to this bit when an external shift clock is selected (CKS1, CKS0 = “11B”) enables data transfer, clears the shift clock counter, and sets serial I/O to delay for input of the external shift clock.</td>
</tr>
<tr>
<td><strong>Bit 0</strong></td>
<td>• This bit is cleared to “0” and the SIOF bit set to “1” when transfer completes.</td>
</tr>
<tr>
<td>Bit 0</td>
<td>• Writing “0” to this bit while transfer is in progress (SST = “1”) aborts the transfer.</td>
</tr>
<tr>
<td><strong>Bit 0</strong></td>
<td>After halting a transfer, data must be set again to the SDR register for data output and transfer restarted (the shift clock counter cleared) for data input.</td>
</tr>
</tbody>
</table>
9.3 Structure of 8-bit Serial

9.3.2 Serial Data Register (SDR)

The serial data registers (SDR) stores the transfer data for 8-bit serial I/O. The register functions as the transmit data register for serial output operation and as the receive data register for serial output operation.

Figure 9.3.2a. shows the bit structure of the serial data register.

<table>
<thead>
<tr>
<th>Register</th>
<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>SDR</td>
<td>001DH</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>XXXXXXXXXXXs</td>
</tr>
</tbody>
</table>

R/W : Readable and writable
X : Indeterminate

- Serial output operation
  The register functions as the transmit data register. When serial I/O transfer starts (SMR: SST = "1"), the 8-bit serial I/O performs serial transfer of the data written in the register.

- Serial input operation
  The register function as the receive data register. When serial I/O transfer starts (SMR: SST = "1"), the received serial transfer data is stored in this register.

- During serial I/O transfer
  Do not write a data to the SDR register during serial I/O transfer operating. Also, the read value has no meaning.
9.4 8-bit Serial I/O Interrupts

The 8-bit serial I/O can generate interrupt requests after completion of the serial input and output of the 8-bit data.

### Interrupt for Serial Output Operation

The 8-bit serial I/O performs the serial input operation and serial output operation at the same time. When the serial transfer starts, the data in the serial data register (SDR) is input and output one bit at a time, synchronized with the cycle of the selected shift clock. The interrupt request flag bit (SMR: SIOF) is set to “1” on the rising edge of the shift clock of the eighth bit.

At this time, an interrupt request (IRQ6) to the CPU is generated if the interrupt request enable bit is enabled (SMR: SIOE = “1”).

Write “0” to the SIOF bit in the interrupt processing routine to clear the interrupt request. The SIOF bit is set after completing 8-bit serial output, regardless of the SIOE bit value.

**Note:** The interrupt request flag bit is not set (SMR: SIOF = “1”) if serial transfer is stopped (SMR: SST = “0”) at the same time as serial data transfer completes for the serial I/O operation. An interrupt request is generated immediately if the SIOF bit is “1” when the SIOE bit is changed from disabled to enabled (“0” → “1”).

### Register and Vector Table for 8-bit Serial I/O Interrupts

#### Table 9.4a  Register and Vector Table for 8-bit Serial I/O Interrupts

<table>
<thead>
<tr>
<th>Interrupt</th>
<th>Interrupt setting register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRQ6</td>
<td>ILR2 (007DH) L61 (bit5) L60 (bit4)</td>
<td>FFE6H FFEFH</td>
</tr>
</tbody>
</table>

**Reference:** See Section 3.4.2, “Interrupt Processing” for details on the interrupt operation.
9.5 Operation of Serial Output

The 8-bit serial I/O can perform serial output of 8-bit data synchronized with a shift clock.

■ Serial Output Operation

Serial output can operate using an internal or external shift clock. When serial output operation is enabled, the contents of the SDR register are output to the serial data output pin (SO). Serial input is performed at the same time.

● Internal shift clock

Figure 9.5a shows the settings required to operate serial output using an internal shift clock.

Figure 9.5a Serial Output Settings (When Using Internal Shift Clock)

Activating serial output operation outputs the contents of the SDR register to the SO pin, synchronized with the falling edge of the selected internal shift clock. At this time, the device being communicated with (a serial input) must be waiting for input of the external shift clock.

● External shift clock

Figure 9.5b shows the settings required to operate serial output using an external shift clock.

Figure 9.5b Serial Output Settings (When Using External Shift Clock)

Enabling serial output operation outputs the contents of the SDR register to the SO pin, synchronized with the falling edge of the external shift clock. When serial output completes, reset the SDR register and enable operation (SMR: SST = “1”) promptly for output of the next data.

When the device being communicates with has completed the serial input operation (on the rising edge), hold the external shift clock at the “H” level while waiting for next output data (idle state).
Figure 9.5c shows the 8-bit serial output operation.

**Figure 9.5c 8-bit Serial Output Operation**

- **Operation at Completion of Serial Output**
  
The 8-bit serial I/O sets the interrupt request flag bit (SMR: SIOF = 1) and clears the serial I/O transfer start bit (SMR: SST = “0”) on the rising edge of the shift clock after the serial data of the eighth bit is input or output.
9.6 Operation of Serial Input

The 8-bit serial I/O can perform serial input of 8-bit data synchronized with a shift clock.

### Serial Input Operation

Serial input can operate using an internal or external shift clock. When serial input operation is enabled, the contents of the SDR register are input from the serial data input pin (SI). Serial output is performed at the same time.

#### Internal shift clock

Figure 9.6a shows the settings required to operate serial input using an internal shift clock.

![Figure 9.6a Serial Input Settings (When Using Internal Shift Clock)](image)

Starting serial input operation stores the value of the serial data input pin (SI) to the SDR register, synchronized with the rising edge of the selected internal shift clock. At this time, the device being communicated with (a serial output) must have data set in the SDR register and be waiting for input of the external shift clock.

#### External shift clock

Figure 9.6b shows the settings required to operate serial input using an external shift clock.

![Figure 9.6b Serial Input Settings (When Using External Shift Clock)](image)

Enabling serial input operation stores the data on the SI pin to the SDR register, synchronized with the rising edge of the external shift clock. When serial input completes, read the SDR register and enable operation (SMR: SST = “1”) promptly to input next data.

During this time, hold the external shift clock at the “H” level while waiting for the next data (idle state).
Figure 9.6c shows the 8-bit serial input operation.

The 8-bit serial I/O sets the interrupt request flag bit (SMR: SIOF = "1") and clears the serial I/O transfer start bit (SMR: SST = "0") on the rising edge of the shift clock after the serial data of the eighth bit is input or output.

**Operation at Completion of Serial Input**
- The 8-bit serial I/O sets the interrupt request flag bit (SMR: SIOF = "1") and clears the serial I/O transfer start bit (SMR: SST = "0") on the rising edge of the shift clock after the serial data of the eighth bit is input or output.
9.7 States in Each Mode during 8-bit Serial I/O Operation

This section describes the operation of the 8-bit serial I/O when the device changes to sleep or stop or watch mode or an operation halt request occurs during operation.

■ Using Internal Shift Clock
  - Operation in sleep mode
    In sleep mode, serial I/O operation does not halt but transfer continues, as shown in Figure 9.7a.

  ![Figure 9.7a Operation in Sleep Mode (Internal Shift Clock)](image)

  ■ Operation in stop and watch modes
    In stop or watch mode, serial I/O operation halts and transfer aborts, as shown in Figure 9.7b. As operation restarts after wake-up from stop or watch mode, initialize the 8-bit serial I/O depending on the state of the device with the 8-bit serial I/O is communicating..

  ![Figure 9.7b Operation in Stop and Watch Mode Halt (Internal Shift Clock)](image)

  ■ Operation during halt
    Halting operation during transfer (SMR: SST = “0”) halts the transfer and clears the shift clock counter, as shown in Figure 9.7c. Therefore, the device being communicated with must also be initialized. In serial output operation, set data to the SDR register again before reactivating.

  ![Figure 9.7c Operation during Halt (Internal Shift Clock)](image)

■ Using External Shift Clock
  - Operation in sleep mode
    In sleep mode, serial I/O operation does not halt but transfer continues, as shown in Figure 9.7d
Operation in stop and watch modes

In stop or watch mode, serial I/O operation halts and transfer aborts, as shown in Figure 9.7e. As operation restarts after wake-up from stop or watch mode. This causes an error to occur on the device with which the 8-bit serial I/O is communicating. Initialize the 8-bit serial I/O after wake-up from stop mode.

Operation during halt

Halting operation during transfer (SMR: SST = “0”) halts the transfer and clears the shift clock counter, as shown in Figure 9.7f. Therefore, the device being communicated with must also be initialized. In serial output operation, set the SDR register again before reactivating. If an external clock is input at this time, the SO pin output changes.
9.8 Notes on Using 8-bit Serial I/O

This section lists points to note using when the 8-bit serial I/O.

- **Notes on Using 8-Bit Serial I/O**
  - **Error on starting serial transfer**
    Activating the serial transfer by software (SMR: SST = “1”) is not synchronized with the falling edge (output) or rising edge (input) of the shift clock, there is a delay of up to one cycle of the selected shift clock before the first serial data I/O occurs.
  - **Malfunction due to noise**
    In serial data transfer, malfunction of the serial I/O may occur if unwanted pulses (pulses exceeding the hysteresis width) occur on the shift clock due to external noise.
  - **Notes on setting by program**
    - Write to the serial mode register (SMR) and serial input register (SDR) when serial I/O is stopped (SMR: SST = “0”).
    - Do not modify other SMR register bits when starting/enabling serial I/O transfer (SMR: SST = “1”).
    - When using an external shift clock and when serial data output is enabled (SMR: SOE = “1”), the output level on the SO pin when the external shift clock is the most significant bit (when MSB first is selected) or least significant bit (when LSB first is selected). This applies even if serial transfer is stopped (SMR: SST = “0”).
    - The interrupt request flag bit (SMR: SIOF) is not set if serial I/O transfer is stopped (SMR: SST = “0”) at the same time as serial transfer data completes.
    - Interrupt processing cannot return if the SIOF bit is “1” and the interrupt requests enable bit is enabled (SIOE = “1”). Always clear the SIOF bit.
  - **Transfer Speed of serial I/O**
    The serial data output pin (SO) for serial I/O is an N-ch open-drain output, and is not suitable for high-speed transfer. Care is required when using serial I/O with a high speed shift clock.
  - **Idle state of shift clock**
    Hold the external shift clock at the “H” level during the delay time between transfers of 8-bit data (idle state). When set as the shift clock output (SMR: SCKE = “1”), the internal shift clock (SMR2: CKS1, CKS0 = other than “11”) output an “H” level during the idle state.

Figure 9.8a shows the idle state of the shift clock.

![Figure 9.8a Idle State of Shift Clock](image)
9.9 Connection Example for 8-bit Serial I/O

This section shows an example of connecting together two MB89160/160A/160L Series 8-bit serial I/O and performing bidirectional serial I/O.

Bi Directional Serial I/O Performing

![Connection Diagram](image)

Figure 9.9a Connection Example for 8-bit Serial I/O (Interface between Two MB89160/160A/160L)

Figure 9.9b Operation of Bi Directional Serial I/O

SST: The SST bit is the serial I/O transfer start bit in the serial mode register (SMR).

1. If the SO, SI, and SCK pins only are connected, there is no direct method of confirming whether SIO-B has enabled serial transfer. Therefore, SIO-A must use a software timer or similar to delay time for a sufficient time for SIO-B to enable serial transfer.

2. Data is not transferred correctly if SIO-A starts data transfer when SIO-B has not enabled serial transfer.

3. An interrupt request is generated after 8-bit data have been transferred.
9.10 Program Example for 8-bit Serial I/O

This section gives program example for 8-bit serial I/O.

### Program Example of Serial Output

#### Processing description

- Outputs 8 bits of serial data (55H) from the SO pin of serial I/O, then generates an interrupt when transfer is completed.
- The interrupt processing routine resets the transfer data and continues output.
- Operates as an internal shift clock and outputs the shift clock from the SCK pin.
- With a main clock master oscillator $F_{CH}$ of 4.2 MHz, the highest speed clock selected by the speed-shift function (1 instruction cycle = $4/F_{CH}$), and a 32 $t_{inst}$ shift clock, the data transfer rate will be as follows.

  Transfer speed = $4.2 \text{ MHz} / 4 / 32 = 32.8 \text{ kbps}$,

  Interrupt cycle = $8 \times 32 \times 4 / 4.2 \text{ MHz} = 243.8 \mu s$.

#### Coding example

```assembly
SMR EQU 001CH ; Serial mode register
SDR EQU 001DH ; Serial data register
SIOF EQU SMR:7 ; Define the interrupt request flag bit.
SST EQU SMR:0 ; Define the serial I/O transfer start bit.
ILR2 EQU 007DH ; Address of the interrupt level setting register
INT_V DSEG ABS ; [DATA SEGMENT]
ORG 0FFEEH
IRQ6 DW WARI ; Set interrupt vector.
INT_V ENDS
CSEG ; [CODE SEGMENT]
; Stack pointer (SP) etc. are already initialized.
CLR ; Disable interrupts.
CLRB SST ; Stop serial I/O transfer.
MOV ILR2,#11011111B ; 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
; output (SCK), enable serial data output (SO), select
; 32 $t_{inst}$, LSB first.
SETB SST ; Start serial I/O transfer.
SETI ; Enable interrupts.

-----Main program-----------------------------------------------

-----User processing-----------------------------------------------

-----Interrupt processing routine-----------------------------------------------

-----END-----------------------------------------------
```

208  CHAPTER 9  8-BIT SERIAL I/O  MB89160/160A/160L Series
Program Example for Serial Input

Processing description

- Inputs 8 bits of serial data from the SI pin of serial I/O, then generates an interrupt when transfer is completed.
- The interrupt processing routine reads the transferred data and continues input.
- Serial I/O uses the external shift clock. The shift clock is input from the SCK pin.

Coding example

```assembly
; Program Example for Serial Input

; Processing description

; Inputs 8 bits of serial data from the SI pin of serial I/O, then generates an interrupt when transfer is completed.
; The interrupt processing routine reads the transferred data and continues input.
; Serial I/O uses the external shift clock. The shift clock is input from the SCK pin.

; Coding example

DDR2 EQU 0005H ; Serial mode register
SMR EQU 001CH ; Serial data register
SDR EQU 001DH ; Serial data register
SIOF EQU SMR:7 ; Define the interrupt request flag bit.
SST EQU SMR:0 ; Define the serial I/O transfer start bit.
ILR2 EQU 007DH ; Address of the interrupt level setting register 2

; DATA SEGMENT

DATA SEGMENT
ORG 0FFEEH
IRQ6 DW WARI ; Set interrupt vector.
INT_V ENDS

; CODE SEGMENT

CSEG ; Stack pointer (SP) etc. are already initialized.

MOV DDR2,#00000000B ; Set P25/SCK and P23/SI pin as an input.
CLRI ; Disable interrupts.
CLRB SST ; Stop serial I/O transfer.
MOV ILR2,#11011111B ; Set interrupt level (level 1)
MOV SMR,#01001100B ; Clear interrupt request flag, enable interrupt request output, set shift clock input SCK, disable serial data output (SO) select the external shift clock, LSB first.
SETB SST ; Enable serial I/O transfer.
SETI ; Enable interrupts.

; DATA SEGMENT

WARI CLRB SIOF ; Clear interrupt request flag.
PUSHW A
XCHW A,T
PUSHW A
MOV A,SDR ; Read transfer data.
SETB SST ; Enable serial I/O transfer.

; User processing

PEND

; END

END
```

Main program

; Stack pointer (SP) etc. are already initialized.

MOV DDR2,#00000000B ; Set P25/SCK and P23/SI pin as an input.
CLRI ; Disable interrupts.
CLRB SST ; Stop serial I/O transfer.
MOV ILR2,#11011111B ; Set interrupt level (level 1)
MOV SMR,#01001100B ; Clear interrupt request flag, enable interrupt request output, set shift clock input SCK, disable serial data output (SO) select the external shift clock, LSB first.
SETB SST ; Enable serial I/O transfer.
SETI ; Enable interrupts.

; DATA SEGMENT

WARI CLRB SIOF ; Clear interrupt request flag.
PUSHW A
XCHW A,T
PUSHW A
MOV A,SDR ; Read transfer data.
SETB SST ; Enable serial I/O transfer.

; User processing

PEND

; END
This chapter describes the functions and operation of the buzzer output.

10.1 Overview of Buzzer Output ........................................ 212
10.2 Block Diagram of Buzzer Output ............................... 213
10.3 Structure of Buzzer Output ........................................ 214
10.4 Buzzer Register (BZCR) ............................................ 215
10.5 Program Example for Buzzer Output ......................... 216
10.1 Overview of Buzzer Output

The buzzer output can select from seven different output frequencies (square waves) and can be used for applications such as sounding a buzzer to confirm key input. The function uses the same output pin as the remote control transmit output.

Buzzer Output Function

The buzzer output function outputs a signal (square wave) suitable for applications such as sounding a buzzer to confirm an operation.

- For buzzer output, one of seven output frequencies can be selected, or the output disabled.
- Four divide-by-n outputs are supplied from the timebase timer and three from the watch prescaler, for selection as the buzzer output signal.

Note: Since divided outputs of the timebase timer and timeclock prescaler are fed as the buzzer output signal, the buzzer output will be affected when the signal source selected for it (timebase timer or watch prescaler) is cleared.

Check: Since the timebase timer stops when the main clock oscillator stops (during subclock mode), do not select the divided output of the timebase timer as the buzzer output when subclock mode is used.

Similarly, do not select the watch prescaler as the buzzer source in a chip in which the single clock option is selected.

Table 10.1a lists the seven output frequencies (square waves) that can be selected for the buzzer output function.

<table>
<thead>
<tr>
<th>Clock supply source</th>
<th>Buzzer output cycle</th>
<th>Square wave output (Hz)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Timebase Timer</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( F_{CH}/2^2 )</td>
<td></td>
<td>( F_{CH}/2^3 ) (1.025 kHz)</td>
</tr>
<tr>
<td>( F_{CH}/2^3 )</td>
<td></td>
<td>( F_{CH}/2^4 ) (2.051 kHz)</td>
</tr>
<tr>
<td>( F_{CH}/2^4 )</td>
<td></td>
<td>( F_{CH}/2^5 ) (4.102 kHz)</td>
</tr>
<tr>
<td>( F_{CH}/2^5 )</td>
<td></td>
<td>( F_{CH}/2^6 ) (8.203 kHz)</td>
</tr>
<tr>
<td>Watch Prescaler</td>
<td></td>
<td></td>
</tr>
<tr>
<td>( F_{CL}/2^2 )</td>
<td></td>
<td>( F_{CL}/2^3 ) (1.024 kHz)</td>
</tr>
<tr>
<td>( F_{CL}/2^3 )</td>
<td></td>
<td>( F_{CL}/2^4 ) (2.048 kHz)</td>
</tr>
<tr>
<td>( F_{CL}/2^4 )</td>
<td></td>
<td>( F_{CL}/2^5 ) (4.096 kHz)</td>
</tr>
</tbody>
</table>

**Note:** For a 4.2 MHz main clock source oscillation and if the buzzer register (BZCR) selects a timebase timer divided output of \( F_{CH}/2^{10} \) (Bz2, Bz1, Bz0 = 011b), the output frequency of the BZ pin is calculated as follows:

\[
\text{Output frequency} = \frac{F_{CH}}{2^{10}} = \frac{4.2 \text{ MHz}}{1024} = 4.102 \text{ kHz}
\]
10.2 Block Diagram of Buzzer Output

The buzzer output consists of the following two blocks:
- Buzzer output selector
- Buzzer register (BZCR)

### Block Diagram of Buzzer Output

**Figure 10.2a Block Diagram of Buzzer Output**

- **Buzzer output selector**
  Selects one of the four frequencies output from the timebase timer or three frequencies output from the watch prescaler.

- **BZCR register**
  The BZCR register to set the buzzer output frequency and enable buzzer output.
  Buzzer output is enabled if an output frequency is specified (other than “0000”) in the BZCR register.
10.3 Structure of Buzzer Output

This section describes the pin, pin block diagram, and register of the buzzer output.

- **Buzzer Output Pin**

  The buzzer output uses the P30/BZ/RCO pin. This pin can function either as an output-only port (P30) or as the remote control transmit output pin (RCO) and the buzzer output pin (BZ).

  **BZ:** This pin outputs a buzzer square wave with the specified frequency.

  Setting a buzzer output frequency in the buzzer output register (BZCR: BZ1, BZ0 = other than "00B") automatically sets the P30/BZ pin as the BZ pin regardless of the output latch value or remote control transmit frequency generator circuit settings.

- **Block Diagram of Buzzer Output Pin**

  ![Figure 10.3a Block Diagram of P30/BZ/RCO Pin](image)

  SPL: Pin state specification bit in the standby control register (STBC)

- **Buzzer Output Register**

  ![Figure 10.3b Buzzer Output Register](image)

  BZCR (Buzzer Register) Address: 0010H

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<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>
</tr>
<tr>
<td>BZ2</td>
<td>BZ1</td>
<td>BZ0</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
</tr>
<tr>
<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>
</tbody>
</table>

  R/W : Readable and writable

  — : Unused

  X : Indeterminate
10.4 Buzzer Register (BZCR)

The buzzer register (BZCR) is used to select the buzzer output frequency and also enables buzzer output.

---

### Buzzer Register (BZCR)

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0010H</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>BZ2</td>
<td>BZ1</td>
<td>BZ0</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>BZ2</th>
<th>BZ1</th>
<th>BZ0</th>
<th>Buzzer selection bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>Disables buzzer output. Makes pin available for use as a general-purpose port (P30).</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>Functions as a buzzer output pin (BZ).</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>Timebase timer outputs: F(CH)/2^9</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>F(CH)/2^8</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>0</td>
<td>Watch prescaler outputs: F(CL)/2^5</td>
</tr>
<tr>
<td>1</td>
<td>0</td>
<td>1</td>
<td>F(CL)/2^4</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>F(CL)/2^3</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>F(CL)/2^2</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>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>The read value is indeterminate.</td>
</tr>
<tr>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>Writing to these bits has no effect on the operation.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
<th>BZ2, BZ1, BZ0: Buzzer selection bits</th>
</tr>
</thead>
<tbody>
<tr>
<td>—</td>
<td>—</td>
<td>—</td>
<td>These bits selects buzzer output and enable output.</td>
</tr>
<tr>
<td>—</td>
<td>—</td>
<td>—</td>
<td>Setting &quot;000B&quot; disables the buzzer output and sets the pin as a general-purpose port (P30). However, if setting as the remote control transmit frequency output pin (RCO), that setting has precedence. Setting other than &quot;000B&quot; sets the pin as the buzzer output (BZ) pin and outputs a square wave of the selected frequency.</td>
</tr>
<tr>
<td>—</td>
<td>—</td>
<td>—</td>
<td>Selects one of four divided outputs from the timebase timer or three from the timeclock prescaler.</td>
</tr>
</tbody>
</table>

**Check:** When using this pin as the BUZ pin, set the RCO register (RCR2) to "00H" to disable the RCO output.

**Note:** Do not select a timebase timer division output in subclock mode. The subclock oscillator operates in the main-stop mode. Therefore, if the pin state specification bit (STBC: SPL) is "0," the buzzer output can be used even in main-stop mode by selecting one of the watch prescaler divided-by-\(n\) outputs (BZ2, BZ1, BZ0 = 101B to 111B).
This section gives a program example for the buzzer output.

- **Program Example for Buzzer Output**
  - **Processing description**
    - Output a buzzer output of approximately 1.025 kHz to the BZ pin, then turn the buzzer output “OFF”.
    - For a 4.2 MHz main clock source oscillation and selecting $2^{12}/F_{CH}$ ($F_{CH}$: main clock oscillation), the buzzer output frequency is as follows:
      
      $\text{Buzzer output frequency} = \frac{4.2 \text{ MHz}}{2^{12}}$
      
      $= \frac{4.2 \text{ MHz}}{4096}$
      
      $= 1.025 \text{ kHz}$
  
  - **Coding example**
    
    BZCR EQU 0010H ; Buzzer register
    CSEG ; [CODE SEGMENT]
    BUZON MOV BZCR,#00000001B ; Buzzer output “ON”.
    BUZOFF MOV BZCR,#00000000B ; Buzzer output “OFF”.
    ENDS
    END
CHAPTER 11
EXTERNAL INTERRUPT CIRCUIT 1 (EDGE)

This chapter describes the functions and operation of the external interrupt circuit 1 (edge).

11.1 Overview of External Interrupt Circuit 1...................... 218
11.2 Block Diagram of External Interrupt Circuit 1 ............. 219
11.3 Structure of External Interrupt Circuit 1 ...................... 220
11.4 External Interrupt Circuit 1 Interrupts ......................... 224
11.5 Operation of External Interrupt Circuit 1..................... 225
11.6 Program Example for External Interrupt Circuit 1....... 226
11.1 Overview of External Interrupt Circuit 1

The external interrupt circuit 1 detects edges on the signals input to the four external interrupt pins and generates the corresponding interrupt requests to the CPU.

- **External Interrupt Circuit 1 Function (Edge Detection)**

  The external interrupt circuit 1 function detects specified edges on signals input to the external interrupt pins and to generate interrupt requests to the CPU. These interrupts can wake up the CPU from standby mode and change the device to the normal operating state (main-run or sub-run mode).

  - **External interrupt pins:** 4 pins (P10/INT10 to P13/INT13)
  - **External interrupt source:** Inputs a specified edge (rising edge or falling edge) on the signal input to an external interrupt pin.
  - **Interrupt control:** Enable or disable to input external interrupts and to output in interrupt requests, by the external interrupt 1 control register (EIE1).
  - **Interrupt flags:** Detects specified edges by the external interrupt request flag bits in the external interrupt 1 flag register (EIF1).
  - **Interrupt requests:** Separate interrupt requests are generated for each external interrupt source (IRQ0, IRQ1, IRQ2, and IRQ3).
11.2 Block Diagram of External Interrupt Circuit 1

The external interrupt circuit 1 consists of the following three blocks:
- Edge detector
- External interrupt 1 control register (EIE1)
- External interrupt 1 flag register (EIF1)

**Edge detector circuit**

The signals received at the external interrupt pins (INT10 to INT13) are either inverted or not inverted, based on the state of the corresponding signal inversion (SIV) bit of the EIE1 register. If external interrupt inputs are enabled, a falling edge is detected in one of the inverted or non-inverted signals, and the interrupt request flag bit for that INT pin (IF10 to IF13) is set to “1.”

**EIE1 register**

The states of the signal inversion bits (SIV0 to SIV3) of this register determine whether the corresponding external interrupt input signal will be inverted. The external interrupt enable bits (IE10 to IE13) simultaneously enable or disable both the interrupt inputs and the interrupt request outputs.

**EIF1 register**

The external interrupt request flag bits (IF10 to IF13) of this register are used to check interrupt request status and clear the interrupt requests.
11.3 Structure of External Interrupt Circuit 1

This section describes the pins, pin block diagram, registers, and interrupt sources of the external interrupt circuit 1.

External Interrupt Circuit 1 Pins

The external interrupt circuit 1 uses four external interrupt pins. The external interrupt pins can function either as external interrupt inputs (hysteresis inputs) or as general-purpose I/O ports.

When P10/INT10 to P13/INT13 Pins are set as inputs in their port 1 data direction register (DDR1), and the corresponding external interrupt inputs are enabled in the external interrupt 1 control register (EIE1) they operate as external interrupt input pins (INT10 to INT13). When they are being used as the port, the pin states, can be read from the port data register (PDR1) at any time.

Table 11.3a lists the pins associated with external interrupt circuit 1.

<table>
<thead>
<tr>
<th>External interrupt pin</th>
<th>When used as external interrupt input (interrupt input enabled)</th>
<th>When used as general-purpose I/O port (interrupt input disabled)</th>
</tr>
</thead>
<tbody>
<tr>
<td>P10/INT10</td>
<td>INT10 (EIE1:IE10=1, DDR1:bit0=0)</td>
<td>P10 (EIE1:IE10=0)</td>
</tr>
<tr>
<td>P11/INT11</td>
<td>INT11 (EIE1:IE11=1, DDR1:bit1=0)</td>
<td>P11 (EIE1:IE11=0)</td>
</tr>
<tr>
<td>P12/INT12</td>
<td>INT12 (EIE1:IE12=1, DDR1:bit2=0)</td>
<td>P12 (EIE1:IE12=0)</td>
</tr>
<tr>
<td>P13/INT13</td>
<td>INT13 (EIE1:IE13=1, DDR1:bit3=0)</td>
<td>P13 (EIE1:IE13=0)</td>
</tr>
</tbody>
</table>

Block Diagram of External Interrupt Circuit 1 Pins

![Figure 11.3a Block Diagram of External Interrupt Circuit 1 Pins](image)

Note: Pins with a pull-up register (optional) go to the "H" level during a reset or in stop and watch mode (SPL = 1).
### External Interrupt Circuit 1 Registers

<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>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0030H</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>00000000</td>
</tr>
</tbody>
</table>

**EIE1** (External interrupt 1 control register)

<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>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0031H</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>IF13</td>
<td>IF12</td>
<td>IF11</td>
<td>XXXX0000</td>
</tr>
</tbody>
</table>

**EIF1** (External interrupt 1 flag register)

<table>
<thead>
<tr>
<th></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>
<th>R/W</th>
<th>R/W</th>
</tr>
</thead>
</table>

R/W: Readable and writable
—: Unused
X: Indeterminate

---

### External Interrupt Circuit 1 Interrupt Sources

**IRQ0**: External interrupt circuit generates an interrupt request (IRQ0) if an edge of the selected polarity is input to the external interrupt pin (INT0) when external interrupt is enabled (EIC1: EIE0 = “1”).

**IRQ1**: External interrupt circuit generates an interrupt request (IRQ1) if an edge of the selected polarity is input to the external interrupt pin (INT1) when external interrupt is enabled (EIC1: EIE1 = “1”).

**IRQ2**: External interrupt circuit generates an interrupt request (IRQ2) if an edge of the selected polarity is input to the external interrupt pin (INT2) when external interrupt is enabled (EIC2: EIE2 = “1”).

**IRQ3**: External interrupt circuit generates an interrupt request (IRQ3) if an edge of the selected polarity is input to the external interrupt pin (INT3) when external interrupt is enabled (EIC2: EIE3 = “1”).
11.3 Structure of External Interrupt Circuit 1

11.3.1 External Interrupt 1 Control Register (EIE1)

The external interrupt 1 control register (EIE1) is used to select the inversion or non-inversion of interrupt input signal and to select enable or disable interrupts for external interrupt pins (INT10 to INT13).

External Interrupt 1 Control Register (EIE1)

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0030H</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>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>

### External Interrupt Enable Bits

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>Disables external interrupt input and interrupt request output</td>
</tr>
<tr>
<td>1</td>
<td>Enables external interrupt input and interrupt request output</td>
</tr>
</tbody>
</table>

### External Interrupt Input Signal Inversion Bits

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>External interrupt input signal not inverted (detect falling edge at pin).</td>
</tr>
<tr>
<td>1</td>
<td>External interrupt input signal inverted (detect rising edge at pin).</td>
</tr>
</tbody>
</table>

**Table 11.3.1a External Interrupt 1 Control Register (EIE1) Bits vs. Interrupts Pins**

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 3</th>
<th>Bit 6</th>
<th>Bit 2</th>
<th>Bit 5</th>
<th>Bit 1</th>
<th>Bit 0</th>
<th>External interrupt pin</th>
<th>Interrupt</th>
</tr>
</thead>
<tbody>
<tr>
<td>SIV3</td>
<td>IE13</td>
<td>SIV2</td>
<td>IE12</td>
<td>SIV1</td>
<td>IE11</td>
<td>IE10</td>
<td>INT13</td>
<td>IRQ3</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>INT12</td>
<td>IRQ2</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>INT11</td>
<td>IRQ1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>INT10</td>
<td>IRQ0</td>
</tr>
</tbody>
</table>

**Table 11.3.1b External Interrupt 1 Control Register (EIE1) Bits**

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
<th>SIV3 to SIV0: External Interrupt Input Signal Inversion Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• Select inversion/non-inversion of signal input to external interrupt pin.</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• A value of “0” has the effect of selecting falling edge detection for that pin, and a “1” selects rising edge detection.</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 3</th>
<th>Bit 2</th>
<th>Bit 1</th>
<th>Bit 0</th>
<th>IE13 to IE10: External Interrupt Enable Bits</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Enables or disables both the external interrupt input, and the interrupt request output to the CPU. An interrupt request output when both this bit and the corresponding external interrupt request flag bit (IF13 to IF10) are “1.”</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Note:</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• When using an external interrupt pin, set it as an input by writing its bit in the port 1 data direction register (DDR1) to “0.”</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• The state of the external interrupt pin can always be read directly out of the port 1 data register (PDR1) regardless of the state of this interrupt enable bit.</td>
</tr>
</tbody>
</table>

**Note:**
- When using an external interrupt pin, set it as an input by writing its bit in the port 1 data direction register (DDR1) to “0.”
- The state of the external interrupt pin can always be read directly out of the port 1 data register (PDR1) regardless of the state of this interrupt enable bit.
11.3 Structure of External Interrupt Circuit 1

11.3.2 External Interrupt 1 Flag Register ({EIF1})

External Interrupt 1 flag register ({EIF1}) is used to hold the IRQ state when an interrupt edge has been detected, and to clear the interrupt.

### External Interrupt 1 Flag Register ({EIF1})

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0031H</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>IF13</td>
<td>IF12</td>
<td>IF11</td>
<td>IF10</td>
<td>XXXX0000B</td>
</tr>
</tbody>
</table>

**R/W : Readable and writable**

— : Unused

X : Indeterminate

---

**Table 11.3.2a External Interrupt 1 Flag Register ({EIF1}) Bits**

<table>
<thead>
<tr>
<th>Bit</th>
<th>External interrupt pin</th>
<th>Interrupt</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 3</td>
<td>IF13</td>
<td>INT13</td>
</tr>
<tr>
<td>Bit 2</td>
<td>IF12</td>
<td>INT12</td>
</tr>
<tr>
<td>Bit 1</td>
<td>IF11</td>
<td>INT11</td>
</tr>
<tr>
<td>Bit 0</td>
<td>IF10</td>
<td>INT10</td>
</tr>
</tbody>
</table>

**Table 11.3.2b External Interrupt 1 Flag Register ({EIF1}) Bits**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7, Bit 6, Bit 5, Bit 4</td>
<td>Unused bits: This read value is indeterminate. Writing to this bit has no effect on the operation.</td>
</tr>
<tr>
<td>Bit 3, Bit 2, Bit 1, Bit 0</td>
<td>IF13 to IF10: External interrupt request flag bits</td>
</tr>
<tr>
<td>Bit 3</td>
<td>IF13 to IF10, External interrupt request flag bits</td>
</tr>
<tr>
<td>Bit 2, Bit 1, Bit 0</td>
<td>When a rising or falling edge, as specified by the state of the external interrupt in put signal inversion bit (SIV3 to SIV0), the corresponding IRQ flag bit is set to “1.” When both this bit and the corresponding interrupt enable bit (EIE1: IE13 to IE10) are set to “1,” that IRQ is sent to the CPU. Writing “0” clears this bit. Writing “1” has no effect and does not change the bit value. Note: When the external interrupt enable bits (EIE1: IE13 to IE10) are cleared to “0,” the external interrupt inputs are disabled at the same time, which means that in this state, these bits will not be changed even when the specified edge is input.</td>
</tr>
</tbody>
</table>

---

Figure 11.3.2a External Interrupt 1 Flag Register ({EIF1})
11.4 External Interrupt Circuit 1 Interrupts

The external interrupt circuit 1 can generate interrupt requests when it detects a specified edge on the signal input to an external interrupt pin.

Interrupts for External Interrupt Circuit 1 Operation

If external interrupts are enabled (EIE1: IE10 to IE30 = 1) and the specified edge is detected at the external interrupt input, the corresponding external IRQ flag bit (EIF1: IF10 to IF13) is set to “1”, and the corresponding IRQ (IRQ0 to IRQ3) sent to the CPU. Write “0” to the corresponding external interrupt request flag bit in the interrupt processing routine to clear the interrupt request.

Check: When enabling interrupts (EIE1: IE10 to IE13 = “1”) after wake-up from a reset, always clear the corresponding external interrupt request flag bit (EIE1: IF10 to IF13 = “0”) in advance.

Also, interrupt processing cannot return if the external interrupt request flag bit is “1” and the interrupt request enable bit is enabled. In the interrupt processing routine, always clear the external interrupt request flag bit.

Notes:
- Changing a signal inversion bit from the “non-invert” to the “invert” state while the INT pin is “H”, or from “invert” to “non-invert” while the pin is LOW, will cause the external interrupt request flag bit (EIF1: IF10 to IF13) to be set immediately. Changing an external interrupt bit from “disable” to “enable” (EIE1: IE10 to IE13: 0 → 1) may also set the external IRQ flag bit. For this reason, you should make the inversion or enable bit changes with interrupts in the disabled state, then clear IRQ flags before enabling interrupts again.
- An interrupt request is generated immediately if the external interrupt request flag bit is “1” when the external interrupt enable bit is changed from disabled to enabled (“0” → “1”).
- Wake-up from stop mode by an interrupt is possible using only the external interrupt circuit 1 and 2.
- Perform with interrupts disabled, then clear external IRQ flags before enabling interrupts.

Register and Vector Table for External Interrupt Circuit 1 Interrupts

<table>
<thead>
<tr>
<th>IRQ</th>
<th>Interrupt level setting register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>Register</td>
<td>Setting bits</td>
</tr>
<tr>
<td>IRQ0</td>
<td>ILR1 (007CH)</td>
<td>L01 (bit1)</td>
</tr>
<tr>
<td>IRQ1</td>
<td></td>
<td>L11 (bit3)</td>
</tr>
<tr>
<td>IRQ2</td>
<td></td>
<td>L21 (bit5)</td>
</tr>
<tr>
<td>IRQ3</td>
<td></td>
<td>L31 (bit7)</td>
</tr>
</tbody>
</table>

Reference: See section 3.4.2, “Interrupt Processing” for details on the interrupt operation.
11.5 Operation of External Interrupt Circuit 1

The external interrupt circuit 1 sends an interrupt request to the CPU when it detects a specified edge at one of its external interrupt pins.

### Operation of External Interrupt Circuit 1

Figure 11.5a shows the settings required to operate the external interrupt circuit 1.

#### Figure 11.5a External Interrupt Circuit 1 Settings

The input signals from the external interrupt pins (INT10 to INT13) are either inverted or not, depending on the state of the applicable external interrupt signal inversion bit (EIE1: SIV0 to SIV3). If the external interrupt enable bit (EIE1: IE10 to IE13) is “1”, the corresponding external interrupt request flag bit (EIF1: IF10 to IF13) will be set to “1” when a falling edge is detected in the inverted/non-inverted signal.

Figure 11.5b shows the external interrupt 1 operation (for signals received at pin INT10).

#### Figure 11.5b Operation of External Interrupt 1 (INT10)

**Note:** The pin state can be read directly from the port data register (PDR1), even when used as an external interrupt pins.
11.6 Program Example for External Interrupt Circuit 1

This section gives a program example for the external interrupt circuit 1.

- Program Example for External Interrupt Circuit 1
  - Processing description
    - Generates interrupts on detecting a falling edge input to the INT0 pin.
  - Coding Example
    
    DDREQU0003H ; Address of the port 1 data direction register
    EIE1EQU0030H ; Address of the external interrupt 1 control register
    EIF1EQU0031H ; Address of the external interrupt 1 flag register
    IE10EQUEIE1:0 ; Define the external interrupt enable bit.
    SIV0EQUEIE1:4 ; Define the external interrupt signal inversion bit.
    IF10EQUEIE1:0 ; Define the external interrupt request flag bit.
    ILR1EQU007CH ; Address of the set interrupt level settings register

    INT_VDSEGABS ; [DATA SEGMENT]
    ORG 0FFFFAH
    IRQ0DW WARI ; Set interrupt vector.
    INT_VENDS

    ;--------Main program-----------------------------------------------
    CSEG ; [CODE SEGMENT]
    : ; Stack pointer (SP) etc. are already initialized.
    CLR1 ; Disable interrupts.
    MOV ILR1,#11111110B ; Set interrupt priority to level 2.
    MOV DDR0,#00000000B ; Set P10/INT10 pin as input.
    CLRB SIV0 ; Select falling edge.
    SETB IE10 ; Enable INT10 interrupt input.
    CLRB IF10 ; Clear external interrupt request flag.
    SETI ; Enable interrupts.
    :

    ;--------Interrupt processing routine ---------------------------------
    WARI CLR1BIF10 ; Clear external interrupt request flag.
    PUSHWA
    XCHWA,T
    PUSHWA
    : ; User processing
    :
    POPWA
    XCHWA,T
    POPWA
    RETI

    ENDS

    ;-----------------------------------------------------------------------
    END

---
This chapter describes the functions and operation of the external interrupt circuit 2 (level).

12.1 Overview of External Interrupt Circuit 2...................... 228
12.2 Block Diagram of External Interrupt Circuit 2 ............. 229
12.3 Structure of External Interrupt Circuit 2 ...................... 230
12.4 External Interrupt Circuit 2 Interrupt ......................... 234
12.5 Operation of External Interrupt Circuit 2..................... 235
12.6 Program Example for External Interrupt Circuit 2....... 236
12.1 Overview of External Interrupt Circuit 2

The external interrupt circuit 2 detects the level of the signals input to the eight external interrupt pins and generates the interrupt requests to the CPU.

- External Interrupt Circuit 2 Function (Level Detection)

  The external interrupt circuit 2 function detects the signals of the “L” levels input to the external interrupt pins and to generate interrupt request to the CPU. These interrupts can wake up the CPU from standby mode and change the device to the normal operating state (main-run or sub-run mode).

  - External interrupt pins: 8 pins (P00/INT20 to P07/INT27)
  - External interrupt sources: “L” level signal input to an external interrupt pin.
  - Interrupt control: Enables or disables to input external interrupt controlled by external interrupt 2 control register (EIE2)
  - Interrupt flag: IRQ flag bit of external interrupt 2 flag register (EIF2). Flag set when there is an IRQ.
  - Interrupt request: IRQ4 is generated if any enabled external interrupt pin goes LOW.
12.2 Block Diagram of External Interrupt Circuit 2

The external interrupt circuit 2 consists of the following three blocks:

- Interrupt request generator
- External interrupt 2 control register (EIE2)
- External interrupt 2 flag register (EIF2)

Interrupt request generator

The interrupt request generator generates CPU interrupt requests based on signals input at external interrupt pins (INT20 to INT27) and the external interrupt enable bits.

EIE2 register

External interrupt input enable bits (IE20 to IE27) enable/disable “L” level signals input at the corresponding external interrupt input pins.

EIF2 register

The interrupt request flag bit of this register (IF20) is used to hold (and clear) interrupt request signals.
12.3 Structure of External Interrupt Circuit 2

This section describes the pins, pin block diagram, registers, and interrupt sources of the external interrupt circuit 2.

■ External Interrupt Circuit 2 Pins

The external interrupt circuit 2 uses eight external interrupt pins. The external interrupt pins can function either as external interrupt inputs (hysteresis inputs) or as general-purpose I/O ports.

When P00/INT20 to P07/INT27 pins are set as inputs in the port 0 data direction register (DDR0), and the corresponding external interrupt inputs are enabled in the external interrupt 2 control register (EIE2) they operate as external interrupt input pins (INT20 to INT27). When they are being used as the input port, the pin states can be read from the port data register (PDR0) at any time.

Table 12.3a lists the external interrupt circuit 2 pins.

<table>
<thead>
<tr>
<th>External interrupt pin</th>
<th>When used as an external interrupt input (Interrupt Input Enabled)</th>
<th>When used as General-purpose I/O port (Interrupt Input Disabled)</th>
</tr>
</thead>
<tbody>
<tr>
<td>P00/INT20</td>
<td>INT20 (EIE2:IE20=1, DDR0:bit0=0)</td>
<td>P00 (EIE2:IE20=0)</td>
</tr>
<tr>
<td>P01/INT21</td>
<td>INT21 (EIE2:IE21=1, DDR1:bit1=0)</td>
<td>P01 (EIE2:IE21=0)</td>
</tr>
<tr>
<td>P02/INT22</td>
<td>INT22 (EIE2:IE22=1, DDR2:bit2=0)</td>
<td>P02 (EIE2:IE22=0)</td>
</tr>
<tr>
<td>P03/INT23</td>
<td>INT23 (EIE2:IE23=1, DDR3:bit3=0)</td>
<td>P03 (EIE2:IE23=0)</td>
</tr>
<tr>
<td>P04/INT24</td>
<td>INT24 (EIE2:IE24=1, DDR4:bit4=0)</td>
<td>P04 (EIE2:IE24=0)</td>
</tr>
<tr>
<td>P05/INT25</td>
<td>INT25 (EIE2:IE25=1, DDR5:bit5=0)</td>
<td>P05 (EIE2:IE25=0)</td>
</tr>
<tr>
<td>P06/INT26</td>
<td>INT26 (EIE2:IE26=1, DDR6:bit6=0)</td>
<td>P06 (EIE2:IE26=0)</td>
</tr>
<tr>
<td>P07/INT27</td>
<td>INT27 (EIE2:IE27=1, DDR7:bit7=0)</td>
<td>P07 (EIE2:IE27=0)</td>
</tr>
</tbody>
</table>
- **Block Diagram of External Interrupt Circuit 2 Pins**

![Diagram of External Interrupt Circuit 2 Pins]

**Note:** Pins with a pull-up resistor (optional) go to the “H” level during a reset or in stop and watch modes (SPL = “1”).

- **External Interrupt Circuit 2 Registers**

  **EIE2 (External interrupt 2 control register)**
  
<table>
<thead>
<tr>
<th>Address</th>
<th>Bit 0</th>
<th>Bit 1</th>
<th>Bit 2</th>
<th>Bit 3</th>
<th>Bit 4</th>
<th>Bit 5</th>
<th>Bit 6</th>
<th>Bit 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0032H</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
  
  Initial value: 00000000B

  **EIF2 (External interrupt 2 flag register)**
  
<table>
<thead>
<tr>
<th>Address</th>
<th>Bit 0</th>
<th>Bit 1</th>
<th>Bit 2</th>
<th>Bit 3</th>
<th>Bit 4</th>
<th>Bit 5</th>
<th>Bit 6</th>
<th>Bit 7</th>
</tr>
</thead>
<tbody>
<tr>
<td>0033H</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>
  
  Initial value: 00000000B

  R/W: Readable and writable
  X: Unused
  —: Indeterminate

- **External Interrupt Circuit 2 Interrupt Sources**

  **IRQ4:** IRQ4 is generated if any one of external interrupt pins INT20 to INT27 goes to “L” with a “1” in the external interrupt input enable bit for that pin.
12.3  Structure of External Interrupt Circuit 2

12.3.1  External Interrupt 2 Control Register (EIE2)

The external interrupt 2 control register (EIE2) is used to enable/disable input of external interrupt pins (INT20 to INT27).

**External Interrupt 2 Control Register (EIE2)**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Pin</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>IE27</td>
<td>00000000B</td>
</tr>
<tr>
<td>6</td>
<td>IE26</td>
<td>00000000B</td>
</tr>
<tr>
<td>5</td>
<td>IE25</td>
<td>00000000B</td>
</tr>
<tr>
<td>4</td>
<td>IE24</td>
<td>00000000B</td>
</tr>
<tr>
<td>3</td>
<td>IE23</td>
<td>00000000B</td>
</tr>
<tr>
<td>2</td>
<td>IE22</td>
<td>00000000B</td>
</tr>
<tr>
<td>1</td>
<td>IE21</td>
<td>00000000B</td>
</tr>
<tr>
<td>0</td>
<td>IE20</td>
<td>00000000B</td>
</tr>
</tbody>
</table>

Table 12.3.1a  External Interrupt 2 Control Register (EIE2) Bits vs. Pins

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>7</td>
<td>IE27 to IE20: External Interrupt Enable Bits</td>
</tr>
<tr>
<td>6</td>
<td>IE26 to IE21: External Interrupt Enable Bits</td>
</tr>
<tr>
<td>5</td>
<td>IE25 to IE23: External Interrupt Enable Bits</td>
</tr>
<tr>
<td>4</td>
<td>IE24 to IE22: External Interrupt Enable Bits</td>
</tr>
<tr>
<td>3</td>
<td>IE23 to IE22: External Interrupt Enable Bits</td>
</tr>
<tr>
<td>2</td>
<td>IE22 to IE21: External Interrupt Enable Bits</td>
</tr>
<tr>
<td>1</td>
<td>IE21 to IE20: External Interrupt Enable Bits</td>
</tr>
<tr>
<td>0</td>
<td>IE20 to IE27: External Interrupt Enable Bits</td>
</tr>
</tbody>
</table>

R/W : Readable and writable
- : Unused
X : Indeterminate
- : Initial value

Note: When using a pin for external interrupts, set it as an input by writing its bit in the port 0 data direction register (DDR0) to "0." The state of the pin can always be read directly out of the port 0 data register (PDR0) regardless of the state of this external interrupt enable bit.
12.3 Structure of External Interrupt Circuit 2

12.3.2 External Interrupt 2 Flag Register (EIF2)

External Interrupt 2 flag register (EIF2) is used to hold the IRQ state when a level interrupt has been detected, and clear the interrupt.

**External Interrupt 2 Flag Register (EIF2)**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7</td>
<td>Unused Bits</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></td>
</tr>
<tr>
<td>Bit 0</td>
<td>IF20: External interrupt request flag bit</td>
</tr>
</tbody>
</table>

- **Bit 0**: IF20: External interrupt request flag bit
  - **Bit Function**: The read value is indeterminate. Writing to this bit has no effect on the operation.
  - **Note**: Writing “0” to the external interrupt enable bits of the external interrupt 2 control register (EIE2: IE20 to IE27) simply disables the corresponding external interrupt input; it does not clear the interrupt request. IRQ4 will continue to be sent to the CPU until it is cleared by writing “0” to the IF20 bit.
12.4 External Interrupt Circuit 2 Interrupt

The external interrupt circuit 2 interrupt trigger event is the detection of a “L” level at the external interrupt pin.

- Interrupts for External Interrupt Circuit 2 Operation

  If a “L” is detected at an enabled external interrupt pin, the external interrupt request flag bit (EIF2: IF20) is set to “1,” and an interrupt request (IRQ4) to the CPU is generated. Write “0” to the IF20 bit in the interrupt processing routine to clear the interrupt request.

  Once the external interrupt request flag bit (IF20) is set to “1,” IRQ4 continues to be asserted as long as the flag set. Disabling the interrupt input by writing the IE bit (IE20 to IE27) of the EIE2 register to “0” will not clear the interrupt request. Always clear the IF20 bit.

  Also, if the external interrupt pin stays “L”, writing “0” to the IF20 bit without disabling the external interrupt input will not clear the interrupt either, because IF20 will immediately be set again by the “L” pin. After an interrupt request is generated, then, either the input must be disabled, or the external IRQ signal de-asserted.

  Check: When enabling interrupts of CPU after wake-up from a reset, clear the IF20 bit in advance.

  Note: Wake-up from stop mode by an interrupt is possible using only the external interrupt circuit 1 and 2.

- Register and Vector Table for External Interrupt Circuit 2 Interrupts

  Table 12.4a Registers and Vector Table for External Interrupt Circuit 2 Interrupts

<table>
<thead>
<tr>
<th>Interrupt</th>
<th>Interrupt level setting register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRQ4</td>
<td>ILR2 (007DH)</td>
<td>FFF2H</td>
</tr>
<tr>
<td></td>
<td>L41 (Bit 1)</td>
<td>L40 (Bit 0)</td>
</tr>
</tbody>
</table>

Reference: See Section 3.4.2, “Interrupt Processing” for details on the interrupts operation.
12.5 Operation of External Interrupt Circuit 2

The external interrupt circuit 2 sends an interrupt request to the CPU when it detects a “L” at one of its external interrupt pins.

- Operation of External Interrupt Circuit 2

Figure 12.5a shows the settings required to operate the external interrupt circuit 2.

<table>
<thead>
<tr>
<th>EIE2</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></td>
<td>E27</td>
<td>E26</td>
<td>E25</td>
<td>E24</td>
<td>E23</td>
<td>E22</td>
<td>E21</td>
<td>E20</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>EIF2</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>F20</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>DDR0</th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

- Figure 12.5a External Interrupt Circuit 2 Settings

If a “L” is applied to one of the INT20 to INT27 pins with the corresponding external interrupt input enable bit (IE20 to IE27) in the “enable” state, the circuit sends an IRQ4 interrupt request to the CPU.

Figure 12.5b shows external interrupt circuit 2 operation (for a signal received at INT20).

- Figure 12.5b Operation of External Interrupt 2 (INT20)

Note: The pin state can be read directly from the port data register (PDR0) even when the pin is being used as an external interrupt input.
12.6 Program Example for External Interrupt Circuit 2

This section gives a Program example for the external interrupt circuit 2.

- **Program Example for External Interrupt Circuit 2**

  - **Processing description**
    - Generates interrupts on detecting a “L” input the INT20 pin.

  - **Coding example**

    ```
    DDR0EQU0001H ; Address of port 0 data direction register
    EIE2EQU0032H ; Address of external interrupt 2 control register
    EIF2EQU0033H ; Address of external interrupt 2 flag register
    IF20EQUEIF2:0 ; Define the external interrupt request flag bit.
    ILR2EQU007DH ; Address of the set interrupt level setting register 2
    INT_VDSEGABS ; [DATA SEGMENT]
    ORG 0FFF2H
    IRQ4DWWARI ; Set interrupt vector.
    INT_VENDS
    
    ;------------------Main processing-----------------------------------------------
    CSEG ; [CODE SEGMENT]
    ; Stack pointer (SP) etc. are already initialized.
    
    ; Disable interrupts.
    CLR IF20
    ; Clear external interrupt request flag.
    MOV ILR2,#11111110B ; Set interrupt priority (level 2).
    MOV DDR0,#00000000B ; Set P00/INT20 pin as input.
    MOV EIE2,#00000001B ; Enable external interrupt input at INT20 pin.
    SETI ; Enable interrupts.
    
    ;----------------------Interrupt processing routine-----------------------------
    WARIMOVEIE2,#00000000B ; Disable external interrupt input at INT20 pin.
    CLRB IF20 ; Clear external interrupt request flag.
    PUSHWA
    XCHWA,T
    PUSHWA
    
    ; User processing
    
    ; POPWA
    XCHWA,T
    POPWA
    RETI
    ENDS
    
    ;-----------------------------------------------------------------------------
    END
    ```

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

13.1 Overview of A/D Converter ........................................ 238
13.2 Block Diagram of A/D Converter ................................ 240
13.3 Structure of A/D Converter ....................................... 242
13.4 A/D Converter Interrupts .......................................... 249
13.5 Operation of A/D Converter ...................................... 250
13.6 Notes on Using A/D Converter .................................... 252
13.7 Program Example for A/D Converter ............................ 254
13.1 Overview of A/D Converter

The A/D converter can be selected to function either as an 8-bit successive approximation type A/D conversion or as a sense function. The sense function performs a high-speed comparison between the input voltage and a set voltage. Both functions select one input signal from the eight analog input pin channels and can be activated either by software, by an internal clock, or by an 8/16-bit timer output. The sense function can be used in applications such as checking battery capacitance or temperature fluctuation.

■ A/D Conversion Function

The A/D conversion function converts the analog voltage (input voltage) input to an analog input pin to an 8-bit digital value.

- Selects one input from eight analog input pins.
- Conversion speed is 44 instruction cycles (41.9 µs with highest main clock and 4.2 MHz source oscillation) (except MB89160L series)
- Conversion speed for MB89160L series is 52 instruction cycles (49.5 µs with highest main clock and 4.2 MHz source oscillation)
- Generates an interrupt when A/D conversion completes.
- Conversion completion can also be determined by software.

The following methods are available to activate A/D conversion:

- Activation by software
- Continuous activation by a timebase timer output (divide-by-2 main clock source oscillation)
- Continuous activation by an 8/16-bit timer/counter output.

■ Sense Function

The sense function compares the analog voltage (input voltage) input to an analog input pin with the voltage (compare voltage) corresponding to the value set in the A/D data register (ADCD), and determines which voltage is higher or lower.

- Selects one input from eight analog input pins.
- Compare speed is 12 instruction cycles (11.4 µs in main clock mode, with the highest clock speed selected, and a clock oscillator frequency of 4.2 MHz).
- Generates an interrupt when the comparison condition is satisfied.

The following methods are available to activate the sense function:

- Activation by software
- Continuous activation by a timebase timer output (divide-by-2 main clock source oscillation)
- Continuous activation by an 8/16-bit timer/counter output.
13.2 Block Diagram of A/D Converter

The A/D converter consists of the following nine blocks:

- Clock selector (input clock selector for A/D converter activation)
- Analog channel selector
- Sample hold circuit
- D/A converter
- Comparator
- Controller
- A/D data register (ADCD)
- A/D control register 1 (ADC1)
- A/D control register 2 (ADC2)

---

Figure 13.2a Block Diagram of A/D Converter

T01: 8/16-bit timer/counter 16-bit timer output (Timer 1 output)

FCH: Main clock source oscillation
Clock selector
Selects the clock used to activate the A/D conversion or sense function when continuous
activation is enabled (ADC2: EXT = "1").

Analog channel selector
Selects one of the eight analog input channels.

Sample hold circuit
Holds the input voltage selected by the analog channel selector. The circuit samples and holds
the input voltage immediately after the A/D conversion or sense function is activated. This
allows A/D conversion (or comparison) to proceed without being affected by input voltage
fluctuation.

D/A converter
Generates the voltage corresponding to the value set in the ADCD register.

Comparator
Compares the sampled and held input voltage with the output voltage of the D/A converter, and
determines which voltage is higher or lower.

Controller
The controller has two functions:
• For the A/D conversion function, the controller successively determines the value of each bit of
  the ADCD register, starting from the most significant bit and proceeding to the least significant
  bit, based on the greater-than/less-than signal from the comparator. When conversion is
  complete, the circuit sets the interrupt request flag bit (ADC1: ADI).
• For the sense function, the controller sets the interrupt request flag bit (ADI) if the greater-
  than/less-than signal from the comparator matches the compare condition setting bit (SIFM)
  in the ADC1 register.

ADCD register
The ADCD register has two functions:
• Stores the A/D conversion result for the A/D conversion function.
• For the sense function, the data for the voltage that is compared with the input voltage is
  written to this register.

ADC1 register
The ADC1 register is used to enable or disable each function, select the analog input pin, check
statuses, and control interrupts.

ADC2 register
The ADC2 register is used to select the input clock, enable or disable interrupts, and select
functions.

A/D Converter Power Supply Voltage

AVcc
The A/D converter power supply pin. Use at the same voltage as Vcc. When high A/D
conversion resolution is required, take measures to ensure that the noise on Vcc is not present
on AVcc, or use a separate power supply. Connect this pin to the power supply, even if the A/D
converter is not used.

AVss
The A/D converter ground pin. Use at the same voltage as Vss. When high A/D conversion
accuracy is required, take measures to ensure that the noise on Vss is not present on AVss.
Connect this pin to ground (GND), even if the A/D converter is not used.

AVR
Reference voltage input pin for the A/D converter. The A/D converter performs 8-bit A/D
conversion between AVR and AVss.
Connect to AVss if the A/D converter is not used.
13.3 Structure of A/D Converter

This section describes the pins, pin block diagrams, registers, and an interrupt source for the A/D converter.

- A/D Converter Pins

The A/D converter function uses the P50/AN0 to P57/AN7 pins. These pins can function as either output-only ports of the N-ch open-drain outputs (P50 to P57), as the analog input pins (AN0 to AN7).

AN0 to AN7:

The analog voltages to be converted (A/D conversion function) or compared (sense function) are applied to these pins.

To select the analog input function for one of these pins, you set the corresponding bit of the port data register (PDR5) to “1,” to turn off the port output transistor, then set the analog input channel select bits (ADC1: ANS0 to ANS3) to select the pin as the analog input channel. Pins that are not needed for analog inputs can still be used as output port pins, even while the A/D converter is being used.

- Block Diagram of A/D Converter Pin

**Figure 13.3a Block Diagram of P57/AN7 to P50/AN0 Pins**

Check: Do not set with a pull-up resistor as an option for any of the P57/AN7 to P50/AN0 pins if using the A/D converter.
A/D Converter Registers

<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>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>002Dh</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>00000000s</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</td>
<td>R/W</td>
<td>R/W</td>
<td></td>
</tr>
</tbody>
</table>

ADC2 (A/D control register 2)

<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>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>002Eh</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>ADCK</td>
<td>ADIE</td>
<td>ADMD</td>
<td>EXT</td>
<td>XXX00001s</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</td>
<td>R/W</td>
<td>R/W</td>
<td></td>
</tr>
</tbody>
</table>

ADCD (A/D data register)

<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>
<th>Initial Value</th>
</tr>
</thead>
<tbody>
<tr>
<td>002Fh</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>XXXXXXXXs</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 : Readable and writable
R  : Read-only
—  : Unused
X  : Indeterminate
* 1 : For MB89160L series, ANS3 is not used. The read and initial value of ANS3 are indeterminate.

Figure 13.3b A/D Converter Registers

A/D Converter Interrupt Source

IRQB: The A/D converter generates an interrupt request if an interrupt request output is enabled (ADC2: ADIE = "1") when A/D conversion completes or the sense function detects the specified condition.
### 13.3 A/D Converter configuration

#### 13.3.1 A/D Control Register 1 (ADC1)

A/D control register 1 (ADC1) is used to enable or disable the functions, select the analog input pin, and check the state of the A/D converter.

#### A/D Control Register 1 (ADC1)

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>002DH</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R/W</td>
<td>R</td>
<td>R/W</td>
<td>R/W</td>
<td>00000000H*1</td>
</tr>
</tbody>
</table>

- **AD**
  - **A/D converter activation bit**
  - Only applies when software activation is specified (ADC2: EXT = "0").
  - 0: Does not activate the A/D conversion or sense function.
  - 1: Activates the A/D conversion or sense function.

- **SIFM**
  - **Compare condition setting bit**
  - Only applies when the sense function is selected (ADC2: ADMD = "1").
  - 0: Sets the interrupt request flag bit when the input voltage is less than the compare voltage.
  - 1: Sets the interrupt request flag bit when the input voltage is greater than the compare voltage.

- **ADMV**
  - **Conversion-in-progress flag bit**
  - 0: Conversion or comparison not currently in progress.
  - 1: Conversion or comparison in progress.

- **ADI**
  - **Interrupt request flag bit**
  - Read
    - For the A/D conversion function:
      - 0: Conversion not complete. Specified condition has not occurred. Clears this bit.
      - 1: Conversion completes. Specified condition occurred. No effect. The bit does not change.
  - Write

**AN53 AN52 AN51 AN50**: Analog input channel selection bits

<table>
<thead>
<tr>
<th></th>
<th>AN0</th>
<th>AN1</th>
<th>AN2</th>
<th>AN3</th>
<th>AN4</th>
<th>AN5</th>
<th>AN6</th>
<th>AN7</th>
<th>Not available</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>×</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>×</td>
</tr>
<tr>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>×</td>
</tr>
</tbody>
</table>

*1 For MB89160L series, ANS3 is not used. Writing to ANS3 has no effect on operation. The read and initial value of ANS3 are indeterminate.

---

*Figure 13.3.1a A/D Control Register 1 (ADC1)*
## Table 13.3.1a A/D Control Register 1 (ADC1) Bits

<table>
<thead>
<tr>
<th>Bit 7</th>
<th>Bit 6</th>
<th>Bit 5</th>
<th>Bit 4</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>ANS3 to ANS0:</strong> Analog input channel selection bits</td>
<td>These bits select which of the AN0 to AN7 pins to use as the analog input pin. When using software activation (ADC2: EXT = &quot;0&quot;), these bits can be modified to at the same time as activating the A/D conversion or sense function (AD = &quot;1&quot;).</td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Check:</strong> Always set ANS3 to &quot;0&quot;.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>If ANS3 is &quot;1&quot;, no pin is selected as the analog input pin. Also, disable general-purpose port output corresponding to the analog input pin.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>Do not modify these bits when the ADMV bit is set to &quot;1&quot;.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Note:</strong> Pins not used as analog inputs can be used as general-purpose ports.</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td><strong>Note:</strong> For MB89160L series, ANS3 is not used. Writing to ANS3 has no effect on operation. The read and initial value of ANS3 is indeterminate.</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 3</th>
<th><strong>ADI:</strong> Interrupt request flag bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>• For the A/D conversion function:</td>
<td></td>
</tr>
<tr>
<td>This bit is set to &quot;1&quot; when the A/D conversion is completed.</td>
<td></td>
</tr>
<tr>
<td>• For the sense function:</td>
<td></td>
</tr>
<tr>
<td>This bit is set to &quot;1&quot; when the input voltage satisfies the condition set in the compare condition setting bit (SIFM).</td>
<td></td>
</tr>
<tr>
<td>• An interrupt request is output for either function when both this bit and the interrupt request enable bit (ADC2: ADIE) are &quot;1&quot;.</td>
<td></td>
</tr>
<tr>
<td>• Writing &quot;0&quot; clears this bit. Writing &quot;1&quot; has no effect and does not change the bit value.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 2</th>
<th><strong>ADMV:</strong> Conversion-in-progress flag bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>This bit indicates whether or not the A/D conversion function is currently performing a conversion or the sense function is currently performing a voltage comparison. The bit is set to &quot;1&quot; when a conversion or comparison is in progress.</td>
<td></td>
</tr>
<tr>
<td><strong>Note:</strong> This bit is read-only. The write value has no meaning and has no effect on the operation.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 1</th>
<th><strong>SIFM:</strong> Compare condition setting bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>• This bit has no meaning for the A/D conversion function.</td>
<td></td>
</tr>
<tr>
<td>• For the sense function:</td>
<td></td>
</tr>
<tr>
<td>This bit sets the comparison condition for the input voltage and compare voltage that generates an interrupt source.</td>
<td></td>
</tr>
<tr>
<td>An interrupt request is generated (ADI = &quot;1&quot;) when the input voltage is less than the compare voltage if the bit is &quot;0&quot;, and when the input voltage is greater than the compare voltage if the bit is &quot;1&quot;.</td>
<td></td>
</tr>
<tr>
<td>No interrupt request is generated if the input voltage and compare voltage are equal.</td>
<td></td>
</tr>
<tr>
<td>When using software activation (ADC2: EXT = &quot;0&quot;), this bit can be modified to at the same time as starting the sense function (AD = &quot;1&quot;).</td>
<td></td>
</tr>
<tr>
<td><strong>Check:</strong> Do not modify these bits when the ADMV bit is set to &quot;1&quot;.</td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Bit 0</th>
<th><strong>AD:</strong> A/D converter activation bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>• This bit activates the A/D conversion or sense function by software.</td>
<td></td>
</tr>
<tr>
<td>• Writing &quot;1&quot; to this bit activates the A/D conversion or sense function when continuous activation is not specified (ADC2: EXT = &quot;0&quot;).</td>
<td></td>
</tr>
<tr>
<td><strong>Check:</strong> Writing &quot;0&quot; to this bit does not stop the A/D conversion or sense function.</td>
<td></td>
</tr>
<tr>
<td>• The read value is always &quot;0&quot;.</td>
<td></td>
</tr>
<tr>
<td>• This bit has no meaning when continuous activation is specified.</td>
<td></td>
</tr>
</tbody>
</table>
13.3  A/D Converter configuration

13.3.2  A/D Control Register 2 (ADC2)

A/D control register 2 (ADC2) is used to select the A/D converter functions, select the input clock, enable or disable interrupts and continuous activation, and check the state of the A/D converter.

**A/D Control Register 2 (ADC2)**

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>002EH</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>ADCK</td>
<td>ADIE</td>
<td>ADMD</td>
<td>EXT</td>
<td>RESV</td>
<td>XXX00001H</td>
</tr>
</tbody>
</table>

- **ADCK**: Input clock selection bit
  - 0: Timebase timer output (divide-by-2 source oscillation)
  - 1: 8/16-bit timer 16-bit time output

- **ADIE**: Interrupt request enable bit
  - 0: Disables interrupt request output.
  - 1: Enables interrupt request output.

- **ADMD**: Function selection bit
  - 0: A/D conversion function
  - 1: Sense function

- **EXT**: Continuous activation enable bit
  - 0: Activates by the AD bit in the ADC1 register.
  - 1: Activates continuously by the clock selected in the ADCK bit.

- **RESV**: Reserved bit
  - Always write “1” to this bit.

![Figure 13.3.2a A/D Control Register 2 (ADC2)](image-url)
<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7 Bit 6 Bit 5</td>
<td>Unused bits</td>
</tr>
<tr>
<td>Bit 4</td>
<td>ADCK: Input clock selection 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: Continuous activation enable bit</td>
</tr>
<tr>
<td>Bit 0</td>
<td>RESV1: Reserved bit</td>
</tr>
</tbody>
</table>

### Bit 7 Bit 6 Bit 5: Unused bits
- The read value is indeterminate.
- Writing to these bits has no effect on the operation.

### Bit 4: ADCK: Input clock selection bit
This bit selects the input clock used to activate the A/D conversion or sense function when continuous activation is specified (EXT = “1”). Setting this bit to “0” selects the timebase timer output (divide-by-2<sup>8</sup> main clock source oscillation). Setting this bit to “1” selects the 16-bit timer output (TO1) in 8/16 timer/counter.

**Check:** In the subclock mode, the main clock oscillator is stopped, which means that the timebase timer output cannot be used to trigger continuous mode conversions/comparisons.

### Bit 3: ADIE: Interrupt request enable bit
This bit enables or disables an interrupt request output to the CPU. An interrupt request is output when both this bit and the interrupt request flag bit (ADC1: ADI) are “1”.

### Bit 2: ADMD: Function selection bit
This bit switches between the A/D conversion function and sense function. The A/D converter operates as the A/D conversion function when this bit is set to “0” and as the sense function when this bit is set to “1”.

**Check:** Do not modify this bit when the conversion-in-progress bit (ADC1: ADMV) is set to “1”. Also, clear the interrupt request flag bit (ADC1: ADI = “0”) at switching functions.

### Bit 1: EXT: Continuous activation enable bit
This bit selects whether to activate the A/D conversion and sense functions by software or to operate continuously synchronized with an input clock. Setting this bit to “0” enables software activation by the A/D converter activation bit (ADC1: AD). Setting this bit to “1” enables continuous activation on the rising edge of the clock selected in the input clock selection bit (ADC2: ADCK).

### Bit 0: RESV1: Reserved bit
**Check:**
- Always write “1” to this bit.
- The read value is always “1”.
13.3 A/D Converter configuration

13.3.3 A/D Data Register (ADCD)

The A/D data register stores the A/D conversion result for the A/D conversion function. The compare voltage data is written to this register for the sense function.

- **A/D Data Register (ADCD)**

  Figure 13.3.3a shows the bit structure of the A/D data register (ADCD).

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>002Fh</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>R</td>
<td>W</td>
<td>XXXXXXXXb</td>
</tr>
<tr>
<td></td>
<td>R</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td>W</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

  R : Read-only  
  W : Write-only  
  X : Indeterminate

  **Figure 13.3.3a A/D Data Register (ADCD)**

  - **For A/D conversion function**

    The conversion result is decided approximately 44 instruction cycles (52 instruction cycle for MB89165L series) after A/D conversion is activated. The data of conversion is stored in this register. The value of the register is indeterminate while A/D conversion is in progress. The register is read-only for the A/D conversion function.

  - **For sense function**

    Before activating the sense function, set the data corresponding to the voltage to be compared (compare voltage).

    As the register is write-only when the sense function is selected, bit manipulation instructions cannot be used. Confirm operation stopped (ADC2: EXT = "0", ADC1: ADMV = "0") before writing to this register.

- **Example of ADCD Register Setting for Sense Function**

<table>
<thead>
<tr>
<th>Compare voltage (V)</th>
<th>5.0</th>
<th>4.0</th>
<th>3.0</th>
<th>2.0</th>
<th>1.0</th>
<th>0.0</th>
</tr>
</thead>
<tbody>
<tr>
<td>ADCD register set value (AVcc=AVR=5.0V, AVss=0.0V)</td>
<td>FFh</td>
<td>CDh</td>
<td>9Ah</td>
<td>66h</td>
<td>33h</td>
<td>00h</td>
</tr>
<tr>
<td>ADCD register set value (AVcc=AVR=3.0V, AVss=0.0V)</td>
<td>----</td>
<td>----</td>
<td>FFh</td>
<td>ABh</td>
<td>55h</td>
<td>00h</td>
</tr>
</tbody>
</table>

Table 13.3.3a Example of ADCD Register Setting for Sense Function
13.4 A/D Converter Interrupts

The A/D converter has the following two interrupts:

- Conversion completion for the A/D conversion function
- Match of the input voltage and the comparison condition

Interrupt for A/D Conversion Function

When A/D conversion completes, the interrupt request flag bit (ADC1: ADI) is set to “1”. At this time, an interrupt request (IRQB) to the CPU is generated if the interrupt request enable bit is enabled (ADC2: ADIE = “1”). Write “0” to the ADI bit in the interrupt processing routine to clear the interrupt request.

The ADI bit is set after completion of A/D conversion, regardless of the ADIE bit value.

Note: An interrupt request is generated immediately if the ADI bit is “1” when the ADIE bit is changed from disabled to enabled (“0” → “1”).

Interrupt for Sense Function

When the specified comparison condition is satisfied after completion of comparison of the input voltage and compare voltage, the interrupt request flag bit (ADC1: ADI) is set to “1”.

At this time, an interrupt request (IRQB) to the CPU is generated if the interrupt request enable bit is enabled (ADC2: ADIE = “1”). Write “0” to the ADI bit in the interrupt processing routine to clear the interrupt request.

The ADI bit is set when the comparison condition is satisfied, regardless of the ADIE bit value.

Note: An interrupt request is generated immediately if the ADI bit is “1” when the ADIE bit is changed from disabled to enabled (“0” → “1”).

Register and Vector Table for A/D Converter Interrupt

<table>
<thead>
<tr>
<th>Interrupt</th>
<th>Interrupt level settings register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRQB</td>
<td>ILR3 (007EH)</td>
<td>FFE4H FF5EH</td>
</tr>
</tbody>
</table>

Register Setting bits: LB1 (Bit 7) LB0 (Bit 6)

Reference: See Section 3.4.2, “Interrupt Processing” for details on the operation of interrupt.
13.5 Operation of A/D Converter

The A/D conversion and sense functions of the A/D converter can be activated by software or can be activated continuously.

Activating A/D Conversion Function

Software activation

Figure 13.5a shows the settings required for software activation of the A/D conversion function.

<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>ANS3</td>
<td>ANS2</td>
<td>ANS1</td>
<td>ANS0</td>
<td>ADI</td>
<td>ADMV</td>
<td>SIFM</td>
<td>AD</td>
</tr>
<tr>
<td>0</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>1</td>
</tr>
<tr>
<td>ADC2</td>
<td></td>
<td></td>
<td></td>
<td>ADCK</td>
<td>ADIE</td>
<td>ADMD</td>
<td>EXT</td>
</tr>
<tr>
<td>×</td>
<td>×</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>RESV</td>
<td></td>
</tr>
<tr>
<td>ADCD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 13.5a A/D Conversion Function (Software Activation) Settings

On activation, the A/D converter starts the operation of the A/D conversion function. The A/D conversion function can be reactivated while conversion is in progress.

Continuous activation

Figure 13.5b shows the settings required for continuous activation of the A/D conversion function.

<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>ANS3</td>
<td>ANS2</td>
<td>ANS1</td>
<td>ANS0</td>
<td>ADI</td>
<td>ADMV</td>
<td>SIFM</td>
<td>AD</td>
</tr>
<tr>
<td>0</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>ADC2</td>
<td></td>
<td></td>
<td></td>
<td>ADCK</td>
<td>ADIE</td>
<td>ADMD</td>
<td>EXT</td>
</tr>
<tr>
<td>×</td>
<td>×</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>RESV</td>
<td></td>
</tr>
<tr>
<td>ADCD</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Figure 13.5b A/D Conversion Function (Continuous Activation) Settings

When continuous activation is enabled, the rising edge of the selected input clock activates the A/D conversion, starting operation of the A/D conversion function. When continuous activation is disabled (ADC2: EXT = “0”), continuous activation halts but software activation is available.

Operation of A/D Conversion Function

The following describes the operation of the A/D converter. From activation to completion of A/D conversion requires approximately 44 instruction cycles (52 instruction cycles for MB89160L series).

1. On activation, A/D conversion sets the conversion-in-progress flag bit (ADC1: ADMV = “1”) and connects the sample hold circuit to the specified analog input pin.
2. The internal sample hold capacitor captures the voltage at the analog input pin for approximately 8 instruction cycles. The capacitor holds the voltage until the A/D conversion completes.
3. The comparator compares the voltage captured by the sample hold capacitor with the A/D converter reference voltage starting from the most significant bit (MSB) and ending with the least significant bit (LSB), and transfers each bit sequentially to the ADCD register.
4. When the complete result has been transferred to the ADCD register, the conversion-in-progress flag bit is cleared (ADC1: ADMV = “0”) and the interrupt request flag bit is set (ADC1: ADI = “1”).
## Activating Sense Function

### Software activation

Figure 13.5c shows the settings required for software activation of the sense function.

### Continuous activation

Figure 13.5d shows the settings required for continuous activation of the sense function.

### Operation of Sense Function

The following describes the operation of the sense function. From activation to completion of the sense function requires approximately 12 instruction cycles.

1. On activation, the sense function sets the conversion-in-progress flag bit (ADC1: ADMV = “1”) and connects the sample hold circuit to the specified analog input pin.
2. The internal sample hold capacitor captures the voltage at the analog input pin for approximately 8 instruction cycles. The capacitor holds the voltage until the comparison completes.
3. The comparator compares the voltage captured by the sample hold capacitor with the voltage corresponding to the value set in the ADCD register.
4. When voltage comparison completes, the interrupt request flag bit is set (ADC1: ADI = “1”) if the input voltage matches the condition specified by the compare condition setting bit (ADC1: SIFM). The ADI bit does not change if the input voltage does not match the specified condition or if the input voltage and set voltage are equal.

**Note:** For the sense function, an interrupt request is not generated when comparison completes if the comparison condition is not matched. Whether or not comparison has completed can be determined by checking whether the conversion-in-progress flag bit (ADC1: ADMV) is “0”. 

---

### Figure 13.5c Sense Function (Software Activation) Settings

On activation the sense function starts the operation of the sense function.

### Figure 13.5d Sense Function (Continuous Activation) Settings

When continuous activation is enabled, the rising edge of the selected input clock activates the sense function, starting operation of the sense function. When continuous activation is disabled (ADC2: EXT = “0”), continuous activation stops but software activation is available.

---

### ADC1, ADC2, and ADCD Settings

<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>ANS3</td>
<td>ANS2</td>
<td>ANS1</td>
<td>ANS0</td>
<td>ADI</td>
<td>ADMV</td>
<td>SIFM</td>
<td>AD</td>
</tr>
<tr>
<td>0</td>
<td>⊗</td>
<td>⊗</td>
<td>⊗</td>
<td>⊗</td>
<td>⊗</td>
<td>⊗</td>
<td>1</td>
</tr>
<tr>
<td>ADCK</td>
<td>ADIE</td>
<td>ADMD</td>
<td>EXT</td>
<td>RESV</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>×</td>
<td>⊗</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>ANS3</th>
<th>ANS2</th>
<th>ANS1</th>
<th>ANS0</th>
<th>ADI</th>
<th>ADMV</th>
<th>SIFM</th>
<th>AD</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>⊗</td>
<td>⊗</td>
<td>⊗</td>
<td>⊗</td>
<td>⊗</td>
<td>⊗</td>
<td>×</td>
</tr>
<tr>
<td>ADCK</td>
<td>ADIE</td>
<td>ADMD</td>
<td>EXT</td>
<td>RESV</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>⊗</td>
<td>⊗</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Sets the compare voltage.
13.6 Notes on Using A/D Converter

This section lists points to note when using the A/D converter.

- **Input impedance of analog input pins**
  The A/D converter contains a sample hold circuit as shown in Figure 13.6a to fetch analog input voltage into the sample hold capacitor for eight instruction cycles after activating A/D conversion (or the sense function). For this reason, if the output impedance of the external circuit for the analog input is high, analog input voltage might not stabilize within the analog input sampling period. Therefore, it is recommended to keep the output impedance of the external circuit low (below 10 kΩ). Note that if the impedance cannot be kept low, it is recommended to connect an external capacitor of about 0.1µF for the analog input pin.

- **Notes on setting by program**
  - For the A/D conversion function, the ADCD register maintains previous value until the next A/D conversion is activated. However, the content of the ADCD register becomes indeterminate immediately after activating A/D conversion.
  - Do not re-select the analog input channel (ADC1: ANS3 to ANS0) or do not switch between the A/D conversion and sense functions (ADC2: ADMD) while the A/D conversion or sense function is operating. Particularly, when continuous activation is enabled, only perform such operations after disabling continuous activation (ADC2: EXT = “0”) and waiting for the conversion-in-progress flag bit (ADC1: ADMV) to go to “0”. Stop operation before modifying the compare condition setting bit (ADC1: SIFM) in the same way when the sense function is operating.
  - When using the sense function, stop operation before writing to the ADCD register.
  - Clear the interrupt request flag bit (ADC1: ADI = “0”) before switching between the A/D conversion and sense functions.
  - A reset or activation of stop mode stops the A/D converter and initializes all registers.
  - Interrupt processing cannot return if the interrupt request flag bit (ADC1: ADI) is “1” and the interrupt request enable bit is enabled (ADC2: ADIE = “1”). Always clear the ADI bit.

- **Note on interrupt requests**
  The interrupt request flag bit (ADC1: ADI) is not set if A/D conversion is reactivated (ADC1: AD = “1”) at the same time as the previous A/D conversion completes, or if the sense function is reactivated (ADC1: AD = “1”) at the same time as the comparison condition is satisfied.
- **Error**
  The smaller the AVR – AVss, the greater the error would become relatively.

- **Turn-on sequence for A/D converter power supply and analog inputs**
  Always apply the A/D converter power supply (AVcc, AVss) and analog inputs (AN0 to AN7) at the same time or after turning on the digital power supply (Vcc).
  Similarly, when power supply is turned off, always turn off the A/D converter power supply (AVcc, AVss) and analog inputs (AN0 to AN7) at the same time or before turning off the digital power supply (Vcc).
  Take care that AVcc, AVss, and the analog inputs do not exceed the digital power supply voltage when turning the A/D converter power supply on or off.

- **Conversion time**
  A/D conversion function conversion time and sense function comparison time are affected by the clock mode, oscillator frequency, and main clock speed (speed shift function).

- **Continuous activation input clock**
  The 8/16-bit timer/counter output, which can be selected for continuous activation (ADC2: EXT = 1), is affected by the clock mode and speed shift function. The timebase timer output, which can also be selected, is not affected by the speed shift function, but the timebase timer output cannot be used in subclock mode because the main clock (which drives the timebase timer) is stopped in that mode. Note also that the cycle time is affected (for one cycle) when the timebase timer is cleared.
13.7 Program Example for A/D Converter

This section gives program examples for the A/D conversion and sense functions of the 8-bit A/D converter.

Program Example for A/D Conversion Function

Processing description

- Performs software-activated A/D conversion of the analog voltage input to the AN0 pin. The example does not use interrupts and detects conversion completion within the program loop.

Coding example

```assembly
 ; Port 5 data register
PDR5 EQU 000FH

 ; A/D control register 1
ADC1 EQU 002DH

 ; A/D control register 2
ADC2 EQU 002EH

 ; A/D data register
ADCD EQU 002FH

 ; Define the AN0 analog input pin.
AN0 EQU PDR5:0

 ; Define the interrupt request flag bit.
ADI EQU ADC1:3

 ; Define the conversion-in-progress flag bit.
ADMV EQU ADC1:2

 ; A/D converter activation bit (software activation)
AD EQU ADC1:0

 ; Define the continuous activation enable bit.
EXT EQU ADC2:1

;-------------------------------Main program-----------------------------------------------------------------------------------------------

CSEG ; [CODE SEGMENT]

SETB AN0 ; Set P50/AN0 pin as an analog input pin (AN0).

CLRI ; Disable interrupts.

CLRB EXT ; Disable continuous activation.

AD_WAIT

BBS ADMV,AD_WAIT ; Loop to check that the A/D converter is stopped.

MOV ADC1,#00000000B ; Select analog input channel 0 (AN0),

MOV ADC2,#00000001B ; clear interrupt request flag, and do not activate by software.

MOV A,ADCD ; Read A/D conversion data.

ENDS

;------------------------------------------------------------------------------------------------------------------------------

END
```
Program Example for Sense Function

Processing description

• Generate an interrupt if the analog voltage input to the AN0 pin is less than 3.0 V.
• Perform continuous activation of the sense function synchronized with pulses (Timebase timer output (from divided by $F_{CH}/2$)).
• For analog power supply voltage ($AV_{CC}$) = reference voltage ($AV_{R}$) = 5.0 V, an ADCD register value of 0A9H gives a compare voltage of 3.0 V. With a main clock oscillator frequency of 4.2 MHz, the continuous activation cycle time would be $2^{16}/4.2$ MHz = approx. 61.0 µs.

Coding example

PDR5 EQU 000FH ; Port 5 data register
ADC1 EQU 002DH ; A/D control register 1
ADC2 EQU 002EH ; A/D control register 2
ADCD EQU 002FH ; A/D data register
AN0 EQU PDR5:0 ; Define the AN0 analog input pin.
ADI EQU ADC1:3 ; Define the interrupt request flag bit.
ADMV EQU ADC1:2 ; Define the conversion-in-progress flag bit.
AD EQU ADC1:0 ; A/D converter activation bit (software activation)
EXT EQU ADC2:1 ; Define the continuous activation enable bit.
ILR3 EQU 007EH ; Set interrupt level setting register.
INT_V DSEG ABS ; [DATA SEGMENT]
ORG 0FFE8H
IORB
INT_V ENDS

---Main program-----------------------------------------------

CSEG ; [CODE SEGMENT]

SETB AN0 ; Set P50/AN0 pin as an analog input pin.
CLRI ; Disable interrupts.
MOV ILR3,#01111111B ; Set interrupt level (level 1).
CLRB EXT ; Disable continuous activation.

AD_WAIT
BBS ADMV,AD_WAIT ; Loop to check that the A/D converter is halted.
MOV ADCD,#9AH ; Set compare voltage data (3.0 V).
MOV ADC1,#00000000B ; Select analog input channel 0 (AN0), clear interrupt request flag, set compare condition (interrupt if the input voltage is lower), and do not activate by software.
MOV ADC2,#00001111B ; Select timebase timer output as A/D clock, enable interrupt request output, select the sense function, and enable continuous activation.

SETI ; Enable interrupts.

---Interrupt processing routine-------------------------------------------

WAR1 CLRB ADI ; Clear interrupt request flag.
PUSHW A
XCHW A,T
PUSHW A

User processing

POP W A
XCHW A,T
POPW A
RETI
INT_V ENDS

END
CHAPTER 14

WATCH PRESCALER

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

14.1 Overview of Watch Prescaler ..................................... 258
14.2 Block Diagram of Watch Prescaler............................. 259
14.3 Watch Prescaler Control Register (WPCR)................. 261
14.4 Watch Prescaler Interrupt........................................... 263
14.5 Operation of Watch Prescaler .................................... 265
14.6 Notes on Using Watch Prescaler ............................... 267
14.7 Program Example for Watch Prescaler ...................... 269
14.1 Overview of Watch Prescaler

The watch prescaler provides interval timer functions. Four different interval times can be selected. The watch rescaler uses a 15-bit free-run counter which counts-up in sync with a subclock generated by the clock generator.

The watch prescaler also provides the timer output for the subclock oscillation stabilization delay time and the operating clock for watchdog timer.

- **Interval Timer Function (Watch Interrupt)**

  - The interval timer function generates repeated interrupts at fixed intervals with the subclock used as the count clock.
  - Interrupts are generated by watch prescaler interval timer divided clock outputs.
  - The interval timer divided clock output (interval time) can be selected from different settings.
  - The watch prescaler counter can be cleared.

  Table 14.1a lists the available interval times for the watch prescaler.

<table>
<thead>
<tr>
<th>Subclock Cycle Time</th>
<th>Interval time</th>
</tr>
</thead>
<tbody>
<tr>
<td>1/FCL (approx. 30.5 μs)</td>
<td>2^15/FCL (31.25 ms)</td>
</tr>
<tr>
<td></td>
<td>2^14/FCL (0.25 s)</td>
</tr>
<tr>
<td></td>
<td>2^13/FCL (0.50 s)</td>
</tr>
<tr>
<td></td>
<td>2^12/FCL (1.00 s)</td>
</tr>
</tbody>
</table>

FCL: Subclock source oscillation

The values enclosed in parentheses ( ) are for a 32.768 kHz subclock source oscillation.

Check: The watch prescaler cannot be used in devices in which a single clock option has been selected.

- **Clock Supply Function**

  The watch prescaler has the following clock supply functions:

  - The timer output used for the subclock oscillation stabilization delay time (one value)
  - The clock used for the watchdog timer (one value)
  - The clock used for the buzzer output (three values)

Table 14.1b lists the cycles of the clocks that the watch prescaler supplies to various peripherals.

<table>
<thead>
<tr>
<th>Subclock destination</th>
<th>Subclock cycle</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>Subclock oscillation stabilization delay time</td>
<td>2^15/FCL (1.00 s)</td>
<td>Do not switch to the subclock mode during the oscillator stabilization wait time.</td>
</tr>
<tr>
<td>Watchdog timer</td>
<td>2^14/FCL (0.50 s)</td>
<td>count-up clock for the watchdog timer</td>
</tr>
<tr>
<td>Buzzer output</td>
<td>2^13/FCL to 2^5/FCL (approx. 0.24 to 0.98 ms)</td>
<td>See Chapter 10, “Buzzer Output.”</td>
</tr>
</tbody>
</table>

FCL: Subclock source oscillation

The values enclosed in parentheses ( ) are for a 32.768 kHz subclock source oscillation.

Note: The oscillation stabilization delay time should be used as a guideline since the oscillation cycle is unstable immediately after oscillation starts.
14.2 Block Diagram of Watch Prescaler

The watch prescaler consists of the following four blocks:

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

### Block Diagram of Watch Prescaler

![Block Diagram of Watch Prescaler](image)

- **Watch prescaler counter**: A 15-bit up-counter that uses the subclock source oscillation clock as its count clock.
- **Counter clear circuit**: In addition to being cleared by setting the WPCR register (WCLR = "0"), the counter is cleared when the device changes to sub-stop mode (STBC : STP = “1”) and by power-on reset (optional).
- **Interval timer selector**: This circuit selects one of four divided clock outputs of the watch prescaler counter as the interval timer output. The falling edge of the selected output is the event that triggers the watch interrupt.
- **WPCR register**: The WPCR register is used to select the interval time bit, clear the counter, control interrupts, and check the state of the watch prescaler.
14.3 Watch Prescaler Control Register (WPCR)

The watch prescaler control register (WPCR) is used to select the interval timer bit, clear the counter, control interrupts, and check state of the watch prescaler.

### Watch Prescaler Control Register (WPCR)

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0005h</td>
<td>WIF</td>
<td>WIE</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>WS1</td>
<td>WS0</td>
<td>WCLR</td>
<td>00XXX000&lt;sup&gt;b&lt;/sup&gt;</td>
</tr>
</tbody>
</table>

- **WCLR** Watch prescaler clear bit
  - **Read**
    - 0: Clears the watch prescaler
    - 1: Reading always return "1". No effect. The bit does not change.

- **WS1** Watch interrupt interval time select bits
  - 0 0: 2<sup>15</sup>/FCL
  - 0 1: 2<sup>14</sup>/FCL
  - 1 0: 2<sup>13</sup>/FCL
  - 1 1: 2<sup>12</sup>/FCL

- **WIE** Interrupt Request Enable Bit
  - 0: Disables interrupt request output
  - 1: Enables interrupt request output

- **WIF** Watch interrupt request flag bit
  - 0: Have no interval interrupt
  - 1: Have interval interrupt
    - **Read**
      - 0: Have no interval interrupt
      - 1: Have interval interrupt
    - **Write**
      - 0: Clears this bit.
      - 1: No effect. The bit does not change.

---

<sup>b</sup> Initial value

---

**Figure 14.3a Watch Prescaler Control Register (WPCR)**
### Table 14.3a Watch Prescaler Control Register (WPCR) Bits

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7</td>
<td><strong>WIF:</strong> Watch interrupt request flag bit</td>
</tr>
<tr>
<td></td>
<td>• Set to “1” by the falling edge of the selected interval timer divided output.</td>
</tr>
<tr>
<td></td>
<td>• An interrupt request is output when both this bit and the interrupt request enable bit (WIE) are “1”.</td>
</tr>
<tr>
<td></td>
<td>• Writing “0” clears this bit. Writing “1” has no effect and does not change the bit value.</td>
</tr>
<tr>
<td>Bit 6</td>
<td><strong>WIE:</strong> Interrupt request enable bit</td>
</tr>
<tr>
<td></td>
<td>• This bit enables or disables an interrupt request output to the CPU. An interrupt request is output when both this bit and the watch interrupt request flag bit (WIF) are “1”.</td>
</tr>
<tr>
<td>Bit 5</td>
<td><strong>Unused bits</strong></td>
</tr>
<tr>
<td>Bit 4</td>
<td></td>
</tr>
<tr>
<td>Bit 3</td>
<td></td>
</tr>
<tr>
<td>Bit 2</td>
<td><strong>WS1, WS0:</strong> Watch interrupt interval time selection bits</td>
</tr>
<tr>
<td>Bit 1</td>
<td>• Select interval timer cycle.</td>
</tr>
<tr>
<td></td>
<td>• Specify which bit of the watch prescaler counter (or which divided output) will be used for the interval timer.</td>
</tr>
<tr>
<td></td>
<td>• One of four interval times may be selected.</td>
</tr>
<tr>
<td>Bit 0</td>
<td><strong>WCLR:</strong> Watch prescaler clear bit</td>
</tr>
<tr>
<td></td>
<td>• Bit used to clear the watch prescaler counter.</td>
</tr>
<tr>
<td></td>
<td>• Writing “0” to this bit clears the counter to 0000H. Writing “1” has no effect and does not change the bit value.</td>
</tr>
<tr>
<td></td>
<td><strong>Note:</strong> The read value is always “1”.</td>
</tr>
</tbody>
</table>
14.4 Watch Prescaler Interrupt

The watch prescaler generates an interrupt request at the falling edge of the specific divided output (interval timer function).

- **Interrupts for Interval Timer Function (Watch Interrupt)**

  The watch prescaler counter counts up, clocked by the subclock source oscillation. Unless the system is in main-stop mode, the watch interrupt request flag is set to “1” (WPCR: WIF = 1) at the end of the selected time interval. At this time, an interrupt request (IRQ8) to the CPU is generated if the interrupt request enable bit is enabled (WPCR: WIE = “1”). Write “0” to the WIF bit in the interrupt processing routine to clear the interrupt request. The WIF bit is set when the specified divide output falls, regardless of the WIE bit value.

  **Check:** When enabling an interrupt request output (WIE = “1”) after wake-up from a reset, always clear the WIF bit (WIF = “0”) at the same time.

  **Notes:**
  - An interrupt request is generated immediately if the WIF bit is “1” when the WIF bit is changed from disabled to enabled (“0” → “1”).
  - The WIF bit is not set if the counter cleared (WPCR: WCLR = “0”) at the same time as an overflow on the specified bit occurs.

- **Oscillation Stabilization Delay Time and Watch Interrupt**

  If the interval time is set shorter than the subclock oscillation stabilization delay time, an watch interrupt request from the watch prescaler (WPCR: WIF = “1”) is generated at the time when CPU wakes up from sub-stop mode by an external interrupt. In this case, disable the watch prescaler interrupt (WPCR: WIE = “0”) when changing to sub-stop mode.

- **Register and Vector Table for Watch Prescaler Interrupt**

  Table 14.4a lists the register and vector table for watch prescaler interrupt.

<table>
<thead>
<tr>
<th>Interrupt</th>
<th>Interrupt level setting register</th>
<th>Vector table address</th>
</tr>
</thead>
<tbody>
<tr>
<td>IRQ8</td>
<td>ILR3 (007Eh)</td>
<td>FFEA (bit1)</td>
</tr>
<tr>
<td></td>
<td></td>
<td>FFEB (bit0)</td>
</tr>
</tbody>
</table>

**Reference:** See Section 3.4.2 “Interrupt Processing” for details on the interrupt operations.
Memo
14.5 Operation of Watch Prescaler

The watch prescaler has the interval timer function and the clock supply function.

- Operation of Interval Timer Function (Watch Prescaler)

Figure 14.5a shows the settings required to operate the interval timer function.

<table>
<thead>
<tr>
<th>WPCR</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></td>
<td>WIF</td>
<td>WIE</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>WS1</td>
<td>WS0</td>
<td>WCLR</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 14.5a Interval Timer Function Settings

Provided the subclock is oscillating, the watch prescaler 15-bit counter continues to count-up using the subclock as its count clock.

After being cleared (WCLR = “0”), the counter restarts counting-up from “0000h”. When the counter reaches a full count of “7FFFh”, the next count takes it “0000h”, and it continues to count-up. As the count proceeds, a falling edge will eventually occur at the selected divided clock output. At this time, unless the system is in main clock stop mode, the watch prescaler sets the watch interrupt request flag bit (WIF) to “1”. Consequently, the watch prescaler generates interrupt requests at fixed intervals (the selected interval time), based on the time that the counter is cleared.

- Operation of Clock supply Function

The watch prescaler is also used as a timer to generate the subclock oscillation stabilization delay time. The time between the counter cleared state and the falling edge of the MSB output is used the subclock oscillation stabilization delay time \(2^{15}/F_{CL}\), where \(F_{CL}\) is subclock source oscillation).

The watch prescaler also provides the clock for the watchdog timer and buzzer output. Clearing the watch prescaler counter affects the operation of the buzzer output. When the watch prescaler is selected as the clock source for the watchdog timer (WDTC: CS = 1) both counters are cleared simultaneously.
**Operation of Watch Prescaler**

Figure 14.5b shows counter states when the interval timer is operating in subclock mode and the system goes into the sleep and stop modes, and when there is a counter clear request.

For the case when the interrupt interval time selection bits in the watch prescaler control register (WPCR: WS1, WS0) are “set to 11” ($2^{15}/F_{CL}$).
14.6 Notes on Using Watch Prescaler

This section lists points to note when using the watch prescaler. The watch prescaler cannot be used in devices in which the single-clock option has been selected.

Notes on Using Watch Prescaler

- **Notes on setting bits by program**
  
  The system cannot recover from interrupt processing if the interrupt request flag bit (WPCR: WIF) is “1” and the interrupt request enable bit is enabled (WPCR: WIE = “1”). Always clear the WIF bit.

- **Clearing Watch prescaler**
  
  In addition to being cleared by the watch prescaler clear bit (WPCR: WCLR = “0”), the watch prescaler is cleared wherever the subclock oscillation stabilization delay time is required.

  When the watch prescaler is selected as a count clock of the watchdog timer (WDTC: CS = “1”), clearing the watch prescaler also clears the watchdog timer.

- **Using as timer for oscillator Stabilization delay time**
  
  As the subclock source oscillation is stopped when the power is turned on and during sub-stop mode, the watch prescaler provides the oscillation stabilization delay time after the oscillator starts. Do not switch clock modes from main clock to subclock during this delay time (immediately after power on, etc.)

  The subclock oscillation stabilization delay time is fixed.

  **Reference:** See Section 3.6.1, “Oscillation Stabilization delay Time” for details.

- **Notes on watch interrupt**
  
  In main-stop mode, the watch prescaler counter operates, but no interrupt requests are generated.

- **Notes on peripheral functions that provides a clock supply from watch prescaler**
  
  As the clock derived from the watch prescaler restarts output from the its initial state when the watch prescaler counter is cleared, the “H” level may be shorter or the “L” level longer by a maximum of half cycle. The clock of the watchdog timer also restarts output from its initial state.

  However, as the watchdog timer counter is cleared at the same time, the watchdog timer operates in normal cycle.

  Figure 14.6a shows the effect on the buzzer output of clearing the watch prescaler.
Figure 14.6a Effect on Buzzer Output Due to Clearing of Watch Prescaler

For the case when the buzzer selection bits in the buzzer register (BZCR: BZ2, BZ1, BZ0) are set to "101. (Divide-by-32 subclock source oscillation, 1024 Hz output at 32.768 Hz operation).
14.7 Program Example for Watch Prescaler

This section gives program example for the watch prescaler.

- **Program Example for the Watch Prescaler**
  - **Processing description**
    Generates repeated watch interrupts at \(2^{15}/F_{CL}\) (\(F_{CL}\) = subclock source oscillation) intervals. At this time, the interval time is 1 second (at 32,768 kHz operation).
  - **Coding example**

```
WPCR EQU 000BH ; Address of watch prescaler control register
WIF  EQU WPCR:7 ; Define the watch interrupt request flag bit.
ILR3 EQU 007EH ; Address of the interrupt level setting register 2
INT_V DSEG  ABS ; [DATA SEGMENT]
ORG 0FFEAH
IRQ8 DW WARI ; Set interrupt vector.
INT_V ENDS

;----------------------------------------
CSEG ; [CODE SEGMENT]
; Stack pointer (SP) etc. are already initialized.
:
CLRI ; Disable interrupts.
MOV ILR3, #11111110B ; Set interrupt priority (level 2).
MOV WPCR, #01000110B ; Clear interrupt request flag, enable interrupt request output, select \(2^{15}/F_{CL}\), and clear watch prescaler.
SETI ; Enable interrupts.
:
;----------------------------------------
END

;----------------------------------------
END

Main program ;----------------------------------------

Interrupt program ;----------------------------------------

WARI CLRB WIF ; Clear interrupt request flag.
PUSHW A
XCHW A,T
PUSHW A
:
User processing
:
POPW A
XCHW A,T
POPW A
RETI
ENDS

;----------------------------------------
```
CHAPTER 15
REMOTE CONTROL TRANSMIT OUTPUT GENERATOR (6-BIT PPG)

This chapter describes the functions and operation of the remote control transmit output generator.

15.1 Overview of Remote Control Transmit Output Generator ............... 270
15.2 Block Diagram of Remote Control Transmit Output Generator ...... 274
15.3 Structure of Remote Control Transmit Output Generator .............. 276
15.4 Operation of Remote Control Transmit Output Generator .......... 279
15.5 Notes on Using Remote Control Transmit Output Generator ........ 280
15.6 Program Example for Remote Control Transmit Output Generator. 282
15.1 Overview of Remote Control Transmit Output Generator

The remote control transmit generator is a 6-bit binary counter that can select one of four clocks as its count clock. Both the cycle of the output waveform and its “H” state pulse width can be set, which allows the circuit to be used as a 6-bit PPG. The circuit uses the same output pin as the buzzer output.

- Remote Control Transmit Output Generation Function
  - Generates frequencies for use by a remote control unit, and outputs the signal at the RCO pin.
  - The cycle and “H” state pulse width of the output waveform can be set separately.
  - The count clock can be selected from four different internal clocks.
  - The frequencies can generate with a cycle among 2 and 2⁶ -1 times the count clock cycle.
  - Shares its output pin with the buzzer output, the buzzer output having precedence.

Table 15.1a lists the available range of “H” state pulse widths.

- Calculation example for the remote control transmit output generator cycle and “H” state pulse width (when a 1/8/32 \( t_{\text{inst}} \) clock is selected for count clock cycle).

Assume a main clock source oscillation (\( F_{CH} \)) of 4.2 MHz, and a 1 \( t_{\text{inst}} \) clock selected for count clock cycle. Also assume main clock mode, and the highest clock speed selected from the system clock control register (SYCC: SCS = CS1 = CS0 = 1). (This makes the instruction cycle time \( 4/F_{CH} \).) Then, for the indicated comparison values, the output waveform cycle and “H” state pulse width can be calculated as follows:

<table>
<thead>
<tr>
<th>Internal count clock cycle</th>
<th>Output cycle</th>
<th>Output “H” pulse width*</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.5 ( t_{\text{inst}} )</td>
<td>1 ( t_{\text{inst}} ) to 31.5 ( t_{\text{inst}} )</td>
<td>0.5 ( t_{\text{inst}} ) to 31.5 ( t_{\text{inst}} )</td>
</tr>
<tr>
<td>1 ( t_{\text{inst}} )</td>
<td>2 ( t_{\text{inst}} ) to 63 ( t_{\text{inst}} )</td>
<td>1 ( t_{\text{inst}} ) to 62 ( t_{\text{inst}} )</td>
</tr>
<tr>
<td>8 ( t_{\text{inst}} )</td>
<td>16 ( t_{\text{inst}} ) to 504 ( t_{\text{inst}} )</td>
<td>8 ( t_{\text{inst}} ) to 496 ( t_{\text{inst}} )</td>
</tr>
<tr>
<td>32 ( t_{\text{inst}} )</td>
<td>64 ( t_{\text{inst}} ) to 2016 ( t_{\text{inst}} )</td>
<td>32 ( t_{\text{inst}} ) to 1984 ( t_{\text{inst}} )</td>
</tr>
</tbody>
</table>

\( t_{\text{inst}} \): instruction cycle (affected by clock mode, etc.)

*: Can also output a steady “L” or “H” state (0% or 100% duty cycle).

If the “H” pulse width setting is equal to or greater than the cycle setting, the output will be a steady “H” state.
Calculation example for the remote control transmit output generator cycle and “H” state pulse width (when a 0.5 t\text{\textsubscript{inst}} clock is selected for count clock cycle)

Assume a main clock source oscillation (F\text{\textsubscript{CH}}) of 4.2 MHz, and a 0.5 t\text{\textsubscript{inst}} clock selected for count clock cycle. Also assume main clock mode, and the highest clock speed selected from the system clock control register (SYCC; SCS = CS1 = CS0 = 1). (This makes the instruction cycle time 4/F\text{\textsubscript{CH}}.) Then, for the indicated comparison values, the output waveform cycle and “H” state pulse width can be calculated as follows:

- Cycle comparison value = 011110 B (30 clock cycles)
- Pulse width comparison value = 001010 B (10 clock cycles)

Cycle = \[(cycle \ comparison \ value + 1) \times count \ clock \ cycle\]
= \[011110 B \times 0.5 \times \frac{4}{F\text{\textsubscript{CH}}}\]
= \[31 \times 0.475 \text{ ms}\]
= \[14.725 \text{ ms}\]

“H” pulse width = \[(“H” \ pulse \ width \ comparison \ value + 1) \times count \ clock \ cycle\]
= \[001010 B \times 0.5 \times \frac{4}{F\text{\textsubscript{CH}}}\]
= \[11 \times 0.475 \text{ ms}\]
= \[5.225 \text{ ms}\]

If the “H” pulse width setting is equal to or greater than the cycle setting, the output will be a steady “H” state.
# 6-bit PPG Function (when a 0.5 \( t_{\text{inst}} \) clock selected for count clock cycle)

Because the cycle and "H" pulse width of its output waveform can be set separately, the remote control generator can be used as a 6-bit PPG. The duty ratio is from 1.56% to 100%. The valid range of "H" pulse width comparison settings, however, is from "0" to the cycle comparison setting. This means that the lower the cycle comparison setting (the shorter the cycle of the output waveform), the lower the resolution (the larger the minimum duty ratio step size).

For a cycle comparison setting of "1," for example, the possible "H" pulse width comparison settings would be "0" and "1" which would result in a resolution of 1/2. The duty ratios for these settings would be 50% and 100%, or a minimum duty ratio step of 50%.

The output cycle and duty ratio are calculated as follows:

\[
\text{Output cycle} = (\text{cycle comparison value} + 1) \times 0.5 \ t_{\text{inst}}.
\]

\[
\text{Duty ratio} (\%) = \left( \frac{\text{"H" pulse width compare value} + 1}{\text{cycle compare value} + 1} \right) \times 100
\]

Table 13.1b shows the available output cycle, resolution and the minimum steps for duty ratio.

---

**Table 13.1b 6-Bit PPG Resolution and Output Cycles (0.5 \( t_{\text{inst}} \) count clock)**

<table>
<thead>
<tr>
<th>Cycle comparison value</th>
<th>&quot;H&quot; pulse width comparison value setting range</th>
<th>Output cycle Count clock = 0.5 ( t_{\text{inst}} )</th>
<th>Resolution</th>
<th>Duty ratio Minimum step</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Output &quot;H&quot;</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>0 to 1</td>
<td>1 ( t_{\text{inst}} )</td>
<td>1/2</td>
<td>50.0%*</td>
</tr>
<tr>
<td>2</td>
<td>0 to 2</td>
<td>1.5 ( t_{\text{inst}} )</td>
<td>1/3</td>
<td>33.3%*</td>
</tr>
<tr>
<td>3</td>
<td>0 to 3</td>
<td>2.0 ( t_{\text{inst}} )</td>
<td>1/4</td>
<td>25.0%*</td>
</tr>
<tr>
<td>4</td>
<td>0 to 4</td>
<td>2.5 ( t_{\text{inst}} )</td>
<td>1/5</td>
<td>20.0%*</td>
</tr>
<tr>
<td>5</td>
<td>0 to 5</td>
<td>3.0 ( t_{\text{inst}} )</td>
<td>1/6</td>
<td>16.7%*</td>
</tr>
<tr>
<td>6</td>
<td>0 to 6</td>
<td>3.5 ( t_{\text{inst}} )</td>
<td>1/7</td>
<td>14.3%*</td>
</tr>
<tr>
<td>7</td>
<td>0 to 7</td>
<td>4.0 ( t_{\text{inst}} )</td>
<td>1/8</td>
<td>12.5%*</td>
</tr>
<tr>
<td>8</td>
<td>0 to 8</td>
<td>4.5 ( t_{\text{inst}} )</td>
<td>1/9</td>
<td>11.1%*</td>
</tr>
<tr>
<td>9</td>
<td>0 to 9</td>
<td>5.0 ( t_{\text{inst}} )</td>
<td>1/10</td>
<td>10.0%*</td>
</tr>
<tr>
<td>10</td>
<td>0 to 10</td>
<td>5.5 ( t_{\text{inst}} )</td>
<td>1/11</td>
<td>9.09%*</td>
</tr>
</tbody>
</table>

| 15                     | 0 to 15                                       | 8.0 \( t_{\text{inst}} \)                       | 1/16       | 6.25%*                 |
| 20                     | 0 to 20                                       | 10.5 \( t_{\text{inst}} \)                      | 1/21       | 4.76%*                 |
| 25                     | 0 to 25                                       | 13.0 \( t_{\text{inst}} \)                      | 1/26       | 3.85%*                 |
| 30                     | 0 to 30                                       | 15.5 \( t_{\text{inst}} \)                      | 1/31       | 3.23%*                 |
| 40                     | 0 to 40                                       | 20.5 \( t_{\text{inst}} \)                      | 1/41       | 2.44%*                 |
| 50                     | 0 to 50                                       | 25.5 \( t_{\text{inst}} \)                      | 1/51       | 1.96%*                 |
| 60                     | 0 to 60                                       | 30.5 \( t_{\text{inst}} \)                      | 1/61       | 1.64%*                 |
| 63                     | 0 to 63                                       | 32 \( t_{\text{inst}} \)                        | 1/64       | 1.56%*                 |

*\( t_{\text{inst}}\): Instruction cycle time

*: Steady "H" output when "H" pulse width comparison value is equal to period comparison value.
6-bit PPG Function (when a 1/8/32 tinst clock selected for count clock cycle)

Because the cycle and "H" pulse width of its output waveform can be set separately, the remote control generator can be used as a 6-bit PPG. The duty ratio is from 1.6% to 100%. The valid range of "H" pulse width comparison settings, however, is from “1” to the cycle comparison setting. This means that the lower the cycle comparison setting (the shorter the cycle of the output waveform), the lower the resolution (the larger the minimum duty ratio step size).

For a cycle comparison setting of “2,” for example, the possible "H" pulse width comparison settings would be “1,” and “2,” which would result in a resolution of 1/2. The duty ratios for these settings would be 50%, and 100%, or a minimum duty ratio step of 50%.

The output cycle and duty ratio are calculated as follows:

Output cycle = cycle comparison value × selected count clock cycle
Duty ratio (%) = ("H" pulse width compare value/cycle compare value) × 100

Table 13.1b shows the available output cycle, resolution and the minimum steps for duty ratio.

<table>
<thead>
<tr>
<th>Cycle comparison value</th>
<th>&quot;H&quot; pulse width comparison value setting range</th>
<th>Output cycle (Count Clock)</th>
<th>Resolution</th>
<th>Duty ratio Minimum step</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>0</td>
<td>Prohibited setting</td>
<td></td>
<td></td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>1 to 2</td>
<td>2 tinst</td>
<td>16 tinst</td>
<td>64 tinst</td>
</tr>
<tr>
<td>3</td>
<td>1 to 3</td>
<td>3 tinst</td>
<td>24 tinst</td>
<td>96 tinst</td>
</tr>
<tr>
<td>4</td>
<td>1 to 4</td>
<td>4 tinst</td>
<td>32 tinst</td>
<td>128 tinst</td>
</tr>
<tr>
<td>5</td>
<td>1 to 5</td>
<td>5 tinst</td>
<td>40 tinst</td>
<td>160 tinst</td>
</tr>
<tr>
<td>6</td>
<td>1 to 6</td>
<td>6 tinst</td>
<td>48 tinst</td>
<td>192 tinst</td>
</tr>
<tr>
<td>7</td>
<td>1 to 7</td>
<td>7 tinst</td>
<td>56 tinst</td>
<td>224 tinst</td>
</tr>
<tr>
<td>8</td>
<td>1 to 8</td>
<td>8 tinst</td>
<td>64 tinst</td>
<td>256 tinst</td>
</tr>
<tr>
<td>9</td>
<td>1 to 9</td>
<td>9 tinst</td>
<td>72 tinst</td>
<td>288 tinst</td>
</tr>
<tr>
<td>10</td>
<td>1 to 10</td>
<td>10 tinst</td>
<td>80 tinst</td>
<td>320 tinst</td>
</tr>
<tr>
<td>15</td>
<td>1 to 15</td>
<td>15 tinst</td>
<td>120 tinst</td>
<td>480 tinst</td>
</tr>
<tr>
<td>20</td>
<td>1 to 20</td>
<td>20 tinst</td>
<td>160 tinst</td>
<td>640 tinst</td>
</tr>
<tr>
<td>25</td>
<td>1 to 25</td>
<td>25 tinst</td>
<td>200 tinst</td>
<td>800 tinst</td>
</tr>
<tr>
<td>30</td>
<td>1 to 30</td>
<td>30 tinst</td>
<td>240 tinst</td>
<td>960 tinst</td>
</tr>
<tr>
<td>40</td>
<td>1 to 40</td>
<td>40 tinst</td>
<td>320 tinst</td>
<td>1280 tinst</td>
</tr>
<tr>
<td>50</td>
<td>1 to 50</td>
<td>50 tinst</td>
<td>400 tinst</td>
<td>1600 tinst</td>
</tr>
<tr>
<td>60</td>
<td>1 to 60</td>
<td>60 tinst</td>
<td>480 tinst</td>
<td>1920 tinst</td>
</tr>
<tr>
<td>63</td>
<td>1 to 63</td>
<td>63 tinst</td>
<td>504 tinst</td>
<td>2016 tinst</td>
</tr>
</tbody>
</table>

tinst: instruction cycle time
*: If "H" pulse width comparison setting is “00H”, a 0.5 tinst long “H” pulse will be outputted. Steady “H” output when “H” pulse width comparison value is equal to period comparison value.
15.2 Block Diagram of Remote Control Transmit Output Generator

The remote control transmit output generator consists of the following five blocks:

- Count clock selector
- 6-bit counter
- Comparator circuit
- Remote control register 1 (RCR1)
- Remote control register 2 (RCR2)

![Block Diagram of Remote Control Transmit Output Generator](image-url)
• **Count clock selector**
  Selects a count-up clock for the 6-bit counter from the four available internal count clocks.

• **6-bit counter**
  The 6-bit counter counts-up, on the count clock selected by the count clock selector. The counter can be cleared by clearing the output enable bit of the RCR2 register (RCR2: RCEN = 0).

• **Comparison circuit**
  The comparison circuit holds a "H" state until the count in the 6-bit counter matches the setting in the "H" pulse width compare register. Then it holds the "L" state until the counter count matches the setting in the cycle compare register, at which time the counter is cleared to all zeros and continues counting.

• **Remote control register 1 (RCR1)**
  RCR1 is used to select the counter clock for remote control transmit output, and set the output "H" pulse width comparison value.

• **Remote control register 2 (RCR2)**
  RCR2 is used to enable/disable outputs for remote control transmit output, and set the output cycle comparison value.
15.3 Structure of Remote Control Transmit Output Generator

The section describes the pin, pin block diagram and register of the remote control transmit output generator.

- **Remote Control Transmit Output Generator Pin**

  The remote control transmit output generator uses the P30/BZ/RCO pin. The pin can function as an output-only port (P30), as the buzzer output pin (BZ), or as the remote control output (RCO).

  **RCO:**

  When the remote control transmit output enable bit is set to “1” (RCR2: RCEN = 1), this pin functions as the remote control transmit output pin, outputting a waveform having a "H" state pulse width and cycle as set. The buzzer output, however, has higher precedence for use of the pin. Therefore, when using the remote control output, always write "00" to the buzzer register (BZCR) to turn off the buzzer output.

- **Block Diagram of Remote Control Transmit Output Generator Pin**

  ![Block Diagram of P30/BZ/RCO Pin](image)

- **Remote Control Transmit Output Generator Registers**

  ![Remote Control Transmit Generator Registers](image)

  **Figure 15.3b Remote Control Transmit Generator Registers**
15.3 Structure of Control Transmit Output Generator

15.3.1 Remote Control Register 1 (RCR1)

Remote control register 1 is used to select the counter clock, and set the "H" pulse width.

- **Remote Control Register 1 (RCR1)**

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0014h</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>00000000h</td>
</tr>
</tbody>
</table>

**Bit 7, Bit 6**  
RCK1, RCK0  
Count clock selection bits

**Bit 5**  
HSC5 to HSC0  
"H" pulse width setting bits

**Bit 4**  
HSC5 to HSC0  
"H" pulse width comparison value

**Bit 3**  
RCK1, RCK0  
Count clock selection bits

**Bit 2**  
R/W : Readable and writable

**Bit 1**  
Inst : Instruction cycle

**Note:**
- When the count clock is 0.5\( \text{t}_{\text{inst}} \)
  - The setting value between "000000" and "111110" (00 to 3E\( \text{H} \)) is always set a value less than the cycle comparison value. If they are set equal to or greater than the setting, a steady "H" is output.
- When the count clock is 1/8/32\( \text{t}_{\text{inst}} \)
  - The setting value between "000001" and "111110" (01 to 3E\( \text{H} \)) is always set a value less than the cycle comparison value. If the value is set to "000000", a 0.5\( \text{t}_{\text{inst}} \) long "H" pulse will be outputted. If they are set equal to or greater than the setting, a steady "H" is output.

**Table 15.3.1a Remote Control Register 1 (RCR1) Bits**

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7, Bit 6</td>
<td>RCK2, RCK1: Count clock selection bits</td>
</tr>
<tr>
<td>Bit 5</td>
<td>HSC5 to HSC0: &quot;H&quot; pulse width setting bits</td>
</tr>
<tr>
<td>Bit 4, Bit 3, Bit 2, Bit 1, Bit 0</td>
<td>These bits set the number of counts for which the remote control output generator output is to remain &quot;H&quot;. (The &quot;H&quot; pulse width comparison value to be matched by the count in the counter.) Note:</td>
</tr>
<tr>
<td></td>
<td>• When the count clock is 0.5( \text{t}_{\text{inst}} )</td>
</tr>
<tr>
<td></td>
<td>- The setting value between &quot;000000&quot; and &quot;111110&quot; (00 to 3E( \text{H} )) is always set a value less than the cycle comparison value. If they are set equal to or greater than the setting, a steady &quot;H&quot; is output.</td>
</tr>
<tr>
<td></td>
<td>• When the count clock is 1/8/32( \text{t}_{\text{inst}} )</td>
</tr>
<tr>
<td></td>
<td>- The setting value between &quot;000001&quot; and &quot;111110&quot; (01 to 3E( \text{H} )) is always set a value less than the cycle comparison value. If the value is set to &quot;000000&quot;, a 0.5( \text{t}_{\text{inst}} ) long &quot;H&quot; pulse will be outputted. If they are set equal to or greater than the setting, a steady &quot;H&quot; is output.</td>
</tr>
</tbody>
</table>
15.3 Structure of Control Transmit Output Generator

15.3.2 Remote Control Register 2 (RCR2)

Remote control register 2 is used to enable/disable outputs, and set the output cycle period.

Remote Control Register 2 (RCR2)

Remote control register 2 is used to enable/disable outputs, and set the output cycle period.

Remote Control Register 2 (RCR2)

Table 15.3.2a Remote Control Register 2 (RCR2) Bits

<table>
<thead>
<tr>
<th>Address (0015h)</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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>RCEN — SCL5 SCL4 SCL3 SCL2 SCL1 SCL0</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>0x00000000</td>
</tr>
</tbody>
</table>

RCEN: Output enable bit

- **0**: Disable output; clear counter.
- **1**: Enable output; start counter.

SCL5 to SCL0: Cycle setting bits

These bits set the length of the output cycle in terms of counter counts. (The cycle period comparison value to be matched by the count in the counter.)

**Note:**

- When the count clock is 0.5 \( t_{\text{inst}} \):
  - The setting value is between "000001" and "111111" (01 to 3FH). The RCO output will remain the previous state until the cycle comparison value is matched by the count value in the counter. Then the RCO output will output "H" state.
- When the count clock is 1/8/32 \( t_{\text{inst}} \):
  - The setting value is between "000010" and "111111" (02 to 3FH). The RCO output will remain the previous state until the cycle comparison value is matched by the count value in the counter. Then the RCO output will output "H" state.

If this setting value is set to "01" and the "H" pulse comparison value is set to "00", RCO will output a 0.5 \( t_{\text{inst}} \) long "H" pulse.
### 15.4 Operation of Remote Control Transmit Output Generator

The remote control transmit output generator (6-bit PPG) generates a remote control transmit output in which the cycle and "H" state pulse width of the output can be set separately.

#### Operation of Remote Control Transmit Output Generator

Figure 15.4a shows the settings required to operate the remote control transmit output generator.

<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>RCR1</td>
<td>RCK1</td>
<td>RCK0</td>
<td>HSC5</td>
<td>HSC4</td>
<td>HSC3</td>
<td>HSC2</td>
<td>HSC1</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>RCR2</td>
<td>RCEN</td>
<td>—</td>
<td>SCL5</td>
<td>SCL4</td>
<td>SCL3</td>
<td>SCL2</td>
<td>SCL1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>BZCR</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>—</td>
<td>BZ2</td>
<td>BZ1</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Q: Used bit</th>
</tr>
</thead>
<tbody>
<tr>
<td>1: Set &quot;1&quot;.</td>
</tr>
<tr>
<td>0: Set &quot;0&quot;.</td>
</tr>
</tbody>
</table>

**Figure 15.4a Remote Control Transmit Output Generator Settings**

When the remote control transmit output generator output is enabled, its 6-bit counter starts counting up from "zero" in sync with the selected count clock. The RCO pin starts out "H", and remains in that state until the count in the counter matches the "H" state pulse width comparison value. When this happens, the RCO pin goes "L" and stays there until the count matches the cycle period comparison value, at which time the 6-bit counter is cleared to zero and continues to count. The fact that the pulse width and period can be set separately enables the circuit to be used as a 6-bit PPG.

Figure 15.4b shows the operation of the remote control transmit output generator.

**Figure 15.4b Operation of Remote Control Transmit Output Generator**

- **Counter count**
  - **Cycle setting value** (RCR2: SCL0 to SCL5)
  - **High pulse width setting value** (RCR1: HSC0 to HSC5)

- **Period**
  - When count clock is 0.5 \( f_{	ext{inst}} \)
    - \( \text{Period} = 0.5 f_{\text{inst}} \times (\text{Period comparison value} + 1) \)
  - When count clock is 1/8/32 \( f_{\text{inst}} \)
    - \( \text{Period} = \text{Count clock period} \times \text{Period comparison value} \)

- **Pulse width**
  - When count clock is 0.5 \( f_{\text{inst}} \)
    - \( \text{Pulse width} = 0.5 f_{\text{inst}} \times (\text{High pulse width comparison value} + 1) \)
  - When count clock is 1/8/32 \( f_{\text{inst}} \)
    - \( \text{Pulse width} = \text{Count clock period} \times \text{High pulse width comparison value} \)
15.5 Notes on Using Remote Control Transmit Output Generator

This section lists points to note when using the remote control unit transmit output generator.

- "H" pulse width restrictions
  Generally, the "H" pulse width setting bits of remote control register 1 (RCR1: HSC5~0) must always be set less than the cycle setting bits of remote control register 2 (RCR2: SCL5~0).
  For any clock, when SCL5~0 is “00H”, HSC5~0 is equal to SCL5:0, or HSC5~0 is larger than SCL5~0, the RCO pin will remain a steady "H" level.
  For any clock, when HSC5~0 is “00H” and SCL5~0 is not equal to “00H”, the RCO pin will start to output "L" level and then output a 0.5 \(t_{\text{inst}}\) long "H" pulse after every cycle match is detected.

- Resolution
  When count clock is 0.5 \(t_{\text{inst}}\), the maximum "H" pulse width resolution is 1/64 of the cycle (the resolution when the cycle setting is “111111” (3FH)). Reducing the time of the cycle reduces the resolution, with the minimum resolution of 1/2 occurring with a cycle setting of “000001” (01H). When count clock is 1/8/32 \(t_{\text{inst}}\), the maximum "H" pulse width resolution is 1/63 of the cycle (the resolution when the cycle setting is “111111” (3FH)). Reducing the time of the cycle reduces the resolution, with the minimum resolution of 1/2 occurring with a cycle setting of “000010” (02H).

- Changing settings during operation
  Direct comparisons are performed between the 6-bit counter of the remote control transmit generator and the "H" pulse width setting bits (RCR1: HSC5 to HSC0), and between the counter and the cycle setting bits (RCR2: SCL5 to SCL0). Therefore, if a setting is reduced in mid-count, the cycle time \(t_1\) may be long until the counter overflows in the worst case and the takes effect in the next cycle. Similarly, the "H" state pulse width \(t_2\) may be long in the worst case until the next end-of-cycle match is detected.

Figure 15.5a illustrates what happens when settings are changed during remote control transmit output generator operation.
Errors

Activating the counter by program is not synchronized with the start of counting-up using the selected count clock. Therefore, the time from activating the counter until a match with the “H” pulse comparison value and Cycle comparison value are detected may be shorter than the theoretical time by a maximum of one cycle of the count clock.

Figure 15.5b shows the error that occurs on starting counter operation.

![Figure 15.5a Changing Settings during Operation (Remote Control)](image)

*1: Since the current count in the counter is less than the new setting, the new setting takes effect for the current cycle.
*2: Since cycle is changed to a value less than the current count, the counter counts all the way to counter overflow and takes effect in the next cycle (in the worst case).
*3: Since the new “H” pulse width setting is less that the current count, the pulse width match is not detected until the next cycle (in the worst case).
*4: The length of the t1 and t2 is depended on when the new cycle setting value and “H” pulse width setting value. It will vary from case to case.

![Figure 15.5b Error during activating Operation (Remote Control)](image)
15.6 Program Example for Remote Control Transmit Output Generator

This section gives a program example for the remote control transmit output generator.

- Programming Example for Remote Control Transmit Output
  - Processing description
    - Generate the remote control transmit Output at a cycle of approximately 28.6 µs and a 33% duty ratio.
    - With a main clock frequency \((F_{CH})\) of 4.2 MHz, the "H"est clock speed (speed shift function), and the 1 tinst clock selected (1 instruction cycle time = \(4/F_{CH}\)), the comparison value for a cycle of approximately 28.6 µs is as found as follows:
      
      \[
      \text{Cycle comparison value (RCR2: SCL5 to SCL0)} = \frac{28.6 \mu s}{1 \times 4/4.2 \text{ MHz}} = 30
      \]
      
    - The comparison value for a "H" state pulse width to provide a 33% duty ratio is found as follows:
      
      "H" pulse width comparison value (RCR1: HSC5 to HSC0) = \(\frac{33}{100} \times \text{cycle comparison value} = 0.33 \times 30 = 10\)
      
      (This is an approximately 9.5 µs "H" pulse width.)
  - Coding example
    
    BZCREQU0010H ; Buzzer register
    RCR1EQU0014H ; Remote control register 1
    RCR2EQU0015H ; Remote control register 2
    
    CSEG ; [CODE SEGMENT]
    
    MOV BZCR,#00000000B ; Disable buzzer output.
    MOV RCR1,#01001010B ; Select 1 tinst count clock, set "H" pulse width comparison value.
    MOV RCR2,#10011110B ; Enable output and start counter, set cycle period, comparison value.
    
    ENDS
  
  END
CHAPTER 16

LCD CONTROLLER/DRIVER

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

16.1 Overview of LCD Controller/Driver ......................... 284
16.2 Block Diagram of LCD Controller/Driver ................... 286
16.3 Structure of LCD Controller/Driver ......................... 294
16.4 Operation of LCD Controller/Driver ......................... 300
16.5 Program Example for LCD Controller/Driver .............. 308
16.1 Overview of LCD Controller/Driver

The LCD controller/driver includes 12 bytes of on-chip display data in memory, the contents of which control an LCD display via 24 segment and 4 common outputs. The function can drive an LCD display panel directly, using one of three selectable duty ratios.

### LCD Controller/Driver Function

The LCD controller/driver function displays the contents of a display data memory directly to the LCD panel (liquid crystal display) by segment and common outputs.

- Built-in booster (MB89160A only)
- Built-in driving resistor for LCD driving voltage. Can be connected the external divider resistor (MB89160 and MB89160L only).
- Up to 24 segment outputs (SEG0 to SEG23) and four common outputs (COM0 to COM3) may be used.
- Built-in display RAM: 12 bytes (24 \( \times \) 4 bits)
- Three selectable duty ratios (1/2, 1/3, and 1/4). Not all duty ratios are available with all bias settings, however.
- Either the main or subclock can be selected as the drive clock.
- LCD can be driven directly.

Table 16.1a shows the duty ratios available with each bias setting.

<table>
<thead>
<tr>
<th>Part number</th>
<th>Bias</th>
<th>1/2 duty ratio</th>
<th>1/3 duty ratio</th>
<th>1/4 duty ratio</th>
</tr>
</thead>
<tbody>
<tr>
<td>MB89160 &amp; MB89160L Series</td>
<td>1/2 bias</td>
<td>◯</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td></td>
<td>1/3 bias</td>
<td>×</td>
<td>◯</td>
<td>◯</td>
</tr>
<tr>
<td>MB89160A Series</td>
<td>1/2 bias</td>
<td>×</td>
<td>×</td>
<td>×</td>
</tr>
<tr>
<td>(internal voltage</td>
<td>1/3 bias</td>
<td>×</td>
<td>◯</td>
<td>◯</td>
</tr>
<tr>
<td>booster)</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

◯: Recommended mode
×: Do not use

**Check:**
- The single-clock option is not available in devices that have internal voltage boosters (MB89160A series).
- The 1/2 bias mode cannot be used with devices that have internal voltage boosters (MB89160A series) because it requires an external divider resistor.
- If P60/SEG8 to P67/SEG15, P40/SEG16 to P47/SEG23, P70/COM2, and P71/COM3 are set as output-only port pins (general-purpose outputs) as a mask option, they cannot be used as LCD segment and common outputs.
- When P70/COM2 and P71/COM3 are used as output-only port pins, the 1/3 and 1/4 duty ratio output modes cannot be used.
Memo
16.2 Block Diagram of LCD Controller/Driver

The LCD controller/driver is made up of the eight blocks listed below. Functionally, the circuit can be broken into two major sections: the controller section, which generates LCD segment and common signals based on the current contents of display RAM, and the driver section, which develops sufficient drive to operate the display.

- LCDC control register 1 (LCR1)
- Display RAM
- Prescaler
- Timing controller
- V/I converter
- Common output driver
- Segment output driver
- Booster (MB89160A series) or divider resistors (MB89160 & MB89160L series)

Figure 16.2a Block Diagram of LCD Controller/Driver

*: This block is a voltage booster only in MB89160A series devices. In MB89160 series devices, it contains on-chip divider resistors.
● LCDC control register 1 (LCR1)
   This register is used to select the frame clock (the clock used to generate the frame cycle), enable/disable operation in watch mode, control the LCD drive supply voltage, select display blanking/non-blanking, select the display mode, and select the LCD clock cycle.

● Display RAM
   This 24 × 4-bit block of RAM controls the segment output signals. Its contents are automatically read out to the segment outputs in sync with the timing of the selected common signal.

● Prescaler
   The prescaler selects settings from 2 clocks and 4 frequencies to generate the frame frequency.

● Timing controller
   This block controls the segment and common signals based on the frame frequency and LCR1 register settings.

● V/I converter
   This circuit generates alternating current waveforms from the voltage signals it receives from the timing controller to drive the LCD.

● Common output driver
   Contains the drivers for the LCD common pins.

● Segment output driver
   Contains the drivers for the LCD segment pins.

● Voltage booster (MB89160A series only)
   The voltage booster develops the LCD drive voltage. It generates a voltage 2 to 3 times that of the reference voltage. Either an internal 1.5 V reference voltage or an external reference voltage can be used. Devices with voltage boosters do not have on-chip divider resistors, in which case V0 to V3, C0, and C1 pins are all used to connect external capacitors.

● Divider resistors (MB89160 & MB89160L series only)
   The LCD drive voltage can be developed across a voltage divider made up of these internal resistors. External resistors may also be used. When a device without an internal voltage booster is selected, the V0 to V3 pins are used to connect divider resistors and C0 and C1 are used as output-only port pins (P32 and P33).

● LCD Controller/Driver Supply Voltage

   ● Devices with internal voltage boosters (MB89160A series only)
     The LCD driver supply voltage is generated by the internal voltage booster, which boosts a reference voltage supplied either via the V1 pin or by an internal reference voltage generator.

   ● Devices without internal voltage boosters (MB89160 & MB89160L series only)
     The LCD driver supply voltage is taken from a voltage divider. The divider can be made up of internal or external resistors connected to the V0 to V3 pins.
16.2 Block Diagram of LCD Controller/Driver

16.2.1 LCD Controller/Driver Power Supply Voltage (Devices with Internal Voltage Booster)

In devices that have an internal voltage booster, the supply voltage for the LCD driver is developed by the reference voltage generator and voltage booster. V0 to V3, C0, and C1 pins are used to connect external capacitors. V1 can also be used to connect an external reference voltage.

- Devices with Internal Voltage Boosters (MB89160A Series)

  Only MB89160A series devices have on-chip voltage boosters and reference voltage generators.

  - Reference voltage generator

    The reference voltage generator outputs a constant 1.5 V regardless of variations in the operating voltage. If desired, instead of the internal reference voltage supply, an external circuit can be used to supply the reference voltage to the voltage booster. The external reference voltage is connected to the V1 pin.

    Initially, the voltage booster is connected to the internal reference voltage generator. If you are using an external reference voltage, be sure to disconnect the internal reference voltage generator from the booster (Set LCR1: VSEL = 1).

  - Voltage booster

    From a 32 kHz input clock (subclock) and the reference voltage, the internal voltage booster configured as shown in Figure 16.2.1a generates a voltage equal to twice or three times the reference voltage (depending on the switch position). Since the circuit uses the subclock, it does not work in modes in which the subclock is stopped (during sub-stop mode, etc.). Similarly, it cannot be used in devices for which the single-clock option is selected. When the voltage booster is used, capacitors as shown in Figure 16.2.1a should be connected to pins V0 to V3, C0, and C1. Since the voltage booster cannot generate 1/2 bias voltage, do not select 1/2 duty ratio output mode (LCR1: MS1, MS0 = 01b).
Figure 16.2.1a Reference Voltage Generator/Voltage Booster External Connections

Check: When applying an external reference voltage at the V1 pin, check the applicable data sheet for information on the maximum voltage, etc. to be used.
16.2 Block Diagram of LCD Controller/Driver

16.2.2 LCD Controller/Driver Internal Divider Resistors (Devices without Voltage Boosters)

In devices that have internal divider resistors instead of a voltage booster, the LCD driver supply voltage is taken from an internal voltage divider. (External divider resistors may also be used.)

- **Internal Divider Resistors (MB89160 & MB89160L Series)**

  Devices that do not have internal voltage boosters have internal divider resistors instead. In these devices, external divider resistors may also be connected at pins V0 through V3.

  The selection of internal or external resistors is made by the drive supply voltage control bit of LCDC control register 1 (LCR1: VSEL). VSEL = 1 connects the internal resistors. Set VSEL to "1" when you want to use the internal resistors only (when no external resistors are connected).

  The LCDC enable is inactive when LCD operation is stopped (LCR1: MS1 = MS0 = 00B), and when operation is stopped (LCR1: LCEN = 0) in watch mode (STBC: TMD = 1). Pin V2 and V1 should be shorted together when using the 1/2 bias setting.

  Figure 16.2.2a shows an equivalent circuit of the internal voltage divider.

  ![Internal Voltage Divider Equivalent Circuit](image)

  **Vo to V3**: Voltages at V0 to V3 pins.

  **Figure 16.2.2a Internal Voltage Divider Equivalent Circuit**
Use of Internal Voltage Divider Resistors

Figure 16.2.2b shows the voltage divider circuits for 1/2 and 1/3 bias. As shown in this figure, in the 1/2 bias mode (with LCDC enabled) V2 and V1 will be 1/2 of V3 (V3 is the LCD operating voltage, which is VCC/2 in this configuration). In the 1/3 bias mode, V1 is 1/3 of V3, and V2 is 2/3 of V3.

![Figure 16.2.2b Use of Internal Voltage Divider Resistors](image)

Display Brightness Adjustment when Internal Divider Resistors Are Used

When internal divider resistors do not provide sufficient LCD display brightness, connect an external brightness adjust variable resistor between VCC and V3 as shown in Figure 16.2.2c.

![Figure 16.2.2c Use of Internal Voltage Divider Resistors with Brightness Adjustment](image)

Note: During LCD operation, the 2R internal resistance will be in the divider circuit, and VR will be in parallel with this resistor.
16.2 Block Diagram of LCD Controller/Driver

16.2.3 LCD Controller/Driver External Divider Resistors (Devices without Internal Voltage Boosters)

External voltage divider resistors can also be used with devices that have internal divider resistors (devices without internal voltage boosters). Display brightness can be adjusted by a variable resistor connected between the Vcc and V3 pins.

External Divider Resistors (MB89160 & MB89160L Series)

When you are using a device without a voltage booster, but do not wish to use the internal divider resistors, external voltage divider resistors can be connected at the LCD drive voltage supply pins (V0 to V3). Figure 16.2.3a shows connections for external divider resistors for the two biasing modes, and Table 16.2.3a lists the corresponding LCD drive voltages.

![Figure 16.2.3a External Voltage Divider Resistor Connections](image)

<table>
<thead>
<tr>
<th>Table 16.2.3a LCD Drive Voltages and Biasing Modes</th>
</tr>
</thead>
<tbody>
<tr>
<td>V3</td>
</tr>
<tr>
<td>------------------</td>
</tr>
<tr>
<td>1/2 bias</td>
</tr>
<tr>
<td>1/3 bias</td>
</tr>
</tbody>
</table>

V0 to V3: Voltages at pins V0 to V3.
V_LCD: LCD operating voltage
Using External Divider Resistors

Internally, the V0 pin is connected through a transistor to VSS (GND). Therefore, when external voltage divider resistors are used, the current flow to the external resistors with the LCD controller off can be cut off by connecting the VSS end of the divider to the V0 pin only. Figure 16.2.3b shows an external voltage divider resistor connection.

1. To preclude the external voltage divider from being affected by the internal divider resistors, the LCDC control register drive voltage control bit (LCR1: VSEL) must be written to “0” to isolate it from the entire internal voltage divider.

2. With the internal voltage divider thus isolated, writing the display mode select bits (MS1 and MS0) of the LCR1 register to any state other than “00" will turn on the LCDC enable transistor (Q1), causing current to flow in the external divider resistors.

3. Writing the display mode select bits (MS1 and MS0) to “00s” will turn off the LCDC enable transistor (Q1), and current will stop flowing in the external resistors.

**Note:** The resistance of RX in the external divider depends on the LCD used. Select an appropriate value.

---

**Figure 16.2.3b External Voltage Divider Resistor Connections**

(1) To preclude the external voltage divider from being affected by the internal divider resistors, the LCDC control register drive voltage control bit (LCR1: VSEL) must be written to “0” to isolate it from the entire internal voltage divider.

(2) With the internal voltage divider thus isolated, writing the display mode select bits (MS1 and MS0) of the LCR1 register to any state other than “00s” will turn on the LCDC enable transistor (Q1), causing current to flow in the external divider resistors.

(3) Writing the display mode select bits (MS1 and MS0) to “00s” will turn off the LCDC enable transistor (Q1), and current will stop flowing in the external resistors.

**Note:** The resistance of RX in the external divider depends on the LCD used. Select an appropriate value.
16.3 Structure of LCD Controller/Driver

This section describes the pins, pin block diagrams, register, and display RAM of the LCD controller/driver.

**LCD Controller/Driver Pins**

The LCD controller/driver uses 4 common output pins (COM0 to COM3), 24 segment output pin (SEG0 to SEG23) and 2 external capacitor pins (C0 and C1), and 4 LCD driving power supply pins (V0 to V3).

- **COM0, COM1, P70/COM2, and P71/COM3 Pins**
  
  P70/COM2 and P71/COM3 pins can function either as output-only ports (P70 and P71) and LCD common output pins (COM2 and COM3). The selection, however is made as a mask option.

  **Check:** When the pins are used as LCD common outputs, the corresponding port data register bits (PDR7: bits 0 and 1) should be set to “1” to turn the output transistor “off.” (COM0 and COM1 are dedicated LCD common output pins.)

- **SEG0 to SEG7, P60/SEG8 to P67/SEG15, and P40/SEG16 to P47/SEG23**

  P60/SEG8 to P67/SEG15 and P40/SEG16 to P47/SEG23 pins can function either as output-only ports (P60 to P67 and P40 to P47) and LCD segment output pins (SEG8 to SEG15 and SEG16 to SEG23). The selection, however is made as a mask option.

  **Check:** When these pins are used as LCD segment outputs, the corresponding port data registers (PDR6 and PDR4) should be set to all “1s” to turn the output transistors off. (SEG0 to SEG7 are dedicated LCD segment output pins.)

- **P33/C0 and P32/C1**

  In devices that have internal voltage boosters (MB89160A series), these pins (C0 and C1) are used to connect the voltage booster capacitors. In devices that do not have voltage boosters, these pins function as output-only port pins P33 and P32.

  **Check:** When these pins are used to connect external capacitors, the corresponding port data register bits (PDR3, bits 2 and 3) should be written to “1” to turn off the output transistors at the pins.

- **V0 to V3**

  In devices that have voltage boosters (MB89160A series), these pins are used to connect external capacitors. In devices without boosters, they are the LCD driving power supply pins.

**Block Diagrams of LCD Controller/Driver Pin**

![Figure 16.3a Block Diagram of LCD Controller/Driver Pin (Dedicated Common/Segment Output Pins)](image-url)
Figure 16.3b Block Diagram of LCD Controller-Driver Pin (Dual Function Common/Segment Output Pins)

Check: Do not select the pull-up resistor option on pins used for common or segment outputs or capacitor connection pins.

**LCD Controller/Driver Register**

<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>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>0072h</td>
<td>CSS</td>
<td>LCEN</td>
<td>VSEL</td>
<td>BK</td>
<td>MS1</td>
<td>MS0</td>
<td>FP1</td>
<td>FP0</td>
<td>00010000b</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>

R/W : Readable and writable
X : Indeterminate

**LCD Controller/Driver RAM**

LCD controller/driver has 24 × 4-bit of internal display RAM in which the data used to generate the segment output signals is stored.
16.3 Structure of LCD Controller/Driver

16.3.1 LCDC Control Register (LCR1)

LCDC Control Register 1 (LCR1) is used to select the frame cycle, enable/disable operation in watch mode, control the LCD drive supply voltage, select display blanking/non-blanking, and select the display mode.

<table>
<thead>
<tr>
<th>Address</th>
<th>CSS</th>
<th>LCEN</th>
<th>VSEL</th>
<th>BK</th>
<th>MS1</th>
<th>MS0</th>
<th>FP1</th>
<th>FP0</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>90H</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>00010000b</td>
</tr>
</tbody>
</table>

- **FP1** Frame cycle Selection bits
  - 00: Main clock (CSS=0), Subclock (CSS=1)
  - 01: Main clock (CSS=2), Subclock (CSS=1)
  - 10: Main clock (CSS=4), Subclock (CSS=1)
  - 11: Main clock (CSS=6), Subclock (CSS=1)

- **FP0** Frame cycle Selection bits
  - 00: Main clock (CSS=0), Subclock (CSS=1)
  - 01: Main clock (CSS=2), Subclock (CSS=1)
  - 10: Main clock (CSS=4), Subclock (CSS=1)
  - 11: Main clock (CSS=6), Subclock (CSS=1)

Frame cycle Selection bits:

<table>
<thead>
<tr>
<th>Frame cycle</th>
<th>Main clock (CSS=0)</th>
<th>Subclock (CSS=1)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0000</td>
<td>256 Hz</td>
<td>256 Hz</td>
</tr>
<tr>
<td>0001</td>
<td>128 Hz</td>
<td>128 Hz</td>
</tr>
<tr>
<td>0010</td>
<td>64 Hz</td>
<td>64 Hz</td>
</tr>
<tr>
<td>0011</td>
<td>32 Hz</td>
<td>32 Hz</td>
</tr>
<tr>
<td>0100</td>
<td>16 Hz</td>
<td>16 Hz</td>
</tr>
<tr>
<td>0101</td>
<td>8 Hz</td>
<td>8 Hz</td>
</tr>
<tr>
<td>0110</td>
<td>4 Hz</td>
<td>4 Hz</td>
</tr>
<tr>
<td>0111</td>
<td>2 Hz</td>
<td>2 Hz</td>
</tr>
<tr>
<td>1000</td>
<td>1 Hz</td>
<td>1 Hz</td>
</tr>
<tr>
<td>1001</td>
<td>0.5 Hz</td>
<td>0.5 Hz</td>
</tr>
<tr>
<td>1010</td>
<td>0.25 Hz</td>
<td>0.25 Hz</td>
</tr>
<tr>
<td>1011</td>
<td>0.125 Hz</td>
<td>0.125 Hz</td>
</tr>
<tr>
<td>1100</td>
<td>0.0625 Hz</td>
<td>0.0625 Hz</td>
</tr>
<tr>
<td>1101</td>
<td>0.03125 Hz</td>
<td>0.03125 Hz</td>
</tr>
<tr>
<td>1110</td>
<td>0.015625 Hz</td>
<td>0.015625 Hz</td>
</tr>
<tr>
<td>1111</td>
<td>0.0078125 Hz</td>
<td>0.0078125 Hz</td>
</tr>
</tbody>
</table>

- **MS1**, **MS0** Display mode selection bits
  - 00: Stop LCD operation
  - 01: 1/2 duty ratio output mode (time division N = 2)
  - 10: 1/3 duty ratio output mode (time division N = 3)
  - 11: 1/4 duty ratio output mode (time division N = 4)

- **BK** Display blanking selection bit
  - 0: Display unblanked
  - 1: Display blanked

- **VSEL** Drive supply voltage control bit
  - 0: External divider resistors used (internal divider resistors isolated)
  - 1: Internal divider resistors used

- **LCEN** Watch mode operation enable bit
  - 0: Stop in watch mode
  - 1: Run in watch mode (also)

- **CSS** Frame cycle generate clock selection bit
  - 0: Main clock
  - 1: Subclock

Figure 16.3.1a LCDC Control Register 1 (LCR1)
Table 16.3.1a  LCDC Control Register (LCR1) Bit Functions

<table>
<thead>
<tr>
<th>Bit</th>
<th>Function</th>
</tr>
</thead>
<tbody>
<tr>
<td>Bit 7</td>
<td>CSS: Frame cycle generation clock selection bit</td>
</tr>
<tr>
<td></td>
<td>Selects the frame clock, which generates the frame cycle for LCD display. “0” selects the output of the timebase timer derived from the main clock divided by ( F_{CL} ), “1” selects the subclock as the frame clock.</td>
</tr>
<tr>
<td></td>
<td>Check: The timebase timer output may not be selected as the frame clock in the main-stop and subclock modes because the main clock oscillator is stopped in those modes.</td>
</tr>
<tr>
<td></td>
<td>Note: When the timebase timer output is selected, the frame clock is not affected when clock speed is changed via the speed shift function. (The timebase timer’s count clock is not supplied through the speed shift function.)</td>
</tr>
<tr>
<td>Bit 6</td>
<td>LCEN: Watch mode operation enable bit</td>
</tr>
<tr>
<td></td>
<td>Determines whether the LCD controller/driver will operate in watch mode. If this bit is “1,” the LCD display will continue to operate after the system goes to watch mode; if it is “0,” the LCD will cease operation.</td>
</tr>
<tr>
<td></td>
<td>Check: To use the display in watch mode, the subclock must be selected as the frame clock (CSS = 1).</td>
</tr>
<tr>
<td>Bit 5</td>
<td>VSEL: LCD drive supply voltage control bit</td>
</tr>
<tr>
<td></td>
<td>• In devices that have an internal divider resistor, the VSEL bit controls the divider current path continuity. A “1” in this bit completes the divider current path; a “0” opens it. This bit must be “0” when external divider resistors are used.</td>
</tr>
<tr>
<td></td>
<td>• In devices with internal voltage boosters, this bit deselects the internal reference voltage generator when an external reference voltage source is used. A “1” deselects the internal reference voltage generator, “0” connects it to the booster. This bit must be written to “1” before applying the external voltage.</td>
</tr>
<tr>
<td>Bit 4</td>
<td>BK: Display blanking selection bit</td>
</tr>
<tr>
<td></td>
<td>Blanks/unblanks the LCD. Setting this bit to “1” (blank) outputs a “deselect” waveform to the LCD segments (which blanks the display).</td>
</tr>
<tr>
<td>Bit 3</td>
<td>MS1, MS0: Display mode selection bits</td>
</tr>
<tr>
<td>Bit 2</td>
<td>Select one of three output waveform duty ratio modes. The mode selected affects the common pins used. Setting both bits to “0” turns “off” the display (stops LCD controller/driver display operation).</td>
</tr>
<tr>
<td></td>
<td>Check: Before going to a mode in which the selected frame cycle generate clock oscillator is stopped (stop mode, etc.), these bits should be written to “00s” to turn off the display.</td>
</tr>
<tr>
<td>Bit 1</td>
<td>FP1, FP0: Frame cycle selection bits</td>
</tr>
<tr>
<td>Bit 0</td>
<td>These bits select one of four LCD display frame cycles.</td>
</tr>
<tr>
<td></td>
<td>Check: To determine this register setting, calculate the optimum frame frequency for the LCD module you are using. Note that the frame cycle is a function of main clock frequency.</td>
</tr>
</tbody>
</table>
16.3 Structure of LCD Controller/Driver

16.3.2 Display RAM

Display RAM consists of 24 × 4-bit (12 bytes) of display data memory used to generate the segment output signals.

- Display RAM and Output Pins

The contents of display RAM are automatically read out and output via the segment outputs in sync with the selected common signal timing. A “1” bit is converted to a “select” (display on) voltage and a “0” to a “deselect” (display off) voltage. Since the operation of the LCD is not directly related to the operation of the CPU, display RAM read/write timing can be set by the user. The SEG8 to SEG23 pins that are not made dedicated segment outputs by mask option selection may be used as general-purpose output-only port pins, and the RAM that goes with those pins may be used as regular RAM. (See Table 16.3.2a.)

Table 16.3.2b shows the relationship between duty ratio mode, common outputs, and display RAM.

Figure 16.3.2a shows which display RAM bits are associated with each segment and common output pin.
Table 16.3.2a Segment Outputs, Display RAM Locations, and Sharing Port Pins

<table>
<thead>
<tr>
<th>Segment/Common Output Pins Used (Mask Option)</th>
<th>Corresponding Display RAM Area</th>
<th>General-Purpose Ports Sharing Same Pins</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEG0 to SEG7 (8 pins) COM0, COM1</td>
<td>60H to 63H</td>
<td>P60 to P67, P40 to P47, P70, P71 (16 + 2 pins)</td>
</tr>
<tr>
<td>SEG0 to SEG11 (12 pins) COM0, COM1</td>
<td>60H to 65H</td>
<td>P64 to P67, P40 to P47, P70, P71 (12 + 2 pins)</td>
</tr>
<tr>
<td>SEG0 to SEG15 (16 pins) COM0 to COM3</td>
<td>60H to 67H</td>
<td>P40 to P47 (8 pins)</td>
</tr>
<tr>
<td>SEG0 to SEG19 (20 pins) COM0 to COM3</td>
<td>60H to 69H</td>
<td>P40 to P47 (4 pins)</td>
</tr>
<tr>
<td>SEG0 to SEG23 (24 pins) COM0 to COM3</td>
<td>60H to 6BH</td>
<td>None</td>
</tr>
</tbody>
</table>

Note: Locations in the display RAM area that are not required for display data can be used as regular RAM.

Table 16.3.2b Common Outputs and Display RAM Bits Used in Each Duty Ratio Mode

<table>
<thead>
<tr>
<th>Duty Ratio Setting</th>
<th>Common Outputs Used</th>
<th>Display Data Bit Used</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td>bit7 bit6 bit5 bit4 bit3 bit2 bit1 bit0</td>
</tr>
<tr>
<td>1/2</td>
<td>COM0 to COM1 (2 pins)</td>
<td>— — ◦ ◦ — — ◦ ◦</td>
</tr>
<tr>
<td>1/3*</td>
<td>COM0 to COM2 (3 pins)</td>
<td>— ◦ ◦ ◦ — ◦ ◦ ◦</td>
</tr>
<tr>
<td>1/4*</td>
<td>COM0 to COM3 (4 pins)</td>
<td>◦ ◦ ◦ ◦ ◦ ◦ ◦ ◦</td>
</tr>
</tbody>
</table>

◦: Used
—: Not used
*: The LCD common output option must be selected for pins COM2 and COM3 (mask option).
16.4 Operation of LCD Controller/Driver

The LCD controller/driver provides the necessary control and drive for an LCD display.

- Operation of LCD Controller/Driver

Figure 16.4a shows the settings required to operate the LCD display.

![LCD Controller/Driver Settings](image)

Once the above settings have been made, if the selected clock for frame cycle generation is running, LCD panel driving waveforms reflecting the contents of display RAM will be output at the segment and common output pins (COM0 to COM3 and SEG0 to SEG23).

Although the clock for frame period generation can be switched even while the LCD is displaying data, the display may flicker when the switching occurs. This can be avoided by temporarily blanking the display (LCR1: BK = 1), etc. while switching.

The display driving output is a two-frame a.c. waveform for which the bias level and display duty cycle is selected by settings.

When the P70/COM2 and P71/COM3 pins are set as COM outputs, deselection levels are output in the waveforms at the COM2 and COM3 outputs in 1/2 duty ratio operation, and at the COM3 output in 1/3 duty ratio operation.

When LCD display operation is stopped (LCR1: MS1 = MS0 = 00 B ), and during reset, all COM and SEG output pins are taken "L".

Check: If the selected frame cycle generate clock were to stop while the LCD display is operating, the circuit that converts the waveform from d.c. to a.c. would also stop, causing a d.c. voltage to be applied to the liquid crystal cells. The LCD display must be therefore be stopped before the clock is stopped. The conditions under which the main clock (timebase timer) and subclock are stopped are a function of the clock mode and standby mode. Also note that when the timebase timer is selected as the frame clock source (LCR1: CSS = 0), clearing the timebase timer will affect the frame cycle.

- LCD Driving Waveforms

It is characteristic of LCDs that applying d.c. drive to the panel can cause electrochemical degradation of the material used in the LCD cells. For this reason, the LCD controller/driver includes a circuit to convert the original driving waveform to a two-frame a.c. output waveform (zero d.c. bias) to drive the LCD. There are three types of output waveform:

- 1/2 bias, 1/2 duty ratio output waveform (only devices without voltage boosters)
- 1/3 bias, 1/3 duty ratio output waveform
- 1/3 bias, 1/4 duty ratio output waveform
Memo
16.4 Operation of LCD Controller/Driver

16.4.1 Output Waveforms during LCD Controller/Driver Operation (1/2 Duty Ratio)

The display drive output is a multiplex drive-type two-frame a.c. waveform. In the 1/2 duty ratio mode, the only common outputs are COM0 and COM1. (COM2 and COM3 are not used.) 1/2 duty ratio operation cannot be selected in devices that have internal voltage boosters (MB89160A series).

- **1/2 Bias, 1/2 duty output waveform**

The maximum potential difference exists between a segment output and the corresponding common output when the segment (LCD cell) is “turned on.” Figure 16.4.1a shows the output waveforms for the display RAM contents listed in Table 16.4.1a.

Table 16.4.1a Display RAM Contents Example

<table>
<thead>
<tr>
<th>Segment</th>
<th>Display RAM contents</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>COM3</td>
</tr>
<tr>
<td>SEGn</td>
<td>—</td>
</tr>
<tr>
<td>SEGn+1</td>
<td>—</td>
</tr>
</tbody>
</table>

---: Not used

![Figure 16.4.1a Output Waveforms, 1/2 Bias and 1/2 Duty Ratio Example](image)

V0 to V3: V0 to V3 pin voltages
LCD panel connections and display data example (1/2 duty ratio drive mode)

Example) Using segments to represent "5."

Figure 16.4.1b Segment/Common Connections, Data States and Corresponding Display
In the 1/3 duty ratio mode, the COM0, COM1 and COM2 outputs are used by the display. COM3 is not used.

- 1/3 bias, 1/3 duty output waveform

The maximum potential difference exists between a segment output and the corresponding common output when the segment (LCD cell) is “turned on.” Figure 16.4.2a shows the output waveforms for the display RAM contents listed in Table 16.4.2a.

### Table 16.4.2a Display RAM Contents Example

<table>
<thead>
<tr>
<th>Segment</th>
<th>COM3</th>
<th>COM2</th>
<th>COM1</th>
<th>COM0</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEGn</td>
<td>—</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>SEGn+1</td>
<td>—</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

---: Not used

---

Figure 16.4.2a  Output Waveforms, 1/3 Bias and 1/3 Duty Ratio Example
LCD panel connections and display data example (1/3 duty ratio drive mode)

Example) Using segments to represent "5."

<table>
<thead>
<tr>
<th>Address</th>
<th>COM3</th>
<th>COM2</th>
<th>COM1</th>
<th>COM0</th>
</tr>
</thead>
<tbody>
<tr>
<td>nH</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>n+1H</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

*0 to *8: Indicate corresponding display RAM bits. (Bits 3 and 7 and *2 are not used.)

0: OFF
1: ON

LCD Display
Bit States for Numerals "0" through "9"

0. | 1 | 0 | 1 | 0 | 1 | 1 | 1 |
1. | 0 | 1 | 1 | 1 | 1 |
2. | 1 | 1 | 1 | 1 |
3. | 0 | 0 | 0 | 1 | 1 |
4. | 1 | 1 | 1 | 1 |
5. | 0 | 0 | 0 | 1 | 1 |
6. | 1 | 1 | 1 | 1 |
7. | 0 | 0 | 0 | 1 | 1 |
8. | 1 | 1 | 1 | 1 |
9. | 0 | 0 | 0 | 1 | 1 |

In 1/3 duty ratio operation, to be able to define two digits in three bytes, the data stored in two bytes, with the first byte starting at bit 0, and second byte starting at bit 4.
16.4 Operation of LCD Controller/Driver

16.4.3 Output Waveforms during LCD Controller/Driver Operation (1/4 Duty Ratio)

In the 1/4 duty ratio mode, all four common outputs, COM0, COM1, COM2, and COM3 are used.

- 1/3 bias, 1/4 duty output waveforms

The maximum potential difference exists between a segment output and the corresponding common output when the segment (LCD cell) is "turned on." Figure 16.4.3a shows the output waveforms for the display RAM contents listed in Table 16.4.3a.

Table 16.4.3a Display RAM Contents Example

<table>
<thead>
<tr>
<th>Segment</th>
<th>Display RAM contents</th>
</tr>
</thead>
<tbody>
<tr>
<td>SEGn</td>
<td>COM3</td>
</tr>
<tr>
<td>SEGn+1</td>
<td>0</td>
</tr>
</tbody>
</table>

Figure 16.4.3a Output Waveforms, 1/3 Bias and 1/4 Duty Ratio Example
8-segment LCD panel connections and Display data (1/4 duty ratio drive mode)

**Example** Using segments to represent "5."

*0 to *7: Indicate corresponding display RAM bits.

0: OFF
1: ON

---

**Figure 16.4.3b Segment/Common Connections, Data States and Corresponding Display**

<table>
<thead>
<tr>
<th>Address</th>
<th>COM3</th>
<th>COM2</th>
<th>COM1</th>
<th>COM0</th>
<th>SEGn</th>
<th>SEGn+1</th>
</tr>
</thead>
<tbody>
<tr>
<td>n</td>
<td>bit3</td>
<td>bit2</td>
<td>bit1</td>
<td>bit0</td>
<td>bit7</td>
<td>bit6</td>
</tr>
<tr>
<td>00H</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>01H</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>02H</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>03H</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>04H</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>05H</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>06H</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>07H</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
</tbody>
</table>

---

**Table: LCD Display Bit States for Numerals "0" through "9"**

<table>
<thead>
<tr>
<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>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>0</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>0</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>1</td>
<td>0</td>
<td>1</td>
</tr>
</tbody>
</table>

---

16
16.5 Program Example for LCD Controller/Driver

This section gives a program example for LCD controller/driver.

● Processing description

The process writes LCD display data to display RAM. The data is that required to display the numbers ‘0’ through ‘9’ in an LCD panel connected as shown in Figure 16.4.3b. The settings are as follows:

- Internal voltage divider resistors are selected in a device with no voltage booster (LCR1: VSEL = 1)
- 1/3 bias and 1/4 duty ratio are used.
- The subclock (LCR1: CSS = 1) is selected as the clock for frame cycle generation.
- The frame frequency is set at 32 Hz (LCR1: FP1, FP0 = 11b)
- Operation is stopped in watch mode.

● Coding example

LCRAMEQU0060H; Starting address of LCD display RAM
LCR1EQU0072H; Address of LCDC control register 1 (LCR1)

LCDSEG CSEG; 8-segment LCD display data
LCDDATADB11011111B; "0"
   DB 11001000B; "1"
   DB 11110110B; "2"
   DB 11111100B; "3"
   DB 11101001B; "4"
   DB 01111101B; "5"
   DB 01111111B; "6"
   DB 11011001B; "7"
   DB 11111111B; "8"
   DB 11111101B; "9"
   DB 00000000B; END

LCDSESEGENDS

;------------------Main program--------------------------------------------------------
CSEG : [CODE SEGMENT]

: MOVWEP,#LCRAM; Set LCD display RAM address.
: MOVWIX,#LCDDATA; Set LCD display data table address.

LCDSET
: MOV A,@IX+00H
: MOV @EP,A
: INCW EP
: INCW IX
: BNZ LCDSET; Continue until data end (00H) is detected.
: MOV LCR1,#10101111B; Set LCR1 and turn LCD display on.

: ENDS
:-------------------------------------------------------------------------------------------------------------------------------
END

-------------------------------------------------------------------------------------------------------------------------------
The appendices are include an I/O map and the instruction list.

<table>
<thead>
<tr>
<th>Section</th>
<th>Description</th>
<th>Page</th>
</tr>
</thead>
<tbody>
<tr>
<td>A</td>
<td>I/O Map</td>
<td>310</td>
</tr>
<tr>
<td>B</td>
<td>Instructions</td>
<td>312</td>
</tr>
<tr>
<td>B.1</td>
<td>Addressing</td>
<td>314</td>
</tr>
<tr>
<td>B.2</td>
<td>Special Instructions</td>
<td>318</td>
</tr>
<tr>
<td>B.3</td>
<td>$^2\text{MC-8L Instructions}$</td>
<td>320</td>
</tr>
<tr>
<td>B.4</td>
<td>Instruction Map</td>
<td>326</td>
</tr>
<tr>
<td>B.5</td>
<td>Bit Manipulation Instructions (SETB, CLR\text{B})</td>
<td>327</td>
</tr>
<tr>
<td>C</td>
<td>Mask Options</td>
<td>326</td>
</tr>
<tr>
<td>D</td>
<td>Programming Specifications for One-time PROM and EPROM Microcontrollers</td>
<td>330</td>
</tr>
<tr>
<td>D.1</td>
<td>Programming Yield and Erasure</td>
<td>332</td>
</tr>
<tr>
<td>D.2</td>
<td>Programming to the EPROM with Piggyback/Evaluation Device</td>
<td>333</td>
</tr>
<tr>
<td>E</td>
<td>Pin States</td>
<td>334</td>
</tr>
</tbody>
</table>
### I/O Map

Table A lists the addresses of the registers of used by the internal peripheral functions of the MB89160/160A/160L series.

#### I/O Map

<table>
<thead>
<tr>
<th>Address</th>
<th>Register name</th>
<th>Register description</th>
<th>Read/Write</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>00H</td>
<td>PDR0</td>
<td>Port 0 data register</td>
<td>R/W</td>
<td>XXMMMMMMXX</td>
</tr>
<tr>
<td>01H</td>
<td>DDR0</td>
<td>Port 0 data direction register</td>
<td>W</td>
<td>00000000</td>
</tr>
<tr>
<td>02H</td>
<td>PDR1</td>
<td>Port 1 data register</td>
<td>R/W</td>
<td>XXMMMMMMXX</td>
</tr>
<tr>
<td>03H</td>
<td>DDR1</td>
<td>Port 1 data direction register</td>
<td>W</td>
<td>00000000</td>
</tr>
<tr>
<td>04H</td>
<td>PDR2</td>
<td>Port 2 data register</td>
<td>R/W</td>
<td>XXMMMMMMXX</td>
</tr>
<tr>
<td>05H</td>
<td>DDR2</td>
<td>Port 2 data direction register</td>
<td>W</td>
<td>00000000</td>
</tr>
<tr>
<td>06H</td>
<td>(Vacancy)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>07H</td>
<td>SYCC</td>
<td>System clock control register</td>
<td>R/W</td>
<td>XXMMMMMMXX</td>
</tr>
<tr>
<td>08H</td>
<td>STBC</td>
<td>Standby control register</td>
<td>R/W</td>
<td>00000000</td>
</tr>
<tr>
<td>09H</td>
<td>WDTC</td>
<td>Watchdog timer control register</td>
<td>R/W</td>
<td>00000000</td>
</tr>
<tr>
<td>0AH</td>
<td>TBTC</td>
<td>Timebase timer control register</td>
<td>R/W</td>
<td>00000000</td>
</tr>
<tr>
<td>0BH</td>
<td>WPCR</td>
<td>Watch prescaler control register</td>
<td>R/W</td>
<td>00000000</td>
</tr>
<tr>
<td>0CH</td>
<td>PDR3</td>
<td>Port 3 data register</td>
<td>R/W</td>
<td>XXMMMMMMXX</td>
</tr>
<tr>
<td>0DH</td>
<td>(Vacancy)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0EH</td>
<td>PDR4</td>
<td>Port 4 data register</td>
<td>R/W</td>
<td>11111111</td>
</tr>
<tr>
<td>0FH</td>
<td>PDR5</td>
<td>Port 5 data register</td>
<td>R/W</td>
<td>11111111</td>
</tr>
<tr>
<td>10H</td>
<td>BZCR</td>
<td>Buzzer register</td>
<td>R/W</td>
<td>XXMMMMMMXX</td>
</tr>
<tr>
<td>11H</td>
<td>(Vacancy)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12H</td>
<td>PDR6</td>
<td>Port 6 data register</td>
<td>R/W</td>
<td>11111111</td>
</tr>
<tr>
<td>13H</td>
<td>PDR7</td>
<td>Port 7 data register</td>
<td>R/W</td>
<td>XXMMMMMMXX</td>
</tr>
<tr>
<td>14H</td>
<td>RCR1</td>
<td>Remote control transmission register 1</td>
<td>R/W</td>
<td>00000000</td>
</tr>
<tr>
<td>15H</td>
<td>RCR2</td>
<td>Remote control transmission register 2</td>
<td>R/W</td>
<td>00000000</td>
</tr>
<tr>
<td>16H to 17H</td>
<td>(Vacancy)</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18H</td>
<td>T2CR</td>
<td>Timer 2 control register</td>
<td>R/W</td>
<td>X0000XXX</td>
</tr>
<tr>
<td>19H</td>
<td>T1CR</td>
<td>Timer 1 control register</td>
<td>R/W</td>
<td>X0000XXX</td>
</tr>
<tr>
<td>1AH</td>
<td>T2DR</td>
<td>Timer 2 data register</td>
<td>R/W</td>
<td>XXMMMMMMXX</td>
</tr>
<tr>
<td>1BH</td>
<td>T1DR</td>
<td>Timer 1 data register</td>
<td>R/W</td>
<td>XXMMMMMMXX</td>
</tr>
<tr>
<td>1CH</td>
<td>SMR</td>
<td>Serial mode register</td>
<td>R/W</td>
<td>00000000</td>
</tr>
<tr>
<td>1DH</td>
<td>SDR</td>
<td>Serial data register</td>
<td>R/W</td>
<td>XXMMMMMMXX</td>
</tr>
<tr>
<td>1EH</td>
<td>CNTR1</td>
<td>PWM 1 control register</td>
<td>R/W</td>
<td>00000000</td>
</tr>
<tr>
<td>1FH</td>
<td>COMR1</td>
<td>PWM 1 compare register</td>
<td>W</td>
<td>XXMMMMMMXX</td>
</tr>
<tr>
<td>20H</td>
<td>CNTR2</td>
<td>PWM 2 control register</td>
<td>R/W</td>
<td>00000000</td>
</tr>
<tr>
<td>21H</td>
<td>COMR2</td>
<td>PWM 2 compare register</td>
<td>W</td>
<td>XXMMMMMMXX</td>
</tr>
</tbody>
</table>
Table Ab I/O Map (Continued)

<table>
<thead>
<tr>
<th>Address (Hex)</th>
<th>Register name</th>
<th>Register description</th>
<th>Read/Write</th>
<th>Initial value</th>
</tr>
</thead>
<tbody>
<tr>
<td>22H to 2CH</td>
<td>(Vacancy)</td>
<td>XXXXXXXXX B</td>
<td>R/W</td>
<td></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>XXX00001a</td>
</tr>
<tr>
<td>2FH</td>
<td>ADCD</td>
<td>A/D data register</td>
<td>R/W</td>
<td>XXXXXXX00a</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>XXXXXXX00a</td>
</tr>
<tr>
<td>32H</td>
<td>EIE2</td>
<td>External interrupt 2 control register</td>
<td>R/W</td>
<td>00000000a</td>
</tr>
<tr>
<td>33H</td>
<td>EIF2</td>
<td>External interrupt 2 flag register</td>
<td>R/W</td>
<td>XXXXXXX00a</td>
</tr>
<tr>
<td>34H to 5FH</td>
<td>(Vacancy)</td>
<td>XXXXXXXXX B</td>
<td>R/W</td>
<td></td>
</tr>
<tr>
<td>60H to 68H</td>
<td>VRAM</td>
<td>Display RAM</td>
<td>R/W</td>
<td>XXXXXXXXXa</td>
</tr>
<tr>
<td>6CH to 71H</td>
<td>(Vacancy)</td>
<td>XXXXXXXXX B</td>
<td>R/W</td>
<td></td>
</tr>
<tr>
<td>72H</td>
<td>LCR1</td>
<td>LCD controller/driver control register 1</td>
<td>R/W</td>
<td>00010000 B</td>
</tr>
<tr>
<td>73H to 7BH</td>
<td>(Vacancy)</td>
<td>XXXXXXXXX B</td>
<td>R/W</td>
<td></td>
</tr>
<tr>
<td>7CH</td>
<td>ILR1</td>
<td>Interrupt level setting register 1</td>
<td>W</td>
<td>11111111a</td>
</tr>
<tr>
<td>7DH</td>
<td>ILR2</td>
<td>Interrupt level setting register 2</td>
<td>W</td>
<td>11111111a</td>
</tr>
<tr>
<td>7EH</td>
<td>ILR3</td>
<td>Interrupt level setting register 3</td>
<td>W</td>
<td>11111111a</td>
</tr>
<tr>
<td>7FH</td>
<td>ITR</td>
<td>Interrupt test register</td>
<td>Access prohibited</td>
<td>XXXXXXX00a</td>
</tr>
</tbody>
</table>

* : For MB89165L series, the Initial value is X0000000B

- **Read/write access symbols**
  - R/W: Readable and writable
  - R: Read-only
  - W: Write-only

- **Initial value symbols**
  - 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 undefined.
  - M: The initial value of this bit is determined by mask option.

**Check:** Do not use vacancies.
B  Instructions

This appendix describes the F2MC-8L instruction set.

- Instruction List Symbols

Table Ba lists the meaning of the symbols and Table Bb lists the meanings of the columns used in Section B.3, “F2MC-8L Instructions”

<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: b</td>
<td>Bit direct address (8:3 bits)</td>
</tr>
<tr>
<td>rel</td>
<td>Branch relative address (8 bits)</td>
</tr>
<tr>
<td>@</td>
<td>Register indirect (Example: @A, @IX, @EP)</td>
</tr>
<tr>
<td>A</td>
<td>Accumulator A (Whether its length is 8 or 16 bits is determined by the instruction in use.)</td>
</tr>
<tr>
<td>AH</td>
<td>Upper 8 bits of the accumulator A (8 bits)</td>
</tr>
<tr>
<td>AL</td>
<td>Lower 8 bits of the accumulator A (8 bits)</td>
</tr>
<tr>
<td>T</td>
<td>Temporary accumulator T (Whether its length is 8 or 16 bits is determined by the instruction in use.)</td>
</tr>
<tr>
<td>TH</td>
<td>Upper 8 bits of the temporary accumulator T (8 bits)</td>
</tr>
<tr>
<td>TL</td>
<td>Lower 8 bits of the temporary accumulator T (8 bits)</td>
</tr>
<tr>
<td>IX</td>
<td>Index register IX (16 bits)</td>
</tr>
<tr>
<td>EP</td>
<td>Extra pointer EP (16 bits)</td>
</tr>
<tr>
<td>PC</td>
<td>Program counter PC (16 bits)</td>
</tr>
<tr>
<td>SP</td>
<td>Stack pointer SP (16 bits)</td>
</tr>
<tr>
<td>PS</td>
<td>Program status PS (16 bits)</td>
</tr>
<tr>
<td>dr</td>
<td>Accumulator A or index register IX (16 bits)</td>
</tr>
<tr>
<td>CCR</td>
<td>Condition code register CCR (8 bits)</td>
</tr>
<tr>
<td>RP</td>
<td>Register bank pointer RP (5 bits)</td>
</tr>
<tr>
<td>Ri</td>
<td>General-purpose register Ri (8 bits, i = 0 to 7)</td>
</tr>
<tr>
<td>·</td>
<td>Indicates that the very × is the immediate data. (Whether its length is 8 or 16 bits is determined by the instruction in use.)</td>
</tr>
<tr>
<td>(x)</td>
<td>Indicates that the contents of × is the target of accessing. (Whether its length is 8 or 16 bits is determined by the instruction in use.)</td>
</tr>
<tr>
<td>(x)</td>
<td>The address indicated by the contents of × is the target of accessing. (Whether its length is 8 or 16 bits is determined by the instruction in use.)</td>
</tr>
</tbody>
</table>
### Table Bb Instruction List Columns

<table>
<thead>
<tr>
<th>Column</th>
<th>Description</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mnemonic</td>
<td>Assembler notation of an instruction</td>
</tr>
<tr>
<td>~</td>
<td>Number of instructions</td>
</tr>
<tr>
<td>#</td>
<td>Number of bytes</td>
</tr>
<tr>
<td>Operation</td>
<td>Operation of an instruction</td>
</tr>
</tbody>
</table>
| TL, TH, AH  | A content change when each of the TL, TH, and AH instructions is executed. Symbols in the column indicate the following:  
• "—" indicate no change.  
• dh is the 8 upper bits of operation description data.  
• AL and AH must become the contents of AL and AH immediately before the instruction is executed.  
• 00 becomes 00. |
| N, Z, V, C  | An instruction of which the corresponding flag will change. If + is written in this column, the relevant instruction will change its corresponding flag. |
| OP Code     | Code of an instruction. If an instruction is more than one code, it is written according to the following rule:  
Example: 48 to 4F ← This indicates 48, 49 ... 4F. |
B.1 Addressing

The F\textsuperscript{2}MC-8L supports 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

Addressing Modes

- **Direct addressing**
  
  Indicated by “dir” in the instruction list. Used to access the area between “0000H” and “00FFH”. For direct addressing, the upper one byte of the address is “00H” and the operand specifies the lower one byte. Figure B.1a shows an example.

  \[
  \text{MOV } 12\text{H, A}
  \]

  ![Figure B.1a Direct Addressing](image)

- **Extended addressing**
  
  Indicated by “ext” in the instruction list. Used to access the entire 64-Kbyte area. For extended addressing, the first operand specifies the upper one byte of the address and the second operand specifies the lower one byte. Figure B.1b shows an example.

  \[
  \text{MOVW A, 1234\text{H}}
  \]

  ![Figure B.1b Extended Addressing](image)

- **Bit direct addressing**
  
  Indicated by “dir: b” in the instruction list. Used to access the area between “0000H” and “00FFH” in bit units. For bit direct addressing, the upper one byte of the address is “00H”, the operand specifies the lower one byte of the address, and the lower three bits of the operation code specify the bit position. Figure B.1c shows an example.

  \[
  \text{SETB 34\text{H, 2}}
  \]

  ![Figure B.1c Bit Direct Addressing](image)
Index addressing
Indicated by "@IX+off" in the instruction list. Used to access the entire 64-Kbyte area. Index addressing generates the address is obtained by adding the sign-extended contents of the first operand to the index register (IX). Figure B.1d shows an example.

![Figure B.1d Index Addressing](image)

Pointer addressing
Indicated by "@EP" in the instruction list. Used to access the entire 64-Kbyte area. Pointer addressing uses the extra pointer (EP) the address. Figure B.1e shows an example.

![Figure B.1e Pointer Addressing](image)

General-purpose register addressing
Indicated by "Ri" in the instruction list. Used to access the general-purpose register area register bank. For general-purpose register addressing, the upper one byte of the address is fixed at "01" and the lower byte is generated from the register bank pointer (RP) and the lower three bits of the operation code. The CPU accesses the resulting address. Figure B.1f shows an example.

![Figure B.1f General-purpose Register Addressing](image)

Immediate addressing
Indicated by "#d8" in the instruction list. Used when immediate data is required. In immediate addressing, the operand is used directly as immediate data. The operation code determines whether the data is byte or word. Figure B.1g shows an example.

![Figure B.1g Immediate Addressing](image)
Vector addressing

Indicated by “vct” in the instruction list. Used to branch to a subroutine address stored in the vector table. For vector addressing, the “vct” data is contained in the operation code. Table B.1 lists the correspondence between “vct” and the resulting address.

Table B.1a  Vector Table Address Corresponding to “vct”

<table>
<thead>
<tr>
<th>#vct</th>
<th>Vector table address (branch destination upper address: lower address)</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>FFCAH : FFCDH</td>
</tr>
<tr>
<td>1</td>
<td>FFCAH : FFCDH</td>
</tr>
<tr>
<td>2</td>
<td>FFCAH : FFCDH</td>
</tr>
<tr>
<td>3</td>
<td>FFCAH : FFCDH</td>
</tr>
<tr>
<td>4</td>
<td>FFCAH : FFCDH</td>
</tr>
<tr>
<td>5</td>
<td>FFCAH : FFCDH</td>
</tr>
<tr>
<td>6</td>
<td>FFCAH : FFCDH</td>
</tr>
<tr>
<td>7</td>
<td>FFCAH : FFCDH</td>
</tr>
</tbody>
</table>

Figure B.1h shows an example.

Figure B.1h  Vector Addressing

Relative addressing

Indicated by “rel” in the instruction list. Used to branch to a destination in the area 128 bytes above or below the program counter (PC). Relative addressing adds the sign-extended contents of the first operand to the PC and stores the result in the PC. Figure B.1i shows an example.

Figure B.1i  Relative Addressing

This example branches to the address containing the BNE operation code and therefore results in an endless loop.

Inherent addressing

Inherent addressing is used for instructions in the instruction list that do not have operands and for which the operation code determines the operation. The operation of inherent addressing depends on the instruction. Figure B.1j shows an example.

Figure B.1j  Inherent Addressing
B Instructions

B.2 Special Instructions

This section describes special instructions, other than addressing.

- **JMP @A**
  
  This instruction moves the address contained in the accumulator (A) to the program counter (PC) and branches to the new address. This instruction can be used to perform an N option branch by placing N branch destination addresses in a table and moving the desired address to the accumulator.

  Figure B.2a shows an outline of the instruction operation.

  ![Figure B.2a JMP @A](image)

- **MOVW A,PC**
  
  This instruction stores the PC contents in the accumulator A. This performs the opposite operation to “JMP @A”. By executing this instruction in the main routine and calling a particular subroutine, the subroutine can determine whether the contents of A match a predetermined value. The subroutine can check whether program runaway has occurred by checking whether or not execution has branched from an expected location.

  Figure B.2b shows an outline of the instruction operation.

  ![Figure B.2b MOVW A,PC](image)

  The content of A after executing this instruction is the address of the next instruction (not the address containing the operation code of this instruction). Accordingly, the value “1234H” stored in A in the example shown in Figure B.2b is the address of the next operation code after “MOVW A, PC”.

- **MULU A**
  
  This instruction performs an unsigned multiplication of AL (lower 8 bits of the accumulator) and TL (lower 8 bits of the temporary accumulator) and stores the 16-bit result in A. The contents of T (temporary accumulator) does not change. The arithmetic operation does not use the pre-execution contents of AH (upper 8 bits of the accumulator) and TH (upper 8 bits of the temporary accumulator). Since the flags remain unchanged, use care when branching is required based on the result of multiplication.

  Figure B.2c shows an outline of the instruction operation.

  ![Figure B.2c MULU A](image)
• **DIVU A**

This instruction divides the 16 bits of T by the 8 bits of AL, treating the data as unsigned. The instruction stores the result in AL and the remainder in TL, both as 8 bit data. AH and TH are both set to “zero”. The arithmetic operation does not use the value of AH prior to instruction execution. The result is not assured for data that produces a result that exceeds 8 bits. As there is no indication that the result exceeded 8 bits, check the data before performing. Since the flags remain unchanged, use care when branching is required based on the result of the division.

Figure B.2d shows an outline of the instruction operation.

<table>
<thead>
<tr>
<th>(Before execution)</th>
<th>(After execution)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A 5678H</td>
<td>A 0034H</td>
</tr>
<tr>
<td>T 1862H</td>
<td>T 0002H</td>
</tr>
</tbody>
</table>

**Figure B.2d DIVU A**

• **XCHW A,PC**

This instruction exchanges the contents of A and PC, and as a result branches to the address corresponding to contents of A before execution. The contents of A after execution assume the address next to the address where the operation code of the “XCHW A,PC” is stored. The instruction can be used to specify a table in the main routine which is used in a subroutine.

Figure B.2e shows an outline of the instruction operation.

<table>
<thead>
<tr>
<th>(Before execution)</th>
<th>(After execution)</th>
</tr>
</thead>
<tbody>
<tr>
<td>A 5678H</td>
<td>A 1235H</td>
</tr>
<tr>
<td>PC 1234H</td>
<td>PC 5678H</td>
</tr>
</tbody>
</table>

**Figure B.2e XCHW A,PC**

The content of A after executing this instruction is the address of the next instruction (not the address containing the operation code of this instruction). Accordingly, the value “1235H” stored in A in the example shown in Figure B.2e is the address of the next operation code after “XCHW A,PC”. Therefore, the value of A is “1235H”, not “1234H”.

Figure B.2f shows an assembly language example.

<table>
<thead>
<tr>
<th>(Main routine)</th>
<th>(Subroutine)</th>
</tr>
</thead>
<tbody>
<tr>
<td>MOVW A, #PUTSUB</td>
<td>PUTSUB XCHW A, EP</td>
</tr>
<tr>
<td>XCHW A, PC</td>
<td>PUSHW A</td>
</tr>
<tr>
<td>DB ‘PUT OUT DATA’,EOL</td>
<td>MOV A, @EP</td>
</tr>
<tr>
<td>MOVW A, 1234H</td>
<td>INCW EP</td>
</tr>
<tr>
<td></td>
<td>MOV IO, A</td>
</tr>
<tr>
<td></td>
<td>CMP A, #EOL</td>
</tr>
<tr>
<td></td>
<td>BNE PTS1</td>
</tr>
<tr>
<td></td>
<td>POPW A</td>
</tr>
<tr>
<td></td>
<td>XCHW A, EP</td>
</tr>
<tr>
<td></td>
<td>JMP @A</td>
</tr>
</tbody>
</table>

**Figure B.2f Example Using XCHW A, PC**
CALLV #vct

This instruction is used to branch to a subroutine address in the vector table. The instruction saves the return address (contents of the PC) to the address corresponding to the SP (stack pointer) branches to the address stored in the vector table using vector addressing. As “CALLV #vct” is a single-byte instruction, using this instruction for commonly used subroutines reduces the overall program size.

Figure B.2g shows an outline of the instruction operation.

![Figure B.2g Execution example of CALLV #3](image)

The content of PC saved to stack area after executing this instruction is the address of next instruction (not the address containing the operation code of this instruction). Accordingly, the value “5679H” saved to stack (1232H, 1233H) in the example shown in Figure B.2g is the address (return address) of the next operation code after “MOVW A,PC”.

```plaintext
PC 5 6 7 8H
SP 1 2 3 4H
1 2 3 2H X XH
1 2 3 3H X XH
FF C 6H F EH
FF C 7H D CH

PC F E D CH
SP 1 2 3 2H
1 2 3 2H 5 6H
1 2 3 3H 7 9H
FF C 6H F EH
FF C 7H D CH
```
B Instructions

B.3 F²MC-8L Instructions

Tables B.3a to B.3d list the F²MC-8L instructions.

Table B.3a Transfer Instructions

<table>
<thead>
<tr>
<th>No.</th>
<th>Mnemonic</th>
<th>~</th>
<th>Operation</th>
<th>TL</th>
<th>TH</th>
<th>AH</th>
<th>NZVC</th>
<th>OP code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>MOV dir, A</td>
<td>3</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>MOV @IX,off, A</td>
<td>4</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>MOV ext, A</td>
<td>4</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>4</td>
<td>MOV @EP, A</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>5</td>
<td>MOV RI, A</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>6</td>
<td>MOV A, #d8</td>
<td>2</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>MOV A, dir</td>
<td>3</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>8</td>
<td>MOV A, @IX+off</td>
<td>4</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>9</td>
<td>MOV A, ext</td>
<td>4</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>10</td>
<td>MOV A, @A</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>11</td>
<td>MOV A, @EP</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>12</td>
<td>MOV A, Ri</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>13</td>
<td>MOV dir, #d8</td>
<td>4</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>14</td>
<td>MOV @IX,off, #d8</td>
<td>5</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>15</td>
<td>MOV @EP, #d8</td>
<td>4</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>MOV RI, #d8</td>
<td>4</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>17</td>
<td>MOVW @IX,off, A</td>
<td>5</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>18</td>
<td>MOVW ext, A</td>
<td>5</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>19</td>
<td>MOVW @EP, A</td>
<td>4</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>20</td>
<td>MOVW EP, A</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>21</td>
<td>MOVW A, #d16</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>22</td>
<td>MOVW A, dir</td>
<td>4</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>23</td>
<td>MOVW A, @IX</td>
<td>5</td>
<td>2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>24</td>
<td>MOVW A, ext</td>
<td>5</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>25</td>
<td>MOVW A, @A</td>
<td>4</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>26</td>
<td>MOVW A, @EP</td>
<td>4</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>27</td>
<td>MOVW A, EP</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>28</td>
<td>MOVW EP, #d16</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>29</td>
<td>MOVW IX, A</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>30</td>
<td>MOVW IX, #d16</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>31</td>
<td>MOVW IX, dir</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>32</td>
<td>MOVW SP, A</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>33</td>
<td>MOVW A, SP</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>34</td>
<td>MOV @A, T</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>35</td>
<td>MOV @A, T</td>
<td>3</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>36</td>
<td>MOVW IX, #d16</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>37</td>
<td>MOVW A, PS</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>38</td>
<td>MOVW PS, A</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>39</td>
<td>MOVW SP, #d16</td>
<td>3</td>
<td>3</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>40</td>
<td>SWAP</td>
<td>2</td>
<td>1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Check: • The automatic transfer to the T register is TL ← AL for instructions that perform a byte transfer to A.
• Operands in more than one operand instruction must be stored in the order in which their mnemonics are written.
### Arithmetic Operation Instructions

#### Table B.3b 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>NZVC</th>
<th>OP code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>ADDC A, Rl</td>
<td>3</td>
<td>1</td>
<td>(A)=+(+(Rl)+C)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>2</td>
<td>ADDC A, #dB</td>
<td>2</td>
<td>2</td>
<td>(A)=+(+(#dB)+C)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>28 to 2F</td>
</tr>
<tr>
<td>3</td>
<td>ADDC A, dir</td>
<td>3</td>
<td>2</td>
<td>(A)=+(+(dir)+C)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>24</td>
</tr>
<tr>
<td>4</td>
<td>ADDC A, @IX+off</td>
<td>4</td>
<td>2</td>
<td>(A)=+(+(IX)+off)+C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>25</td>
</tr>
<tr>
<td>5</td>
<td>SUBC A, @EP</td>
<td>3</td>
<td>1</td>
<td>(A)=-(+(EP))+C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>26</td>
</tr>
<tr>
<td>6</td>
<td>SUBC A, @EP</td>
<td>3</td>
<td>1</td>
<td>(A)=-(+(EP))+C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>27</td>
</tr>
<tr>
<td>7</td>
<td>SUBC A</td>
<td>2</td>
<td>1</td>
<td>(A)=+(+(AL)+C)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>28</td>
</tr>
<tr>
<td>8</td>
<td>SUBC A, Rl</td>
<td>3</td>
<td>1</td>
<td>(A)=+(+(Rl)-C)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>29</td>
</tr>
<tr>
<td>9</td>
<td>SUBC A, #dB</td>
<td>2</td>
<td>2</td>
<td>(A)=+(+(#dB)-C)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>30</td>
</tr>
<tr>
<td>10</td>
<td>SUBC A, dir</td>
<td>3</td>
<td>2</td>
<td>(A)=+(+(dir)-C)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>31</td>
</tr>
<tr>
<td>11</td>
<td>SUBC A, @IX+off</td>
<td>4</td>
<td>2</td>
<td>(A)=+(+(IX)+off)-C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>32</td>
</tr>
<tr>
<td>12</td>
<td>SUBC A, @EP</td>
<td>3</td>
<td>1</td>
<td>(A)=+(+(EP))-C</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>33</td>
</tr>
<tr>
<td>13</td>
<td>SUBC A</td>
<td>3</td>
<td>1</td>
<td>(A)=+(+(T)-C)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>34</td>
</tr>
<tr>
<td>14</td>
<td>SUBC A</td>
<td>2</td>
<td>1</td>
<td>(A)=+(+(TL)+(AL)-C)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>35</td>
</tr>
<tr>
<td>15</td>
<td>INC Ri</td>
<td>4</td>
<td>1</td>
<td>(Ri)=+(+(Ri)+1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>36</td>
</tr>
<tr>
<td>16</td>
<td>INCW EP</td>
<td>3</td>
<td>1</td>
<td>(EP)=+(+(EP)+1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>37</td>
</tr>
<tr>
<td>17</td>
<td>INCW IX</td>
<td>3</td>
<td>1</td>
<td>(IX)=+(+(IX)+1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>38</td>
</tr>
<tr>
<td>18</td>
<td>INCW A</td>
<td>3</td>
<td>1</td>
<td>(A)=+(+(A)+1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>39</td>
</tr>
<tr>
<td>19</td>
<td>DEC Ri</td>
<td>4</td>
<td>1</td>
<td>(Ri)=+(+(Ri)-1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>40</td>
</tr>
<tr>
<td>20</td>
<td>DECW EP</td>
<td>3</td>
<td>1</td>
<td>(EP)=+(+(EP)-1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>41</td>
</tr>
<tr>
<td>21</td>
<td>DECW IX</td>
<td>3</td>
<td>1</td>
<td>(IX)=+(+(IX)-1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>42</td>
</tr>
<tr>
<td>22</td>
<td>DECW A</td>
<td>3</td>
<td>1</td>
<td>(A)=+(+(A)-1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>43</td>
</tr>
<tr>
<td>23</td>
<td>MULU A</td>
<td>19</td>
<td>1</td>
<td>(A)=+(+(AL)+(TL))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>44</td>
</tr>
<tr>
<td>24</td>
<td>DIVU A</td>
<td>21</td>
<td>1</td>
<td>(A)=+(+(T)+(AL), MOD→+(T))</td>
<td>dl</td>
<td>00</td>
<td>00</td>
<td></td>
<td>45</td>
</tr>
<tr>
<td>25</td>
<td>ANDW A</td>
<td>3</td>
<td>1</td>
<td>(A)∧(A)∧(T)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>46</td>
</tr>
<tr>
<td>26</td>
<td>XORW A</td>
<td>3</td>
<td>1</td>
<td>(A)¬(A)∧(T)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>47</td>
</tr>
<tr>
<td>27</td>
<td>XORW A</td>
<td>3</td>
<td>1</td>
<td>(A)¬(A)∧(T)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>48</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>49</td>
</tr>
<tr>
<td>29</td>
<td>CMPW A</td>
<td>3</td>
<td>1</td>
<td>(TL)+(AL)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50</td>
</tr>
<tr>
<td>30</td>
<td>RORC A</td>
<td>2</td>
<td>1</td>
<td>(A)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>51</td>
</tr>
<tr>
<td>31</td>
<td>ROLC A</td>
<td>2</td>
<td>1</td>
<td>(A)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>52</td>
</tr>
<tr>
<td>32</td>
<td>CMP A, #dB</td>
<td>2</td>
<td>2</td>
<td>(A)=+(#dB)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>53</td>
</tr>
<tr>
<td>33</td>
<td>CMP A, dir</td>
<td>2</td>
<td>2</td>
<td>(A)=+(dir)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>54</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>55</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>56</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>57</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>58</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>59</td>
</tr>
<tr>
<td>39</td>
<td>XOR T</td>
<td>2</td>
<td>1</td>
<td>(A)=+(+(AL)∧(T))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>60</td>
</tr>
<tr>
<td>40</td>
<td>XOR T, #dB</td>
<td>2</td>
<td>2</td>
<td>(A)=+(+(AL)∧(T)+dB)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>61</td>
</tr>
<tr>
<td>41</td>
<td>XOR T, dir</td>
<td>2</td>
<td>2</td>
<td>(A)=+(+(AL)∧(T)+dir)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>62</td>
</tr>
<tr>
<td>42</td>
<td>XOR T, @EP</td>
<td>3</td>
<td>1</td>
<td>(A)=+(+(AL)∧(T)+(EP))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>63</td>
</tr>
<tr>
<td>43</td>
<td>XOR T, @IX+off</td>
<td>4</td>
<td>2</td>
<td>(A)=+(+(AL)∧(T)+(IX)+off)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>64</td>
</tr>
<tr>
<td>44</td>
<td>XOR T, Ri</td>
<td>3</td>
<td>1</td>
<td>(A)=+(+(AL)∧(T)+dir)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>65</td>
</tr>
<tr>
<td>45</td>
<td>AND A</td>
<td>2</td>
<td>1</td>
<td>(A)=+(+(TL))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>66</td>
</tr>
<tr>
<td>46</td>
<td>AND A, #dB</td>
<td>2</td>
<td>2</td>
<td>(A)=+(+(AL)+off)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>67</td>
</tr>
<tr>
<td>47</td>
<td>AND A, dir</td>
<td>3</td>
<td>2</td>
<td>(A)=+(+(AL)+dir)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>68</td>
</tr>
<tr>
<td>48</td>
<td>AND A, @EP</td>
<td>3</td>
<td>1</td>
<td>(A)=+(+(AL)+((EP))+C)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>69</td>
</tr>
<tr>
<td>49</td>
<td>AND A, @IX+off</td>
<td>4</td>
<td>2</td>
<td>(A)=+(+(AL)+((IX)+off)+C)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>70</td>
</tr>
<tr>
<td>50</td>
<td>AND A, Ri</td>
<td>3</td>
<td>1</td>
<td>(A)=+(+(AL)+((Ri))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>71</td>
</tr>
<tr>
<td>51</td>
<td>OR A</td>
<td>2</td>
<td>1</td>
<td>(A)=+(+(AL)+((TL))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>72</td>
</tr>
<tr>
<td>52</td>
<td>OR A, #dB</td>
<td>2</td>
<td>2</td>
<td>(A)=+(+(AL)+((#dB))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>73</td>
</tr>
<tr>
<td>53</td>
<td>OR A, dir</td>
<td>3</td>
<td>2</td>
<td>(A)=+(+(AL)+((dir))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>74</td>
</tr>
<tr>
<td>54</td>
<td>OR A, @EP</td>
<td>3</td>
<td>1</td>
<td>(A)=+(+(AL)+((EP))+C)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>75</td>
</tr>
<tr>
<td>55</td>
<td>OR A, @IX+off</td>
<td>4</td>
<td>2</td>
<td>(A)=+(+(AL)+((IX)+off)+C)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>76</td>
</tr>
<tr>
<td>56</td>
<td>OR A, Ri</td>
<td>3</td>
<td>1</td>
<td>(A)=+(+(AL)+((Ri))</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>77</td>
</tr>
<tr>
<td>57</td>
<td>CMP A, dir, #dB</td>
<td>5</td>
<td>3</td>
<td>(dir)+dB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>78</td>
</tr>
<tr>
<td>58</td>
<td>CMP A, @EP, #dB</td>
<td>5</td>
<td>2</td>
<td>((EP))+dB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>79</td>
</tr>
<tr>
<td>59</td>
<td>CMP A, @IX+off, #dB</td>
<td>5</td>
<td>3</td>
<td>((IX)+off)+dB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>80</td>
</tr>
<tr>
<td>60</td>
<td>CMP A, Ri, #dB</td>
<td>4</td>
<td>2</td>
<td>(Ri)+dB</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>81</td>
</tr>
<tr>
<td>61</td>
<td>INCW S#</td>
<td>3</td>
<td>1</td>
<td>(SP)=+(+(SP)+1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>82</td>
</tr>
<tr>
<td>62</td>
<td>DECW SP</td>
<td>3</td>
<td>1</td>
<td>(SP)=+(+(SP)-1)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>83</td>
</tr>
</tbody>
</table>
### Branch Instructions

#### Table B.3c Branch Instructions

<table>
<thead>
<tr>
<th>No.</th>
<th>Mnemonic</th>
<th>~ #</th>
<th>Operation</th>
<th>TL</th>
<th>TH</th>
<th>AH</th>
<th>NZVC</th>
<th>OP code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>BZ/BEQ rel</td>
<td>3 2</td>
<td>if Z=1 then PC–PC+rel</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>FD</td>
</tr>
<tr>
<td>2</td>
<td>BNZ/BNE rel</td>
<td>3 2</td>
<td>if Z=0 then PC–PC+rel</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>FC</td>
</tr>
<tr>
<td>3</td>
<td>BC/BLO rel</td>
<td>3 2</td>
<td>if C=1 then PC–PC+rel</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>F9</td>
</tr>
<tr>
<td>4</td>
<td>BNC/BHS rel</td>
<td>3 2</td>
<td>if C=0 then PC–PC+rel</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>F8</td>
</tr>
<tr>
<td>5</td>
<td>BN rel</td>
<td>3 2</td>
<td>if N=1 then PC–PC+rel</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>FB</td>
</tr>
<tr>
<td>6</td>
<td>BP rel</td>
<td>3 2</td>
<td>if N=0 then PC–PC+rel</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>FA</td>
</tr>
<tr>
<td>7</td>
<td>BLT rel</td>
<td>3 2</td>
<td>if V ∨ N=1 then PC–PC+rel</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>FF</td>
</tr>
<tr>
<td>8</td>
<td>BGE rel</td>
<td>3 2</td>
<td>if V ∨ N=0 then PC–PC+rel</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 3</td>
<td>if (dir,b)=0 then PC–PC+rel</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 3</td>
<td>if (dir,b)=1 then PC–PC+rel</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 1</td>
<td>(PC)–(A)</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>E9</td>
</tr>
<tr>
<td>12</td>
<td>JMP ext</td>
<td>3 2</td>
<td>(PC)–ext</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>E8 to E9</td>
</tr>
<tr>
<td>13</td>
<td>CALLV #vct</td>
<td>6 1</td>
<td>vector call</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>31</td>
</tr>
<tr>
<td>14</td>
<td>CALL ext</td>
<td>6 3</td>
<td>subroutine call</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>F4</td>
</tr>
<tr>
<td>15</td>
<td>XCHW A,PC</td>
<td>3 1</td>
<td>(PC)–(A), (A)–(PC)+1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>16</td>
<td>RET</td>
<td>4 1</td>
<td>return from subroutine</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>20</td>
</tr>
<tr>
<td>17</td>
<td>RETI</td>
<td>6 1</td>
<td>return form interrupt</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>30</td>
</tr>
</tbody>
</table>

### Other Instructions

#### Table B.3d Other Instructions

<table>
<thead>
<tr>
<th>No.</th>
<th>Mnemonic</th>
<th>~ #</th>
<th>Operation</th>
<th>TL</th>
<th>TH</th>
<th>AH</th>
<th>NZVC</th>
<th>OP code</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>PUSHW A</td>
<td>4 1</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 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>50</td>
</tr>
<tr>
<td>3</td>
<td>PUSHW IX</td>
<td>4 1</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 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>51</td>
</tr>
<tr>
<td>5</td>
<td>NOP</td>
<td>1 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>60</td>
</tr>
<tr>
<td>6</td>
<td>CLRC</td>
<td>1 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>81</td>
</tr>
<tr>
<td>7</td>
<td>SETC</td>
<td>1 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>91</td>
</tr>
<tr>
<td>8</td>
<td>CRLI</td>
<td>1 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>80</td>
</tr>
<tr>
<td>9</td>
<td>SETI</td>
<td>1 1</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>90</td>
</tr>
</tbody>
</table>
### B.4 Instruction Map

Table B.4 lists the F^2MC-8L instruction map.

#### Instruction Map

<table>
<thead>
<tr>
<th>A</th>
<th>B</th>
<th>C</th>
<th>D</th>
<th>E</th>
<th>F</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

Table B.4a F^2MC-8L Instruction Map

---

APPENDIX 326
B Instructions

B.5 Bit Manipulation Instructions (SETB, CLRB)

The bit manipulation instructions use a different read operation to the normal operation for some bits of peripheral function registers.

- **Read-modify-write Operation**

Bit manipulation instructions set to “1” (SETB) or clear to “0” (CLRB) the specified bit only of a register or RAM location. However, as the CPU handles data in 8-bit units, the actual operation consists of reading the 8-bit data, modifying the specified bit, then writing the result back to the same address. This is called a read-modify-write operation.

Table B.5 shows the bus operation for bit manipulation instructions.

<table>
<thead>
<tr>
<th>Code</th>
<th>Mnemonic</th>
<th>~</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>CLRB 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></td>
<td></td>
<td>2</td>
<td>dir address</td>
<td>Data</td>
<td>0</td>
<td>1</td>
<td>1</td>
</tr>
<tr>
<td></td>
<td>SETB dir:b</td>
<td></td>
<td>3</td>
<td>dir address</td>
<td>Data</td>
<td>1</td>
<td>0</td>
<td>0</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td>4</td>
<td>N+2</td>
<td>Next instruction</td>
<td>0</td>
<td>1</td>
<td>0</td>
</tr>
</tbody>
</table>

- **Read Source When Executing Bit Manipulation Instructions**

The read source for a read-modify-write of some I/O ports and interrupt request flag bits is different than for a standard read.

- **I/O ports (Bit manipulation instructions)**

For some I/O ports, a standard read reads the I/O pin values whereas a bit manipulation instruction reads the output latch value. This is to prevent unintentionally modifying other output latch bit values and is independent of the pin input/output direction or pin state.

- **Interrupt request flag bits (Bit manipulation instructions)**

For interrupt request flag bits, a standard read reads the flag bit to determine whether an interrupt has occurred. Bit manipulation instructions, however, always read interrupt request flag bits as “1”. This is to prevent unintentionally clearing the flag by writing “0” to the interrupt request flag bit when performing bit manipulation of a different bit.
This appendix lists the mask options for the MB89160/160A/160L series.

## Mask Options

### Table Ca Mask Options

<table>
<thead>
<tr>
<th>No.</th>
<th>Part number</th>
<th>Specifying procedure</th>
<th>Power-on reset</th>
<th>Main clock oscillation stabilization delay time initial value* selection (F&lt;sub&gt;CH&lt;/sub&gt; = 4.2 MHz)</th>
<th>Reset pin output</th>
<th>Clock mode selection</th>
<th>LCD driving power supply</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• 00: 2&lt;sup&gt;4&lt;/sup&gt;/F&lt;sub&gt;CH&lt;/sub&gt; (Approx. 3.8 us)</td>
<td>Selectable</td>
<td>Selectable</td>
<td>Internal voltage divider</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• 01: 2&lt;sup&gt;12&lt;/sup&gt;/F&lt;sub&gt;CH&lt;/sub&gt; (Approx. 1.0 ms)</td>
<td>Selectable</td>
<td>Selectable</td>
<td>Internal voltage divider</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• 10: 2&lt;sup&gt;16&lt;/sup&gt;/F&lt;sub&gt;CH&lt;/sub&gt; (Approx. 15.6 ms)</td>
<td>Selectable</td>
<td>Selectable</td>
<td>Selectable by version number</td>
</tr>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td>• 11: 2&lt;sup&gt;18&lt;/sup&gt;/F&lt;sub&gt;CH&lt;/sub&gt; (Approx. 62.4 ms)</td>
<td>Selectable</td>
<td>Selectable</td>
<td>Internal voltage divider</td>
</tr>
</tbody>
</table>

F<sub>CH</sub>: Main clock source oscillation frequency

*: This option selects the state to which the oscillator stabilization “wait time” bits of the system control register (SYCC: WT1, WT0) are initialized at reset.
Table Cb Mask Options (Segment Options)

<table>
<thead>
<tr>
<th>Port pin names corresponding to LCD controller-driver common and segment outputs</th>
<th>Part number</th>
<th>MB89161/161A</th>
<th>MB89163/163A/163L</th>
<th>MB89165/165A/165L</th>
<th>MB89PV160</th>
</tr>
</thead>
<tbody>
<tr>
<td>P40 to P43, P44 to P47, P50 to P53, P54 to P57, P70 to P71</td>
<td>Specifying procedure</td>
<td>Specify when ordering masking</td>
<td>Select by version number</td>
<td>Select by version number</td>
<td></td>
</tr>
<tr>
<td>○ ○ ○ ○ ○</td>
<td>SEG0 to SEG23 (24 pins) COM0 to COM3 (4 pins)</td>
<td>Specify as SEG=4</td>
<td>-101</td>
<td>-201</td>
<td></td>
</tr>
<tr>
<td>○ ○ ○ ○ ○</td>
<td>SEG0 to SEG19 (20 pins) COM0 to COM3 (4 pins)</td>
<td>Specify as SEG=3</td>
<td>-102</td>
<td>-202</td>
<td></td>
</tr>
<tr>
<td>× × ○ ○ ○</td>
<td>SEG0 to SEG15 (16 pins) COM0 to COM3 (4 pins)</td>
<td>Specify as SEG=2</td>
<td>-103</td>
<td>-204</td>
<td></td>
</tr>
<tr>
<td>× × ○ × ×</td>
<td>SEG0 to SEG11 (12 pins) COM0, COM1 (2 pins)</td>
<td>Specify as SEG=1</td>
<td>-104</td>
<td>-104</td>
<td></td>
</tr>
<tr>
<td>× × × × ×</td>
<td>SEG0 to SEG7 (8 pins) COM0, COM1 (2 pins)</td>
<td>Specify as SEG=0</td>
<td>-105</td>
<td>-101</td>
<td></td>
</tr>
</tbody>
</table>

● Used as common/segment output pins. (Pull-up resistors may not be selected for these pins.)

× Used as output-only port pins. (Except for pins P70 and P71, pull-up resistors may be selected.)

Table Cc Versions

<table>
<thead>
<tr>
<th>Mass production product</th>
<th>One-time PROM product</th>
<th>EPROMs</th>
<th>Piggyback/evaluation product</th>
<th>Number of segment pins</th>
<th>Booster</th>
</tr>
</thead>
<tbody>
<tr>
<td>MB89160A series</td>
<td>MB89P165-201-202-203</td>
<td>MB89W165-201-202-203</td>
<td>–</td>
<td>24 (4 commons)</td>
<td>Yes</td>
</tr>
</tbody>
</table>

Table Cd Ordering Information

<table>
<thead>
<tr>
<th>Part number</th>
<th>Package</th>
<th>Remarks</th>
</tr>
</thead>
<tbody>
<tr>
<td>MB89161-PFV, MB89161A-PFV</td>
<td>80-pin Plastic SQFP (FPT-80P-M05)</td>
<td></td>
</tr>
<tr>
<td>MB89163-PFV, MB89163A-PFV</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MB89165-PFV, MB89165A-PFV</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MB89163L-PFV, MB89165L-PFV</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MB89165-XXX-PFV</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MB89161-FF, MB89161A-FF</td>
<td>80-pin Plastic QFP (FPT-80P-M06)</td>
<td></td>
</tr>
<tr>
<td>MB89163-FF, MB89163A-FF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MB89165-FF, MB89165A-FF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MB89163L-FF, MB89165L-FF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MB89165-XXX-FF</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MB89161-PFS, MB89161A-PFS</td>
<td>80-pin Plastic SQFP (FPT-80P-M11)</td>
<td></td>
</tr>
<tr>
<td>MB89163-PFS, MB89163A-PFS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MB89165-PFS, MB89165A-PFS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MB89163L-PFS, MB89165L-PFS, MB89165-XXX-PFS</td>
<td></td>
<td></td>
</tr>
<tr>
<td>MB89W165-XXX-PF</td>
<td>80-pin Ceramic QFP (FPT-80C-A02)</td>
<td></td>
</tr>
<tr>
<td>MB89PV160-XXX-PF</td>
<td>80-pin Ceramic QFP (FPT-80C-A02)</td>
<td></td>
</tr>
</tbody>
</table>

Reference: For information on XXX, see section “Tables Cc Versions.”
D Programming Specifications for One-time PROM and EPROM Microcontrollers

In EPROM mode, the MB89P165 and MB89W165 function equivalent to the MBM27C256A. This allows the PROM to be programmed with a general-purpose EPROM programmer by using the dedicated adaptor. Note that the electronic signature mode cannot be used.

EPROM Programmer Socket Adaptor

Connect the jumper pin on the adapter to Vss.

Depending on the EPROM programmer, inserting a capacitor of about 0.1 µF between Vpp and Vss or Vcc and Vss can stabilize programming operations.

Table Da lists the EPROM programmer socket adaptors.

<table>
<thead>
<tr>
<th>Package</th>
<th>Compatible socket adaptor</th>
</tr>
</thead>
<tbody>
<tr>
<td>FPT-80P-M05</td>
<td>ROM-80SCF-28DP-8L</td>
</tr>
<tr>
<td>FPT-80P-M06</td>
<td>ROM-80GF-28DP-8L3</td>
</tr>
<tr>
<td>FPT-80P-M11</td>
<td>ROM-80GF2-28DP-8L2</td>
</tr>
</tbody>
</table>

Inquiry:

Sun Hayato Co., Ltd.: TEL. 81-3-3802-5760

Memory Map in EPROM Mode

Figure Da shows the memory map in EPROM mode. Write the option data in the option setting area after consulting the “OTPROM Option Bit Map”.

![Figure D.1a Memory Map in EPROM Mode](image-url)
### OTPROM Options Bit Map

#### Table Db OTPROM Option Bit Map (MB89P165 and MB89W165)

<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>3FF0h</td>
<td>Vacancy</td>
<td>Vacancy</td>
<td>Oscillation stabilization delay time</td>
<td>Vacancy</td>
<td>Readable</td>
<td>Reset pin output</td>
<td>1: Yes</td>
<td>0: No</td>
</tr>
<tr>
<td></td>
<td>11.2u/sFCH</td>
<td>01.2u/sFCH</td>
<td>10.2u/sFCH</td>
<td>00.2u/sFCH</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P07</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>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>P06</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>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>P05</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>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>P04</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>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>P03</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>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>P02</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>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>P01</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>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>P00</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>Pull-up</td>
<td>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>

- Set each bit to “1” to erase.
- Do not write “0” to blank bits.
- Unless written to “0,” blank bits read out as “1.”

**FCH:** Main clock oscillator frequency

### Recommended Screening Conditions

High-temperature aging is recommended as the pre-assembly screening procedure for a product with a blanked OTPROM microcomputer program.

Figure Db shows the screening procedure.

<table>
<thead>
<tr>
<th>Program, verify</th>
<th>Aging</th>
<th>Data verification</th>
<th>Assembly</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>+150°C, 48 Hrs.</td>
<td></td>
<td></td>
</tr>
</tbody>
</table>

**Figure D.2a** Screening Procedure
D. Programming Specifications for One-Time PROM and EPROM Microcontrollers

D.1 Programming Yield and Erasure

This section describes the programming yield and the data erasure on EPROM microcomputer.

- Programming Yield
  
  All bits cannot be programmed at Fujitsu shipping test to a blanked OTPROM microcomputer, due to its nature. For this reason, a programming yield of 100% cannot be assured at all times.

- Notes on Using and Data Erasure on EPROM Microcomputer
  
  - Erasure

  In order to clear all locations of their programmed contents, it is necessary to expose the internal EPROM to an ultraviolet light source. A dosage of 10 W-seconds/cm² is required to completely erase an internal EPROM. This dosage can be obtained by exposure to an ultraviolet lamp (wavelength of 2537 Angstroms (Å)) with intensity of 12000 μW/cm² for 15 to 21 minutes. The internal EPROM should be about one inch from the source and all filters should be removed from the UV light source prior to erasure.

  It is important to note that the internal EPROM and similar devices, will erase with light sources having wavelengths shorter than 4000 Å. Although erasure time will be much longer than with UV source at 2537 Å, nevertheless the exposure to fluorescent light and sunlight will eventually erase the internal EPROM, and exposure to them should be prevented to realize maximum system reliability. If used in such an environment, the package windows should be covered by an opaque label or substance.
D Programming Specifications for One-Time PROM and EPROM Microcontrollers

D.2 Programming to the EPROM with Piggyback/Evaluation Device

This section describes the programming to the EPROM with piggyback/evaluation device.

- **EPROM for Use**
  
  MBM27C256A-20TV

- **Programming Socket Adaptor**
  
  To program to the PROM using an EPROM programmer, use the socket adaptor (manufacturer: Sun Hayato Co., Ltd.) listed below.

  **Table D.2a Programming Socket Adaptor**

<table>
<thead>
<tr>
<th>Package</th>
<th>Adaptor socket part number</th>
</tr>
</thead>
<tbody>
<tr>
<td>LCC-32 (Rectangle)</td>
<td>ROM-32LC-28DP-YG</td>
</tr>
</tbody>
</table>

  Inquiry: Sun Hayato Co., Ltd.: TEL 81-3-5396-9106

- **Memory Space**

  **Figure D.2a Memory Map of Piggyback/Evaluation Device**

- **Programming to EPROM**
  
  1. Set the EPROM programmer to the MBM27C256A.
  2. Load program data into the EPROM programmer at 0006H to 7FFFH.
  3. Program to 0000H to 7FFFH with the EPROM programmer.
This section describes the pin states of the MB89160/160A/160L series in each mode.

### Pin States in Each Mode

<table>
<thead>
<tr>
<th>Pin name</th>
<th>Normal operation</th>
<th>Sleep mode</th>
<th>Stop mode SPL=“0”</th>
<th>Stop mode SPL=“1”</th>
<th>During a reset</th>
</tr>
</thead>
<tbody>
<tr>
<td>P00/INT20 to P07/INT27</td>
<td>Port I/O/external interrupt 2 input</td>
<td>Hold/external interrupt 2 input</td>
<td>Hold/external interrupt 2 input</td>
<td>Hi-z/external interrupt 2 input</td>
<td></td>
</tr>
<tr>
<td>P10/INT10 to P13/INT13</td>
<td>Port I/O/external interrupt 1 input</td>
<td>Hold/external interrupt 1 input</td>
<td>Hold/external interrupt 1 input</td>
<td>Hi-z/external interrupt 1 input</td>
<td></td>
</tr>
<tr>
<td>P14 to P17</td>
<td>Port I/O</td>
<td>Hold</td>
<td>Hold</td>
<td>Hold</td>
<td>Hi-z</td>
</tr>
<tr>
<td>X0, X0A</td>
<td>Oscillator input</td>
<td>Oscillator input</td>
<td>Hi-z</td>
<td>Hi-z</td>
<td>Oscillator input</td>
</tr>
<tr>
<td>X1, X1A</td>
<td>Oscillator output</td>
<td>Oscillator output</td>
<td>“H” output</td>
<td>“H” output</td>
<td>Oscillator output</td>
</tr>
<tr>
<td>MOD0</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>P20/EC</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Hi-z</td>
</tr>
<tr>
<td>P21</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P22/TO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P23/SI</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P24/SQ</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P25/SCK</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P26</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P27/PWM2*</td>
<td>Port or peripheral I/O</td>
<td>Hold/peripheral I/O</td>
<td>Hold</td>
<td>Hi-z</td>
<td></td>
</tr>
<tr>
<td>P30/BZ/RCO</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P31/PWM1*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P32/C1*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P33/CD*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P40 to P47*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P46 to P67*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P70, P71*</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>P50/AN0 to P57/AN7</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>COM0 to COM3*</td>
<td>LCD common output*</td>
<td>LCD common output*</td>
<td>Hold*</td>
<td>Hold*</td>
<td>&quot;L&quot; output</td>
</tr>
<tr>
<td>SEG0 to SEG23*</td>
<td>LCD segment output*</td>
<td>LCD segment output*</td>
<td>Hold*</td>
<td>Hold*</td>
<td></td>
</tr>
</tbody>
</table>

*1: The reset pin can function as an output depending on an option setting.
*2: In devices with internal voltage boosters (A series), these pins cannot be used as port pins because they are used to connect external capacitors.
*3: If segment output is selected, the states of these pins are as indicated for pins SEG8 to SEG15 and SEG16 to SEG23.
*4: If common output is selected, the states of these pins are as indicated for COM2 and COM3.
*5: Pins COM2, COM3 and SEG8 to SEG23 are also used as general-purpose output ports (selected as mask option).
*6: Operate as common/segment output if an LCD controller-driver operating clock is supplied.
*7: Pin states of P27 and P31 are undetermined until the internal clock starts operation.
Hi-z: High impedance
SPL: Pin state specification bit in the standby control register (STBC)
Hold: The pin set as output holds its state (level) before changing to each mode.
Troubleshooting

Follow the checklist given below if problems occur. As the problem may be related to software, also consult the manual of the software you are using.

■ Checklist

<table>
<thead>
<tr>
<th>Symptom</th>
<th>Cause</th>
<th>Measures</th>
<th>Check column</th>
</tr>
</thead>
<tbody>
<tr>
<td>The microcontroller does not operate correctly.</td>
<td>The power supply (Vcc, GND) is not connected.</td>
<td>Connect the power supply (Vcc, GND).</td>
<td></td>
</tr>
<tr>
<td></td>
<td>Not all pin inputs are within the electrical characteristics of the device.</td>
<td>Ensure that all pin inputs are within the electrical characteristics of the device.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The MOD1 and MOD2 pins are not connected for the operation mode being used.</td>
<td>Set the MOD1 and MOD2 pins.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>No reset is applied after turning the power on for a product that does not have the power-on reset option selected.</td>
<td>Input a reset after turning on the power.</td>
<td></td>
</tr>
<tr>
<td></td>
<td>The oscillator is connected but no oscillation is generated when turning on the power or when inputting a reset.</td>
<td>If an oscillation is present: Check that the oscillation is at the frequency of the connected oscillator.</td>
<td></td>
</tr>
<tr>
<td></td>
<td></td>
<td>If no oscillation is present: The device may have entered stop mode (a standby mode). Check the program.</td>
<td></td>
</tr>
<tr>
<td>Performing evaluation using the tools operates correctly, but the one-time or mask ROM product does not operate correctly.</td>
<td>Registers and RAM are not set to their initial settings.</td>
<td>If operation is unstable despite repeatedly turning on the power supply, initial values may not be set.</td>
<td></td>
</tr>
</tbody>
</table>

■ Check Items before Contacting Fujitsu

Collect the following information before contacting the Fujitsu Sales Department or agent:

1) Determine the difference in the pin levels (waveforms) between when the device is operating normally and abnormally.
2) Determine how often the problem occurs, the conditions associated with the problem, the number of items affected, and the effect of voltage, temperature and frequency on the occurrence of the problem.
3) Check the operation on one-time PROM, mask ROM, and piggy-back/evaluation products.
For further information please contact:

Japan
FUJITSU LIMITED
Corporate Global Business Support Division
Electronic Devices
KAWASAKI PLANT, 4-1-1, Kamikodanaka
Nakahara-ku, Kawasaki-shi
Kanagawa 211-8588, Japan
Tel: (044) 754-3763
Fax: (044) 754-3329
http://www.fujitsu.co.jp/

North and South America
FUJITSU MICROELECTRONICS, INC.
Semiconductor Division
3545 North First Street
San Jose, CA 95134-1804, USA
Tel: (408) 922-9000
Fax: (408) 922-9179
Customer Response Center
Mon. - Fri.: 7 am - 5 pm (PST)
Tel: (800) 866-8608
Fax: (408) 922-9179
http://www.fujitsumicro.com/

Europe
FUJITSU MIKROELEKTRONIK GmbH
Am Siebenstein 6-10
D-63303 Dreieich-Buchschlag
Germany
Tel: (06103) 690-0
Fax: (06103) 690-122
http://www.fujitsu-ede.com/

Asia Pacific
FUJITSU MICROELECTRONICS ASIA PTE LTD
#05-08, 151 Lorong Chuan
New Tech Park
Singapore 556741
Tel: (65) 281-0770
Fax: (65) 281-0220
http://www.fmap.com.sg/