# Renesas Electronics America Inc - UPD78F1187AGJ-GAE-AX Datasheet

**Welcome to E-XFL.COM**

## What is "Embedded - Microcontrollers"?

"Embedded - Microcontrollers" refer to small, integrated circuits designed to perform specific tasks within larger systems. These microcontrollers are essentially compact computers on a single chip, containing a processor core, memory, and programmable input/output peripherals. They are called "embedded" because they are embedded within electronic devices to control various functions, rather than serving as standalone computers. Microcontrollers are crucial in modern electronics, providing the intelligence and control needed for a wide range of applications.

## Applications of "Embedded - Microcontrollers"

| Details | |
|---|---|
| Product Status | Active |
| Core Processor | 78K/0R |
| Core Size | 16-Bit |
| Speed | 20MHz |
| Connectivity | 3-Wire SIO, EBI/EMI, I²C, LINbus, UART/USART |
| Peripherals | DMA, LVD, POR, PWM, WDT |
| Number of I/O | 127 |
| Program Memory Size | 384KB (384K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 24K x 8 |
| Voltage - Supply (Vcc/Vdd) | 1.8V ~ 5.5V |
| Data Converters | A/D 16x10b; D/A 2x8b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 144-LQFP |
| Supplier Device Package | - |
| Purchase URL | https://www.e-xfl.com/product-detail/renesas-electronics-america/upd78f1187agj-gae-ax |

### 3.1.6 Data memory addressing

Addressing refers to the method of specifying the address of the instruction to be executed next or the address of the register or memory relevant to the execution of instructions.

Several addressing modes are provided for addressing the memory relevant to the execution of instructions for the 78K0R/KJ3, based on operability and other considerations. For areas containing data memory in particular, special addressing methods designed for the functions of special function registers (SFR) and general-purpose registers are available for use. Figures 3-7 to 3-11 show correspondence between data memory and addressing.

<R> **Figure 3-7. Correspondence Between Data Memory and Addressing (μPD78F1184A)**

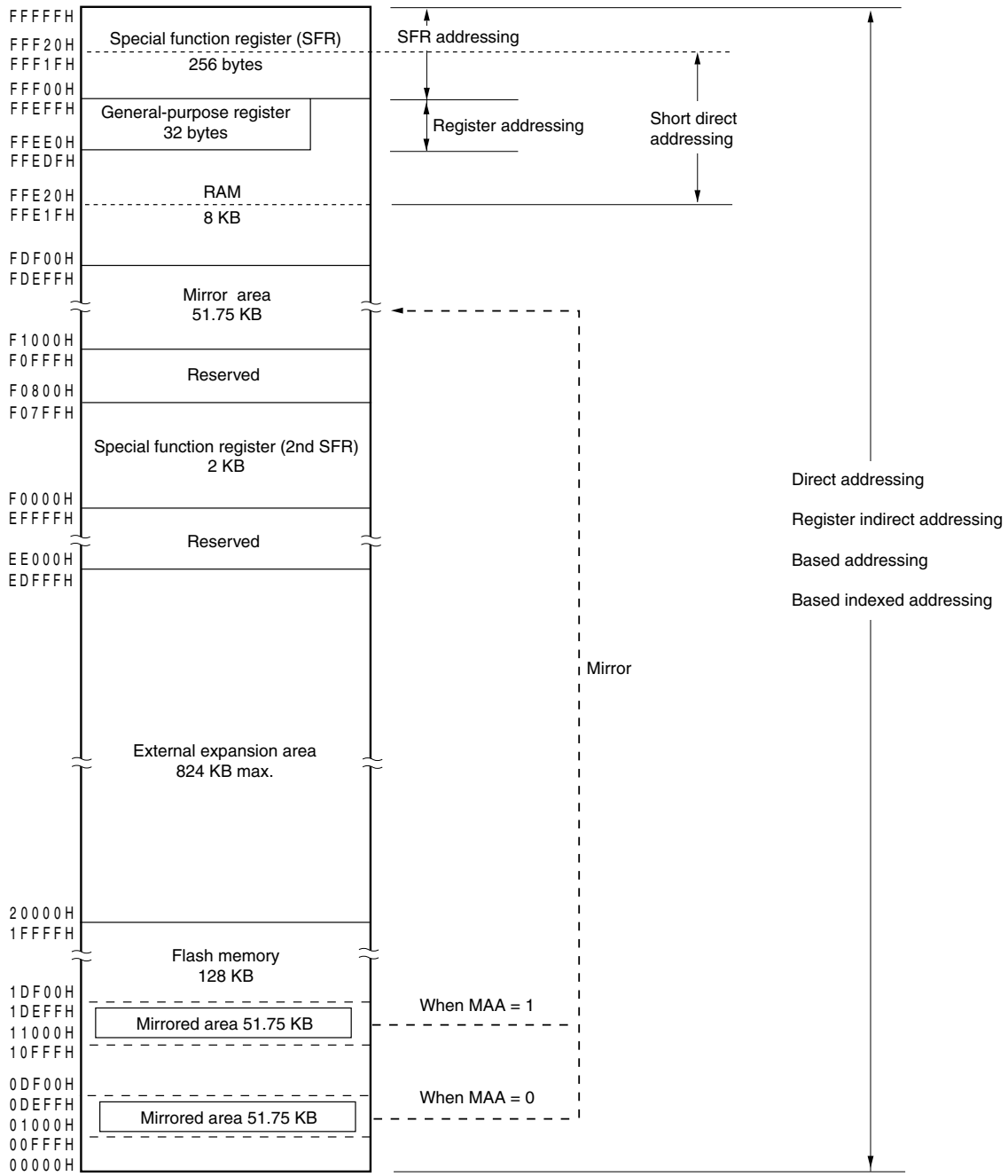
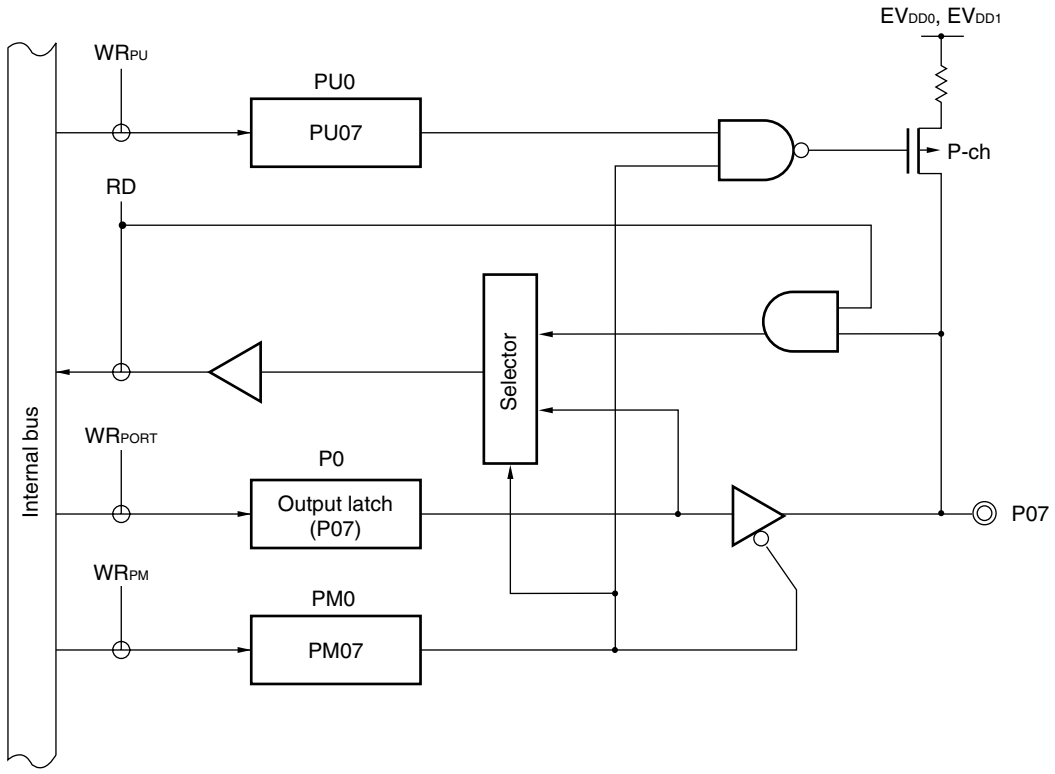


**Figure 4-8. Block Diagram of P07**




P0: Port register 0
PU0: Pull-up resistor option register 0
PM0: Port mode register 0
RD: Read signal
WR××: Write signal

### 4.2.9 Port 8

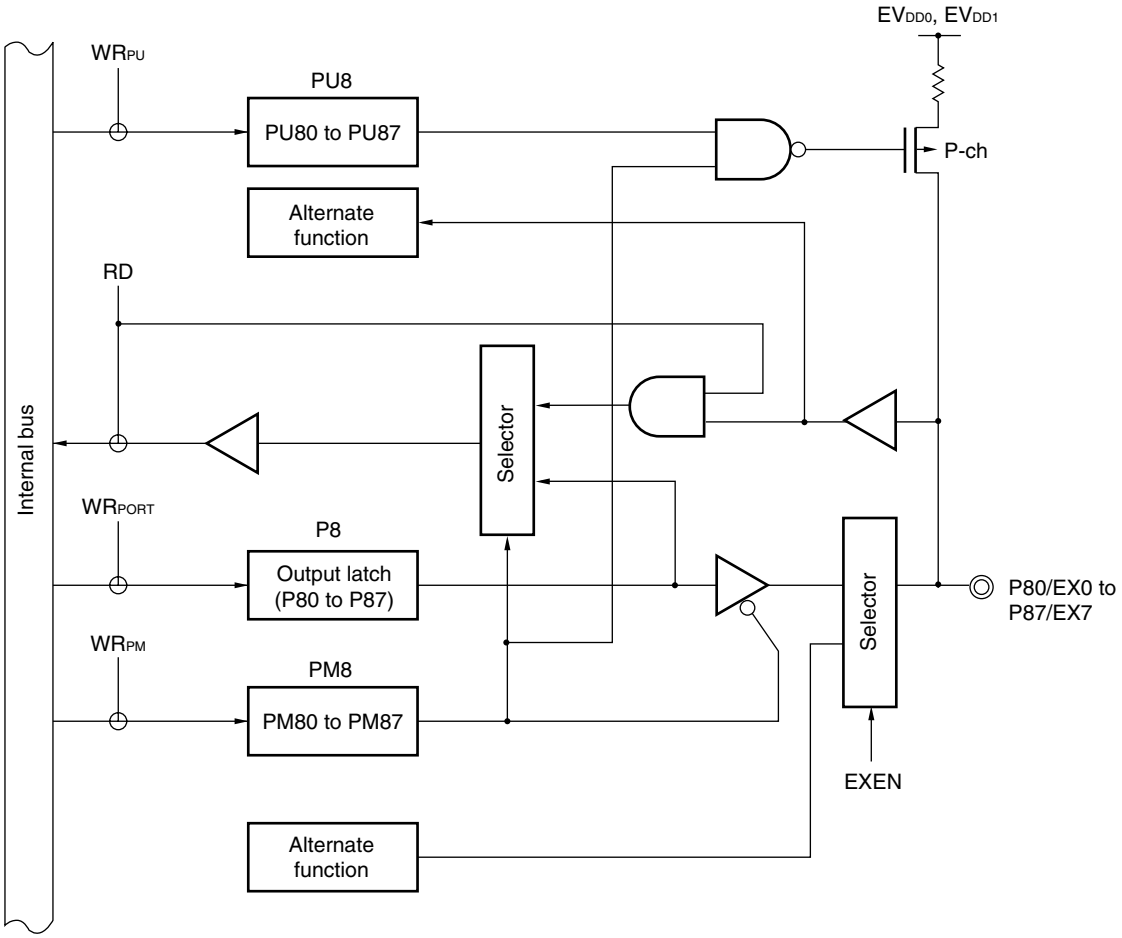
Port 8 is an 8-bit I/O port with an output latch. Port 8 can be set to the input mode or output mode in 1-bit units using port mode register 8 (PM8). When the P80 to P87 pins are used as an input port, use of an on-chip pull-up resistor can be specified in 1-bit units by pull-up resistor option register 8 (PU8).

This port can also be used for external expansion I/O (multiplexed address/data bus, data bus). When the external expansion I/O function is used, it controls the I/O ignoring the settings on port mode register 8 (PM8), port register 8 (P8), and pull-up resistor option register 8 (PU8).

Reset signal generation sets port 8 to input mode.

Figure 4-31 shows a block diagram of port 8.

**Figure 4-31. Block Diagram of P80 to P87**




P8: Port register 8
PU8: Pull-up resistor option register 8
PM8: Port mode register 8
RD: Read signal
WR××: Write signal
EXEN: Bit 7 of memory extension mode control register (MEM)

**(4) Oscillation stabilization time select register (OSTS)**

This register is used to select the X1 clock oscillation stabilization wait time when the STOP mode is released.

When the X1 clock is selected as the CPU clock, the operation automatically waits for the time set using OSTS after the STOP mode is released.

When the internal high-speed oscillation clock is selected as the CPU clock, confirm with OSTC that the desired oscillation stabilization time has elapsed after the STOP mode is released. The oscillation stabilization time can be checked up to the time set using OSTC.

OSTS can be set by an 8-bit memory manipulation instruction.

Reset signal generation sets OSTS to 07H.

Table 6-4 shows transition of the CPU clock and examples of setting the SFR registers.

**Table 6-4. CPU Clock Transition and SFR Register Setting Examples (1/4)**

**(1) CPU operating with internal high-speed oscillation clock (B) after reset release (A)**

| Status Transition | SFR Register Setting |
|---|---|
| (A) → (B) | SFR registers do not have to be set (default status after reset release). |

**(2) CPU operating with high-speed system clock (C) after reset release (A)**

(The CPU operates with the internal high-speed oscillation clock immediately after a reset release (B).)

(Setting sequence of SFR registers) ⟶

| Setting Flag of SFR Register / Status Transition | CMC Register Note 1 | | | CSC Register | OSMC Register | OSTC Register | CKC Register |
|---|---|---|---|---|---|---|---|
| | EXCLK | OSCSEL | AMPH | MSTOP | FSEL | | MCM0 |
| (A) → (B) → (C) (X1 clock: 2 MHz ≤ fx ≤ 10 MHz) | 0 | 1 | 0 | 0 | 0 | Must be checked | 1 |
| (A) → (B) → (C) (X1 clock: 10 MHz < fx ≤ 20 MHz) | 0 | 1 | 1 | 0 | 1 Note 2 | Must be checked | 1 |
| <R> (A) → (B) → (C) (external main clock) | 1 | 1 | × | 0 | 0/1 | Must not be checked | 1 |

**Notes 1.** The CMC and OSMC registers can be written only once by an 8-bit memory manipulation instruction after reset release.

**2.** FSEL = 1 when $f_{CLK}$ > 10 MHz
If a divided clock is selected and $f_{CLK}$ ≤ 10 MHz, use with FSEL = 0 is possible even if fx > 10 MHz.

**Caution Set the clock after the supply voltage has reached the operable voltage of the clock to be set (see CHAPTER 29 ELECTRICAL SPECIFICATIONS (STANDARD PRODUCTS) and CHAPTER 30 ELECTRICAL SPECIFICATIONS ((A) GRADE PRODUCTS)).**

<R> **Remark** ×: don't care

**(3) CPU operating with subsystem clock (D) after reset release (A)**

(The CPU operates with the internal high-speed oscillation clock immediately after a reset release (B).)

(Setting sequence of SFR registers) ⟶

| Setting Flag of SFR Register / Status Transition | CMC Register Note | CSC Register | Waiting for Oscillation Stabilization | CKC Register |
|---|---|---|---|---|
| | OSCSELS | XTSTOP | | CSS |
| (A) → (B) → (D) | 1 | 0 | Necessary | 1 |

**Note** The CMC register can be written only once by an 8-bit memory manipulation instruction after reset release.

**Remark** (A) to (I) in Table 6-4 correspond to (A) to (I) in Figure 6-15.

**Figure 7-51. Example of Set Contents of Registers to Measure Input Pulse Interval**

**(a) Timer mode register mn (TMRmn)**

| | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| TMRmn | CKSmn | | | CCSmn | MAS TERmn | STSmn2 | STSmn1 | STSmn0 | CISmn1 | CISmn0 | | | MDmn3 | MDmn2 | MDmn1 | MDmn0 |
| | 1/0 | 0 | 0 | 0 | 0 | 0 | 0 | 1 | 1/0 | 1/0 | 0 | 0 | 0 | 1 | 0 | 1/0 |

Operation mode of channel n
010B: Capture mode

Setting of operation when counting is started
0: Does not generate INTTMmn when counting is started.
1: Generates INTTMmn when counting is started.

Selection of TImn pin input edge
00B: Detects falling edge.
01B: Detects rising edge.
10B: Detects both edges.
11B: Setting prohibited

Capture trigger selection
001B: Selects the TImn pin input valid edge.

Slave/master selection
0: Cleared to 0 when single-operation function is selected.

Count clock selection
0: Selects operation clock.

Operation clock selection
0: Selects CKm0 as operation clock of channel n.
1: Selects CKm1 as operation clock of channel n.

**(b) Timer output register m (TOm)**

| | Bit n |
|---|---|
| TOm | TOmn |
| | 0 |

0: Outputs 0 from TOmn.

**(c) Timer output enable register m (TOEm)**

| | Bit n |
|---|---|
| TOEm | TOEmn |
| | 0 |

0: Stops TOmn output operation by counting operation.

**(d) Timer output level register m (TOLm)**

| | Bit n |
|---|---|
| TOLm | TOLmn |
| | 0 |

0: Cleared to 0 when TOMmn = 0 (toggle mode).

**(e) Timer output mode register m (TOMm)**

| | Bit n |
|---|---|
| TOMm | TOMmn |
| | 0 |

0: Sets toggle mode.

**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 7),
mn = 00 to 07, 10 to 13

**(3) Real-time counter control register 1 (RTCC1)**

The RTCC1 register is an 8-bit register that is used to control the alarm interrupt function and the wait time of the counter.

RTCC1 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

**Figure 8-4. Format of Real-Time Counter Control Register 1 (RTCC1) (1/2)**

Address: FFF9EH After reset: 00H R/W

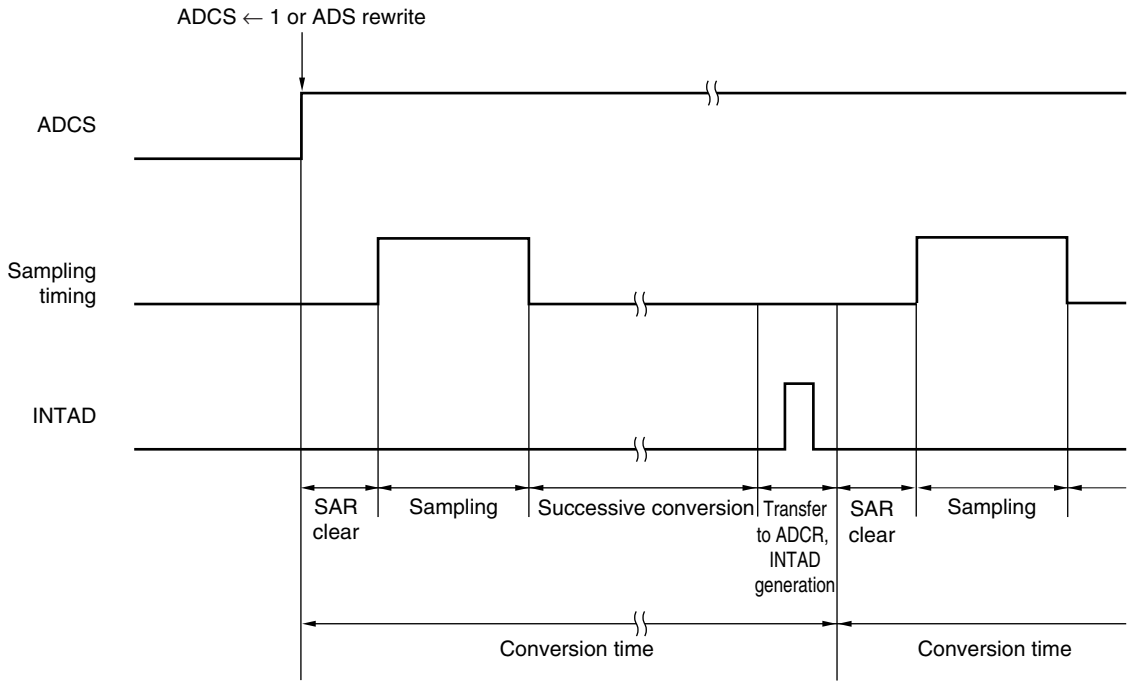
| Symbol | <7> | <6> | 5 | <4> | <3> | 2 | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| RTCC1 | WALE | WALIE | 0 | WAFG | RIFG | 0 | RWST | RWAIT |

| WALE | Alarm operation control |
|---|---|
| 0 | Match operation is invalid. |
| 1 | Match operation is valid. |
| When setting a value to the WALE bit while the counter operates (RTCE = 1) and WALIE = 1, rewrite the WALE bit after disabling interrupt servicing INTRTC by using the interrupt mask flag register. Furthermore, clear the WAFG and RTCIF flags after rewriting the WALE bit. When setting each alarm register (WALIE flag of RTCC1, the ALARMWM register, the ALARMWH register, and the ALARMWW register), set match operation to be invalid (“0”) for the WALE bit. | |

| WALIE | Control of alarm interrupt (INTRTC) function operation |
|---|---|
| 0 | Does not generate interrupt on matching of alarm. |
| 1 | Generates interrupt on matching of alarm. |

| WAFG | Alarm detection status flag |
|---|---|
| 0 | Alarm mismatch |
| 1 | Detection of matching of alarm |
| This is a status flag that indicates detection of matching with the alarm. It is valid only when WALE = 1 and is set to “1” one clock (32.768 kHz) after matching of the alarm is detected. This flag is cleared when “0” is written to it. Writing “1” to it is invalid. | |

**Figure 11-6. A/D Converter Sampling and A/D Conversion Timing**



### 13.1.2 UART (UART0, UART1, UART2, UART3)

This is a start-stop synchronization function using two lines: serial data transmission (TxD) and serial data reception (RxD) lines. It transmits or receives data in asynchronization with the party of communication (by using an internal baud rate). Full-duplex UART communication can be realized by using two channels, one dedicated to transmission (even channel) and the other to reception (odd channel).

[Data transmission/reception]

- Data length of 5, 7, or 8 bits
- Select the MSB/LSB first
- Level setting of transmit/receive data and select of reverse
- Parity bit appending and parity check functions
- Stop bit appending

[Interrupt function]

- Transfer end interrupt/buffer empty interrupt
- Error interrupt in case of framing error, parity error, or overrun error

[Error detection flag]

- Framing error, parity error, or overrun error

The LIN-bus is accepted in UART3 (2 and 3 channels of unit 1)

[LIN-bus functions]

- Wakeup signal detection
- Sync break field (SBF) detection
- Sync field measurement, baud rate calculation

} External interrupt (INTP0) or timer array unit (TAU) is used.

**Figure 13-5. Format of Serial Clock Select Register m (SPSm)**

Address: F0126H, F0127H (SPS0), F0166H, F0167H (SPS1) After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| SPSm | 0 | 0 | 0 | 0 | 0 | 0 | 0 | 0 | PRS m13 | PRS m12 | PRS m11 | PRS m10 | PRS m03 | PRS m02 | PRS m01 | PRS m00 |

| PRS mp3 | PRS mp2 | PRS mp1 | PRS mp0 | Section of operation clock (CKmp)[Note 1] | | | | |
|---|---|---|---|---|---|---|---|---|
| | | | | | $f_{CLK}$ = 2 MHz | $f_{CLK}$ = 5 MHz | $f_{CLK}$ = 10 MHz | $f_{CLK}$ = 20 MHz |
| 0 | 0 | 0 | 0 | $f_{CLK}$ | 2 MHz | 5 MHz | 10 MHz | 20 MHz |
| 0 | 0 | 0 | 1 | $f_{CLK}/2$ | 1 MHz | 2.5 MHz | 5 MHz | 10 MHz |
| 0 | 0 | 1 | 0 | $f_{CLK}/2^2$ | 500 kHz | 1.25 MHz | 2.5 MHz | 5 MHz |
| 0 | 0 | 1 | 1 | $f_{CLK}/2^3$ | 250 kHz | 625 kHz | 1.25 MHz | 2.5 MHz |
| 0 | 1 | 0 | 0 | $f_{CLK}/2^4$ | 125 kHz | 313 kHz | 625 kHz | 1.25 MHz |
| 0 | 1 | 0 | 1 | $f_{CLK}/2^5$ | 62.5 kHz | 156 kHz | 313 kHz | 625 kHz |
| 0 | 1 | 1 | 0 | $f_{CLK}/2^6$ | 31.3 kHz | 78.1 kHz | 156 kHz | 313 kHz |
| 0 | 1 | 1 | 1 | $f_{CLK}/2^7$ | 15.6 kHz | 39.1 kHz | 78.1 kHz | 156 kHz |
| 1 | 0 | 0 | 0 | $f_{CLK}/2^8$ | 7.81 kHz | 19.5 kHz | 39.1 kHz | 78.1 kHz |
| 1 | 0 | 0 | 1 | $f_{CLK}/2^9$ | 3.91 kHz | 9.77 kHz | 19.5 kHz | 39.1 kHz |
| 1 | 0 | 1 | 0 | $f_{CLK}/2^{10}$ | 1.95 kHz | 4.88 kHz | 9.77 kHz | 19.5 kHz |
| 1 | 0 | 1 | 1 | $f_{CLK}/2^{11}$ | 977 Hz | 2.44 kHz | 4.88 kHz | 9.77 kHz |
| 1 | 1 | 1 | 1 | INTTM02 if m = 0, INTTM03 if m = 1[Note 2] | | | | |
| Other than above | | | | Setting prohibited | | | | |

**Notes1.** When changing the clock selected for $f_{CLK}$ (by changing the system clock control register (CKC) value), do so after having stopped (STm = 000FH) the operation of the serial array unit (SAU). When selecting INTTM02 and INTTM03 for the operation clock, also stop the timer array unit 0 (TAU0) (TT0 = 00FFH).

**2.** SAU0 can be operated at a fixed division ratio of the subsystem clock, regardless of the $f_{CLK}$ frequency (main system clock, subsystem clock), by operating the interval timer for which $f_{SUB}/4$ has been selected as the count clock (setting TIS02 (if m = 0) or TIS03 (if m = 1) of the TIS0 register to 1) and selecting INTTM02 and INTTM03 by using the SPSm register in channels 2 and 3 of TAU. When changing $f_{CLK}$, however, SAU and TAU0 must be stopped as described in Note 1 above.

**Cautions 1. Be sure to clear bits 15 to 8 to "0".**

**2. After setting the PER0 register to 1, be sure to set the SPSm register after 4 or more clocks have elapsed.**

**Remarks 1.** $f_{CLK}$: CPU/peripheral hardware clock frequency
$f_{SUB}$: Subsystem clock frequency

**2.** m: Unit number (m = 0, 1), p = 0, 1

**Figure 14-6. Format of IIC Control Register 0 (IICC0) (3/4)**

<R>

| STT0[Note] | Start condition trigger |
|---|---|
| 0 | Do not generate a start condition. |
| 1 | When bus is released (in standby state, when IICBSY = 0):<br>Generate a start condition (for starting as master). When the SCL0 line is high level, the SDA0 line is changed from high level to low level and then the start condition is generated. Next, after the rated amount of time has elapsed, SCL0 is changed to low level (wait state).<br>When a third party is communicating:<br>• When communication reservation function is enabled (IICRSV = 0)<br>Functions as the start condition reservation flag. When set to 1, automatically generates a start condition after the bus is released.<br>• When communication reservation function is disabled (IICRSV = 1)<br>STCF is set to 1 and STT0 is cleared. No start condition is generated.<br>In the wait state (when master device):<br>Generates a restart condition after releasing the wait. |

Cautions concerning set timing

- For master reception: Cannot be set to 1 during transfer. Can be set to 1 only in the waiting period when ACKE0 has been cleared to 0 and slave has been notified of final reception.
- For master transmission: A start condition cannot be generated normally during the acknowledge period. Set to 1 during the wait period that follows output of the ninth clock.
- Cannot be set to 1 at the same time as SPT0.
- Setting STT0 to 1 and then setting it again before it is cleared to 0 is prohibited.

| Condition for clearing (STT0 = 0) | Condition for setting (STT0 = 1) |
|---|---|
| • Cleared by setting SST0 to 1 while communication reservation is prohibited.<br>• Cleared by loss in arbitration<br>• Cleared after start condition is generated by master device<br>• Cleared by LREL0 = 1 (exit from communications)<br>• When IICE0 = 0 (operation stop)<br>• Reset | • Set by instruction |

**Note** The signal of this bit is invalid while IICE0 is 0.

**Remarks 1.** Bit 1 (STT0) becomes 0 when it is read after data setting.

**2.** IICRSV: Bit 0 of IIC flag register (IICF0)
STCF: Bit 7 of IIC flag register (IICF0)

### 14.5.17 Communication operations

The following shows three operation procedures with the flowchart.

**(1) Master operation in single master system**

The flowchart when using the 78K0R/KJ3 as the master in a single master system is shown below.

This flowchart is broadly divided into the initial settings and communication processing. Execute the initial settings at startup. If communication with the slave is required, prepare the communication and then execute communication processing.

**(2) Master operation in multimaster system**

In the $I^2C$ bus multimaster system, whether the bus is released or used cannot be judged by the $I^2C$ bus specifications when the bus takes part in a communication. Here, when data and clock are at a high level for a certain period (1 frame), the 78K0R/KJ3 takes part in a communication with bus released state.

This flowchart is broadly divided into the initial settings, communication waiting, and communication processing. The processing when the 78K0R/KJ3 looses in arbitration and is specified as the slave is omitted here, and only the processing as the master is shown. Execute the initial settings at startup to take part in a communication. Then, wait for the communication request as the master or wait for the specification as the slave. The actual communication is performed in the communication processing, and it supports the transmission/reception with the slave and the arbitration with other masters.

**(3) Slave operation**

An example of when the 78K0R/KJ3 is used as the $I^2C$ bus slave is shown below.

When used as the slave, operation is started by an interrupt. Execute the initial settings at startup, then wait for the INTIIC0 interrupt occurrence (communication waiting). When an INTIIC0 interrupt occurs, the communication status is judged and its result is passed as a flag over to the main processing.

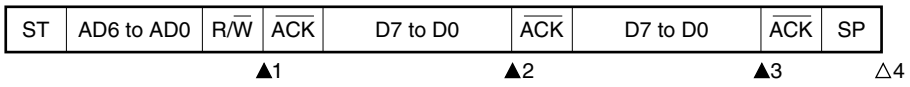
By checking the flags, necessary communication processing is performed.

**(3) Slave device operation (when receiving extension code)**

The device is always participating in communication when it receives an extension code.

**(a) Start ~ Code ~ Data ~ Data ~ Stop**

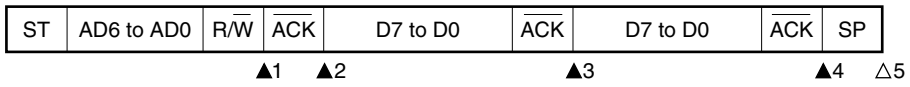
**(i) When WTIM0 = 0**




▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×000B
▲3: IICS0 = 0010×000B
△4: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

**(ii) When WTIM0 = 1**




▲1: IICS0 = 0010×010B
▲2: IICS0 = 0010×110B
▲3: IICS0 = 0010×100B
▲4: IICS0 = 0010××00B
△5: IICS0 = 00000001B

**Remark** ▲: Always generated
△: Generated only when SPIE0 = 1
×: Don't care

# CHAPTER 15 MULTIPLIER

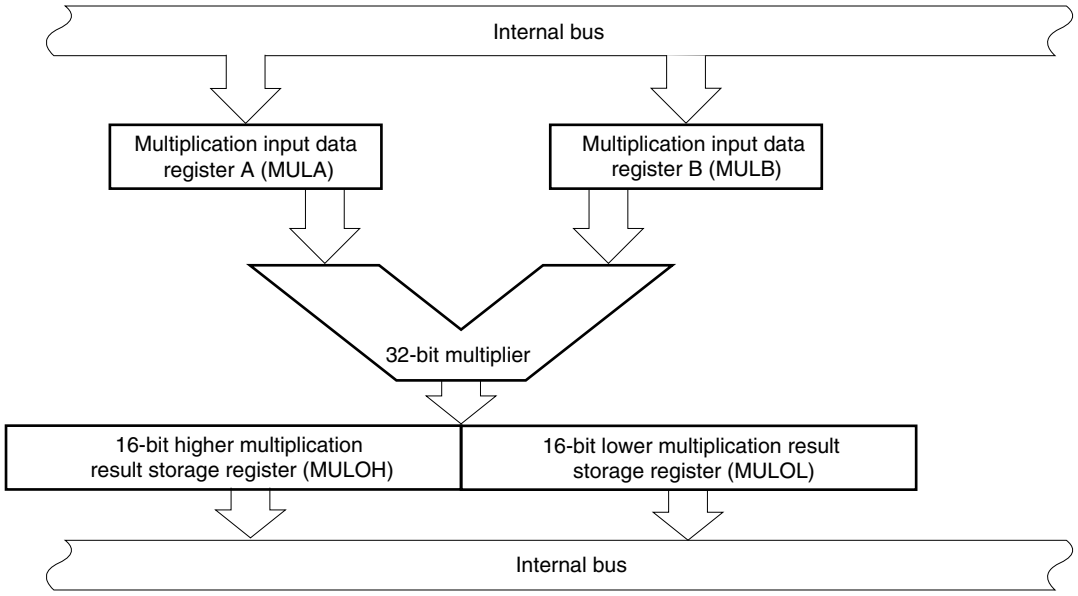
## 15.1 Functions of Multiplier

The multiplier has the following functions.

- Can execute calculation of 16 bits × 16 bits = 32 bits.

Figure 15-1 shows the block diagram of the multiplier.

**Figure 15-1. Block Diagram of Multiplier**



**(2) Interrupt mask flag registers (MK0L, MK0H, MK1L, MK1H, MK2L, MK2H)**

The interrupt mask flags are used to enable/disable the corresponding maskable interrupt servicing.

MK0L, MK0H, MK1L, MK1H, MK2L, and MK2H can be set by a 1-bit or 8-bit memory manipulation instruction. When MK0L and MK0H, MK1L and MK1H, and MK2L and MK2H are combined to form 16-bit registers MK0, MK1, and MK2, they can be set by a 16-bit memory manipulation instruction.

Reset signal generation sets these registers to FFH.

**Remark** If an instruction that writes data to this register is executed, the number of instruction execution clocks increases by 2 clocks.

**Figure 17-3. Format of Interrupt Mask Flag Registers (MK0L, MK0H, MK1L, MK1H, MK2L, MK2H)**

Address: FFFE4H After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| MK0L | PMK5 | PMK4 | PMK3 | PMK2 | PMK1 | PMK0 | LVIMK | WDTIMK |

Address: FFFE5H After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| MK0H | SREMK0 | SRMK0<br>CSIMK01 | STMK0<br>CSIMK00 | DMAMK1 | DMAMK0 | SREMK3 | SRMK3 | STMK3 |

Address: FFFE6H After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| MK1L | TMMK03 | TMMK02 | TMMK01 | TMMK00 | IICMK0 | SREMK1 | SRMK1<br>CSIMK11<br>IICMK11 | STMK1<br>CSIMK10<br>IICMK10 |

Address: FFFE7H After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| MK1H | TMMK04 | SREMK2 | SRMK2<br>CSIMK21<br>IICMK21 | STMK2<br>CSIMK20<br>IICMK20 | KRMK | RTCIMK | RTCMK | ADMK |

Address: FFFD4H After reset: FFH R/W

| Symbol | <7> | <6> | <5> | <4> | <3> | <2> | <1> | <0> |
|---|---|---|---|---|---|---|---|---|
| MK2L | PMK10 | PMK9 | PMK8 | PMK7 | PMK6 | TMMK07 | TMMK06 | TMMK05 |

Address: FFFD5H After reset: FFH R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | <0> |
|---|---|---|---|---|---|---|---|---|
| MK2H | 1 | 1 | 1 | TMMK13 | TMMK12 | TMMK11 | TMMK10 | PMK11 |

| XXMKX | Interrupt servicing control |
|---|---|
| 0 | Interrupt servicing enabled |
| 1 | Interrupt servicing disabled |

**Caution Be sure to set bits 5 to 7 of MK2H to 1.**

## 19.2 Standby Function Operation

### 19.2.1 HALT mode

**(1) HALT mode**

The HALT mode is set by executing the HALT instruction. HALT mode can be set regardless of whether the CPU clock before the setting was the high-speed system clock, internal high-speed oscillation clock, or subsystem clock.

The operating statuses in the HALT mode are shown below.

### 24.1.2 On-chip debug option byte (000C3H/ 010C3H)

- ○ Control of on-chip debug operation
  - On-chip debug operation is disabled or enabled.
- ○ Handling of data of flash memory in case of failure in on-chip debug security ID authentication
  - Data of flash memory is erased or not erased in case of failure in on-chip debug security ID authentication.

**Caution Set the same value as 000C3H to 010C3H when the boot swap operation is used because 000C3H is replaced by 010C3H.**

## 24.2 Format of User Option Byte

The format of user option byte is shown below.

**Figure 24-1. Format of User Option Byte (000C0H/010C0H) (1/2)**

Address: 000C0H/010C0H[Note 1]

| 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|
| WDTINIT | WINDOW1 | WINDOW0 | WDTON | WDCS2 | WDCS1 | WDCS0 | WDSTBYON |

| WDTINIT | Use of interval interrupt of watchdog timer |
|---|---|
| 0 | Interval interrupt is not used. |
| 1 | Interval interrupt is generated when 75% of the overflow time is reached. |

| WINDOW1 | WINDOW0 | Watchdog timer window open period[Note 2] |
|---|---|---|
| 0 | 0 | 25% |
| 0 | 1 | 50% |
| 1 | 0 | 75% |
| 1 | 1 | 100% |

| WDTON | Operation control of watchdog timer counter |
|---|---|
| 0 | Counter operation disabled (counting stopped after reset) |
| 1 | Counter operation enabled (counting started after reset) |

| WDCS2 | WDCS1 | WDCS0 | Watchdog timer overflow time |
|---|---|---|---|
| 0 | 0 | 0 | $2^{10}/f_{IL}$ (3.88 ms) |
| 0 | 0 | 1 | $2^{11}/f_{IL}$ (7.76 ms) |
| 0 | 1 | 0 | $2^{12}/f_{IL}$ (15.52 ms) |
| 0 | 1 | 1 | $2^{13}/f_{IL}$ (31.03 ms) |
| 1 | 0 | 0 | $2^{15}/f_{IL}$ (124.12 ms) |
| 1 | 0 | 1 | $2^{17}/f_{IL}$ (496.48 ms) |
| 1 | 1 | 0 | $2^{18}/f_{IL}$ (992.97 ms) |
| 1 | 1 | 1 | $2^{20}/f_{IL}$ (3971.88 ms) |

## 25.7 Security Settings

The 78K0R/KJ3 supports a security function that prohibits rewriting the user program written to the internal flash memory, so that the program cannot be changed by an unauthorized person.

The operations shown below can be performed using the Security Set command. The security setting is valid when the programming mode is set next.

- Disabling batch erase (chip erase)
  Execution of the block erase and batch erase (chip erase) commands for entire blocks in the flash memory is prohibited by this setting during on-board/off-board programming. Once execution of the batch erase (chip erase) command is prohibited, all of the prohibition settings (including prohibition of batch erase (chip erase)) can no longer be cancelled.

  **Caution After the security setting for the batch erase is set, erasure cannot be performed for the device. In addition, even if a write command is executed, data different from that which has already been written to the flash memory cannot be written, because the erase command is disabled.**

- Disabling block erase
  Execution of the block erase command for a specific block in the flash memory is prohibited during on-board/off-board programming. However, blocks can be erased by means of self programming.

- Disabling write
  Execution of the write and block erase commands for entire blocks in the flash memory is prohibited during on-board/off-board programming. However, blocks can be written by means of self programming.

- Disabling rewriting boot cluster 0
  Execution of the write command, block erase command, and batch erase (chip erase) command for boot cluster 0 (00000H to 00FFFH) in the flash memory is prohibited by this setting.

The batch erase (chip erase), block erase, write commands, and rewriting boot cluster 0 are enabled by the default setting when the flash memory is shipped. Security can be set by on-board/off-board programming and self programming. Each security setting can be used in combination.

All the security settings are cleared by executing the batch erase (chip erase) command.

Table 25-7 shows the relationship between the erase and write commands when the 78K0R/KJ3 security function is enabled.

**Remark** To prohibit writing and erasing during self-programming, use the flash sealed window function (see **25.9.2** for detail).

**Table 28-5. Operation List (15/17)**

| Instruction Group | Mnemonic | Operands | Bytes | Clocks | | Operation | Flag | | |
|---|---|---|---|---|---|---|---|---|---|
| | | | | Note 1 | Note 2 | | Z | AC | CY |
| Call/ return | CALL | rp | 2 | 3 | – | (SP – 2) ← (PC + 2)S, (SP – 3) ← (PC + 2)H, (SP – 4) ← (PC + 2)L, PC ← CS, rp, SP ← SP – 4 | | | |
| | | $!addr20 | 3 | 3 | – | (SP – 2) ← (PC + 3)S, (SP – 3) ← (PC + 3)H, (SP – 4) ← (PC + 3)L, PC ← PC + 3 + jdisp16, SP ← SP – 4 | | | |
| | | !addr16 | 3 | 3 | – | (SP – 2) ← (PC + 3)S, (SP – 3) ← (PC + 3)H, (SP – 4) ← (PC + 3)L, PC ← 0000, addr16, SP ← SP – 4 | | | |
| | | !!addr20 | 4 | 3 | – | (SP – 2) ← (PC + 4)S, (SP – 3) ← (PC + 4)H, (SP – 4) ← (PC + 4)L, PC ← addr20, SP ← SP – 4 | | | |
| | CALLT | [addr5] | 2 | 5 | – | (SP – 2) ← (PC + 2)S, (SP – 3) ← (PC + 2)H, (SP – 4) ← (PC + 2)L , PCS ← 0000, PCH ← (0000, addr5 + 1), PCL ← (0000, addr5), SP ← SP – 4 | | | |
| | BRK | – | 2 | 5 | – | (SP – 1) ← PSW, (SP – 2) ← (PC + 2)S, (SP – 3) ← (PC + 2)H, (SP – 4) ← (PC + 2)L, PCS ← 0000, PCH ← (0007FH), PCL ← (0007EH), SP ← SP – 4, IE ← 0 | | | |
| | RET | – | 1 | 6 | – | PCL ← (SP), PCH ← (SP + 1), PCS ← (SP + 2), SP ← SP + 4 | | | |
| | RETI | – | 2 | 6 | – | PCL ← (SP), PCH ← (SP + 1), PCS ← (SP + 2), PSW ← (SP + 3), SP ← SP + 4 | R | R | R |
| | RETB | – | 2 | 6 | – | PCL ← (SP), PCH ← (SP + 1), PCS ← (SP + 2), PSW ← (SP + 3), SP ← SP + 4 | R | R | R |

**Notes 1.** When the internal RAM area, SFR area, or extended SFR area is accessed, or for an instruction with no data access.

**2.** When the program memory area is accessed.

**Remarks 1.** One instruction clock cycle is one cycle of the CPU clock ($f_{CPU}$) selected by the system clock control register (CKC).

**2.** This number of clocks is for when the program is in the internal ROM (flash memory) area. When fetching an instruction from the internal RAM area, the number of clocks is twice the number of clocks plus 3, maximum (except when branching to the external memory area).

**3.** In products where the external memory area is adjacent to the internal flash area, the number of waits is added to the number of instruction execution clocks placed in the last address (16-byte max.) in the flash memory, in order to use the external bus interface function. This should be done because, during pre-reading of the instruction code, an external memory wait being inserted due to an external memory area exceeding the flash space is accessed. For the number of waits, refer to **5.4 Number of Instruction Wait Clocks for Data Access**.

(2/13)

| Edition | Description | Chapter |
|---|---|---|
| 2nd Edition | Addition of **Notes 3** and change of **Cautions 2** in **Figure 6-6. Format of System Clock Control Register (CKC)** | CHAPTER 6 CLOCK GENERATOR |
| | Addition of **Cautions 5** to **Figure 6-8. Format of Operation Speed Mode Control Register (OSMC)** | |
| | Change of description in **6.3 (8) Internal high-speed oscillator trimming register (HIOTRM)** and addition of **Caution** | |
| | Change of **Figure 6-9. Format of Internal High-Speed Oscillator Trimming Register (HIOTRM)** and addition of **Caution** | |
| | Change of **Figure 6-13. Clock Generator Operation When Power Supply Voltage Is Turned On (When LVI Default Start Function Stopped Is Set (Option Byte: LVIOFF = 1))** | |
| | Change of **Table 7-1. Configuration of Timer Array Unit 0** | CHAPTER 7 TIMER ARRAY UNIT |
| | Change of **Table 7-2. Configuration of Timer Array Unit 1** | |
| | Addition of **Note** to **Figure 7-6. Format of Timer Clock Select Register m (TPSm)** | |
| | Change of description of MASTERmn bit in **Figure 7-7. Format of Timer Mode Register mn (TMRmn) (1/3)** | |
| | Change of **Table 7-3. OVF Bit Operation and Set/Clear Conditions in Each Operation Mode** and addition of **Remark** | |
| | Addition of **Caution** to **Figure 7-17. Format of Timer Input Select Register m (TISm)** | |
| | Addition of description to **7.3 (10) Timer output register m (TOm)** | |
| | Addition of description to **7.3 (12) Timer output mode register m (TOMm)** | |
| | Change of **Remark** in **Figure. 7-22. Format of Input Switch Control Register (ISC)** | |
| | Change of description in **7.3 (14) Noise filter enable register 1, 2 (NFEN1, NFEN2)** | |
| | Change of **7.5.1 TImn edge detection circuit** | |
| | Change of **Figure 8-1. Block Diagram of Real-Time Counter** | CHAPTER 8 REAL-TIME COUNTER |
| | Change of **Caution** in **Figure 8-2. Format of Peripheral Enable Register 0 (PER0)** | |
| | Addition of description to **8.3 (15) Alarm hour register (ALARMWH)** | |
| | Addition of **Note** to **Figure 8-18. Procedure for Starting Operation of Real-Time Counter** | |
| | Change of **Cautions 1** and **Cautions 2** in **9.3 (1) Watchdog timer enable register (WDTE)** | CHAPTER 9 WATCHDOG TIMER |
| | Addition of **Caution 3** to **Table 9-4. Setting Window Open Period of Watchdog Timer** | |
| | Change of **Figure 13-1. Block Diagram of Serial Array Unit 0** | CHAPTER 13 SERIAL ARRAY UNIT |
| | Change of **Figure 13-2. Block Diagram of Serial Array Unit 1** | |
| | Addition of settings and **Note** to **Figure 13-5. Format of Serial Clock Select Register m (SPSm)** | |
| | Addition of **Note** to **Figure 13-7. Format of Serial Communication Operation Setting Register mn (SCRmn) (2/3)** | |
| | Change of **Figure 13-14. Format of Serial Output Enable Register m (SOEm)** | |
| | Addition of description to **13.3 (12) Serial output register m (SOm)** | |
| | Change of **Figure 13-15. Format of Serial Output Register m (SOm)** | |