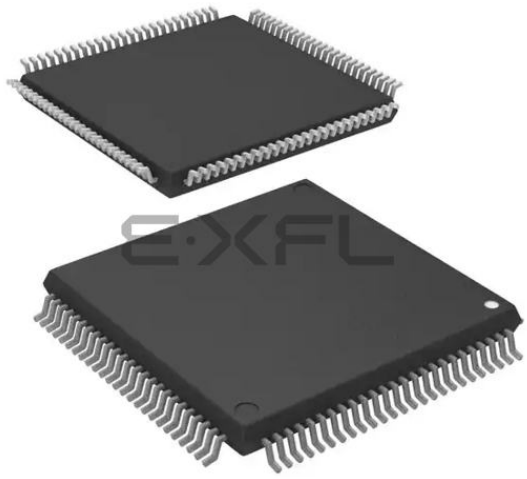
# Renesas Electronics America Inc - UPD78F1849AGC-UEU-AX Datasheet



**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 | CSI, I²C, UART/USART |
| Peripherals | DMA, LVD, POR, PWM, WDT |
| Number of I/O | 80 |
| Program Memory Size | 128KB (128K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 8K x 8 |
| Voltage - Supply (Vcc/Vdd) | 2.7V ~ 5.5V |
| Data Converters | A/D 16x10b |
| Oscillator Type | Internal |
| Operating Temperature | -40°C ~ 85°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 100-LQFP |
| Supplier Device Package | - |
| Purchase URL | https://www.e-xfl.com/product-detail/renesas-electronics-america/upd78f1849agc-ueu-ax |

**Table 3-6. Extended SFR (2nd SFR) List (5/7)**

| Address | Special Function Register (SFR) Name | Symbol | | R/W | Manipulable Bit Range | | | After Reset |
|---|---|---|---|---|---|---|---|---|
| | | | | | 1-bit | 8-bit | 16-bit | |
| F01C0H | Timer counter register 10 | TCR10 | | R | – | – | √ | FFFFH |
| F01C1H | | | | | | | | |
| F01C2H | Timer counter register 11 | TCR11 | | R | – | – | √ | FFFFH |
| F01C3H | | | | | | | | |
| F01C4H | Timer counter register 12 | TCR12 | | R | – | – | √ | FFFFH |
| F01C5H | | | | | | | | |
| F01C8H | Timer mode register 10 | TMR10 | | R/W | – | – | √ | 0000H |
| F01C9H | | | | | | | | |
| F01CAH | Timer mode register 11 | TMR11 | | R/W | – | – | √ | 0000H |
| F01CBH | | | | | | | | |
| F01CCH | Timer mode register 12 | TMR12 | | R/W | – | – | √ | 0000H |
| F01CDH | | | | | | | | |
| F01D0H | Timer status register 10 | TSR10L | TSR10 | R | – | √ | √ | 0000H |
| F01D1H | | – | | | – | – | | |
| F01D2H | Timer status register 11 | TSR11L | TSR11 | R | – | √ | √ | 0000H |
| F01D3H | | – | | | – | – | | |
| F01D4H | Timer status register 12 | TSR12L | TSR12 | R | – | √ | √ | 0000H |
| F01D5H | | – | | | – | – | | |
| F01D8H | Timer channel enable status register 1 | TE1L | TE1 | R | √ | √ | √ | 0000H |
| F01D9H | | – | | | – | – | | |
| F01DAH | Timer channel start register 1 | TS1L | TS1 | R/W | √ | √ | √ | 0000H |
| F01DBH | | – | | | – | – | | |
| F01DCH | Timer channel stop register 1 | TT1L | TT1 | R/W | √ | √ | √ | 0000H |
| F01DDH | | – | | | – | – | | |
| F01DEH | Timer clock select register 1 | TPS1L | TPS1 | R/W | – | √ | √ | 0000H |
| F01DFH | | – | | | – | – | | |
| F01E0H | Timer output register 1 | TO1L | TO1 | R/W | – | √ | √ | 0000H |
| F01E1H | | – | | | – | – | | |
| F01E2H | Timer output enable register 1 | TOE1L | TOE1 | R/W | √ | √ | √ | 0000H |
| F01E3H | | – | | | – | – | | |
| F01E4H | Timer output level register 1 | TOL1L | TOL1 | R/W | – | √ | √ | 0000H |
| F01E5H | | – | | | – | – | | |
| F01E6H | Timer output mode register 1 | TOM1L | TOM1 | R/W | – | √ | √ | 0000H |
| F01B7H | | – | | | – | – | | |
| F0230H | IICA control register 0 | IICCTL0 | | R/W | √ | √ | – | 00H |
| F0231H | IICA control register 1 | IICCTL1 | | R/W | √ | √ | – | 00H |
| F0232H | IICA low-level width setting register | IICWL | | R/W | – | √ | – | FFH |
| F0233H | IICA high-level width setting register | IICWH | | R/W | – | √ | – | FFH |
| F0234H | Slave address register | SVA | | R/W | – | √ | – | 00H |
| F0300H | CEC local address setting register | CADR | | R/W | – | – | √ | 0000H |
| F0302H | CEC control register 1 | CECCTL1 | | R/W | √ | √ | – | 00H |
| F0304H | CEC transmission start bit width setting register | STATB | | R/W | – | – | √ | 0000H |
| F0306H | CEC transmission start bit low width setting register | STATL | | R/W | – | – | √ | 0000H |
| F0308H | CEC transmission logical 0 low width setting register | LGC0L | | R/W | – | – | √ | 0000H |
| F030AH | CEC transmission logical 1 low width setting register | LGC1L | | R/W | – | – | √ | 0000H |

**Figure 4-1. Port Types (78K0R/KF3-C)**

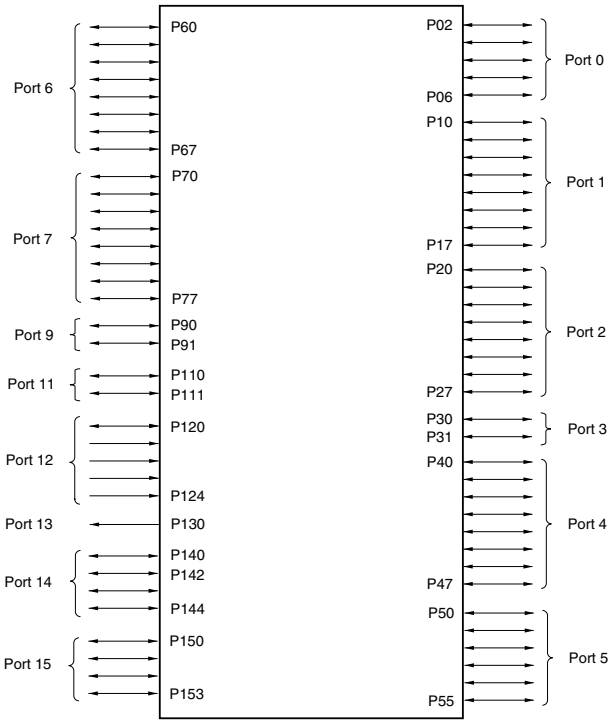



**Figure 4-2. Port Types (78K0R/KG3-C)**

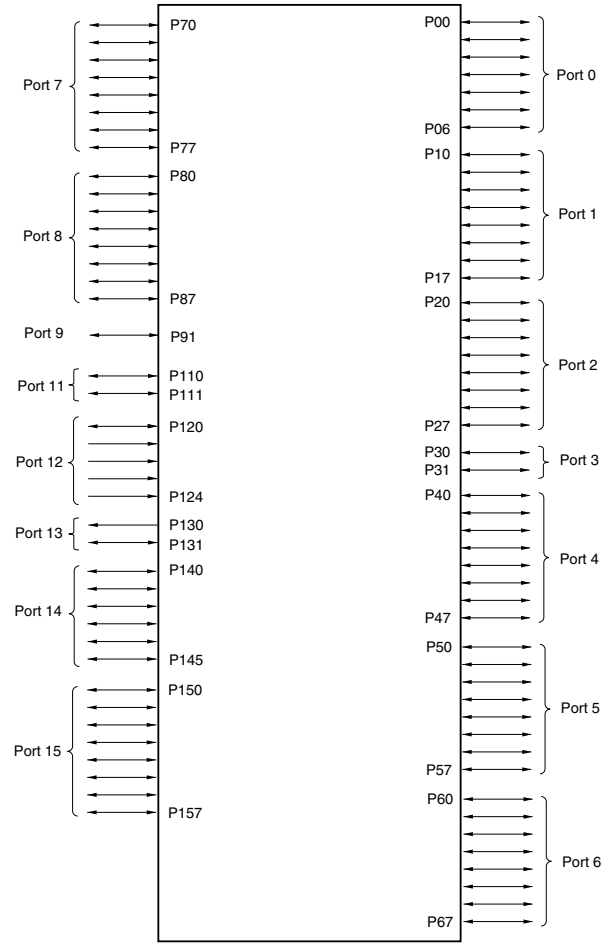
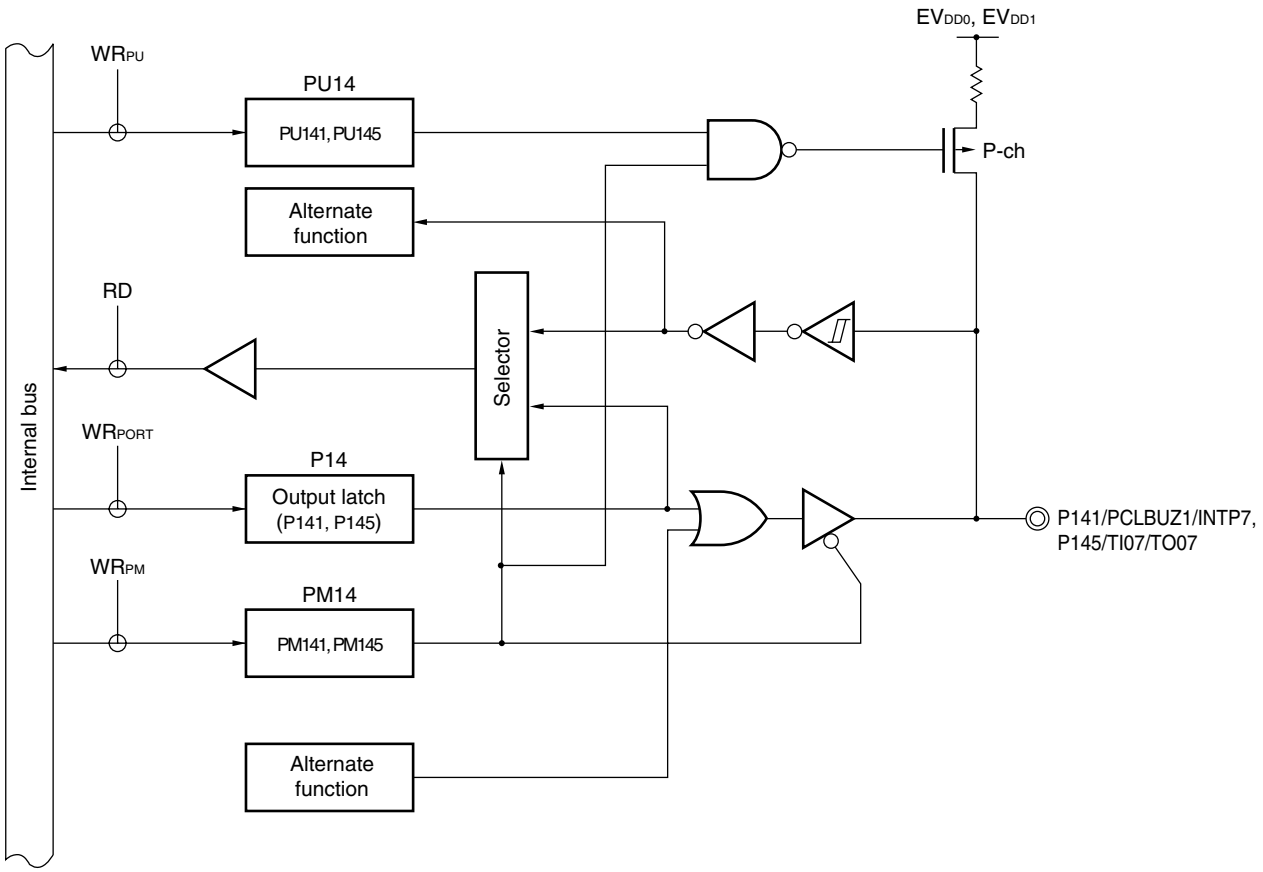


**Figure 4-46. Block Diagram of P141 and P145**

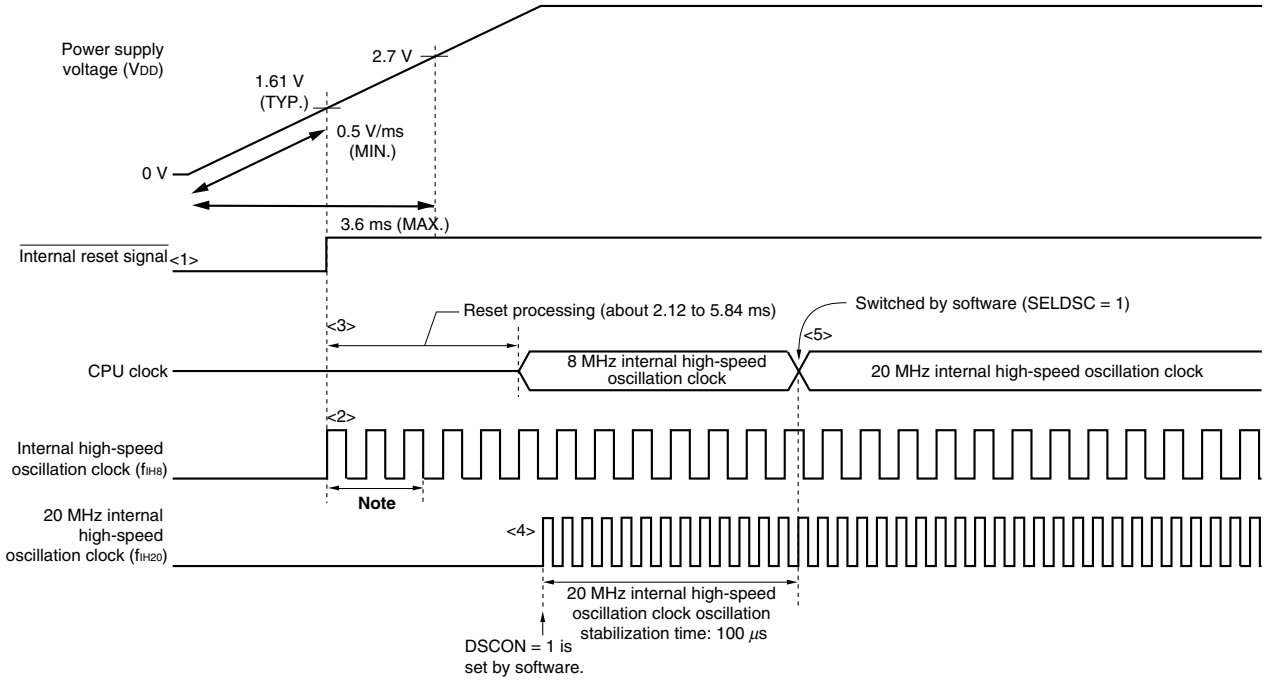



P14: Port register 14
PU14: Pull-up resistor option register 14
PM14: Port mode register 14
RD: Read signal
WR××: Write signal

**Caution 3. Some operations can also be executed while $V_{DD}$ < 2.7 V (For details, figures of CHAPTER 29 ELECTRICAL SPECIFICATIONS AC Characteristics (1) Basic operation.**

**Remark** While the microcontroller is operating, a clock that is not used as the CPU clock can be stopped via software settings. The internal high-speed oscillation clock and high-speed system clock can be stopped by executing the STOP instruction (see **(4)** in **5.6.1 Example of controlling high-speed system clock, (3)** in **5.6.2 Example of controlling internal high-speed oscillation clock**, and **(3)** in **5.6.3 Example of controlling subsystem clock**).

**Figure 5-15. Clock Generator Operation When Power Supply Voltage Is Turned On (When LVI Default Start Function Stopped Is Set (Option Byte: LVIOFF = 1) and Changing to 20 MHz Internal High-Speed Oscillation Clock)**




<1> When the power is turned on, an internal reset signal is generated by the power-on-clear (POC) circuit.

<2> When the power supply voltage exceeds 1.61 V (TYP.), the reset is released and the internal high-speed oscillator automatically starts oscillation.

<3> The CPU starts operation on the internal high-speed oscillation clock after a reset processing such as waiting for the voltage of the power supply or regulator to stabilize has been performed after reset release.

<4> Set DSCON = 1 by software.

<5> Switch the clock by setting SELDSC = 1 by software after waiting for 100 μs.

**Note** The internal reset processing time includes the oscillation accuracy stabilization time of the internal high-speed oscillation clock.

**Cautions 1. To use the 20 MHz internal high-speed oscillation clock, use bits 2 and 1 (FRQSEL2 and FRQSEL1) of the option byte (000C1H) to set the frequency to 20 MHz in advance (for details, see CHAPTER 24 OPTION BYTE).**

### 6.8.3 Operation as multiple PWM output function

By extending the PWM function and using two or more slave channels, many PWM output signals can be produced.

For example, when using two slave channels, the period and duty factor of an output pulse can be calculated by the following expressions.

Pulse period = {Set value of TDRmn (master) + 1} × Count clock period
Duty factor 1 [%] = {Set value of TDRmp (slave 1)}/{Set value of TDRmn (master) + 1} × 100
Duty factor 2 [%] = {Set value of TDRmq (slave 2)}/{Set value of TDRmn (master) + 1} × 100

**Remark** Although the duty factor exceeds 100% if the set value of TDRmp (slave 1) > {set value of TDRmn (master) + 1} or if the {set value of TDRmq (slave 2)} > {set value of TDRmn (master) + 1}, it is summarized into 100% output.

TCRmn of the master channel operates in the interval timer mode and counts the periods.

TCRmp of the slave channel 1 operates in one-count mode, counts the duty factor, and outputs a PWM waveform from the TOmp pin. TCRmp loads the value of TDRmp to TCRmp, using INTTMmn of the master channel as a start trigger, and start counting down. When TCRmp = 0000H, TCRmp outputs INTTMmp and stops counting until the next start trigger (INTTMmn of the master channel) has been input. The output level of TOmp becomes active one count clock after generation of INTTMmn from the master channel, and inactive when TCRmp = 0000H.

In the same way as TCRmp of the slave channel 1, TCRmq of the slave channel 2 operates in one-count mode, counts the duty factor, and outputs a PWM waveform from the TOmq pin. TCRmq loads the value of TDRmq to TCRmq, using INTTMmn of the master channel as a start trigger, and starts counting down. When TCRmq = 0000H, TCRmq outputs INTTMmq and stops counting until the next start trigger (INTTMmn of the master channel) has been input. The output level of TOmq becomes active one count clock after generation of INTTMmn from the master channel, and inactive when TCRmq = 0000H.

When channel 0 is used as the master channel as above, as for the timer array unit 0, up to seven types of PWM signals can be generated, while as for the timer array unit 1, up to two types of PWM signals can be generated.

**Caution To rewrite both TDRmn of the master channel and TDRmp of the slave channel 1, write access is necessary at least twice. Since the values of TDRmn and TDRmp are loaded to TCRmn and TCRmp after INTTMmn is generated from the master channel, if rewriting is performed separately before and after generation of INTTMmn from the master channel, the TOmp pin cannot output the expected waveform. To rewrite both TDRmn of the master and TDRmp of the slave, be sure to rewrite both the registers immediately after INTTMmn is generated from the master channel (This applies also to TDRmq of the slave channel 2) .**

**Remark** m: Unit number, n: Channel number, p: Slave channel number 1 (n+1), q: Slave channel number 2 (n+2)
When m = 0
n = 0, 2, 4
n < p < q ≤ 7 (where p and q are a consecutive integer greater than n)
When m = 1
n = 0
n < p < q ≤ 2 (where p and q are a consecutive integer greater than n ( p = 1, q = 2))

# CHAPTER 7 REAL-TIME COUNTER

## 7.1 Functions of Real-Time Counter

The real-time counter has the following features.

- Having counters of year, month, week, day, hour, minute, and second, and can count up to 99 years.
- Constant-period interrupt function (period: 1 month to 0.5 seconds)
- Alarm interrupt function (alarm: week, hour, minute)
- Interval interrupt function
- Pin output function of 1 Hz
- Pin output function of 512 Hz or 16.384 kHz or 32.768 kHz

## 7.2 Configuration of Real-Time Counter

The real-time counter includes the following hardware.

**Table 7-1. Configuration of Real-Time Counter**

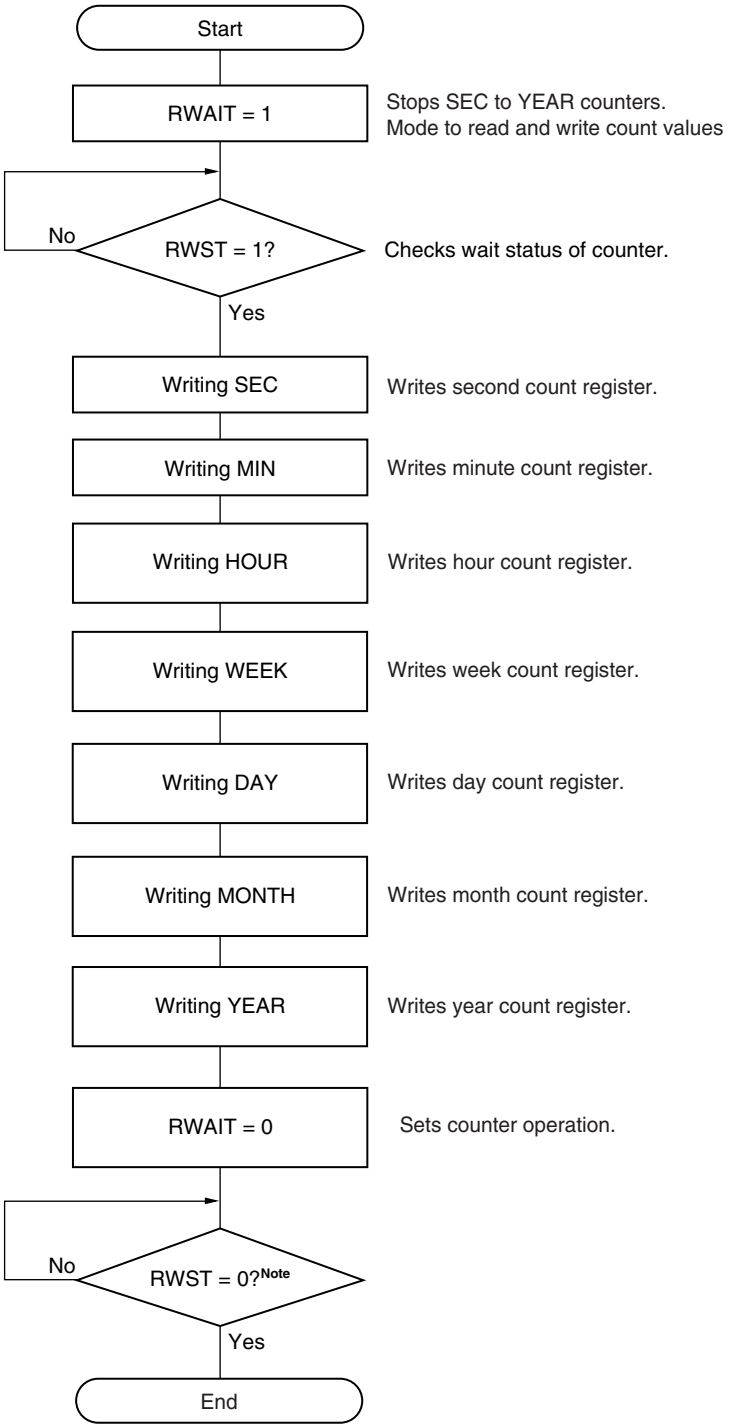
| Item | Configuration |
|---|---|
| Control registers | Peripheral enable register 0 (PER0) |
| | Real-time counter control register 0 (RTCC0) |
| | Real-time counter control register 1 (RTCC1) |
| | Real-time counter control register 2 (RTCC2) |
| | Sub-count register (RSUBC) |
| | Second count register (SEC) |
| | Minute count register (MIN) |
| | Hour count register (HOUR) |
| | Day count register (DAY) |
| | Week count register (WEEK) |
| | Month count register (MONTH) |
| | Year count register (YEAR) |
| | Watch error correction register (SUBCUD) |
| | Alarm minute register (ALARMWM) |
| | Alarm hour register (ALARMWH) |
| | Alarm week register (ALARMWW) |
| | Port mode registers 1 and 3 (PM1, PM3) |
| | Port registers 1 and 3 (P1, P3) |

## 7.3 Registers Controlling Real-Time Counter

The real-time counter is controlled by the following 16 registers.

- Peripheral enable register 0 (PER0)
- Real-time counter control register 0 (RTCC0)
- Real-time counter control register 1 (RTCC1)
- Real-time counter control register 2 (RTCC2)
- Sub-count register (RSUBC)
- Second count register (SEC)
- Minute count register (MIN)
- Hour count register (HOUR)
- Day count register (DAY)
- Week count register (WEEK)
- Month count register (MONTH)
- Year count register (YEAR)
- Watch error correction register (SUBCUD)
- Alarm minute register (ALARMWM)
- Alarm hour register (ALARMWH)
- Alarm week register (ALARMWW)
- Port mode registers 1, 3 (PM1, PM3)
- Port registers 1, 3 (P1, P3)

**Figure 7-22. Procedure for Writing Real-Time Counter**



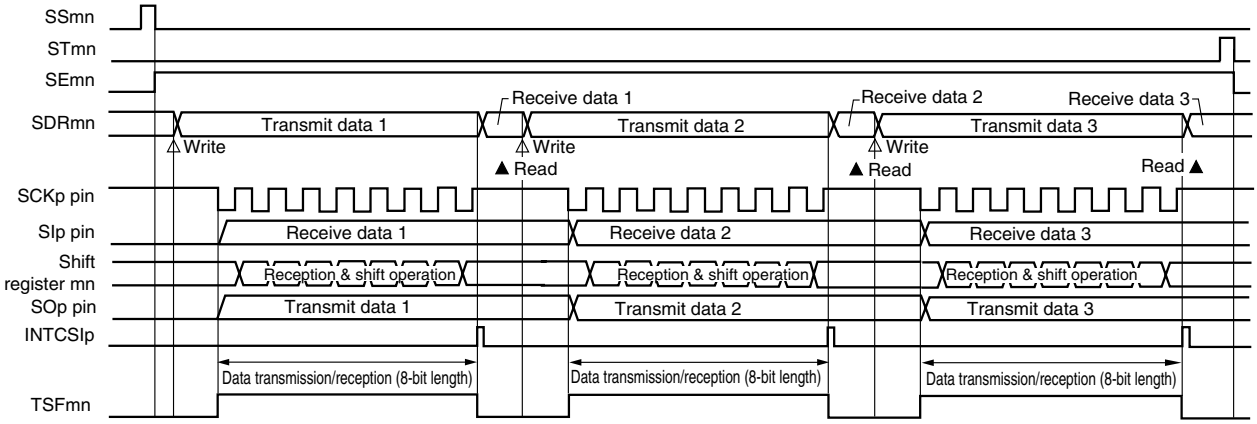

**Note** Be sure to confirm that RWST = 0 before setting STOP mode.

**Caution Complete the series of operations of setting RWAIT to 1 to clearing RWAIT to 0 within 1 second.**

**Remark** SEC, MIN, HOUR, WEEK, DAY, MONTH, and YEAR may be written in any sequence.
All the registers do not have to be set and only some registers may be written.

### (3) Processing flow (in single-transmission/reception mode)

**Figure 11-65. Timing Chart of Slave Transmission/Reception (in Single-Transmission/Reception Mode) (Type 1: DAPmn = 0, CKPmn = 0)**




**Remark** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 2), mn = 00 to 02, 10,
p: CSI number (p = 00, 01, 10, 20)

**Table 11-2. Selection of Operation Clock**

| SMRmn Register | SPSm Register | | | | | | | | Operation Clock ($f_{MCK}$)[Note 1] | |
|---|---|---|---|---|---|---|---|---|---|---|
| CKSmn | PRS m13 | PRS m12 | PRS m11 | PRS m10 | PRS m03 | PRS m02 | PRS m01 | PRS m00 | | $f_{CLK}$ = 20 MHz |
| 0 | X | X | X | X | 0 | 0 | 0 | 0 | $f_{CLK}$ | 20 MHz |
| | X | X | X | X | 0 | 0 | 0 | 1 | $f_{CLK}/2$ | 10 MHz |
| | X | X | X | X | 0 | 0 | 1 | 0 | $f_{CLK}/2^2$ | 5 MHz |
| | X | X | X | X | 0 | 0 | 1 | 1 | $f_{CLK}/2^3$ | 2.5 MHz |
| | X | X | X | X | 0 | 1 | 0 | 0 | $f_{CLK}/2^4$ | 1.25 MHz |
| | X | X | X | X | 0 | 1 | 0 | 1 | $f_{CLK}/2^5$ | 625 kHz |
| | X | X | X | X | 0 | 1 | 1 | 0 | $f_{CLK}/2^6$ | 313 kHz |
| | X | X | X | X | 0 | 1 | 1 | 1 | $f_{CLK}/2^7$ | 156 kHz |
| | X | X | X | X | 1 | 0 | 0 | 0 | $f_{CLK}/2^8$ | 78.1 kHz |
| | X | X | X | X | 1 | 0 | 0 | 1 | $f_{CLK}/2^9$ | 39.1 kHz |
| | X | X | X | X | 1 | 0 | 1 | 0 | $f_{CLK}/2^{10}$ | 19.5 kHz |
| | X | X | X | X | 1 | 0 | 1 | 1 | $f_{CLK}/2^{11}$ | 9.77 kHz |
| | X | X | X | X | 1 | 1 | 1 | 1 | INTTM02 if m = 0[Note 2], Setting prohibited if m = 1 | |
| 1 | 0 | 0 | 0 | 0 | X | X | X | X | $f_{CLK}$ | 20 MHz |
| | 0 | 0 | 0 | 1 | X | X | X | X | $f_{CLK}/2$ | 10 MHz |
| | 0 | 0 | 1 | 0 | X | X | X | X | $f_{CLK}/2^2$ | 5 MHz |
| | 0 | 0 | 1 | 1 | X | X | X | X | $f_{CLK}/2^3$ | 2.5 MHz |
| | 0 | 1 | 0 | 0 | X | X | X | X | $f_{CLK}/2^4$ | 1.25 MHz |
| | 0 | 1 | 0 | 1 | X | X | X | X | $f_{CLK}/2^5$ | 625 kHz |
| | 0 | 1 | 1 | 0 | X | X | X | X | $f_{CLK}/2^6$ | 313 kHz |
| | 0 | 1 | 1 | 1 | X | X | X | X | $f_{CLK}/2^7$ | 156 kHz |
| | 1 | 0 | 0 | 0 | X | X | X | X | $f_{CLK}/2^8$ | 78.1 kHz |
| | 1 | 0 | 0 | 1 | X | X | X | X | $f_{CLK}/2^9$ | 39.1 kHz |
| | 1 | 0 | 1 | 0 | X | X | X | X | $f_{CLK}/2^{10}$ | 19.5 kHz |
| | 1 | 0 | 1 | 1 | X | X | X | X | $f_{CLK}/2^{11}$ | 9.77 kHz |
| | 1 | 1 | 1 | 1 | X | X | X | X | INTTM02 if m = 0[Note 2], Setting prohibited if m = 1 | |
| Other than above | | | | | | | | | Setting prohibited | |

**Notes 1.** 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 for the operation clock, also stop the timer array unit (TAU0) (TT0 = 00FFH).

**2.** SAU0 can be used at the fixed division ratio of the subsystem clock regardless of the $f_{CLK}$ frequency by setting TAU0 and SAU0 as follows.

<TAU0> Select $f_{SUB}/4$ as the input clock of channel 2 of TAU0. (Set TIS02 to 1.)

<SAU0> Select INTTM02 by using the SPS0 register.

However, when changing $f_{CLK}$, SAU0 and TAU0 must be stopped as described in Note 1 above.

**Remarks 1.** X: Don't care

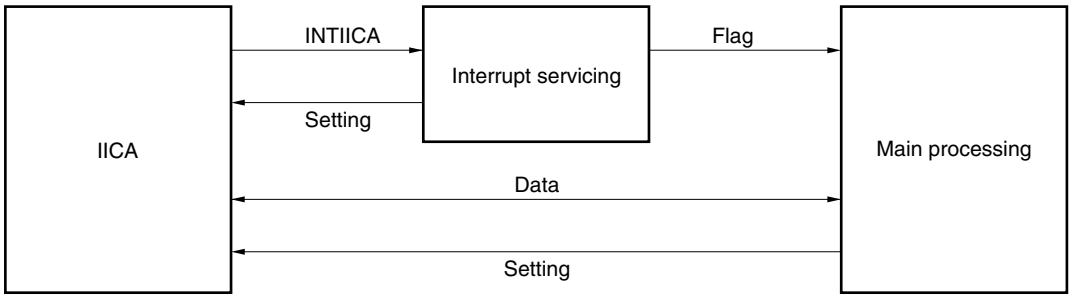
**2.** m: Unit number (m = 0, 1), n: Channel number (n = 0 to 2), mn = 00 to 02, 10

### (3) Slave operation

The processing procedure of the slave operation is as follows.

Basically, the slave operation is event-driven. Therefore, processing by the INTIICA interrupt (processing that must substantially change the operation status such as detection of a stop condition during communication) is necessary.

In the following explanation, it is assumed that the extension code is not supported for data communication. It is also assumed that the INTIICA interrupt servicing only performs status transition processing, and that actual data communication is performed by the main processing.




Therefore, data communication processing is performed by preparing the following three flags and passing them to the main processing instead of INTIICA.

**<1> Communication mode flag**

This flag indicates the following two communication statuses.

- Clear mode: Status in which data communication is not performed
- Communication mode: Status in which data communication is performed (from valid address detection to stop condition detection, no detection of $\overline{\text{ACK}}$ from master, address mismatch)

**<2> Ready flag**

This flag indicates that data communication is enabled. Its function is the same as the INTIICA interrupt for ordinary data communication. This flag is set by interrupt servicing and cleared by the main processing. Clear this flag by interrupt servicing when communication is started. However, the ready flag is not set by interrupt servicing when the first data is transmitted. Therefore, the first data is transmitted without the flag being cleared (an address match is interpreted as a request for the next data).

**<3> Communication direction flag**

This flag indicates the direction of communication. Its value is the same as the TRC bit.

**(4) CEC reception start bit maximum bit width setting register (STATBH)**

STATBH sets the maximum value of the bit width of the start bit during a reception.

STATBH can be set by a 16-bit memory manipulation instruction and rewritten only when CECE = 0.

Reset signal generation clears this register to 0000H.

**Figure 13-26. Format of CEC Reception Start Bit Maximum Bit Width Setting Register (STATBH)**

Address: F0316H After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| STATBH | 0 | 0 | 0 | 0 | 0 | 0 | 0 | STATBH8 to STATBH0 | | | | | | | | |

**Remark** Bit width = (Set values of STATBH8 to STATBH0 + 1) × Clock cycle of $f_{CEC}$

**(5) CEC reception logical 0 minimum low width setting register (LGC0LL)**

LGC0LL is a 9-bit register that sets the minimum value of the low-level width of logical 0 during a reception.

LGC0LL can be set by a 16-bit memory manipulation instruction and rewritten only when CECE = 0.

Reset signal generation clears this register to 0000H.

**Figure 13-27. Format of CEC Reception Logical 0 Minimum Low Width Setting Register (LGC0LL)**

Address: F0318H After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LGC0LL | 0 | 0 | 0 | 0 | 0 | 0 | 0 | LGC0LL8 to LGC0LL0 | | | | | | | | |

**Remark** Low-level width = (Set values of LGC0LL8 to LGC0LL0 + 1) × Clock cycle of $f_{CEC}$

**(6) CEC reception logical 0 maximum low width setting register (LGC0LH)**

LGC0LH is a 9-bit register that sets the maximum value of the low-level width of logical 0 during a reception.

LGC0LH can be set by a 16-bit memory manipulation instruction and rewritten only when CECE = 0.

Reset signal generation clears this register to 0000H.

**Figure 13-28. Format of CEC Reception Logical 0 Maximum Low Width Setting Register (LGC0LH)**

Address: F031AH After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| LGC0LH | 0 | 0 | 0 | 0 | 0 | 0 | 0 | LGC0LH8 to LGC0LH0 | | | | | | | | |

**Remark** Low-level width = (Set values of LGC0LH8 to LGC0LH0 + 1) × Clock cycle of $f_{CEC}$

**(10) CEC reception data bit maximum bit width setting register (DATBH)**

DATBH is a 9-bit register that sets the maximum value of the bit width of the data bit during a reception.

DATBH can be set by a 16-bit memory manipulation instruction and rewritten only when CECE = 0.

Reset signal generation clears this register to 0000H.

**Figure 13-32. Format of CEC Reception Data Bit Maximum Bit Width Setting Register (DATBH)**

Address: F0322H After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| DATBH | 0 | 0 | 0 | 0 | 0 | 0 | 0 | DATBH8 to DATBH0 | | | | | | | | |

**Remark** Bit width = (Set values of DATBH8 to DATBH0 + 1) × Clock cycle of $f_{CEC}$

### 13.5.3 Determining CEC reception data logic (1 or 0)

The timing at which receive data is judged to be 1 or 0 is set by the following register.

**(1) CEC reception data sampling time setting register (NOMT)**

NOMT is a 9-bit register that determines the sampling time of data during a reception.

NOMT can be set by a 16-bit memory manipulation instruction and rewritten only when CECE = 0.

Reset signal generation clears this register to 0000H.

**Figure 13-33. Format of CEC Reception Data Sampling Time Setting Register (NOMT)**

Address: F030EH After reset: 0000H R/W

| Symbol | 15 | 14 | 13 | 12 | 11 | 10 | 9 | 8 | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| NOMT | 0 | 0 | 0 | 0 | 0 | 0 | 0 | NOMT8 to NOMT0 | | | | | | | | |

**Remark** Sampling time = (Set values of NOMT8 to NOMT0 + 1) × Clock cycle of $f_{CEC}$

Set this register within a period of LGC1LH < NOMT < LGC0LL.

**(9) Remote controller receive DH1S compare register m (RMDH1Sm) (Type A, B, or C reception mode only)**

This register is used to detect the high level of remote controller data 1 (short side).

RMDH1Sm is set with an 8-bit memory manipulation instruction.

Reset signal generation sets RMDH1Sm to 00H.

**Figure 14-22. Format of Remote Controller Receive DH1S Compare Register m (RMDH1Sm)**

Address: F033AH, F034CH After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RMDH1Sm | | | | | | | | |

**Cautions 1. Write this register while RMENm = 0.**

**2. This register is shared with the lower eight bits of the RMDB1Sm register for cycle detection (type B1/type C1). If the RMBDm bit of the RMCN2m register is 0, this register is valid as the RMDH1Sm register. If the RMBDm bit is 1, this register is used as the lower eight bits of RMDB1Sm.**

**Remark** m = 02, 13

**(10) Remote controller receive DH1L compare register m (RMDH1Lm) (Type A, B, or C reception mode only)**

This register is used to detect the high level of remote controller data 1 (long side).

RMDH1Lm is set with an 8-bit memory manipulation instruction.

Reset signal generation sets RMDH1Lm to 00H.

**Figure 14-23. Format of Remote Controller Receive DH1L Compare Register m (RMDH1Lm)**

Address: F033BH, F034DH After reset: 00H R/W

| Symbol | 7 | 6 | 5 | 4 | 3 | 2 | 1 | 0 |
|---|---|---|---|---|---|---|---|---|
| RMDH1Lm | | | | | | | | |

**Cautions 1. Write this register while RMENm = 0.**

**2. This register is shared with the higher eight bits of the RMDB1Sm register for cycle detection (type B1/type C1). If the RMBDm bit of the RMCN2m register is 0, this register is valid as the RMDH1Lm register. If the RMBDm bit is 1, this register is used as the higher eight bits of RMDB1Sm.**

**Remark** m = 02, 13

**(3) Multiplication/division data register C (MDCL, MDCH)**

The MDCH and MDCL registers store remainder value of the operation result in the division mode. They are not used in the multiplication mode.

MDCH and MDCL can be read by a 16-bit manipulation instruction.

Reset signal generation clears these registers to 0000H.

**Figure 15-4. Format of Multiplication/Division Data Register C (MDCH, MDCL)**

Address: F00E0H, F00E1H, F00E2H, F00E3H After reset: 0000H, 0000H R

Symbol F00E3H F00E2H

| MDCH | MDCH 15 | MDCH 14 | MDCH 13 | MDCH 12 | MDCH 11 | MDCH 10 | MDCH 9 | MDCH 8 | MDCH 7 | MDCH 6 | MDCH 5 | MDCH 4 | MDCH 3 | MDCH 2 | MDCH 1 | MDCH 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

Symbol F00E1H F00E0H

| MDCL | MDCL 15 | MDCL 14 | MDCL 13 | MDCL 12 | MDCL 11 | MDCL 10 | MDCL 9 | MDCL 8 | MDCL 7 | MDCL 6 | MDCL 5 | MDCL 4 | MDCL 3 | MDCL 2 | MDCL 1 | MDCL 0 |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|

**Caution The MDCH and MDCL values read during division operation processing (while the multiplication/division control register (MDUC) is 81H) will not be guaranteed.**

**Table 15-4. Functions of MDCH and MDCL During Operation Execution**

| DIVMODE | Operation Mode | Setting | Operation Result |
|---|---|---|---|
| 0 | Multiplication mode | – | – |
| 1 | Division mode | – | MDCH: Remainder (higher 16 bits)<br>MDCL: Remainder (lower 16 bits) |

**Remark** DIVMODE: Bit 7 of the multiplication/division control register (MDUC)

The register configuration differs between when multiplication is executed and when division is executed, as follows.

- Register configuration during multiplication

<Multiplier A> <Multiplier B> <Product>

MDAL (bits 15 to 0) × MDAH (bits 15 to 0) = [MDBH (bits 15 to 0), MDBL (bits 15 to 0)]

- Register configuration during division

<Dividend> <Divisor>

[MDAH (bits 15 to 0), MDAL (bits 15 to 0)] ÷ [MDBH (bits 15 to 0), MDBL (bits 15 to 0)] =

<Quotient> <Remainder>

[MDAH (bits 15 to 0), MDAL (bits 15 to 0)] ··· [MDCH (bits 15 to 0), MDCL (bits 15 to 0)]

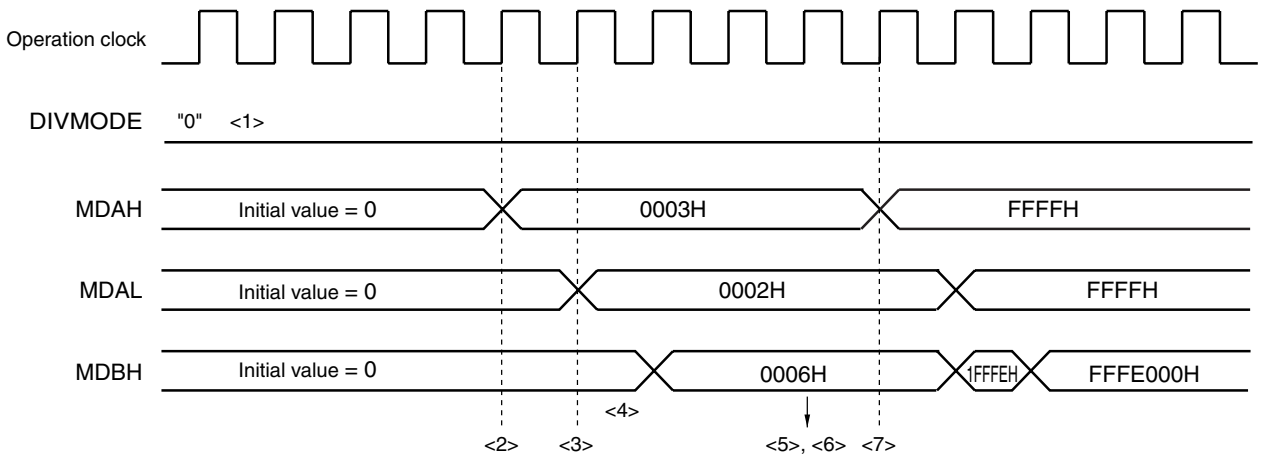
## 15.4 Operations of Multiplier/Divider

### 15.4.1 Multiplication operation

- Initial setting
  - <1> Set bit 7 (DIVMODE) of the multiplication/division control register (MDUC) to 0.
  - <2> Set the multiplicand to the multiplication/division data register A (L) (MDAL).
  - <3> Set the multiplier to the multiplication/division data register A (H) (MDAH).
    (There is no preference in the order of executing steps <2> and <3>. Multiplication operation is automatically started when the multiplier and multiplicand are set to MDAH and MDAL, respectively.)
- During operation processing
  - <4> Wait for at least one clock. The operation will end when one clock has been issued.
- Operation end
  - <5> Read the product (lower 16 bits) from the multiplication/division data register B (L) (MDBL).
  - <6> Read the product (higher 16 bits) from the multiplication/division data register B (H) (MDBH).
    (There is no preference in the order of executing steps <5> and <6>.)
- Next operation
  - <7> To execute multiplication operation next, start from the “Initial setting” for multiplication operation.
  - <8> To execute division operation next, start from the “Initial setting” in **15.4.2 Division operation**.

**Remark** Steps <1> to <7> correspond to <1> to <7> in Figure 15-6.

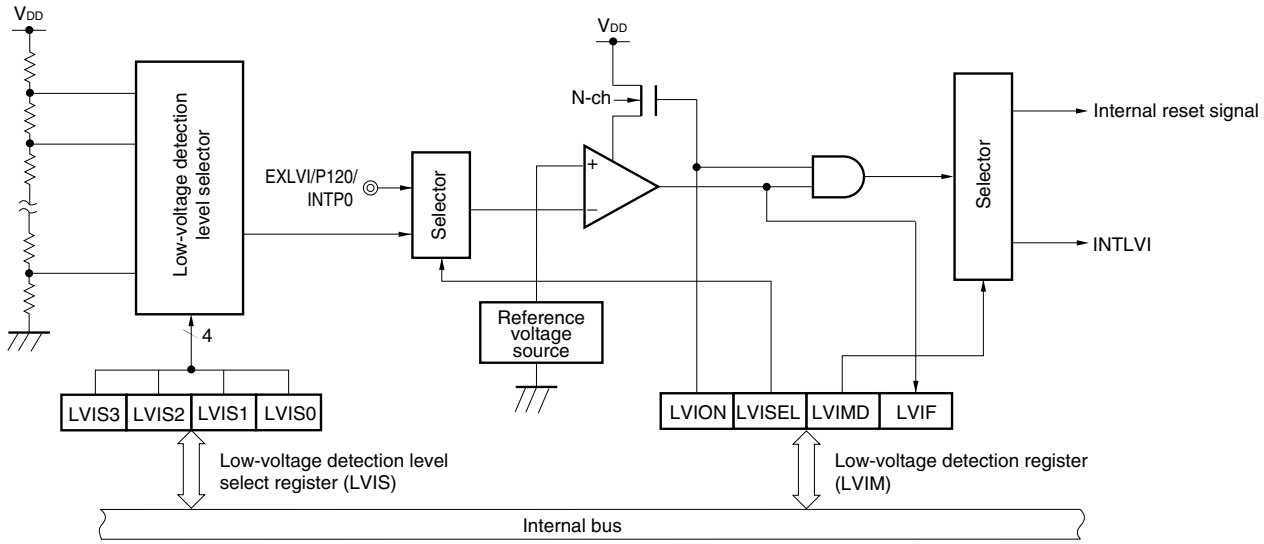
**Figure 15-6. Timing Diagram of Multiplication Operation (0003H × 0002H)**



## 22.2 Configuration of Low-Voltage Detector

The block diagram of the low-voltage detector is shown in Figure 22-1.

**Figure 22-1. Block Diagram of Low-Voltage Detector**




## 22.3 Registers Controlling Low-Voltage Detector

The low-voltage detector is controlled by the following registers.

- Low-voltage detection register (LVIM)
- Low-voltage detection level select register (LVIS)
- Port mode register 12 (PM12)

### (1) Low-voltage detection register (LVIM)

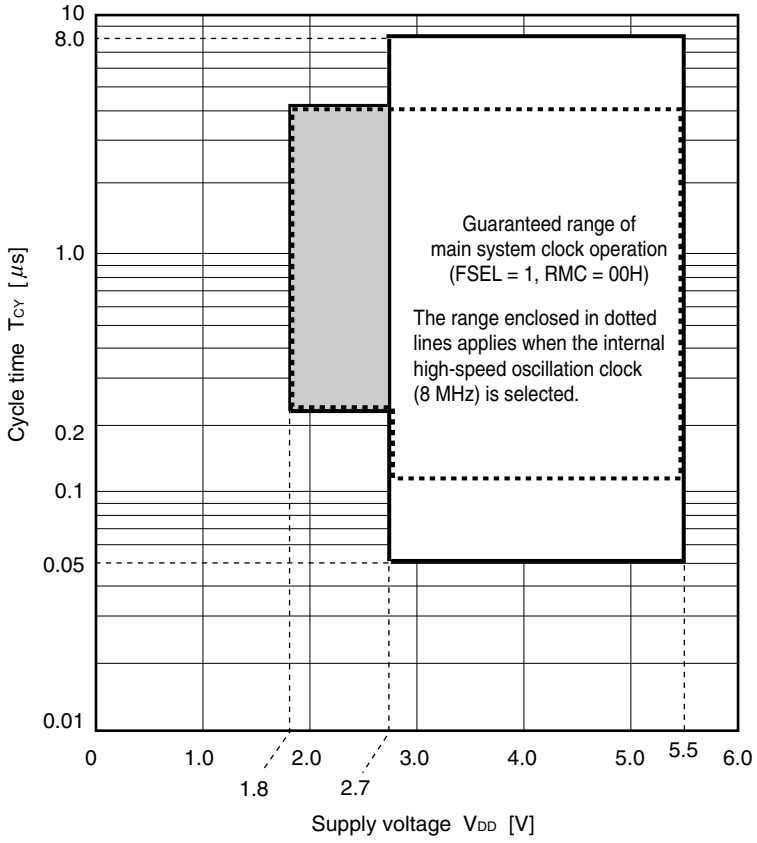
This register sets low-voltage detection and the operation mode.

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

Reset signal generation clears this register to 00H.

### (1) Basic operation (3/6)

**Minimum instruction execution time during main system clock operation (FSEL = 1, RMC = 00H)**




**Cautions 1. To set the FSEL bit to 0, set $f_{CLK}$ to 10 MHz or less in advance.**

**2. Only the following operations are possible in the shaded section above:**

- **Using the CPU (executing instructions)**
  **The CPU clock is a fraction of the 8 MHz internal high-speed oscillation clock ($f_{IH}/2^5$ to $f_{IH}/2$ (250 kHz to 4 MHz)).**
- **Reading or writing the internal RAM**
- **Using the low-voltage detector (LVI)**
- **Using the interval timer function of the timer array unit (TAU)**
- **Specifying the mode of the standby function (STOP or HALT mode)**
- **Setting up the control registers of the clock generator (except those of the clock and buzzer output controllers)**
  **However, the clock can be switched only if the clock cycle time after the clock has been switched will be within the guaranteed operating range.**
- **Using the watchdog timer (WDT) (including the internal low-speed oscillator)**

**Remark** FSEL: Bit 0 of the operation speed mode control register (OSMC)
RMC: Regulator mode control register

78K0R/Kx3-C User's Manual: Hardware

Publication Date: Rev.0.01 Mar 1, 2010
Rev.2.00 Sep 30, 2011

Published by: Renesas Electronics Corporation