**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 | Obsolete |
| Core Processor | - |
| Core Size | - |
| Speed | - |
| Connectivity | - |
| Peripherals | - |
| Number of I/O | - |
| Program Memory Size | - |
| Program Memory Type | - |
| EEPROM Size | - |
| RAM Size | - |
| Voltage - Supply (Vcc/Vdd) | - |
| Data Converters | - |
| Oscillator Type | - |
| Operating Temperature | - |
| Mounting Type | - |
| Package / Case | - |
| Supplier Device Package | - |
| Purchase URL | https://www.e-xfl.com/product-detail/renesas-electronics-america/hd64f3687gfzkv |

- On-chip memory

| Product Classification | | | Model: Standard Version | Model: On-Chip Power-On Reset and Low-Voltage Detecting Circuit Version | ROM | RAM | Remarks |
|---|---|---|---|---|---|---|---|
| Flash memory version (F-ZTAT™ version) | | H8/3687F | HD64F3687 | HD64F3687G | 56 kbytes | 4 kbytes | |
| | | H8/3684F | HD64F3684 | HD64F3684G | 32 kbytes | 4 kbytes | |
| Mask-ROM version | | H8/3687 | HD6433687 | HD6433687G | 56 kbytes | 3 kbytes | |
| | | H8/3686 | HD6433686 | HD6433686G | 48 kbytes | 3 kbytes | |
| | | H8/3685 | HD6433685 | HD6433685G | 40 kbytes | 3 kbytes | |
| | | H8/3684 | HD6433684 | HD6433684G | 32 kbytes | 3 kbytes | |
| | | H8/3683 | HD6433683 | HD6433683G | 24 kbytes | 3 kbytes | |
| | | H8/3682 | HD6433682 | HD6433682G | 16 kbytes | 3 kbytes | |
| EEPROM stacked version (512 bytes) | Flash memory version | H8/3687N | — | HD64N3687G | 56 kbytes | 4 kbytes | |
| | Mask-ROM version | | — | HD6483687G | 56 kbytes | 3 kbytes | |

- General I/O ports
  - — I/O pins: 45 I/O pins (43 I/O pins for H8/3687N), including 8 large current ports ($I_{OL}$ = 20 mA, @$V_{OL}$ = 1.5 V)
  - — Input-only pins: 8 input pins (also used for analog input)
- EEPROM interface (only for H8/3687N)
  - — $I^2C$ bus interface (conforms to the $I^2C$ bus interface format that is advocated by Philips Electronics)
- Supports various power-down states

Note: F-ZTAT™ is a trademark of Renesas Technology Corp.

- Compact package

| Package | Code | Body Size | Pin Pitch |
|---|---|---|---|
| LQFP-64 | FP-64E | 10.0 × 10.0 mm | 0.5 mm |
| QFP-64 | FP-64A | 14.0 × 14.0 mm | 0.8 mm |

Only LQFP-64 (FP-64E) for H8/3687N package

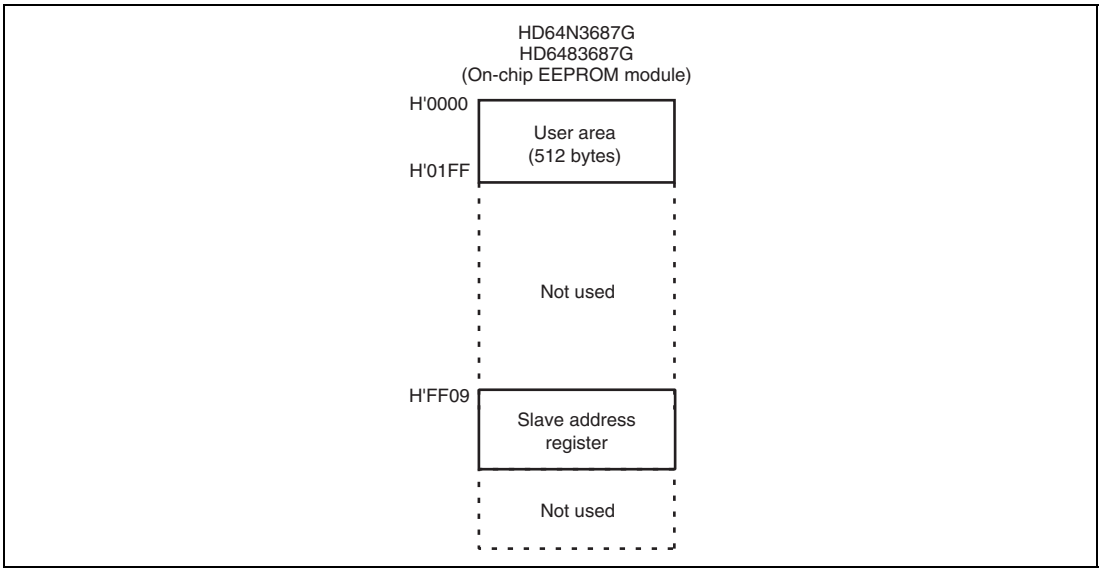



**Figure 2.1 Memory Map (3)**

## 2.5 Addressing Modes and Effective Address Calculation

The following describes the H8/300H CPU. In this LSI, the upper eight bits are ignored in the generated 24-bit address, so the effective address is 16 bits.

### 2.5.1 Addressing Modes

The H8/300H CPU supports the eight addressing modes listed in table 2.10. Each instruction uses a subset of these addressing modes. Addressing modes that can be used differ depending on the instruction. For details, refer to appendix A.4, Combinations of Instructions and Addressing Modes.

Arithmetic and logic instructions can use the register direct and immediate modes. Data transfer instructions can use all addressing modes except program-counter relative and memory indirect. Bit-manipulation instructions use register direct, register indirect, or the absolute addressing mode (@aa:8) to specify an operand, and register direct (BSET, BCLR, BNOT, and BTST instructions) or immediate (3-bit) addressing mode to specify a bit number in the operand.

**Table 2.10 Addressing Modes**

| No. | Addressing Mode | Symbol |
|---|---|---|
| 1 | Register direct | Rn |
| 2 | Register indirect | @ERn |
| 3 | Register indirect with displacement | @(d:16,ERn)/@(d:24,ERn) |
| 4 | Register indirect with post-increment<br>Register indirect with pre-decrement | @ERn+<br>@–ERn |
| 5 | Absolute address | @aa:8/@aa:16/@aa:24 |
| 6 | Immediate | #xx:8/#xx:16/#xx:32 |
| 7 | Program-counter relative | @(d:8,PC)/@(d:16,PC) |
| 8 | Memory indirect | @@aa:8 |

**Register Direct—Rn**

The register field of the instruction specifies an 8-, 16-, or 32-bit general register containing the operand. R0H to R7H and R0L to R7L can be specified as 8-bit registers. R0 to R7 and E0 to E7 can be specified as 16-bit registers. ER0 to ER7 can be specified as 32-bit registers.

### 3.2.4 Interrupt Enable Register 2 (IENR2)

IENR2 enables, timer B1 overflow interrupts.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7, 6 | — | All 0 | — | Reserved<br>These bits are always read as 0. |
| 5 | IENTB1 | 0 | R/W | Timer B1 Interrupt Enable<br>When this bit is set to 1, timer B1 overflow interrupt requests are enabled. |
| 4 to 0 | — | All 1 | — | Reserved<br>These bits are always read as 1. |

When disabling interrupts by clearing bits in an interrupt enable register, or when clearing bits in an interrupt flag register, always do so while interrupts are masked (I = 1). If the above clear operations are performed while I = 0, and as a result a conflict arises between the clear instruction and an interrupt request, exception handling for the interrupt will be executed after the clear instruction has been executed.

### 3.2.5 Interrupt Flag Register 1 (IRR1)

IRR1 is a status flag register for direct transition interrupts, RTC interrupts, and $\overline{IRQ3}$ to $\overline{IRQ0}$ interrupt requests.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | IRRDT | 0 | R/W | Direct Transfer Interrupt Request Flag<br>[Setting condition]<br>When a direct transfer is made by executing a SLEEP instruction while DTON in SYSCR2 is set to 1.<br>[Clearing condition]<br>When IRRDT is cleared by writing 0 |
| 6 | IRRTA | 0 | R/W | RTC Interrupt Request Flag<br>[Setting condition]<br>When the RTC counter value overflows<br>[Clearing condition]<br>When IRRTA is cleared by writing 0 |

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 5, 4 | — | All 1 | — | Reserved<br>These bits are always read as 1. |
| 3 | IRRI3 | 0 | R/W | IRQ3 Interrupt Request Flag<br>[Setting condition]<br>When $\overline{\text{IRQ3}}$ pin is designated for interrupt input and the designated signal edge is detected.<br>[Clearing condition]<br>When IRRI3 is cleared by writing 0 |
| 2 | IRRI2 | 0 | R/W | IRQ2 Interrupt Request Flag<br>[Setting condition]<br>When $\overline{\text{IRQ2}}$ pin is designated for interrupt input and the designated signal edge is detected.<br>[Clearing condition]<br>When IRRI2 is cleared by writing 0 |
| 1 | IRRI1 | 0 | R/W | IRQ1 Interrupt Request Flag<br>[Setting condition]<br>When $\overline{\text{IRQ1}}$ pin is designated for interrupt input and the designated signal edge is detected.<br>[Clearing condition]<br>When IRRI1 is cleared by writing 0 |
| 0 | IRRI0 | 0 | R/W | IRQ0 Interrupt Request Flag<br>[Setting condition]<br>When $\overline{\text{IRQ0}}$ pin is designated for interrupt input and the designated signal edge is detected.<br>[Clearing condition]<br>When IRRI0 is cleared by writing 0 |

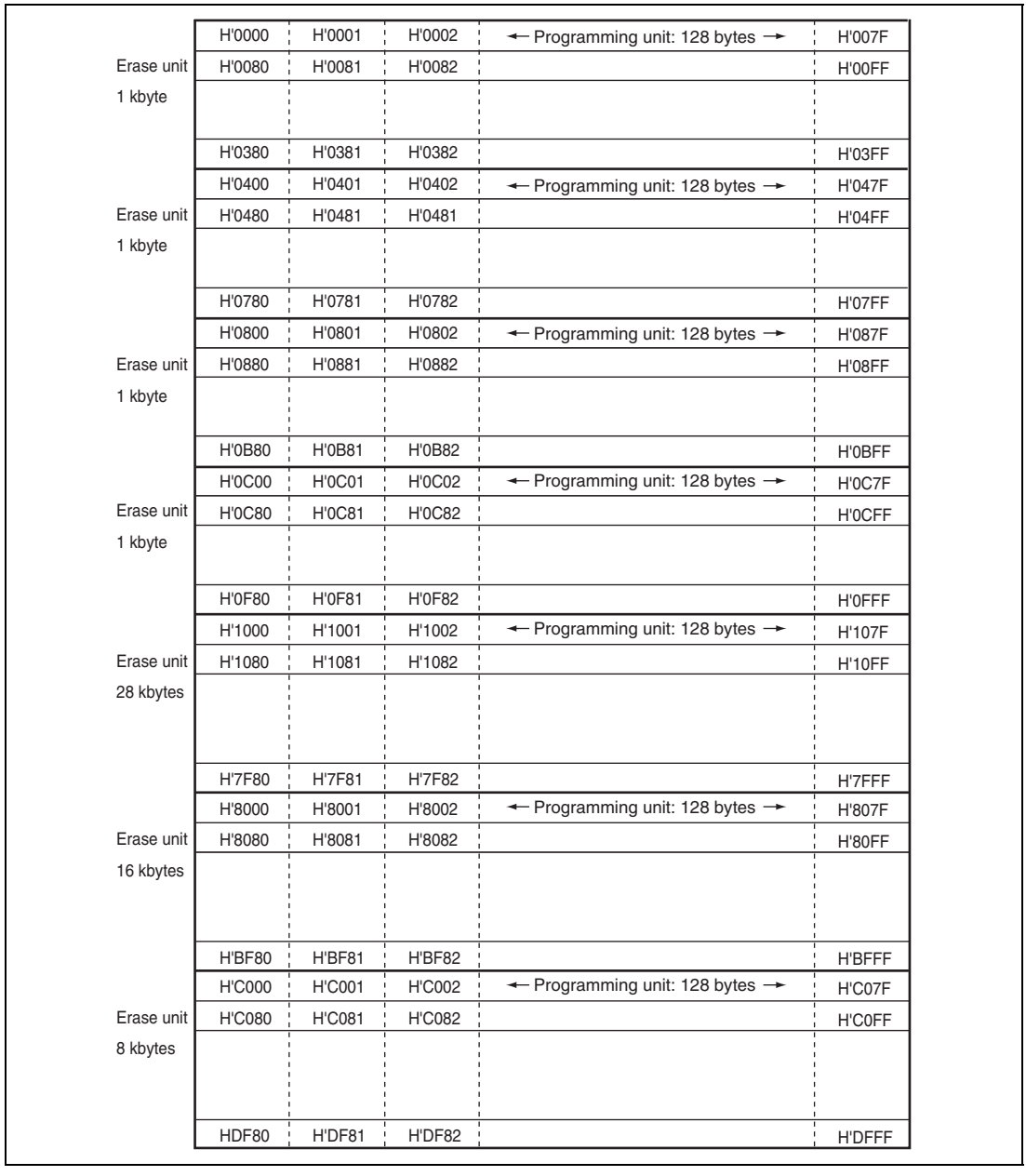



**Figure 7.1 Flash Memory Block Configuration**

### 7.2.4 Flash Memory Power Control Register (FLPWCR)

FLPWCR enables or disables a transition to the flash memory power-down mode when the LSI switches to subactive mode. There are two modes: mode in which operation of the power supply circuit of flash memory is partly halted in power-down mode and flash memory can be read, and mode in which even if a transition is made to subactive mode, operation of the power supply circuit of flash memory is retained and flash memory can be read.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | PDWND | 0 | R/W | Power-Down Disable<br>When this bit is 0 and a transition is made to subactive mode, the flash memory enters the power-down mode. When this bit is 1, the flash memory remains in the normal mode even after a transition is made to subactive mode. |
| 6 to 0 | — | All 0 | — | Reserved<br>These bits are always read as 0. |

### 7.2.5 Flash Memory Enable Register (FENR)

Bit 7 (FLSHE) in FENR enables or disables the CPU access to the flash memory control registers, FLMCR1, FLMCR2, EBR1, and FLPWCR.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | FLSHE | 0 | R/W | Flash Memory Control Register Enable<br>Flash memory control registers can be accessed when this bit is set to 1. Flash memory control registers cannot be accessed when this bit is set to 0. |
| 6 to 0 | — | All 0 | — | Reserved<br>These bits are always read as 0. |

## 10.5 Interrupt Source

There are five kinds of RTC interrupts: week interrupts, day interrupts, hour interrupts, minute interrupts, and second interrupts.

When using an interrupt, initiate the RTC last after other registers are set. Do not set multiple interrupt enable bits in RTCCR2 simultaneously to 1.

When an interrupt request of the RTC occurs, the IRRTA flag in IRR1 is set to 1. When clearing the flag, write 0.

**Table 10.2 Interrupt Source**

| Interrupt Name | Interrupt Source | Interrupt Enable Bit |
|---|---|---|
| Overflow interrupt | Occurs when the free running counter is overflown. | FOIE |
| Week periodic interrupt | Occurs every week when the day-of-week date register value becomes 0. | WKIE |
| Day periodic interrupt | Occurs every day when the day-of-week date register is counted. | DYIE |
| Hour periodic interrupt | Occurs every hour when the hour date register is counted. | HRIE |
| Minute periodic interrupt | Occurs every minute when the minute date register is counted. | MNIE |
| Second periodic interrupt | Occurs every second when the second date register is counted. | SCIE |

## 11.2 Input/Output Pin

Table 11.1 shows the timer B1 pin configuration.

**Table 11.1 Pin Configuration**

| Name | Abbreviation | I/O | Function |
|---|---|---|---|
| Timer B1 event input | TMIB1 | Input | Event input to TCB1 |

## 11.3 Register Descriptions

The timer B1 has the following registers.

- Timer mode register B1 (TMB1)
- Timer counter B1 (TCB1)
- Timer load register B1 (TLB1)

### 13.3.8 General Registers A, B, C, and D (GRA, GRB, GRC, and GRD)

GR are 16-bit registers. Timer Z has eight general registers (GR), four for each channel. The GR registers are dual function 16-bit readable/writable registers, functioning as either output compare or input capture registers. Functions can be switched by TIORA and TIORC.

The values in GR and TCNT are constantly compared with each other when the GR registers are used as output compare registers. When the both values match, the IMFA to IMFD flags in TSR are set to 1. Compare match outputs can be selected by TIORA and TIORC.

When the GR registers are used as input capture registers, the TCNT value is stored after detecting external signals. At this point, IMFA to IMFD flags in the corresponding TSR are set to 1. Detection edges for input capture signals can be selected by TIORA and TIORC.

When PWM mode, complementary PWM mode, or reset synchronous PWM mode is selected, the values in TIORA and TIORC are ignored. Upon reset, the GR registers are set as output compare registers (no output) and initialized to H'FFFF. The GR registers cannot be accessed in 8-bit units; they must always be accessed as a 16-bit unit.

2. 8-bit register

Registers other than TCNT and GR are 8-bit registers that are connected internally with the CPU in an 8-bit width. Figure 13.6 shows an example of accessing the 8-bit registers.

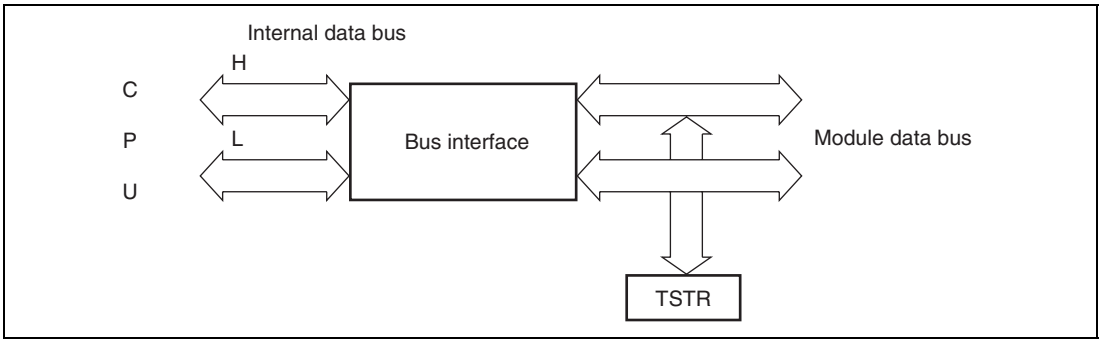



**Figure 13.6 Accessing Operation of 8-Bit Register (between CPU and TSTR (8 bits))**

### 13.4.6 Reset Synchronous PWM Mode

Three normal- and counter-phase PWM waveforms are output by combining channels 0 and 1 that one of changing points of waveforms will be common.

In reset synchronous PWM mode, the FTIOB0 to FTIOD0 and FTIOA1 to FTIOD1 pins become PWM-output pins automatically. TCNT_0 performs an increment operation. Tables 13.4 and 13.5 show the PWM-output pins used and the register settings, respectively.

Figure 13.26 shows the example of reset synchronous PWM mode setting procedure.

**Table 13.4 Output Pins in Reset Synchronous PWM Mode**

| Channel | Pin Name | Input/Output | Pin Function |
|---|---|---|---|
| 0 | FTIOC0 | Output | Toggle output in synchronous with PWM cycle |
| 0 | FTIOB0 | Output | PWM output 1 |
| 0 | FTIOD0 | Output | PWM output 1 (counter-phase waveform of PWM output 1) |
| 1 | FTIOA1 | Output | PWM output 2 |
| 1 | FTIOC1 | Output | PWM output 2 (counter-phase waveform of PWM output 2) |
| 1 | FTIOB1 | Output | PWM output 3 |
| 1 | FTIOD1 | Output | PWM output 3 (counter-phase waveform of PWM output 3) |

**Table 13.5 Register Settings in Reset Synchronous PWM Mode**

| Register | Description |
|---|---|
| TCNT_0 | Initial setting of H'0000 |
| TCNT_1 | Not used (independently operates) |
| GRA_0 | Sets counter cycle of TCNT_0 |
| GRB_0 | Set a changing point of the PWM waveform output from pins FTIOB0 and FTIOD0. |
| GRA_1 | Set a changing point of the PWM waveform output from pins FTIOA1 and FTIOC1. |
| GRB_1 | Set a changing point of the PWM waveform output from pins FTIOB1 and FTIOD1. |

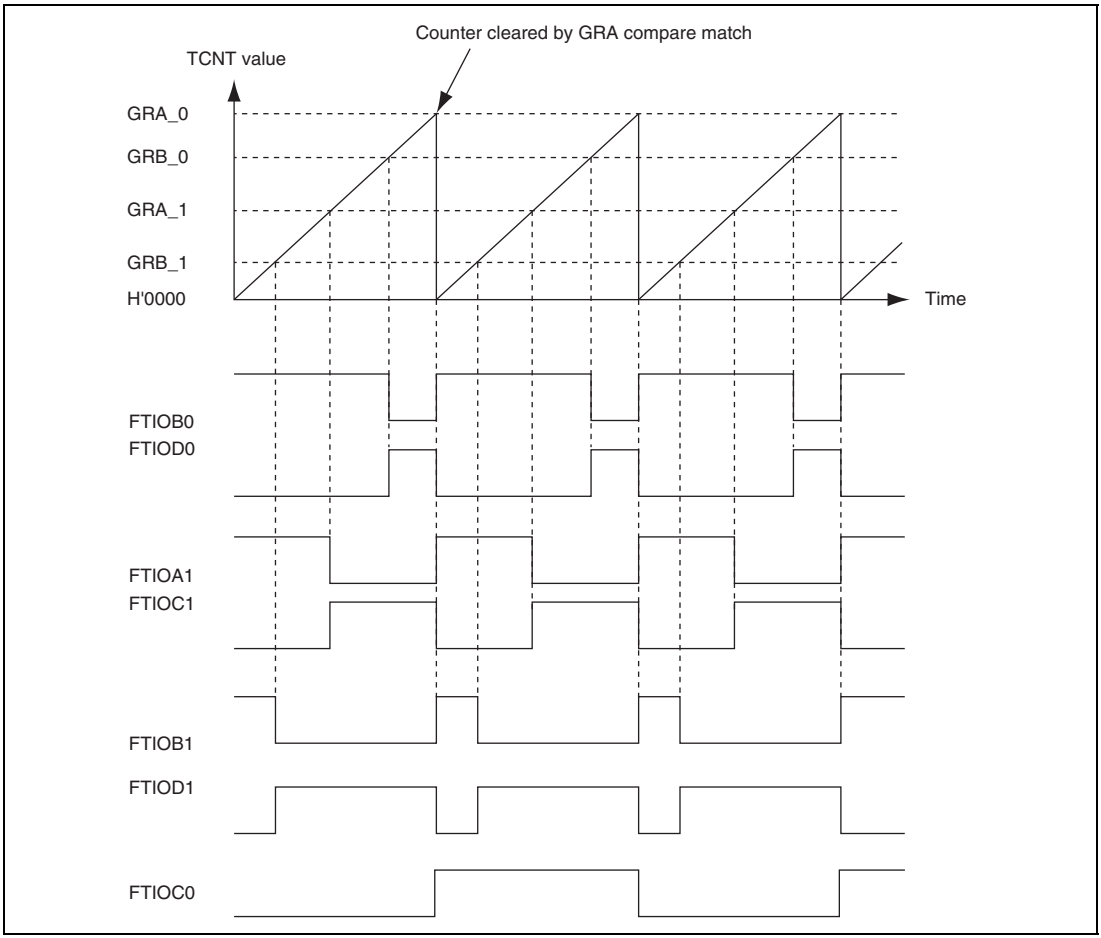



**Figure 13.28 Example of Reset Synchronous PWM Mode Operation (OLS0 = OLS1 = 0)**

In reset synchronous PWM mode, TCNT_0 and TCNT_1 perform increment and independent operations, respectively. However, GRA_1 and GRB_1 are separated from TCNT_1. When a compare match occurs between TCNT_0 and GRA_0, a counter is cleared and an increment operation is restarted from H'0000.

The PWM pin outputs 0 or 1 whenever a compare match between GRB_0, GRA_1, GRB_1 and TCNT_0 or counter clearing occur.

For details on operations when reset synchronous PWM mode and buffer operation are simultaneously set, refer to section 13.4.8, Buffer Operation.

To change duty cycles while a waveform with a duty cycle of 0% or 100% is being output, make sure the following procedure.

- To change duty cycles while a 0%-duty cycle waveform is being output, write to GR while H'0000 ≤ TCNT_1 < previous GR value
- To change duty cycles while a 100%-duty cycle waveform is being output, write to GR while previous GR value< TCNT_0 ≤ GRA_0

Note that changing from a 0%-duty cycle waveform to a 100%-duty cycle waveform and vice versa is not possible.

b. Buffer operation is used and TPSC2 = TPSC1 = TPSC0 = 0

Write H'0000 or a value equal to or more than the GRA_0 value to the buffer register.

- To output a 0%-duty cycle waveform, write a value equal to or more than the GRA_0 value to the buffer register
- To output a 100%-duty cycle waveform, write H'0000 to the buffer register

For details on buffer operation, see section 13.4.8, Buffer Operation.

c. Buffer operation is not used and other than TPSC2 = TPSC1 = TPSC0 = 0

Write a value which satisfies GRA_0 + 1 < GR < H'FFFF to GR directly at the timing shown below.

- To output a 0%-duty cycle waveform, write the value while H'0000 ≤ TCNT_1 < previous GR value
- To output a 100%-duty cycle waveform, write the value while previous GR value< TCNT_0 ≤ GRA_0

To change duty cycles while a waveform with a duty cycle of 0% and 100% is being output, the following procedure must be followed.

- To change duty cycles while a 0%-duty cycle waveform is being output, write to GR while H'0000 ≤ TCNT_1 < previous GR value
- To change duty cycles while a 100%-duty cycle waveform is being output, write to GR while previous GR value< TCNT_0 ≤ GRA_0

Note that changing from a 0%-duty cycle waveform to a 100%-duty cycle waveform and vice versa is not possible.

d. Buffer operation is used and other than TPSC2 = TPSC1 = TPSC0 = 0

Write a value which satisfies GRA_0 + 1 < GR < H'FFFF to the buffer register. A waveform with a duty cycle of 0% can be output. However, a waveform with a duty cycle of 100% cannot be output using the buffer operation. Also, the buffer operation cannot be used to change duty cycles while a waveform with a duty cycle of 100% is being output. For details on buffer operation, see section 13.4.8, Buffer Operation.

### 16.8.4 Receive Data Sampling Timing and Reception Margin in Asynchronous Mode

In asynchronous mode, the SCI3 operates on a basic clock with a frequency of 16 times the transfer rate. In reception, the SCI3 samples the falling edge of the start bit using the basic clock, and performs internal synchronization. Receive data is latched internally at the rising edge of the 8th pulse of the basic clock as shown in figure 16.19. Thus, the reception margin in asynchronous mode is given by formula (1) below.

$$M = \left\{(0.5 - \frac{1}{2N}) - \frac{D - 0.5}{N} - (L - 0.5)\,F\right\} \times 100(\%)$$

... Formula (1)

Legend N : Ratio of bit rate to clock (N = 16)
D : Clock duty (D = 0.5 to 1.0)
L : Frame length (L = 9 to 12)
F : Absolute value of clock rate deviation

Assuming values of F (absolute value of clock rate deviation) = 0 and D (clock duty) = 0.5 in formula (1), the reception margin can be given by the formula.

$M = \{0.5 - 1/(2 \times 16)\} \times 100\ [\%] = 46.875\%$

However, this is only the computed value, and a margin of 20% to 30% should be allowed for in system design.

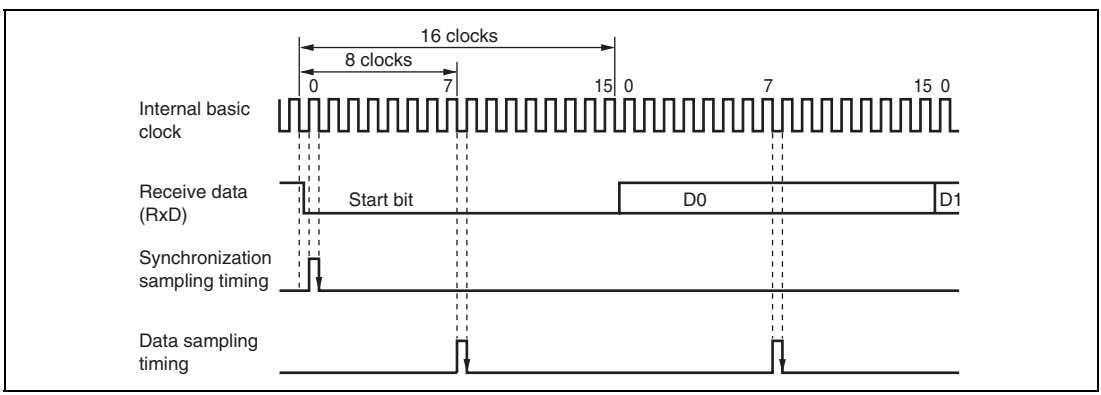
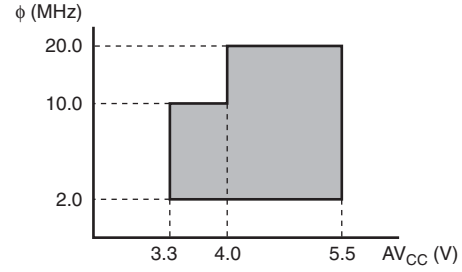
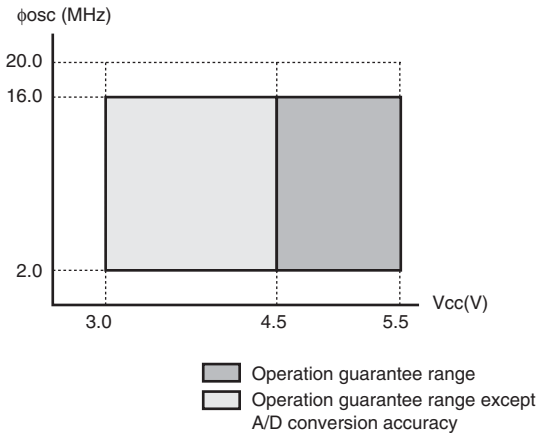



**Figure 16.19 Receive Data Sampling Timing in Asynchronous Mode**

**Analog Power Supply Voltage and A/D Converter Accuracy Guarantee Range**




**Range of Power Supply Voltage and Oscillation Frequency when Low-Voltage Detection Circuit is Used**



| Item | | Symbol | Test Condition | Values Min | Typ | Max | Unit |
|---|---|---|---|---|---|---|---|
| Erasing | Wait time after SWE bit setting*[1] | x | | 1 | — | — | µs |
| | Wait time after ESU bit setting*[1] | y | | 100 | — | — | µs |
| | Wait time after E bit setting*[1]*[6] | z | | 10 | — | 100 | ms |
| | Wait time after E bit clear*[1] | α | | 10 | — | — | µs |
| | Wait time after ESU bit clear*[1] | β | | 10 | — | — | µs |
| | Wait time after EV bit setting*[1] | γ | | 20 | — | — | µs |
| | Wait time after dummy write*[1] | ε | | 2 | — | — | µs |
| | Wait time after EV bit clear*[1] | η | | 4 | — | — | µs |
| | Wait time after SWE bit clear*[1] | θ | | 100 | — | — | µs |
| | Maximum erase count *[1]*[6]*[7] | N | | — | — | 120 | Times |

Notes:
1. Make the time settings in accordance with the program/erase algorithms.
2. The programming time for 128 bytes. (Indicates the total time for which the P bit in flash memory control register 1 (FLMCR1) is set. The program-verify time is not included.)
3. The time required to erase one block. (Indicates the time for which the E bit in flash memory control register 1 (FLMCR1) is set. The erase-verify time is not included.)
4. Programming time maximum value ($t_p$(max.)) = wait time after P bit setting (z) × maximum programming count (N)
5. Set the maximum programming count (N) according to the actual set values of z1, z2, and z3, so that it does not exceed the programming time maximum value ($t_p$(max.)). The wait time after P bit setting (z1, z2) should be changed as follows according to the value of the programming count (n).

   Programming count (n)

   $1 \leq n \leq 6$ z1 = 30 µs

   $7 \leq n \leq 1000$ z2 = 200 µs
6. Erase time maximum value ($t_E$(max.)) = wait time after E bit setting (z) × maximum erase count (N)
7. Set the maximum erase count (N) according to the actual set value of (z), so that it does not exceed the erase time maximum value ($t_E$(max.)).

| Item | Symbol | Applicable Pins | Test Condition | Values Min | Typ | Max | Unit | Notes |
|---|---|---|---|---|---|---|---|---|
| Input low voltage | $V_{IL}$ | $\overline{RES}$, $\overline{NMI}$, $\overline{WKP0}$ to $\overline{WKP5}$, $\overline{IRQ0}$ to $\overline{IRQ3}$, $\overline{ADTRG}$, TMRIV, TMCIV, FTIOA0 to FTIOD0, FTIOA1 to FTIOD1, SCK3, SCK3_2, TRGV | $V_{CC}$ = 4.0 to 5.5 V | –0.3 | — | $V_{CC} \times 0.2$ | V | |
| | | | | –0.3 | — | $V_{CC} \times 0.1$ | | |
| | | RXD, RXD_2, SCL, SDA, P10 to P12, P14 to P17, P20 to P24, P30 to P37, P50 to P57, P60 to P67,. P70 to P72, P74 to P76, P85 to P87, PB0 to PB7 | $V_{CC}$ = 4.0 to 5.5 V | –0.3 | — | $V_{CC} \times 0.3$ | V | |
| | | | | –0.3 | — | $V_{CC} \times 0.2$ | | |
| | | OSC1 | $V_{CC}$ = 4.0 to 5.5 V | –0.3 | — | 0.5 | V | |
| | | | | –0.3 | — | 0.3 | | |
| Output high voltage | $V_{OH}$ | P10 to P12, P14 to P17, P20 to P24, P30 to P37, P50 to P55, P60 to P67, P70 to P72, P74 to P76, P85 to P87 | $V_{CC}$ = 4.0 to 5.5 V<br>$-I_{OH}$ = 1.5 mA | $V_{CC}$ – 1.0 | — | — | V | |
| | | | $-I_{OH}$ = 0.1 mA | $V_{CC}$ – 0.5 | — | — | | |
| | | P56, P57 | $V_{CC}$ = 4.0 to 5.5 V<br>$-I_{OH}$ = 0.1 mA | $V_{CC}$ – 2.5 | — | — | V | |
| | | | $V_{CC}$ =2.7 to 4.0 V<br>$-I_{OH}$ = 0.1 mA | $V_{CC}$ – 2.0 | — | — | | |

| Mnemonic | | Operand Size | Addressing Mode and Instruction Length (bytes) | | | | | | | | | Operation | Condition Code | | | | | | No. of States*1 | |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| | | | #xx | Rn | @ERn | @(d, ERn) | @–ERn/@ERn+ | @aa | @(d, PC) | @@aa | — | | I | H | N | Z | V | C | Normal | Advanced |
| MOV | MOV.W Rs, @–ERd | W | | | | | 2 | | | | | ERd32–2 → ERd32<br>Rs16 → @ERd | — | — | ↕ | ↕ | 0 | — | 6 | |
| | MOV.W Rs, @aa:16 | W | | | | | | 4 | | | | Rs16 → @aa:16 | — | — | ↕ | ↕ | 0 | — | 6 | |
| | MOV.W Rs, @aa:24 | W | | | | | | 6 | | | | Rs16 → @aa:24 | — | — | ↕ | ↕ | 0 | — | 8 | |
| | MOV.L #xx:32, Rd | L | 6 | | | | | | | | | #xx:32 → Rd32 | — | — | ↕ | ↕ | 0 | — | 6 | |
| | MOV.L ERs, ERd | L | | 2 | | | | | | | | ERs32 → ERd32 | — | — | ↕ | ↕ | 0 | — | 2 | |
| | MOV.L @ERs, ERd | L | | | 4 | | | | | | | @ERs → ERd32 | — | — | ↕ | ↕ | 0 | — | 8 | |
| | MOV.L @(d:16, ERs), ERd | L | | | | 6 | | | | | | @(d:16, ERs) → ERd32 | — | — | ↕ | ↕ | 0 | — | 10 | |
| | MOV.L @(d:24, ERs), ERd | L | | | | 10 | | | | | | @(d:24, ERs) → ERd32 | — | — | ↕ | ↕ | 0 | — | 14 | |
| | MOV.L @ERs+, ERd | L | | | | | 4 | | | | | @ERs → ERd32<br>ERs32+4 → ERs32 | — | — | ↕ | ↕ | 0 | — | 10 | |
| | MOV.L @aa:16, ERd | L | | | | | | 6 | | | | @aa:16 → ERd32 | — | — | ↕ | ↕ | 0 | — | 10 | |
| | MOV.L @aa:24, ERd | L | | | | | | 8 | | | | @aa:24 → ERd32 | — | — | ↕ | ↕ | 0 | — | 12 | |
| | MOV.L ERs, @ERd | L | | | 4 | | | | | | | ERs32 → @ERd | — | — | ↕ | ↕ | 0 | — | 8 | |
| | MOV.L ERs, @(d:16, ERd) | L | | | | 6 | | | | | | ERs32 → @(d:16, ERd) | — | — | ↕ | ↕ | 0 | — | 10 | |
| | MOV.L ERs, @(d:24, ERd) | L | | | | 10 | | | | | | ERs32 → @(d:24, ERd) | — | — | ↕ | ↕ | 0 | — | 14 | |
| | MOV.L ERs, @–ERd | L | | | | | 4 | | | | | ERd32–4 → ERd32<br>ERs32 → @ERd | — | — | ↕ | ↕ | 0 | — | 10 | |
| | MOV.L ERs, @aa:16 | L | | | | | | 6 | | | | ERs32 → @aa:16 | — | — | ↕ | ↕ | 0 | — | 10 | |
| | MOV.L ERs, @aa:24 | L | | | | | | 8 | | | | ERs32 → @aa:24 | — | — | ↕ | ↕ | 0 | — | 12 | |
| POP | POP.W Rn | W | | | | | | | | | 2 | @SP → Rn16<br>SP+2 → SP | — | — | ↕ | ↕ | 0 | — | 6 | |
| | POP.L ERn | L | | | | | | | | | 4 | @SP → ERn32<br>SP+4 → SP | — | — | ↕ | ↕ | 0 | — | 10 | |
| PUSH | PUSH.W Rn | W | | | | | | | | | 2 | SP–2 → SP<br>Rn16 → @SP | — | — | ↕ | ↕ | 0 | — | 6 | |
| | PUSH.L ERn | L | | | | | | | | | 4 | SP–4 → SP<br>ERn32 → @SP | — | — | ↕ | ↕ | 0 | — | 10 | |
| MOVFPE | MOVFPE @aa:16, Rd | B | | | | | | 4 | | | | Cannot be used in this LSI | Cannot be used in this LSI | | | | | | | |
| MOVTPE | MOVTPE Rs, @aa:16 | B | | | | | | 4 | | | | Cannot be used in this LSI | Cannot be used in this LSI | | | | | | | |

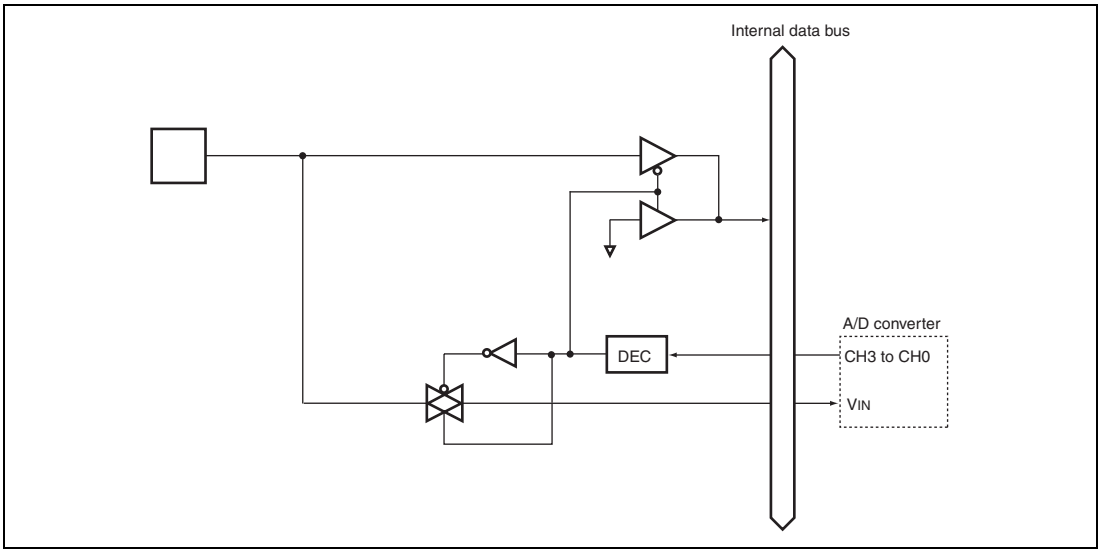



**Figure B.23 Port B Block Diagram (PB7 to PB0)**