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 | H8S/2000 |
| Core Size | 16-Bit |
| Speed | 24MHz |
| Connectivity | SCI, SmartCard, USB |
| Peripherals | DMA, POR, PWM, WDT |
| Number of I/O | 37 |
| Program Memory Size | 32KB (32K x 8) |
| Program Memory Type | FLASH |
| EEPROM Size | - |
| RAM Size | 4K x 8 |
| Voltage - Supply (Vcc/Vdd) | 2.7V ~ 3.6V |
| Data Converters | A/D 6x10b |
| Oscillator Type | External |
| Operating Temperature | -20°C ~ 75°C (TA) |
| Mounting Type | Surface Mount |
| Package / Case | 64-LQFP |
| Supplier Device Package | 64-LFQFP (10x10) |
| Purchase URL | https://www.e-xfl.com/product-detail/renesas-electronics-america/df2210cufp24v |

| | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
|---|---|---|---|---|---|---|---|---|
| I/O | Output | Output | Output | Output | Input | Input | Input | Input |
| P1DDR | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| Read value | 1 | 1 | 1 | 1 | 1 | 0 | 0 | 0 |

The BCLR instruction performs bit manipulation on the read value, which is H'F8 in this example. It clears bit 4 to 0.

| | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
|---|---|---|---|---|---|---|---|---|
| I/O | Output | Output | Output | Output | Input | Input | Input | Input |
| P1DDR | 1 | 1 | 1 | 1 | 0 | 0 | 0 | 0 |
| After bit manipulation | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |

Following bit manipulation the data is written to P1DDR and the BCLR instruction terminates.

| | P17 | P16 | P15 | P14 | P13 | P12 | P11 | P10 |
|---|---|---|---|---|---|---|---|---|
| I/O | Output | Output | Output | Input | Output | Input | Input | Input |
| P1DDR | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |
| Write value | 1 | 1 | 1 | 0 | 1 | 0 | 0 | 0 |

The contents of P1DDR should have been overwritten with a value of H'E0, but in fact a value of H'E8 was written to the register. This changed pin 13, which should have been an input pin, to an output pin. In this example we assumed that the value of bit 1 in P1DDR was 1. However, since the values of bits 7 to 0 in P1DDR are all undefined when read, there is the possibility that individual bit values could be changed from 0 to 1 or from 1 to 0. To prevent this from happening, the recommendations in section 2.9.4, Accessing Registers Containing Write-Only Bits, should be followed when changing the values of registers containing write-only bits.

In addition, the BCLR instruction can be used to clear flags in internal I/O registers to 0. In such cases it is not necessary to read the relevant flag beforehand so long as it is clear that it has been set to 1 by an interrupt processing routine or the like.

### 2.9.4 Accessing Registers Containing Write-Only Bits

Using data transfer instructions or bit manipulation instructions on registers containing write-only bits can result in undefined values being read. To prevent the reading of undefined values, the procedure described below should be used to access registers containing write-only bits.

**Write after Read:** If an external write occurs after an external read while the ICIS0 bit in BCRH is set to 1, an idle cycle is inserted at the start of the write cycle.

Figure 6.23 shows an example of the operation in this case. In this example, bus cycle A is a read cycle from ROM with a long output floating time, and bus cycle B is a CPU write cycle. In (a), an idle cycle is not inserted, and a collision occurs in cycle B between the read data from ROM and the CPU write data. In (b), an idle cycle is inserted, and a data collision is prevented.

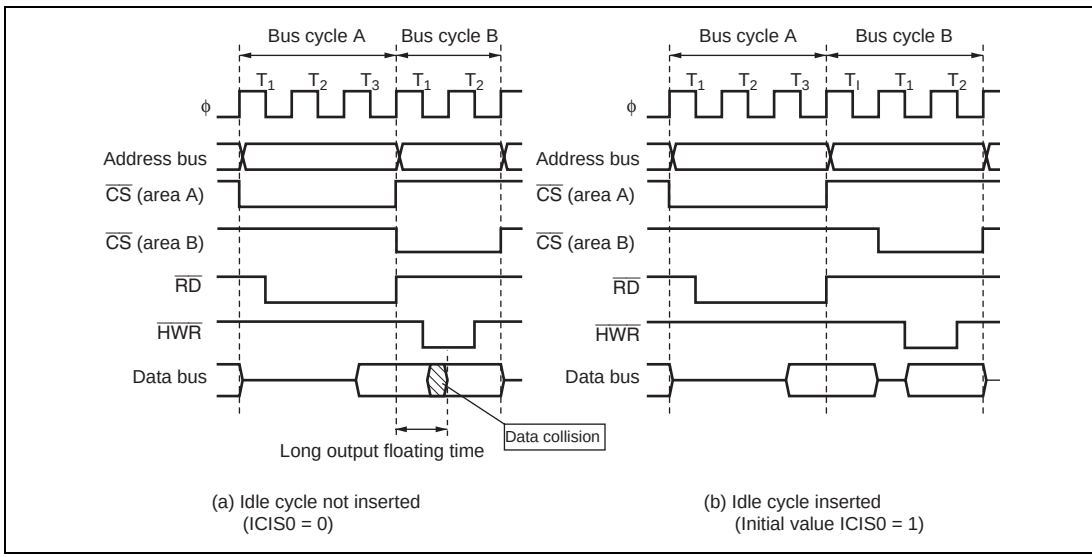



**Figure 6.23 Example of Idle Cycle Operation (2)**

**Relationship between Chip Select ($\overline{CS}$) Signal and Read ($\overline{RD}$) Signal:** Depending on the system's load conditions, the $\overline{RD}$ signal may lag behind the $\overline{CS}$ signal. An example is shown in figure 6.24.

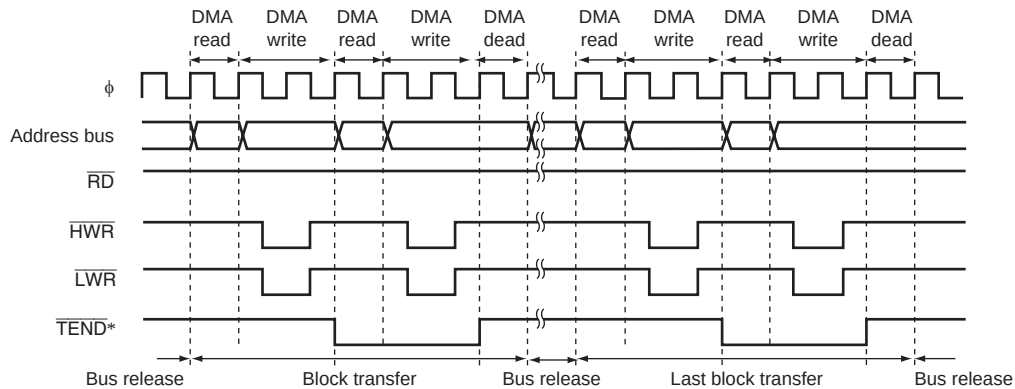
In this case, with the setting for no idle cycle insertion (a), there may be a period of overlap between the bus cycle A $\overline{RD}$ signal and the bus cycle B $\overline{CS}$ signal.

Setting idle cycle insertion, as in (b), however, will prevent any overlap between the $\overline{RD}$ and $\overline{CS}$ signals.

In the initial state after reset release, idle cycle insertion (b) is set.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| | | | | Data Transfer Interrupt Enable B<br>Enables or disables an interrupt to the CPU when transfer is interrupted. If the DTIEB bit is set to 1 when DTME = 0, the DMAC regards this as indicating a break in the transfer, and issues a transfer break interrupt request to the CPU. A transfer break interrupt can be canceled either by clearing the DTIEB bit to 0 in the interrupt handling routine, or by performing processing to continue transfer by setting the DTME bit to 1. |
| 3 | DTIE1B | 0 | R/W | Data Transfer Interrupt Enable 1B<br>Enables or disables the channel 1 transfer break interrupt.<br>0: Transfer break interrupt disabled<br>1: Transfer break interrupt enabled |
| | | | | Data Transfer End Interrupt Enable A<br>Enables or disables an interrupt to the CPU when transfer ends. If the DTIEA bit is set to 1 when DTE = 0, the DMAC regards this as indicating the end of a transfer, and issues a transfer end interrupt request to the CPU. A transfer end interrupt can be canceled either by clearing the DTIEA bit to 0 in the interrupt handling routine, or by performing processing to continue transfer by setting the DTE bit to 1. |
| 2 | DTIE1A | 0 | R/W | Data Transfer End Interrupt Enable 1A<br>Enables or disables the channel 1 transfer end interrupt.<br>0: Transfer end interrupt disabled<br>1: Transfer end interrupt enabled |
| 1 | DTIE0B | 0 | R/W | Data Transfer Interrupt Enable 0B<br>Enables or disables the channel 0 transfer break interrupt.<br>0: Transfer break interrupt disabled<br>1: Transfer break interrupt enabled |
| 0 | DTIE0A | 0 | R/W | Data Transfer End Interrupt Enable 0A<br>Enables or disables the channel 0 transfer end interrupt.<br>0: Transfer end interrupt disabled<br>1: Transfer end interrupt enabled |

**Full Address Mode (Block Transfer Mode):** Figure 7.18 shows a transfer example in which $\overline{\text{TEND}}$* output is enabled and word-size full address mode transfer (block transfer mode) is performed from internal 16-bit, 1-state access space to external 16-bit, 2-state access space.




Note: * This LSI does not support $\overline{\text{TEND}}$ output.

**Figure 7.18 Example of Full Address Mode (Block Transfer Mode) Transfer**

A one-block transfer is performed for one transfer request, and after the transfer the bus is released. While the bus is released, one or more bus cycles are inserted by the CPU.

In the transfer end cycle of each block (the cycle in which the transfer counter reaches 0), a one-state DMA dead cycle is inserted after the DMA write cycle.

One block is transmitted without interruption. NMI generation does not affect block transfer operation.

Note: * This LSI does not support $\overline{\text{TEND}}$ output.

### 8.6.6 Pin Functions

#### Pin Functions of H8S/2218 Group

Port A pins also function as address bus (A19 to A16) output pins and SCI_2 I/O pins. The correspondence between the register specification and the pin functions is shown below.

**Table 8.29 PA3 Pin Function**

| Operating Mode | Modes 4 to 6 | | | | | | Mode 7 | | | | |
|---|---|---|---|---|---|---|---|---|---|---|---|
| AE3 to AE0 | B'11×× | Other than B'11×× | | | | | — | | | | |
| CKE1 in SCR_2 | — | 0 | | | | 1 | 0 | | | | 1 |
| C/$\overline{A}$ in SMR_2 | — | 0 | | | 1 | — | 0 | | | 1 | — |
| CKE0 in SCR_2 | — | 0 | | 1 | — | — | 0 | | 1 | — | — |
| PA3DDR | — | 0 | 1 | — | — | — | 0 | 1 | — | — | — |
| Pin Function | A19 output pin | PA3 input pin | PA3 output pin | SCK2 output pin | SCK2 output pin | SCK2 input pin | PA3 input pin | PA3 output pin | SCK2 output pin | SCK2 output pin | SCK2 input pin |

**Table 8.30 PA2 Pin Function**

| Operating mode | Modes 4 to 6 | | | | Mode 7 | | |
|---|---|---|---|---|---|---|---|
| AE3 to AE0 | B'1011 or B'11×× | Other than B'1011 or B'11×× | | | — | | |
| RE in SCR_2 | — | 0 | | 1 | 0 | | 1 |
| PA2DDR | — | 0 | 1 | — | 0 | 1 | — |
| Pin Function | A18 output pin | PA2 input pin | PA2 output pin | RxD2 input pin | PA2 input pin | PA2 output pin | RxD2 input pin |

Legend:
×: Don't care.

### 14.3.20 USB Interrupt Enable Register 1 (UIER1)

UIER1 enables the interrupt request indicated in the interrupt flag register 1 (UIFR1). When an interrupt flag is set while the corresponding bit in UIER1 is set to 1, an interrupt is requested by asserting the corresponding $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. Either $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ must be selected by the interrupt select register 1 (UISR1).

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 to 4 | — | All 0 | R | Reserved<br>These bits are always read as 0. |
| 3 | — | 0 | R/W | Reserved<br>The write value should always be 0. |
| 2 | EP2READYE | 0 | R/W | Enables the EP2READY interrupt. |
| 1 | EP1TRE | 0 | R/W | Enables the EP1TR interrupt. |
| 0 | EP1EMPTYE | 0 | R/W | Enables the EP1EMPTYE interrupt. |

### 14.3.21 USB Interrupt Enable Register 3 (UIER3)

UIER3 enables the interrupt request indicated in the interrupt flag register 3 (UIFR3). This register is readable/writable while the USB module stop 2 bit (MSTPB0) in MSTPCRB is 1.

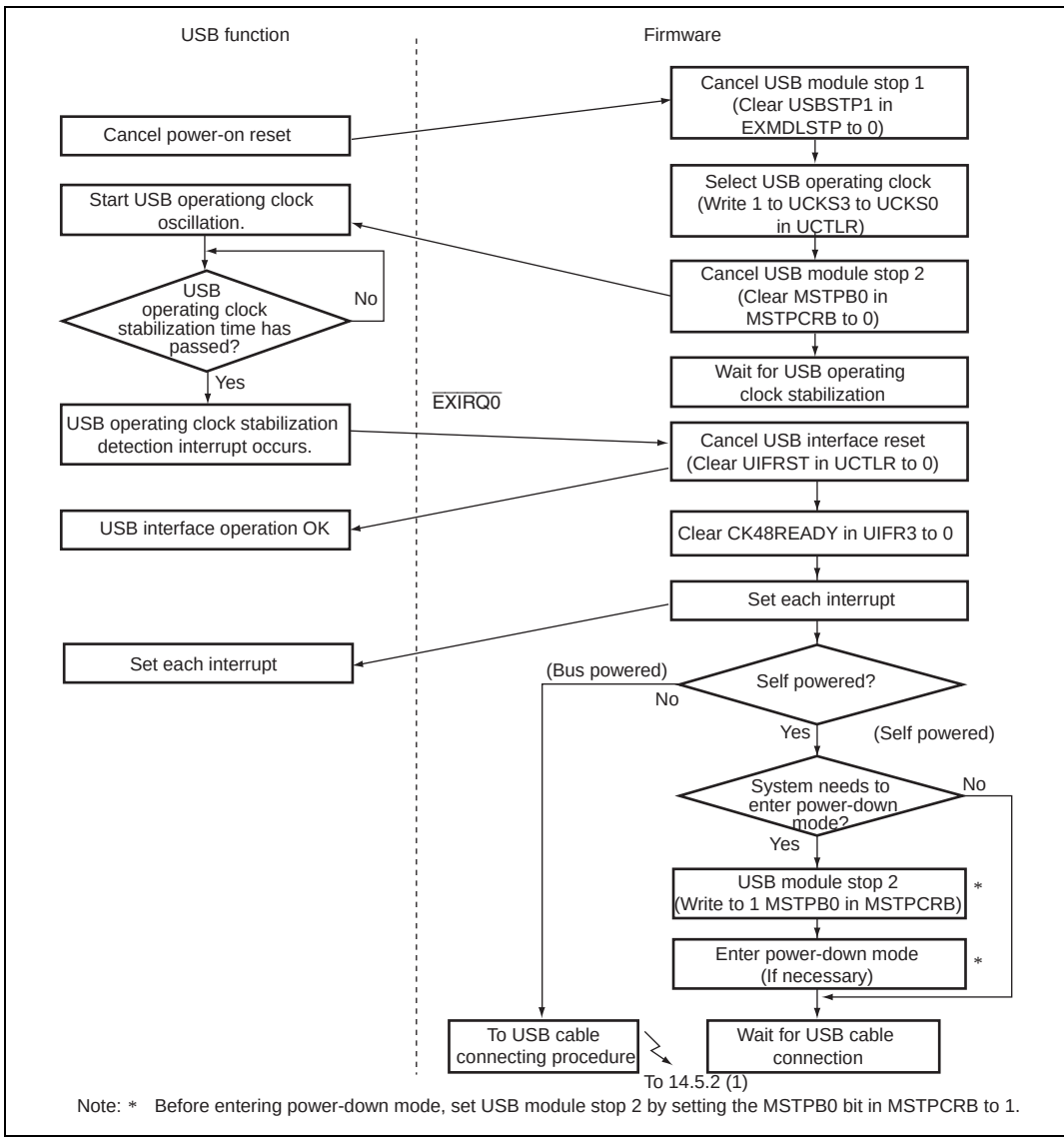
When an interrupt flag is set while the corresponding bit in UIER3 is set to 1, an interrupt is requested by asserting the corresponding $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$. Either $\overline{\text{EXIRQ0}}$ or $\overline{\text{EXIRQ1}}$ must be selected by the interrupt select register 3 (UISR3). Note, however, that the SPRSiE bit is an interrupt enable bit specific to the $\overline{\text{IRQ6}}$ pin and cannot be selected by UISR3.

| Bit | Bit Name | Initial Value | R/W | Description |
|---|---|---|---|---|
| 7 | CK48READYE | 1 | R/W | Enables the CK48READY interrupt. |
| 6 | SOFE | 0 | R/W | Enables the SOF interrupt. |
| 5 | SETCE | 0 | R/W | Enables the SETC interrupt. |
| 4, 3 | — | All 0 | R | Reserved<br>These bits are always read as 0. |
| 2 | SPRSiE | 0 | R/W | Enables the SPRSi interrupt. (only for $\overline{\text{IRQ6}}$) |
| 1 | — | 0 | R | Reserved<br>This bit is always read as 0. |
| 0 | VBUSiE | 0 | R/W | Enables the VBUSi interrupt. |

## 14.5 Communication Operation

### 14.5.1 Initialization

The USB must be initialized as described in the flowchart in figure 14.2.



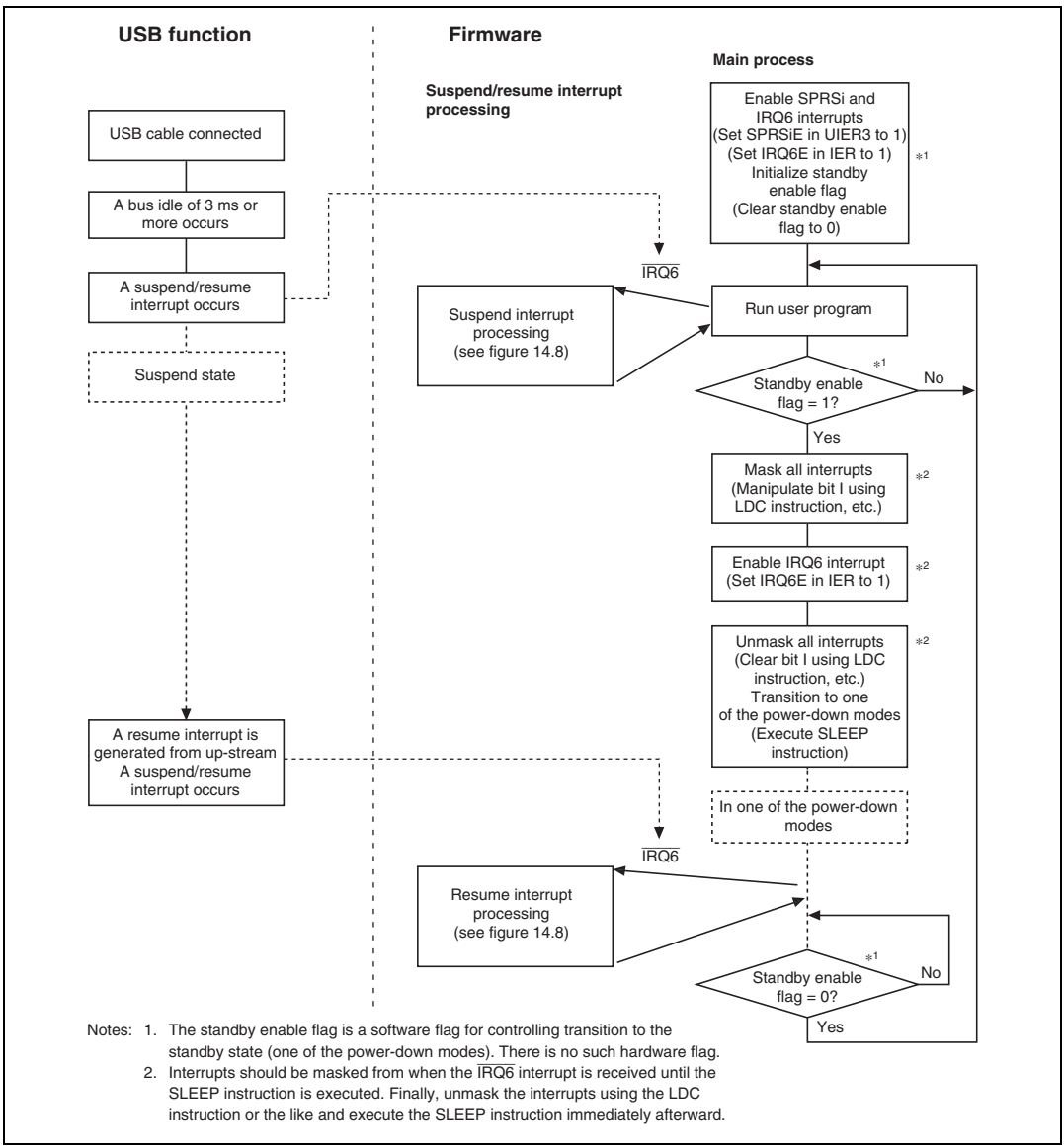

Note: * Before entering power-down mode, set USB module stop 2 by setting the MSTPB0 bit in MSTPCRB to 1.

**Figure 14.2 USB Initialization**

### 14.5.3 Suspend and Resume Operations

#### (1) Suspend and Resume Operations

Figures 14.7 and 14.8 are flowcharts of the suspend and resume operations. If the USB bus enters the suspend state from a non-suspend state, or if it enters a non-suspend state from the suspend state due to a resume signal from up-stream, perform the operations shown below.




Notes: 1. The standby enable flag is a software flag for controlling transition to the standby state (one of the power-down modes). There is no such hardware flag.
2. Interrupts should be masked from when the $\overline{\text{IRQ6}}$ interrupt is received until the SLEEP instruction is executed. Finally, unmask the interrupts using the LDC instruction or the like and execute the SLEEP instruction immediately afterward.

**Figure 14.7 Example Flowchart of Suspend and Resume Operations**

pointer that controls the internal module FIFO is updated and correct operation cannot be guaranteed.

### 14.8.8 Reset

— The manual reset during USB communication operations must not be executed, since the LSI may stop with the state of USD+ and USD- pins maintained.

This USB module uses synchronous reset for some registers. The reset state of these registers must be cancelled after the clock oscillation stabilization time has passed. At initialization, reset must be cancelled using the following procedure:

1. Cancel the USB module stop 1: Clear the USBSTOP1 bit in EXMDLSTP to 0.
2. Select the USB operating clock: Write 1 to the UCKS3 to UCKS0 bits in UCTLR.
3. Cancel the USB module stop 2: Clear the MSTPB0 bit in MSTPCRB to 0.
4. Wait for the USB operating clock stabilization: Wait until the CK48READY bit in UIFR3 is set to 1.
5. Cancel the USB interface reset state: Clear the UIFRST bit in UCTLR to 0.
6. Cancel the UDC core reset state: Clear the UDCRST bit in UCTLR to 0.

For details, see the flowcharts in section 14.5.1, Initialization and section 14.5.2, USB Cable Connection/Disconnection.

— The USB registers are not initialized when the watchdog timer (WDT) triggers a power-on reset. Therefore, the USB may not operate properly after a power-on reset is triggered by the WDT due to CPU runaway or a similar cause. (If a power-on reset is triggered by input of a power-on reset signal from the $\overline{\text{RES}}$ pin, the USB registers are initialized and there is no problem.) Consequently, an initialization routine should be used to write the initial values listed below to the following three registers, thereby ensuring that all the USB registers are properly initialized, immediately following a reset.

UCTLR = H'03, UIER3 = H'80, UIFR3 = H'00

### 14.8.9 EP0 Interrupt Sources Assignment

EP0 interrupt sources assigned to bits 3 to 0 in UIFR0 must be assigned to the same interrupt signal ($\overline{\text{EXIRQx}}$) by setting UISR0. There are no other restrictions on interrupt sources.

## 19.7 Subclock Waveform Generation Circuit

To eliminate noise from the subclock input to OSCI, the subclock is sampled using the dividing clock ϕ. The sampling frequency is set using the NESEL bit of LPWRCR. For details, see section 19.1.2, Low Power Control Register (LPWRCR).

No sampling is performed in subactive mode, subsleep mode, or watch mode.

## 19.8 PLL Circuit for USB

The PLL circuit has the function of doubling or tripling the 16-MHz or 24-MHz clock from the main oscillator to generate the 48-MHz USB operating clock.

When the PLL circuit is used, set the UCKS3 to UCKS0 bits of UCTLR. For details, refer to section 14, Universal Serial Bus (USB).

When the PLL circuit is not used, connect the PLVCC pin to Vcc and the PLLVSS pin to the ground (Vss). Figure 19.9 shows examples of external circuits peripheral to the PLL.

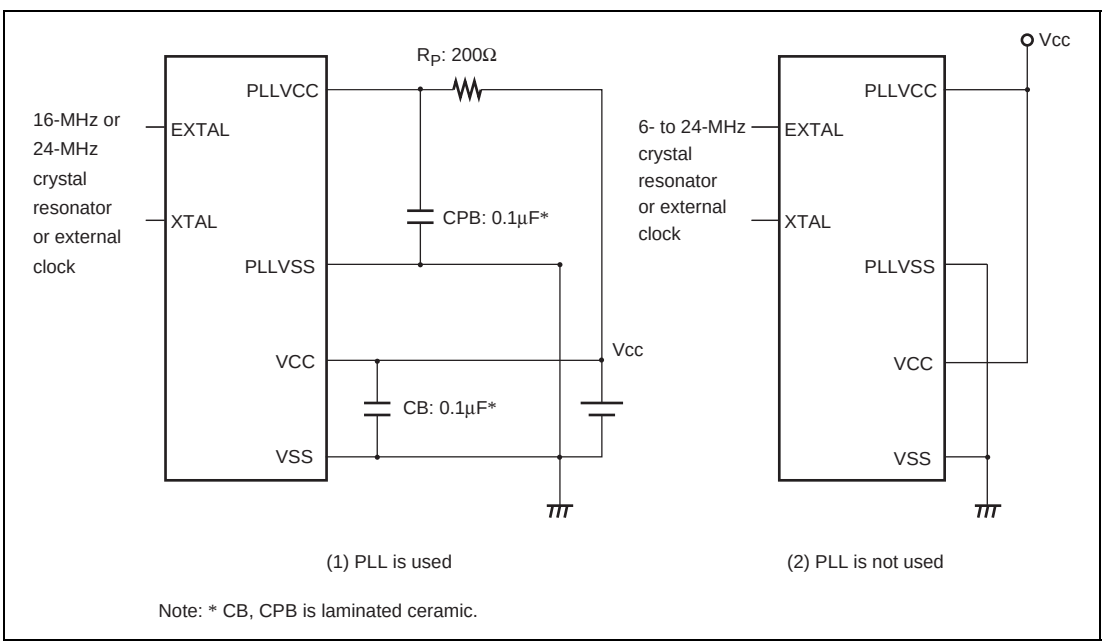



**Figure 19.9 Example of PLL Circuit**

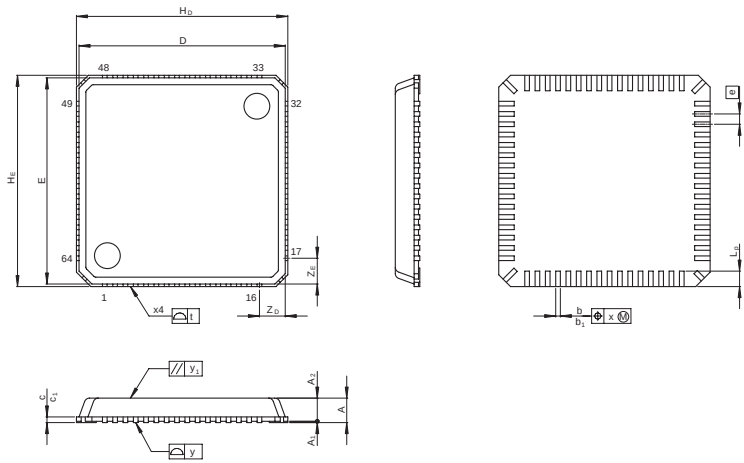
## 21.2 Register Bits

Register bit names of the on-chip peripheral modules are described below.

Each line covers eight bits, so 16-bit registers are shown as two lines and 32-bit registers as four lines.

| Register Name | Bit 7 | Bit 6 | Bit 5 | Bit 4 | Bit 3 | Bit 2 | Bit 1 | Bit 0 | Module |
|---|---|---|---|---|---|---|---|---|---|
| UCTLR | — | USPNDE | UCKS3 | UCKS2 | UCKS1 | UCKS0 | UIFRST | UDCRST | USB |
| UTSTRA | — | — | — | — | — | — | — | — | |
| UDMAR | — | — | — | — | EP2T1 | EP2T0 | EP1T1 | EP1T0 | |
| UDRR | — | — | — | — | — | — | RWUPs | DVR | |
| UTRG0 | — | — | EP2RDFN | EP1PKTE | EP3PKTE | EP0oRDFN | EP0iPKTE | EP0sRDFN | |
| UFCLR0 | — | — | EP2CLR | EP1CLR | EP3CLR | EP0oCLR | EP0iCLR | — | |
| UESTL0 | — | — | EP2STL | EP1STL | EP3STL | — | — | EP0STL | |
| UESTL1 | SCME | — | — | — | — | — | — | — | |
| UEDR0s | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| UEDR0i | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| UEDR0o | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| UEDR3 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| UEDR1 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| UEDR2 | D7 | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| UESZ0o | — | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| UESZ2 | — | D6 | D5 | D4 | D3 | D2 | D1 | D0 | |
| UIFR0 | BRST | — | EP3TR | EP3TS | EP0oTS | EP0iTR | EP0iTS | SetupTS | |
| UIFR1 | — | — | — | — | EP1ALL EMPTYs | EP2 READY | EP1TR | EP1 EMPTY | |
| UIFR3 | CK48 READY | SOF | SETC | — | SPRSs | SPRSi | VBUSs | VBUSi | |
| UIER0 | BRSTE | — | EP3TRE | EP3TSE | EP0oTSE | EP0iTRE | EP0iTSE | SetupTSE | |
| UIER1 | — | — | — | — | — | EP2 READYE | EP1TRE | EP1 EMPTYE | |
| UIER3 | CK48 READYE | SOFE | SETCE | — | — | SPRSiE | — | VBUSiE | |
| UISR0 | BRSTS | — | EP3TRS | EP3TSS | EP0oTSS | EP0iTRS | EP0iTSS | SetupTSS | |

| JEITA Package Code | RENESAS Code | Previous Code | MASS[Typ.] |
|---|---|---|---|
| P-VQFN64-8x8-0.40 | PVQN0064LB-A | TNP-64B/TNP-64BV | 0.12g |




| Reference Symbol | Dimension in Millimeters | | |
|---|---|---|---|
| | Min | Nom | Max |
| D | — | 8.0 | — |
| E | — | 8.0 | — |
| $A_2$ | — | 0.89 | — |
| A | — | — | 0.95 |
| $A_1$ | 0.005 | 0.02 | 0.04 |
| b | 0.13 | 0.18 | 0.23 |
| $b_1$ | — | 0.16 | — |
| e | — | 0.4 | — |
| $L_p$ | 0.50 | 0.60 | 0.70 |
| x | — | — | 0.05 |
| y | — | — | 0.05 |
| $y_1$ | — | — | 0.2 |
| t | — | — | 0.2 |
| $H_D$ | — | 8.2 | — |
| $H_E$ | — | 8.2 | — |
| $Z_D$ | — | 1.0 | — |
| $Z_E$ | — | 1.0 | — |
| c | 0.17 | 0.22 | 0.25 |
| $c_1$ | — | 0.20 | — |

**Figure C.4 TNP-64B and TNP-64BV Package Dimensions**