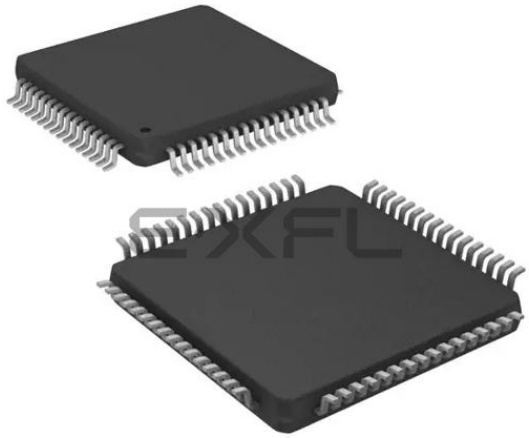
Renesas Electronics America Inc - UPD78F1145AGA-HAB-AX Datasheet



Welcome to E-XFL.COM

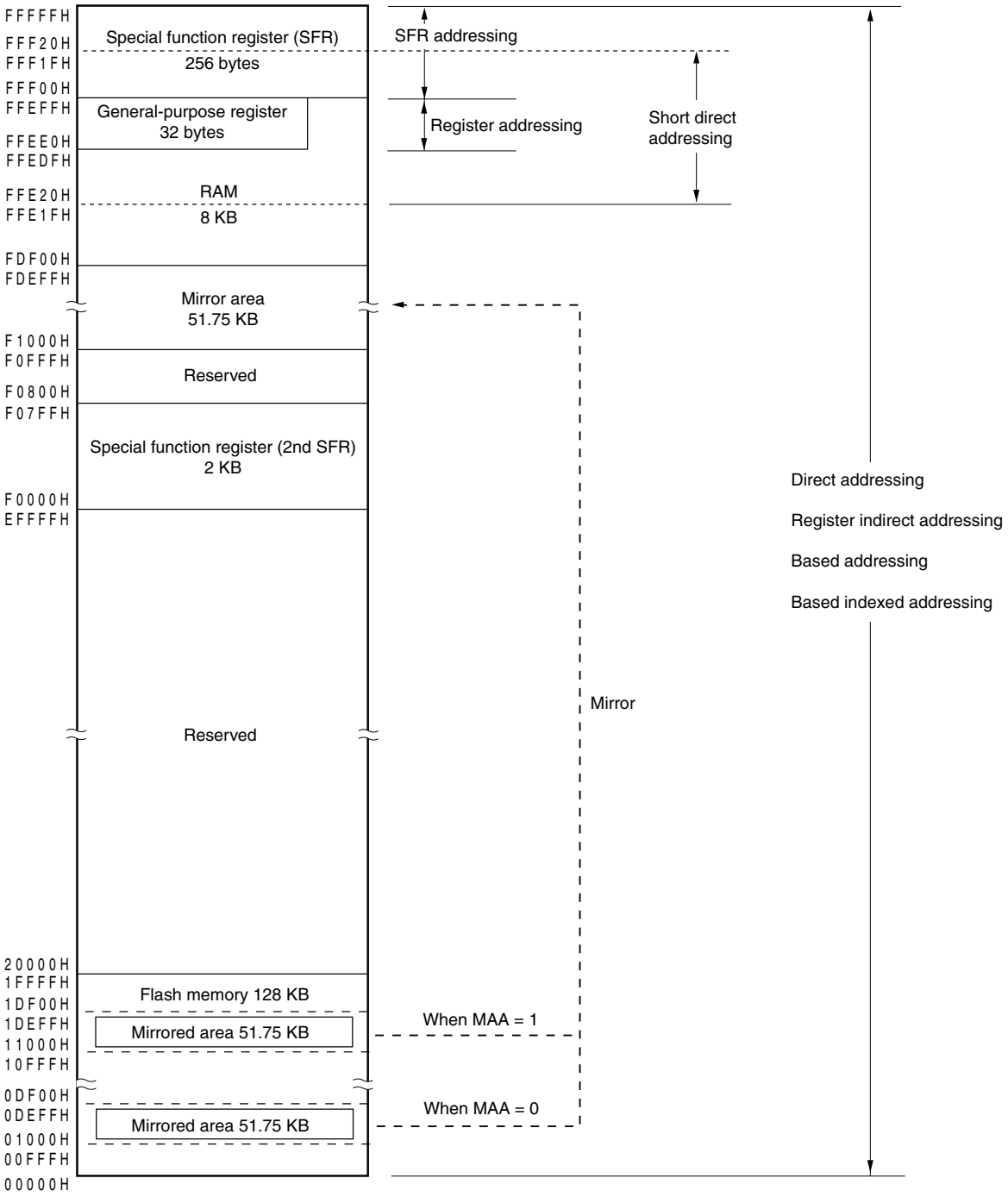
What is "Embedded - Microcontrollers"?

"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, I²C, LINbus, UART/USART
Peripherals	DMA, LVD, POR, PWM, WDT
Number of I/O	50
Program Memory Size	192KB (192K x 8)
Program Memory Type	FLASH
EEPROM Size	-
RAM Size	10K x 8
Voltage - Supply (Vcc/Vdd)	1.8V ~ 5.5V
Data Converters	A/D 8x10b
Oscillator Type	Internal
Operating Temperature	-40°C ~ 85°C (TA)
Mounting Type	Surface Mount
Package / Case	64-TQFP
Supplier Device Package	-
Purchase URL	https://www.e-xfl.com/product-detail/renesas-electronics-america/upd78f1145aga-hab-ax

<R> **Figure 3-9. Correspondence Between Data Memory and Addressing (μPD78F1144, 78F1144A)**



5.2 Configuration of Clock Generator

The clock generator includes the following hardware.

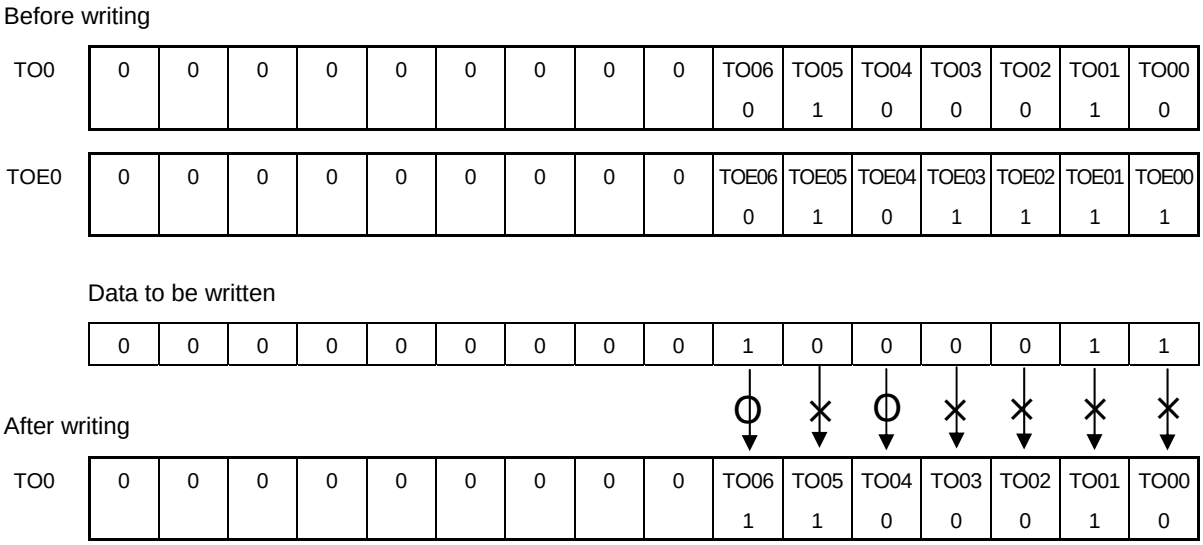
Table 5-1. Configuration of Clock Generator

Item	Configuration
Control registers	Clock operation mode control register (CMC) Clock operation status control register (CSC) Oscillation stabilization time counter status register (OSTC) Oscillation stabilization time select register (OSTS) System clock control register (CKC) Peripheral enable register 0 (PER0) Operation speed mode control register (OSMC) Internal high-speed oscillator trimming register (HIOTRM)
Oscillators	X1 oscillator XT1 oscillator Internal high-speed oscillator Internal low-speed oscillator

6.4.4 Collective manipulation of TO0n bits

In the TO0 register, the setting bits for all the channels are located in one register in the same way as the TS0 register (channel start trigger). Therefore, TO0n of all the channels can be manipulated collectively. Only specific bits can also be manipulated by setting the corresponding TOE0n = 0 to a target TO0n (channel output).

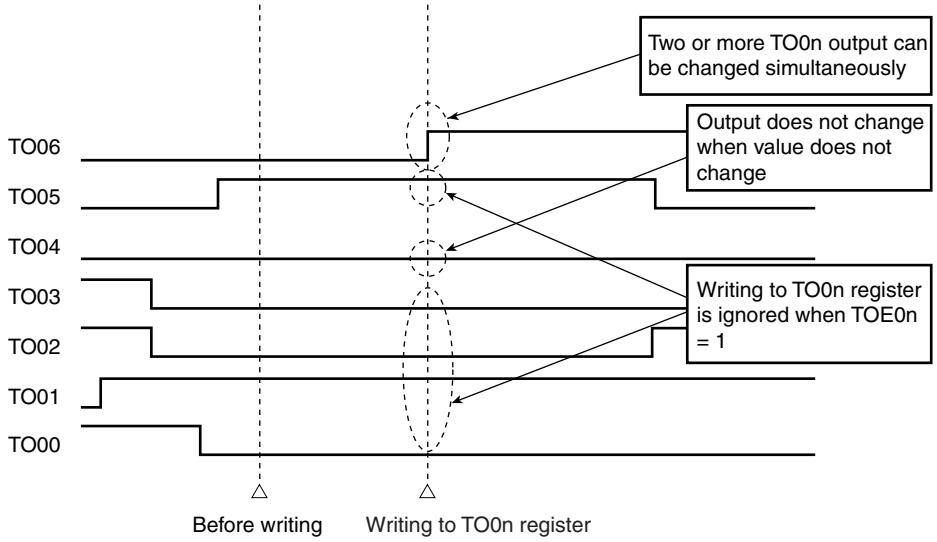
Figure 6-30. Example of TO0n Bits Collective Manipulation



Writing is done only to TO0n bits with TOE0n = 0, and writing to TO0n bits with TOE0n = 1 is ignored.

TO0n (channel output) to which TOE0n = 1 is set is not affected by the write operation. Even if the write operation is done to TO0n, it is ignored and the output change by timer operation is normally done.

Figure 6-31. TO0n Pin Statuses by Collective Manipulation of TOon Bits



(Caution and Remark are given on the next page.)

Caution When TOE0n = 1, even if the output by timer interrupt of each timer (INTTM0n) contends with writing to TO0n, output is normally done to TO0n pin.

Remark n = 0 to 6

6.4.5 Timer Interrupt and TO0n Pin Output at Operation Start

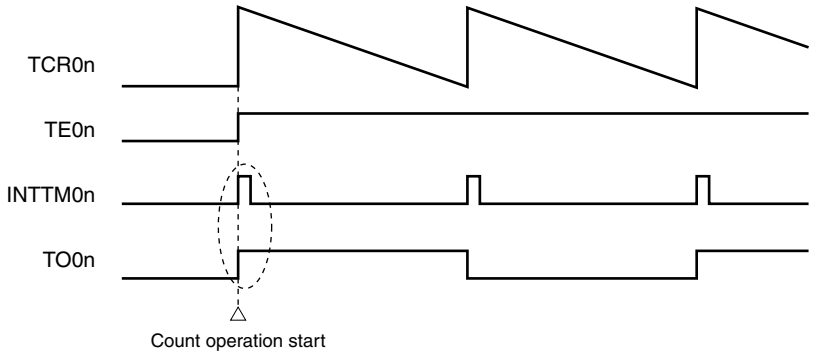
In the interval timer mode or capture mode, the MD0n0 bit in the TMR0n register sets whether or not to generate a timer interrupt at count start.

When MD0n0 is set to 1, the count operation start timing can be known by the timer interrupt (INTTM0n) generation.

In the other modes, neither timer interrupt at count operation start nor TO0n output is controlled.

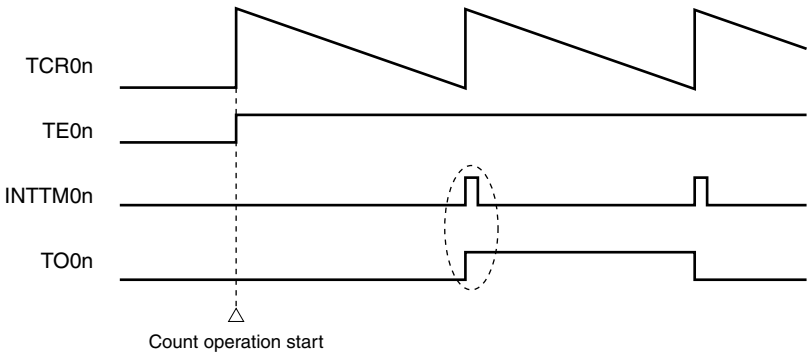
Figures 6-32 and 6-33 show operation examples when the interval timer mode (TOE0n = 1, TOM0n = 0) is set.

Figure 6-32. When MD0n0 is set to 1



When MD0n0 is set to 1, a timer interrupt (INTTM0n) is output at count operation start, and TO0n performs a toggle operation.

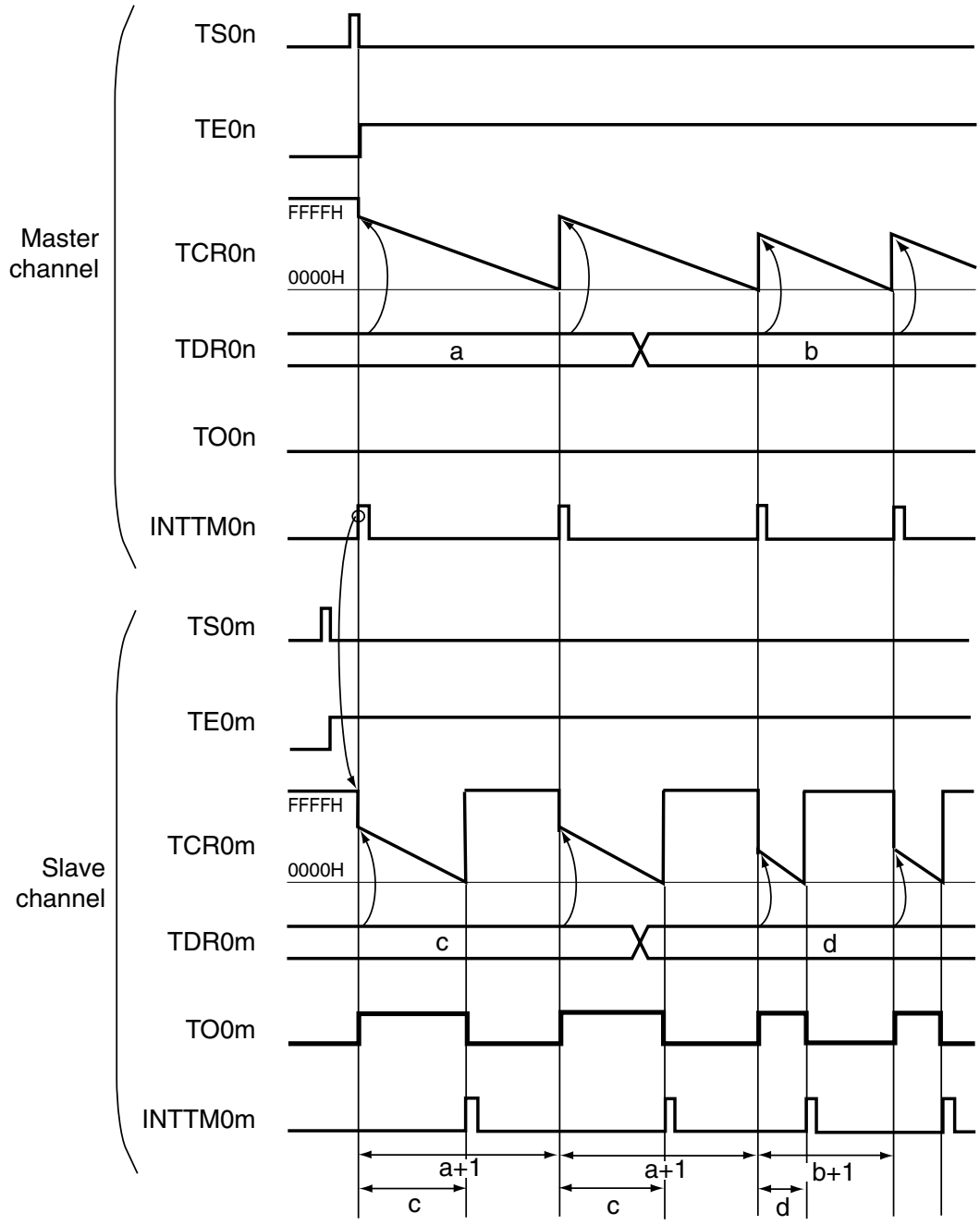
Figure 6-33. When MD0n0 is set to 0



When MD0n0 is set to 0, a timer interrupt (INTTM0n) is not output at count operation start, and TO0n does not change either. After counting one cycle, INTTM0n is output and TO0n performs a toggle operation.

Remark n = 0 to 6

Figure 6-56. Example of Basic Timing of Operation as PWM Function



Remark n = 0, 2, 4
m = n + 1

(1) Peripheral enable register 0 (PER0)

PER0 is used to enable or disable use of each peripheral hardware macro. Clock supply to a hardware macro that is not used is stopped in order to reduce the power consumption and noise.

When the real-time counter is used, be sure to set bit 7 (RTCEN) of this register to 1.

PER0 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

Figure 7-2. Format of Peripheral Enable Register 0 (PER0)

Address: F00F0H After reset: 00H R/W

Symbol	<7>	6	<5>	<4>	<3>	<2>	1	<0>
PER0	RTCEN	0	DACEN	IIC0EN	SAU1EN	SAU0EN	0	TAU0EN

RTCEN	Control of real-time counter (RTC) input clock supply[Note]
0	Stops supply of input clock. • SFR used by the real-time counter (RTC) cannot be written. • The real-time counter (RTC) is in the reset status.
1	Supplies input clock. • SFR used by the real-time counter (RTC) can be read/written.

Note RTCEN is used to supply or stop the clock used when accessing the real-time counter (RTC) register from the CPU. RTCEN cannot control supply of the operating clock (f_{SUB}) to RTC.

Cautions 1. When using the real-time counter, first set RTCEN to 1, while oscillation of the subsystem clock (f_{SUB}) is stable. If RTCEN = 0, writing to a control register of the real-time counter is ignored, and, even if the register is read, only the default value is read.

2. Be sure to clear bits 1, 6 of the PER0 register to 0.

(2) Real-time counter control register 0 (RTCC0)

The RTCC0 register is an 8-bit register that is used to start or stop the real-time counter operation, control the RTCCL and RTC1HZ pins, and set a 12- or 24-hour system and the constant-period interrupt function.

RTCC0 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

CHAPTER 11 SERIAL ARRAY UNIT

The serial array unit has four serial channels per unit and can use two or more of various serial interfaces (3-wire serial (CSI), UART, and simplified I^2C) in combination.

Function assignment of each channel supported by the 78K0R/KE3 is as shown below (channels 2 and 3 of unit 1 are dedicated to UART3 (supporting LIN-bus)).

Unit	Channel	Used as CSI	Used as UART	Used as Simplified I^2C
0	0	CSI00	UART0	–
	1	–		–
	2	CSI10	UART1	IIC10
	3	–		–
1	0	–	–	–
	1	–	–	–
	2	–	UART3 (supporting LIN-bus)	–
	3	–		–

(Example of combination) When “UART1” is used for channels 2 and 3 of unit 0, CSI10 and IIC10 cannot be used, but CSI00 or UART0 can be used.

11.1 Functions of Serial Array Unit

Each serial interface supported by the 78K0R/KE3 has the following features.

11.1.1 3-wire serial I/O (CSI00, CSI10)

This is a clocked communication function that uses three lines: serial clock ($\overline{SCK}$) and serial data (SI and SO) lines.

[Data transmission/reception]

- Data length of 7 or 8 bits
- Phase control of transmit/receive data
- MSB/LSB first selectable
- Level setting of transmit/receive data

[Clock control]

- Master/slave selection
- Phase control of I/O clock
- Setting of transfer period by prescaler and internal counter of each channel

[Interrupt function]

- Transfer end interrupt/buffer empty interrupt

[Error detection flag]

- Overrun error

(15) Noise filter enable register 0 (NFEN0)

NFEN0 is used to set whether the noise filter can be used for the input signal from the serial data input pin to each channel.

Disable the noise filter of the pin used for CSI or simplified I^2C communication, by clearing the corresponding bit of this register to 0.

Enable the noise filter of the pin used for UART communication, by setting the corresponding bit of this register to 1.

When the noise filter is enabled, CPU/peripheral operating clock (f_{CLK}) is synchronized with 2-clock match detection.

NFEN0 can be set by a 1-bit or 8-bit memory manipulation instruction.

Reset signal generation clears this register to 00H.

Figure 11-18. Format of Noise Filter Enable Register 0 (NFEN0)

Address: F0060H After reset: 00H R/W

Symbol	7	6	5	4	3	2	1	0
NFEN0	0	SNFEN30	0	0	0	SNFEN10	0	SNFEN00

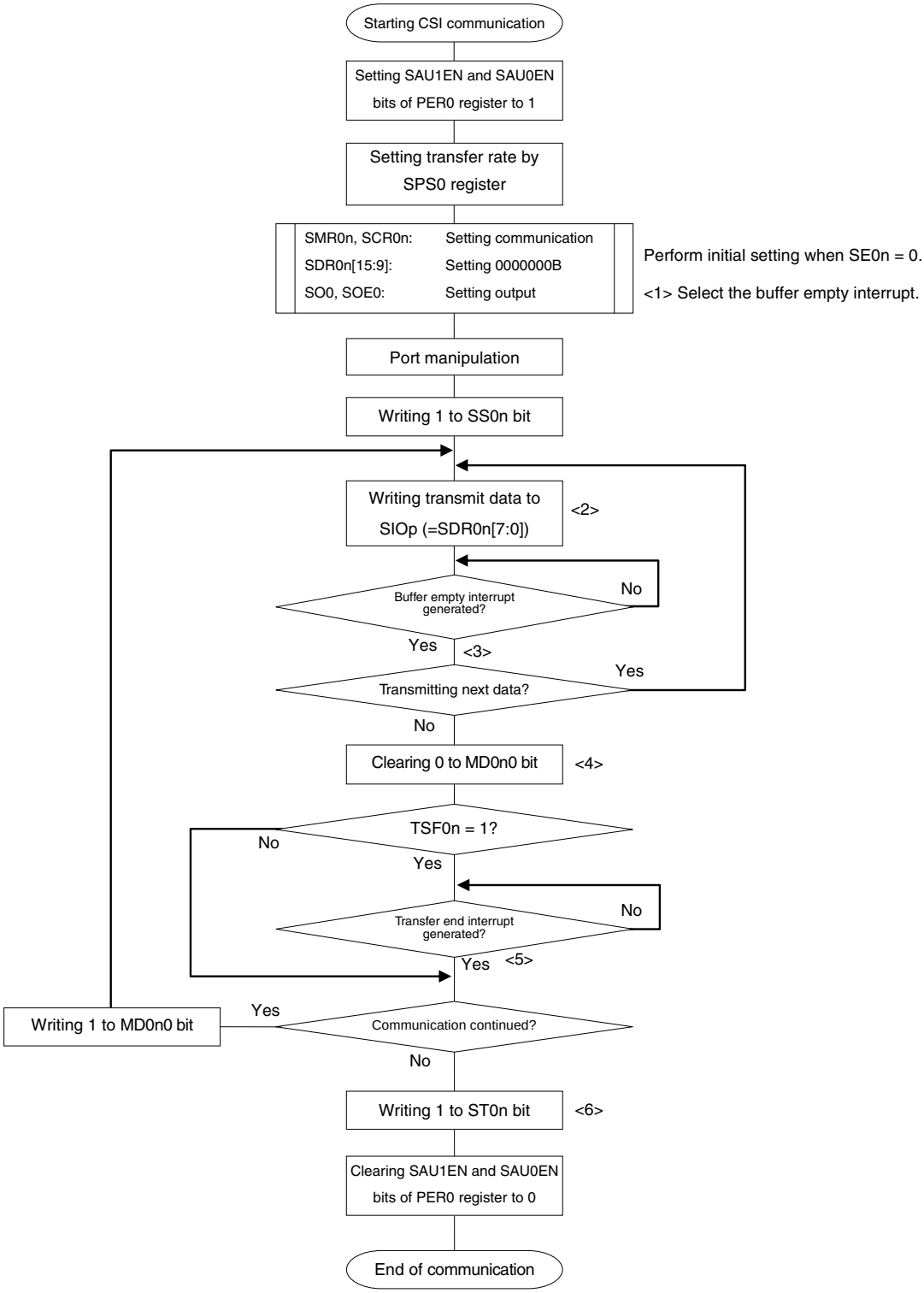
SNFEN30	Use of noise filter of RxD3/P14 pin
0	Noise filter OFF
1	Noise filter ON
Set SNFEN30 to 1 to use the RxD3 pin. Clear SNFEN30 to 0 to use the P14 pin.	

SNFEN10	Use of noise filter of RxD1/SDA10/SI10/P03 pin
0	Noise filter OFF
1	Noise filter ON
Set SNFEN10 to 1 to use the RxD1 pin. Clear SNFEN10 to 0 to use the SDA10, SI10, and P03 pins.	

SNFEN00	Use of noise filter of RxD0/SI00/P11 pin
0	Noise filter OFF
1	Noise filter ON
Set SNFEN00 to 1 to use the RxD0 pin. Clear SNFEN00 to 0 to use the SI00 and P11 pins.	

Caution Be sure to clear bits 7, 5 to 3, and 1 to "0".

Figure 11-55. Flowchart of Slave Transmission (in Continuous Transmission Mode)



<R>

Caution After setting the PER0 register to 1, be sure to set the SPS0 register after 4 or more clocks have elapsed.

Remark <1> to <6> in the figure correspond to <1> to <6> in **Figure 11-54 Timing Chart of Slave Transmission (in Continuous Transmission Mode)**.

(3) SO latch

The SO latch is used to retain the SDA0 pin's output level.

(4) Wakeup controller

This circuit generates an interrupt request (INTIIC0) when the address received by this register matches the address value set to slave address register 0 (SVA0) or when an extension code is received.

(5) Prescaler

This selects the sampling clock to be used.

(6) Serial clock counter

This counter counts the serial clocks that are output or input during transmit/receive operations and is used to verify that 8-bit data was transmitted or received.

(7) Interrupt request signal generator

This circuit controls the generation of interrupt request signals (INTIIC0).

An I^2C interrupt request is generated by the following two triggers.

- Falling edge of eighth or ninth clock of the serial clock (set by WTIM0 bit)
- Interrupt request generated when a stop condition is detected (set by SPIE0 bit)

Remark WTIM0 bit: Bit 3 of IIC control register 0 (IICC0)
SPIE0 bit: Bit 4 of IIC control register 0 (IICC0)

(8) Serial clock controller

In master mode, this circuit generates the clock output via the SCL0 pin from a sampling clock.

(9) Serial clock wait controller

This circuit controls the wait timing.

(10) ACK generator, stop condition detector, start condition detector, and ACK detector

These circuits generate and detect each status.

(11) Data hold time correction circuit

This circuit generates the hold time for data corresponding to the falling edge of the serial clock.

(12) Start condition generator

This circuit generates a start condition when the STT0 bit is set to 1.

However, in the communication reservation disabled status (IICRSV bit = 1), when the bus is not released (IICBSY bit = 1), start condition requests are ignored and the STCF bit is set to 1.

(13) Stop condition generator

This circuit generates a stop condition when the SPT0 bit is set to 1.

14.2 Configuration of DMA Controller

The DMA controller includes the following hardware.

Table 14-1. Configuration of DMA Controller

Item	Configuration
Address registers	• DMA SFR address registers 0, 1 (DSA0, DSA1) • DMA RAM address registers 0, 1 (DRA0, DRA1)
Count register	• DMA byte count registers 0, 1 (DBC0, DBC1)
Control registers	• DMA mode control registers 0, 1 (DMC0, DMC1) • DMA operation control register 0, 1 (DRC0, DRC1)

(1) DMA SFR address register n (DSAn)

This is an 8-bit register that is used to set an SFR address that is the transfer source or destination of DMA channel n.

Set the lower 8 bits of the SFR addresses FFF00H to FFFFFH[Note].

This register is not automatically incremented but fixed to a specific value.

In the 16-bit transfer mode, the least significant bit is ignored and is treated as an even address.

DSAn can be read or written in 8-bit units. However, it cannot be written during DMA transfer.

Reset signal generation clears this register to 00H.

Note Except for address FFFFEH because the PMC register is allocated there.

Figure 14-1. Format of DMA SFR Address Register n (DSAn)

Address: FFFB0H (DSA0), FFFB1H (DSA1) After reset: 00H R/W

	7	6	5	4	3	2	1	0
DSAn								

Remark n: DMA channel number (n = 0, 1)

(2) DMA RAM address register n (DRAn)

This is a 16-bit register that is used to set a RAM address that is the transfer source or destination of DMA channel n.

Addresses of the internal RAM area other than the general-purpose registers (FEF00H to FFEDFH in the case of the μPD78F1142 and 78F1142A) can be set to this register.

Set the lower 16 bits of the RAM address.

This register is automatically incremented when DMA transfer has been started. It is incremented by +1 in the 8-bit transfer mode and by +2 in the 16-bit transfer mode. DMA transfer is started from the address set to this DRAn register. When the data of the last address has been transferred, DRAn stops with the value of the last address +1 in the 8-bit transfer mode, and the last address +2 in the 16-bit transfer mode.

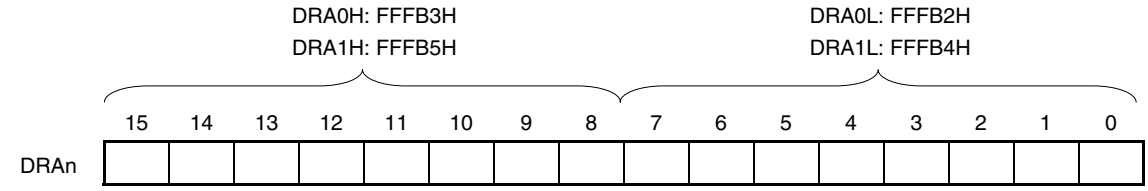
In the 16-bit transfer mode, the least significant bit is ignored and is treated as an even address.

DRAn can be read or written in 8-bit or 16-bit units. However, it cannot be written during DMA transfer.

Reset signal generation clears this register to 0000H.

Figure 14-2. Format of DMA RAM Address Register n (DRAn)

Address: FFFB2H, FFFB3H (DRA0), FFFB4H, FFFB5H (DRA1) After reset: 0000H R/W



(n = 0, 1)

Remark n: DMA channel number (n = 0, 1)

Figure 14-4. Format of DMA Mode Control Register n (DMCn) (2/2)

Address: FFFBAH (DMC0), FFFBBH (DMC1) After reset: 00H R/W

Symbol	<7>	<6>	<5>	<4>	3	2	1	0
DMCn	STGn	DRSn	DSn	DWAITn	IFCn3	IFCn2	IFCn1	IFCn0

IFCn 3	IFCn 2	IFCn 1	IFCn 0	Selection of DMA start source[Note]	
				Trigger signal	Trigger contents
0	0	0	0	–	Disables DMA transfer by interrupt. (Only software trigger is enabled.)
0	0	1	0	INTTM00	End of timer channel 0 count or capture end interrupt
0	0	1	1	INTTM01	End of timer channel 1 count or capture end interrupt
0	1	0	0	INTTM04	End of timer channel 4 count or capture end interrupt
0	1	0	1	INTTM05	End of timer channel 5 count or capture end interrupt
0	1	1	0	INTST0/INTCSI00	UART0 transmission transfer end or buffer empty interrupt/CSI00 transfer end or buffer empty interrupt
0	1	1	1	INTSR0	UART0 reception transfer end
1	0	0	0	INTST1/INTCSI10/INTIIC10	UART1 transmission transfer end or buffer empty interrupt/CSI10 transfer end or buffer empty interrupt/ IIC10 transfer end interrupt
1	0	0	1	INTSR1	UART1 reception transfer end interrupt
1	0	1	0	INTST3	UART3 transmission transfer end or buffer empty interrupt
1	0	1	1	INTSR3	UART3 reception transfer end interrupt
1	1	0	0	INTAD	A/D conversion end interrupt
Other than above				Setting prohibited	

Note The software trigger (STGn) can be used regardless of the IFCn0 to IFCn3 values.

Remark n: DMA channel number (n = 0, 1)

Cautions 1. Be sure to clear bits 4 to 6 of IF1H and bits 1 to 7 of IF2H to 0.

2. When operating a timer, serial interface, or A/D converter after standby release, operate it once after clearing the interrupt request flag. An interrupt request flag may be set by noise.

3. When manipulating a flag of the interrupt request flag register, use a 1-bit memory manipulation instruction (CLR1). When describing in C language, use a bit manipulation instruction such as "IF0L.0 = 0;" or "_asm("clr1 IF0L, 0");" because the compiled assembler must be a 1-bit memory manipulation instruction (CLR1).

If a program is described in C language using an 8-bit memory manipulation instruction such as "IF0L &= 0xfe;" and compiled, it becomes the assembler of three instructions.

```
mov a, IF0L
and a, #0FEH
mov IF0L, a
```

In this case, even if the request flag of another bit of the same interrupt request flag register (IF0L) is set to 1 at the timing between "mov a, IF0L" and "mov IF0L, a", the flag is cleared to 0 at "mov IF0L, a". Therefore, care must be exercised when using an 8-bit memory manipulation instruction in C language.

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

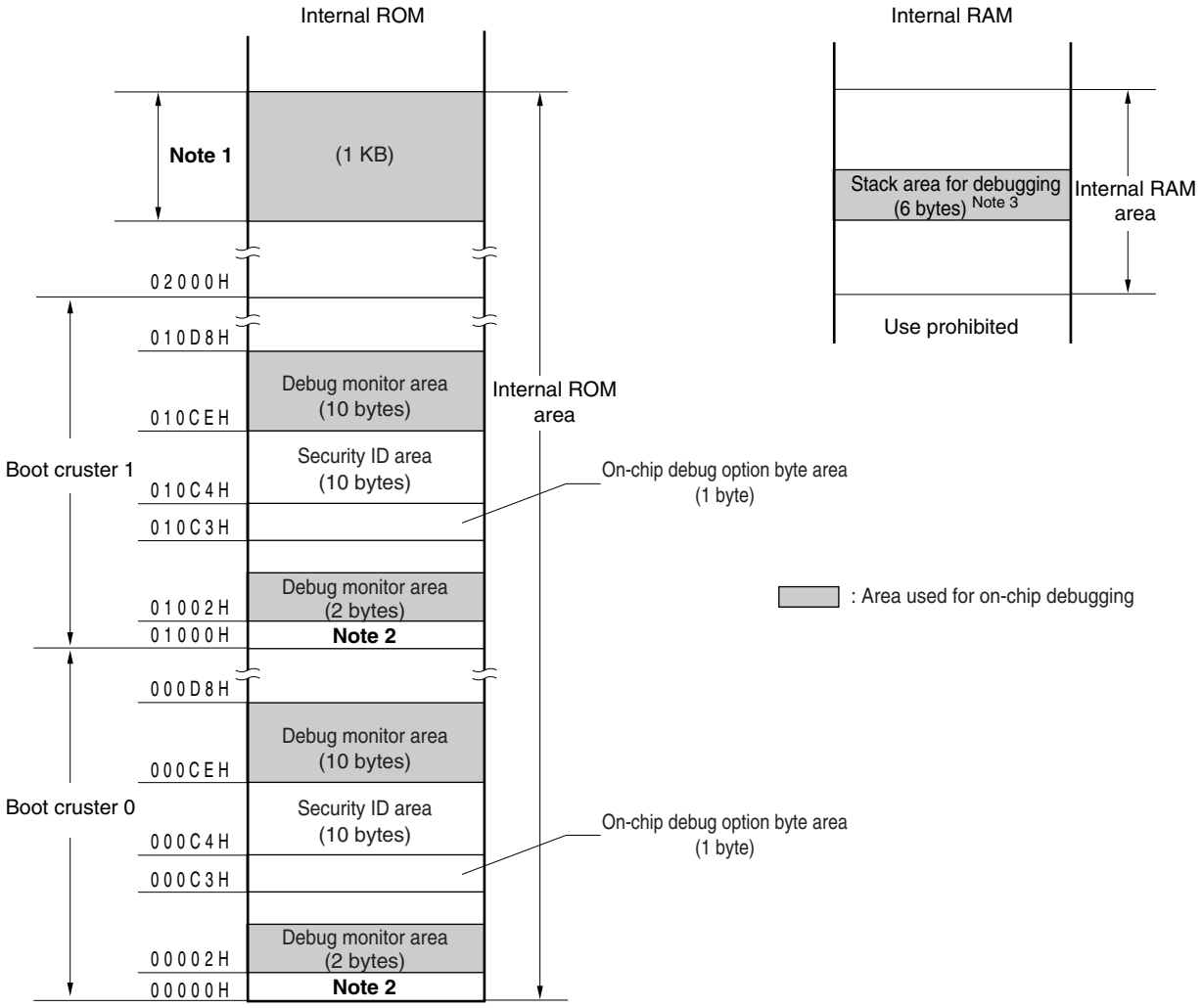
The interrupt mask flags are used to enable/disable the corresponding maskable interrupt servicing.

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 24-2. Memory Spaces Where Debug Monitor Programs Are Allocated



Notes 1. Address differs depending on products as follows.

Products	Internal ROM	Address
μPD78F1142, 78F1142A	64 KB	0FC00H-0FFFFH
μPD78F1143, 78F1143A	96 KB	17C00H-17FFFH
μPD78F1144, 78F1144A	128 KB	1FC00H-1FFFFH
μPD78F1145, 78F1145A	192 KB	2FC00H-2FFFFH
μPD78F1146, 78F1146A	256 KB	3FC00H-3FFFFH

2. In debugging, reset vector is rewritten to address allocated to a monitor program.

3. Since this area is allocated immediately before the stack area, the address of this area varies depending on the stack increase and decrease. That is, 6 extra bytes are consumed for the stack area used.

For details of the way to secure of the memory space, refer to the **QB-MINI2 On-Chip Debug Emulator with Programming Function User's Manual (U18371E)**.

Table 26-5. Operation List (3/17)

Instruction Group	Mnemonic	Operands	Bytes	Clocks		Operation	Flag
				Note 1	Note 2		Z AC CY
8-bit data transfer	MOV	A, ES:[HL]	2	2	5	A ← (ES, HL)	
		ES:[HL], A	2	2	–	(ES, HL) ← A	
		ES:[HL + byte],#byte	4	2	–	((ES, HL) + byte) ← byte	
		A, ES:[HL + byte]	3	2	5	A ← ((ES, HL) + byte)	
		ES:[HL + byte], A	3	2	–	((ES, HL) + byte) ← A	
		A, ES:[HL + B]	3	2	5	A ← ((ES, HL) + B)	
		ES:[HL + B], A	3	2	–	((ES, HL) + B) ← A	
		A, ES:[HL + C]	3	2	5	A ← ((ES, HL) + C)	
		ES:[HL + C], A	3	2	–	((ES, HL) + C) ← A	
		ES:word[B], #byte	5	2	–	((ES, B) + word) ← byte	
		A, ES:word[B]	4	2	5	A ← ((ES, B) + word)	
		ES:word[B], A	4	2	–	((ES, B) + word) ← A	
		ES:word[C], #byte	5	2	–	((ES, C) + word) ← byte	
		A, ES:word[C]	4	2	5	A ← ((ES, C) + word)	
		ES:word[C], A	4	2	–	((ES, C) + word) ← A	
		ES:word[BC], #byte	5	2	–	((ES, BC) + word) ← byte	
		A, ES:word[BC]	4	2	5	A ← ((ES, BC) + word)	
		ES:word[BC], A	4	2	–	((ES, BC) + word) ← A	
		B, ES:!addr16	4	2	5	B ← (ES, addr16)	
		C, ES:!addr16	4	2	5	C ← (ES, addr16)	
		X, ES:!addr16	4	2	5	X ← (ES, addr16)	
	XCH	A, r Note 3	1 (r = X) 2 (other than r = X)	1	–	A ←→ r	
		A, saddr	3	2	–	A ←→ (saddr)	
		A, sfr	3	2	–	A ←→ sfr	
		A, !addr16	4	2	–	A ←→ (addr16)	
		A, [DE]	2	2	–	A ←→ (DE)	
		A, [DE + byte]	3	2	–	A ←→ (DE + byte)	
		A, [HL]	2	2	–	A ←→ (HL)	
		A, [HL + byte]	3	2	–	A ←→ (HL + byte)	
		A, [HL + B]	2	2	–	A ←→ (HL + B)	
		A, [HL + C]	2	2	–	A ←→ (HL + C)	

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.

3. Except r = A

Remarks 1. One instruction clock cycle is one cycle of the CPU clock (f_{CLK}) 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.

A.6 Debugging Tools (Software)

SM+ for 78K0R System simulator	SM+ for 78K0R is Windows-based software. It is used to perform debugging at the C source level or assembler level while simulating the operation of the target system on a host machine. Use of SM+ for 78K0R allows the execution of application logical testing and performance testing on an independent basis from hardware development, thereby providing higher development efficiency and software quality. SM+ for 78K0R should be used in combination with the device file (DF781188).
	Part number: μS××××SM781000
ID78K0R-QB Integrated debugger	This debugger supports the in-circuit emulators for the 78K0R microcontrollers. The ID78K0R-QB is Windows-based software. It has improved C-compatible debugging functions and can display the results of tracing with the source program using an integrating window function that associates the source program, disassemble display, and memory display with the trace result. It should be used in combination with the device file.
	Part number: μS××××ID78K0R-QB

Remark ×××× in the part number differs depending on the host machine and OS used.

μS××××SM781000
μS××××ID78K0R-QB

××××	Host Machine	OS	Supply Medium
AB17	PC-9800 series, IBM PC/AT compatibles	Windows (Japanese version)	CD-ROM
BB17		Windows (English version)	

Chapter	Classification	Function	Details of Function	Cautions	Page
Chapter 12	Soft	Serial interface IIC0	When STCEN = 1	Immediately after I^2C operation is enabled (IICE0 = 1), the bus released status (IICBSY = 0) is recognized regardless of the actual bus status. To generate the first start condition (STT0 (bit 1 of IIC control register 0 (IICC0)) = 1), it is necessary to confirm that the bus has been released, so as to not disturb other communications.	p.509 ☐
			If other I^2C communications are already in progress	IIf I^2C operation is enabled and the device participates in communication already in progress when the SDA0 pin is low and the SCL0 pin is high, the macro of I^2C recognizes that the SDA0 pin has gone low (detects a start condition). If the value on the bus at this time can be recognized as an extension code, $\overline{ACK}$ is returned, but this interferes with other I^2C communications. To avoid this, start I^2C in the following sequence.<1> Clear bit 4 (SPIE0) of IICC0 to 0 to disable generation of an interrupt request signal (INTIIC0) when the stop condition is detected. <2> Set bit 7 (IICE0) of IICC0 to 1 to enable the operation of I^2C. <3> Wait for detection of the start condition. <4> Set bit 6 (LREL0) of IICC0 to 1 before $\overline{ACK}$ is returned (4 to 80 clocks after setting IICE0 to 1), to forcibly disable detection.	p.509 ☐
			Setting transfer clock frequency	Determine the transfer clock frequency by using SMC0, CL01, CL00 (bits 3, 1, and 0 of IICL0), and CLX0 (bit 0 of IICX0) before enabling the operation (IICE0 = 1). To change the transfer clock frequency, clear IICE0 to 0 once.	p.509 ☐
			STT0, SPT0: Bits 1, 0 of IIC control register 0 (IICC0)	Setting STT0 and SPT0 (bits 1 and 0 of IICC0) again after they are set and before they are cleared to 0 is prohibited.	p.509 ☐
			Reserving transmission	When transmission is reserved, set SPIE0 (bit 4 of IICL0) to 1 so that an interrupt request is generated when the stop condition is detected. Transfer is started when communication data is written to IIC0 after the interrupt request is generated. Unless the interrupt is generated when the stop condition is detected, the device stops in the wait state because the interrupt request is not generated when communication is started. However, it is not necessary to set SPIE0 to 1 when MSTS0 (bit 7 of IICS0) is detected by software.	p.509 ☐
Chapter 14	Soft	DMA controller	DBCn: DMA byte count register n	Be sure to clear bits 15 to 10 to “0”.	p.552 ☐
				If the general-purpose register is specified or the internal RAM space is exceeded as a result of continuous transfer, the general-purpose register or SFR space are written or read, resulting in loss of data in these spaces. Be sure to set the number of times of transfer that is within the internal RAM space.	p.552 ☐
			DRCn: DMA operation control register n	The DSTn flag is automatically cleared to 0 when a DMA transfer is completed. Writing the DENn flag is enabled only when DSTn = 0. When a DMA transfer is terminated without waiting for generation of the interrupt (INTDMAn) of DMAn, therefore, set DSTn to 0 and then DENn to 0 (for details, refer to 14.5.5 Forcible termination by software).	p.556 ☐
			Holding DMA transfer pending by DWAITn	When DMA transfer is held pending while using both DMA channels, be sure to hold the DMA transfer pending for both channels (by setting DWAIT0 and DWAIT1 to 1). If the DMA transfer of one channel is executed while that of the other channel is held pending, DMA transfer might not be held pending for the latter channel.	p.570 ☐
			Forced Termination of DMA Transfer	In example 3, the system is not required to wait two clock cycles after DWAITn is set to 1. In addition, the system does not have to wait two clock cycles after clearing DSTn to 0, because more than two clock cycles elapse from when DSTn is cleared to 0 to when DENn is cleared to 0.	p.572 ☐

(2/5)

Page	Description	Classification
CHAPTER 7 REAL-TIME COUNTER (continuation)		
p.270	Change of description of **(7) Minute count register (MIN)**	(c)
p.270	Change of description of **(8) Hour count register (HOUR)**	(c)
p.275	Addition of description of DEV bit to **Figure 7-14. Format of Watch Error Correction Register (SUBCUD)**	(c)
p.277	Addition of **7.3 (17) Port mode register 1, 3 (PM1, PM3)**	(c)
p.278	Change of **Figure 7-19. Procedure for Starting Operation of Real-Time Counter** and addition of **Note**	(c)
p.283	Addition of **Caution** to **7.4.5 1 Hz output of real-time counter**	(c)
p.283	Change of **7.4.6 32.768 kHz output of real-time counter**	(c)
p.283	Change of **7.4.7 512 Hz, 16.384 kHz output of real-time counter**	(c)
CHAPTER 9 CLOCK OUTPUT/BUZZER OUTPUT CONTROLLER		
p.299	Change of **Remark** in **9.4.1 Operation as output pin**	(c)
p.299	Change of **Figure 9-4. Remote Control Output Application Example**	(c)
CHAPTER 10 A/D CONVERTER		
p.304	Change of **Table 10-2. Settings of ADCS and ADCE**	(c)
p.304	Change of **Figure 10-5. Timing Chart When A/D voltage Comparator Is Used**	(c)
p.328	Change of **10.7 Cautions for A/D Converter (1) Operating current in STOP mode**	(c)
p.332	Addition of **10.7 (13) Starting the A/D converter**	(c)
CHAPTER 11 SERIAL ARRAY UNIT		
p.345	Change of MDmn0 bit in **Figure 11-6. Format of Serial Mode Register mn (SMRmn) (2/2)**	(c)
p.347	Addition of **Note** to **Figure 11-7. Format of Serial Communication Operation Setting Register mn (SCRmn) (2/3)**	(c)
p.349	Addition of **Caution** to **Figure 11-8. Format of Serial Data Register mn (SDRmn)**	(c)
p.359	Change of description of **Figure 11-17. Format of Input Switch Control Register (ISC)**	(a)
p.376	Change of interrupt in **11.5.2 Master reception**	(c)
p.377	Change of **Figure 11-32. Example of Contents of Registers for Master Reception of 3-Wire Serial I/O (CSI00, CSI10)**	(c)
p.379	Change of **Figure 11-35. Procedure for Resuming Master Reception**	(c)
p.381	Change of **Figure 11-37. Flowchart of Master Reception (in Single-Reception Mode)**	(c)
p.382	Addition of **Figure 11-38. Timing Chart of Master Reception (in Continuous Reception Mode) (Type 1: DAP0n = 0, CKP0n = 0)**	(c)
p.383	Addition of **Figure 11-39. Flowchart of Master Reception (in Continuous Reception Mode)**	(c)
p.396	Change of **Figure 11-51. Procedure for Resuming Slave Transmission**	(b)
p.398	Change of **Figure 11-53. Flowchart of Slave Transmission (in Single-Transmission Mode)**	(c)
p.400	Change of **Figure 11-55. Flowchart of Slave Transmission (in Continuous Transmission Mode)**	(c)
p.402	Change of **Figure 11-56. Example of Contents of Registers for Slave Reception of 3-Wire Serial I/O (CSI00, CSI10)**	(c)
p.404	Change of **Figure 11-59. Procedure for Resuming Slave Reception**	(c)

Remark "Classification" in the above table classifies revisions as follows.

(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents

(5/5)

Page	Description	Classification
CHAPTER 28 ELECTRICAL SPECIFICATIONS ((A) GRADE PRODUCTS) (continuation)		
pp.774 to 777	Addition of **Remark** to Supply current in **DC Characteristics**	(c)
p.785	Change of **(b) During communication at same potential (CSI mode) (master mode, $\overline{\text{SCKp}}$... internal clock output)** in **Serial interface: Serial array unit**	(b)
p.786	Change of **(c) During communication at same potential (CSI mode) (slave mode, $\overline{\text{SCKp}}$... external clock input)** in **Serial interface: Serial array unit**	(b)
p.788	Addition of **Note** to **(d) During communication at same potential (simplified I^2C mode)** in **Serial interface: Serial array unit**	(c)
pp.794, 795	Change of **(f) During communication at different potential (2.5 V, 3 V) (CSI mode) (master mode, $\overline{\text{SCK10}}$... internal clock output)** in **Serial interface: Serial array unit**	(b)
p.797	Change of **(g) During communication at different potential (2.5 V, 3 V) (CSI mode) (slave mode, $\overline{\text{SCK10}}$... external clock input)** in **Serial interface: Serial array unit**	(b)
p.800	Addition of **Note** to **(h) During communication at different potential (2.5 V, 3 V) (simplified I^2C mode)** in **Serial interface: Serial array unit**	(b)
CHAPTER 29 PACKAGE DRAWINGS		
p.814	Addition of package drawing of **64-PIN PLASTIC FBGA (6x6)**	(d)
CHAPTER 30 RECOMMENDED SOLDERING CONDITIONS		
p.816	Addition of Surface Mounting Type Soldering Conditions of **64-pin plastic FBGA(6 × 6)**	(d)

Remark "Classification" in the above table classifies revisions as follows.

(a): Error correction, (b): Addition/change of specifications, (c): Addition/change of description or note, (d): Addition/change of package, part number, or management division, (e): Addition/change of related documents