TOSHIBA

16 Bit Microcontroller

TLCS-900/L1 Series

TMP91FU62FG
TMP91FU62DFG

Revision 1.1

TOSHIBA CORPORATION

The information contained herein is subject to change without notice.

TOSHIBA is continually working to improve the quality and reliability of its products. Nevertheless, semiconductor devices in general can malfunction or fail due to their inherent electrical sensitivity and vulnerability to physical stress.
It is the responsibility of the buyer, when utilizing TOSHIBA products, to comply with the standards of safety in making a safe design for the entire system, and to avoid situations in which a malfunction or failure of such TOSHIBA products could cause loss of human life, bodily injury or damage to property.
In developing your designs, please ensure that TOSHIBA products are used within specified operating ranges as set forth in the most recent TOSHIBA products specifications.
Also, please keep in mind the precautions and conditions set forth in the “Handling Guide for Semiconductor Devices,” or “TOSHIBA Semiconductor Reliability Handbook” etc.

The TOSHIBA products listed in this document are intended for usage in general electronics applications (computer, personal equipment, office equipment, measuring equipment, industrial robotics, domestic appliances, etc.).
These TOSHIBA products are neither intended nor warranted for usage in equipment that requires extraordinarily high quality and/or reliability or a malfunction or failure of which may cause loss of human life or bodily injury (“Unintended Usage”). Unintended Usage include atomic energy control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, combustion control instruments, medical instruments, all types of safety devices, etc. Unintended Usage of TOSHIBA products listed in this document shall be made at the customer's own risk.

The products described in this document shall not be used or embedded to any downstream products of which manufacture, use and/or sale are prohibited under any applicable laws and regulations.

The information contained herein is presented only as a guide for the applications of our products. No responsibility is assumed by TOSHIBA for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of TOSHIBA or others.

Please contact your sales representative for product-by-product details in this document regarding RoHS comaptibility. Please use these products in this document in compliance with all applicable laws and regulations that regulate ths inclusion or use of controlled substances. Toshiba assumes no liability for damage or losses occurring as a result of noncompliance with applicable laws and regulations.

Revision History

Date	Revision	
2007/01/18	0.2	TENTATIVE
2007/04/27	0.4	Table 1-1 Pin Names and Functions ・WAIT pin deletion. ・HV-monitor → EMU0 ・P00-P07 large-current port
		2.1 RESET 10 system clocks 16us → 1us
		2.3.4 Prescaler Clock Controller
		Table 4-1 Port Functions
		Table 4-2 I/O Port Setting List
		4.3 Port3 (P30 to P33) Deleted The input function of wait control(WAIT) Deleted Note2.
		P40 to P43 function Table.
		4.9.1 Port 90 (TXD0/RXD0), 93 (TXD1/RXD0) 4.9.2 Port91(RXD0/TXD0), 94 (RXD1/TXD1)
		PB0 to PB2 function Table.
		4.12 Open-drain Control
		4.13 Serial channel pin change Control
		14.1 Absolute Maximum Ratings
		Table 2-7 Source of Halt State Clearance and Halt Clearance Operation
		Table 4-2 I/O Port Setting List (Port B)
		4.1 Port 0 (P00 to P07)
		4.2 Port 1 (P10 to P17)
		4.4 Port 4 (P40 to P43)
		Figure 4-12 Port72
		4.13 Serial channel pin change/ Open-drain output Control
		Table 6-1 Registers and Pins for TMRB
		9. 10-bit AD Converter (ADC) VREFH → AVCC
		Figure 9-4 Analog Input Voltage and AD Conversion Result (Typ.)
		13.6.10 Programming the Flash Memory by the Internal CPU ・Read Values in Product ID Mode ・Example: Program to be loaded and executed in RAM
		14.2 DC Electrical Characteristics Low-level output current
		14.3 AD Conversion Characteristics Deleted Analog current for analog reference voltage
		15.Table of SFR's Deleted P4FC register

Date	Revision	
2007/06/07	0.5	14.1 Absolute Maximum Ratings I_{OL}, I_{OH} is corrected
		14.2 DC Electrical Characteristics I_{CC}, I_{DDP-P} is corrected
2007/8/27	1.0	DMAR register (89H) is corrected by RWM prohibition.
		17.2 Points of note j. Releasing the HALT mode by requesting an interruption is deleted.
		2.3.2 Note3 is added
		7.2.1 Plescaler is corrected, and Table 7-2 is corrected
		7.3 Note2 and Note3 are added
		17.2 Points of note j.Clocks for serial channels (SIO) is added
2007/10/10	1.1	6.3 SFR 15. Table of SFR's TB0FFCR, TB1FFCR, TB2FFCR and TB3FFCR register is corrected.

CMOS 16 Bit Microcontroller

TMP91FU62FG/DFG

Product No.	ROM (Flash ROM)	RAM	Package
TMP91FU62FG	96K bytes	4K bytes	LQFP80-P-1212-0.50E
TMP91FU62DFG			QFP80-P-1420-0.80B

1.1 Features

- High-speed 16-bit CPU (900/L1 CPU)
  - Instruction mnemonics are upward-compatible with TLCS-900,900/H,900/L
  - 16 Mbytes of linear address space
  - General-purpose registers and register banks
  - 16-bit multiplication and division instructions; bit transfer and arithmetic instructions
  - Micro DMA: 4 channels (800ns/2 bytes at 20MHz)
- Minimum instruction execution time:200ns (at 20MHz)
- Built-in memory
  - ROM: 96K bytes (Flash ROM)
  - RAM: 4K bytes
- 8-bit timers: 4 channels
- 16-bit timers: 4 channels
- General-purpose serial interface: 4 channels
  - UART/Synchronous mode: 3 channels
  - I^2C bus mode: 1 channels
- 10-bit AD converter (Built-in Sample hold circuit): 16 channels
- Special timer for CLOCK
- Watchdog timer
- Program patch logic: 6 banks

20070701-EN

- Interrupts: 48 interrupts
  - 9 CPU interrupts: Software interrupt instruction and illegal instruction
  - 30 internal interrupts: 7 priority levels are selectable
  - 9 external interrupts: 7 priority levels are selectable (among 1 interrupts are selectable edge mode)
- Input/output ports: 69 pins
- Standby function: Three HALT modes: IDLE2 (Programmable), IDLE1 and STOP
- Clock controller
  - Clock gear function: Select a High-frequency clock fc/1 to fc/16
  - Oscillator for CLOCK (fs = 32.768 kHz)
- Operating voltage

  Flash read operation

  > Vcc=4.5 V - 5.5 V (fc max = 20MHz)

  Flash write/erase operation

  > Vcc=4.75 V - 5.25 V (fc max = 20MHz)
- Package
  - LQFP80-P-1212-0.50E (TMP91FU62FG)
  - QFP80-P-1420-0.80B (TMP91FU62DFG)

1.2 Pin Assignment Diagram

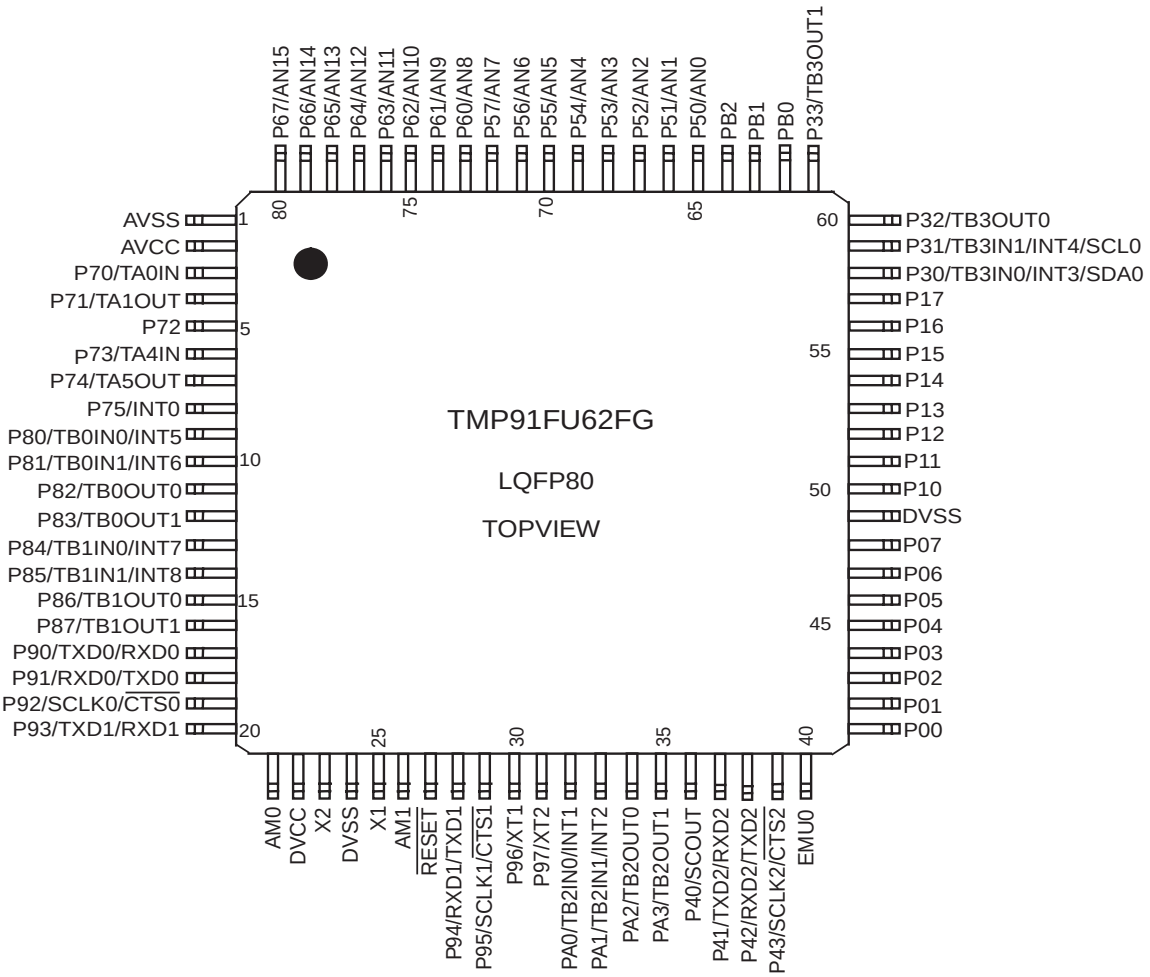


Figure 1-1 Pin Assignment(TMP91FU62FG)

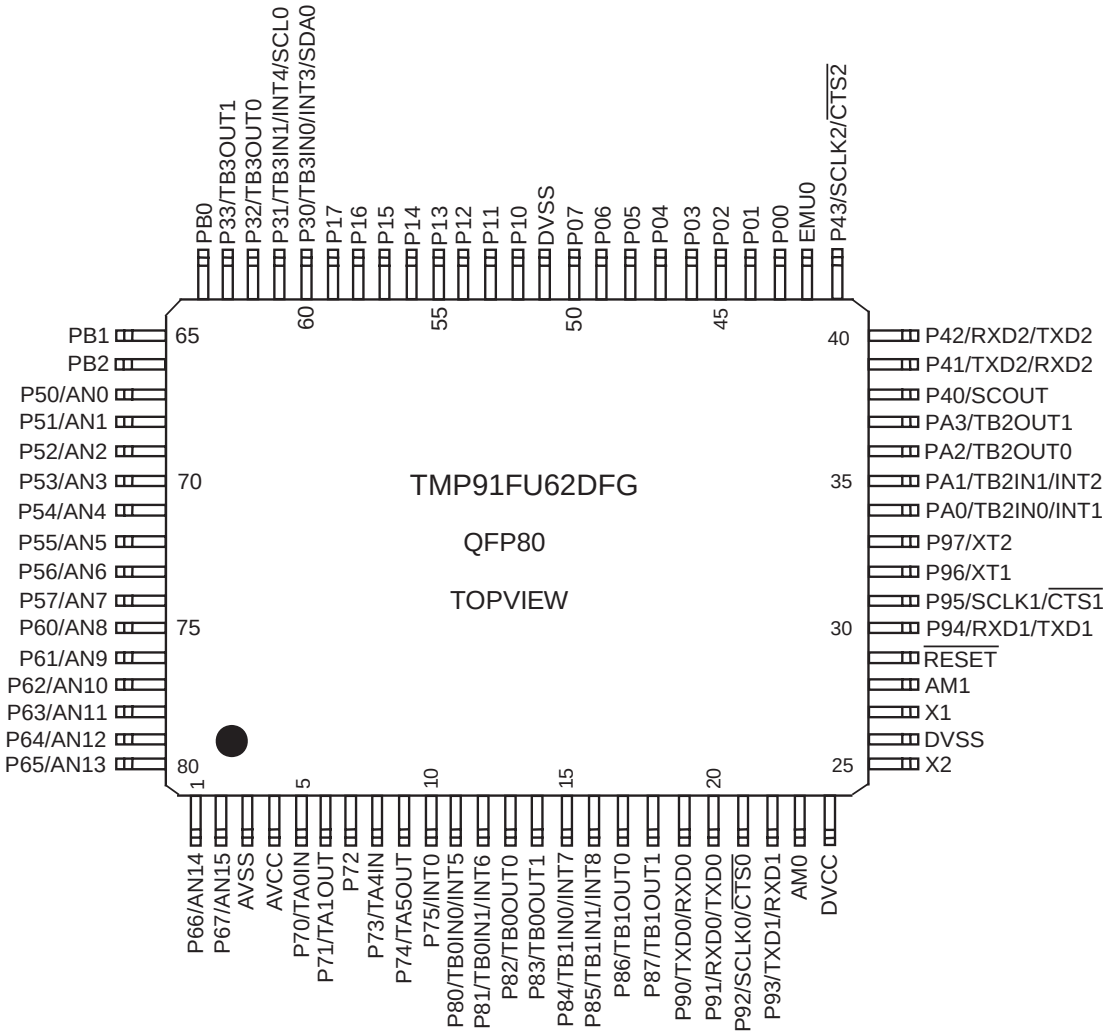


Figure 1-2 Pin Assignment(TMP91FU62DFG)

1.3 Block Diagram

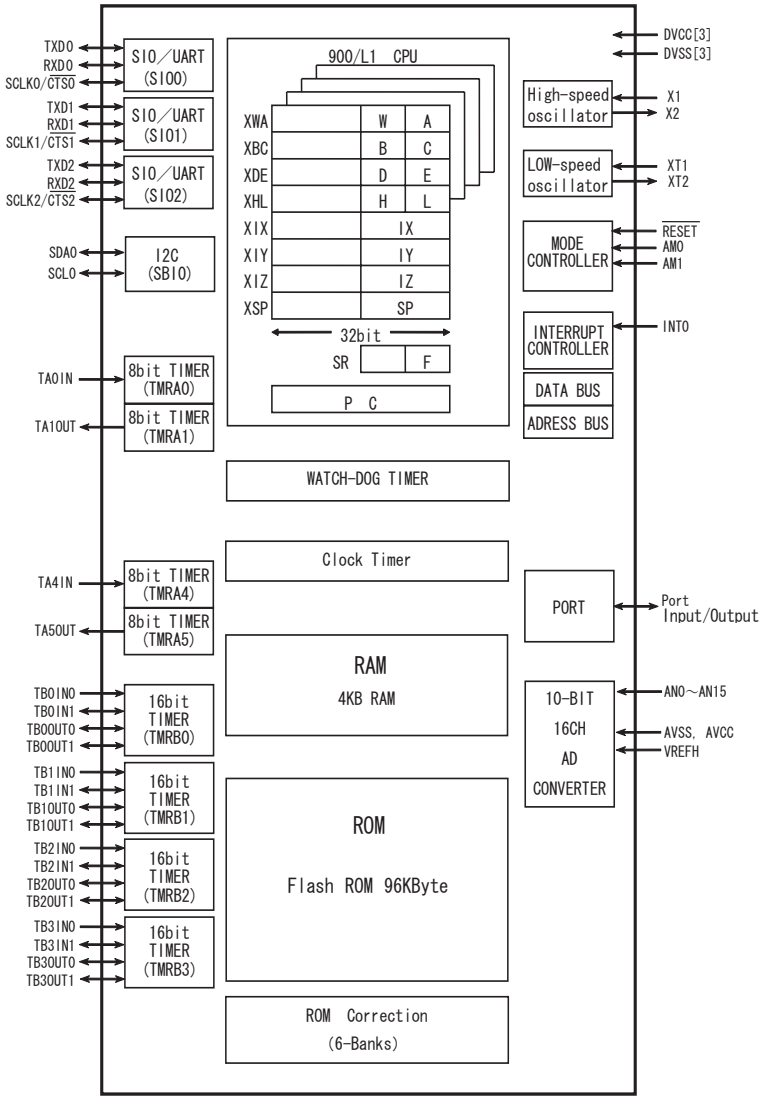


Figure 1-3 Block Diagram

1.4 Pin Names and Functions

Table 1-1 Pin Names and Functions(1/3)

Pin Name	Pin Number	Input / Output	Functions
P00-P07	8	IO	Port 0: I/O port that allows I/O to be selected at the bit level (large-current port)
P10-P17	8	IO	Port 1: I/O port that allows I/O to be selected at the bit level
P30 TB3IN0 INT3 SDA0	1	IO I I IO	Port 30: I/O port 16-bit timer 3 input 0:Timer B3 count/capture trigger Input 0 Interrupt Request Pin 3: Interrupt request pin with programmable rising edge / falling edge. Serial bus interface data 0 in I2C bus Mode.
P31 TB3IN1 INT4 SCL0	1	IO I I IO	Port 31: I/O port 16-bit timer 3 input 1:Timer B3 count/capture trigger Input 1 Interrupt Request Pin 4: Interrupt request on rising edge Serial bus interface clock 0 in I2C bus Mode.
P32 TB3OUT0	1	IO O	Port 32: I/O port 16-bit timer 3 output 0: Timer B3 Output 0
P33 TB3OUT1	1	IO O	Port 33: I/O port 16-bit timer 3 output 1: Timer B3 Output 1
P40 SCOUT	1	IO O	Port 40: I/O port (with pull-up resistor) System Clock Output: Outputs f_{SYS} or fs clock.
P41 TXD2 RXD2	1	IO O I	Port 41: I/O port (with pull-up resistor) Serial Send Data 2 Serial Receive Data 2
P42 RXD2 TXD2	1	IO I O	Port 42: I/O port (with pull-up resistor) Serial Receive Data 2 Serial Send Data 2
P43 SCLK2 $\overline{CTS2}$	1	IO IO I	Port 43: I/O port (with pull-up resistor) Serial Clock I/O 2 Serial Data Send Enable 2 (Clear to Send)
P50-57 AN0-AN7	8	IO I	Port 5: I/O port Analog input: Pin used to input to AD converter
P60-67 AN8-AN15	8	IO I	Port 6: I/O port Analog input: Pin used to input to AD converter
P70 TA0IN	1	IO I	Port 70: I/O port 8-bit timer 0 input: Timer A0 Input
P71 TA1OUT	1	IO O	Port 71: I/O port 8-bit timer 1 output:Timer A1 Output
P72	1	IO	Port 72: I/O port
P73 TA4IN	1	IO I	Port 73: I/O port 8-bit timer 4 input: Timer A4 Input
P74 TA5OUT	1	IO O	Port 74: I/O port 8-bit timer 5 output:Timer A5 Output
P75 INT0	1	IO I	Port 75: I/O port Interrupt Request Pin 0: Interrupt request pin with programmable level / rising edge / falling edge.
P80 TB0IN0 INT5	1	IO I I	Port 80: I/O port 16-bit timer 0 input 0:Timer B0 count/capture trigger Input 0 Interrupt Request Pin 5: Interrupt request pin with programmable rising edge / falling edge.

Table 1-1 Pin Names and Functions(2/3)

Pin Name	Pin Number	Input / Output	Functions
P81 TB0IN1 INT6	1	IO I I	Port 81: I/O port 16-bit timer 0 input 1:Timer B0 count/capture trigger Input 1 Interrupt Request Pin 6: Interrupt request on rising edge
P82 TB0OUT0	1	IO O	Port 82: I/O port 16-bit timer 0 output 0: Timer B0 Output 0
P83 TB0OUT1	1	IO O	Port 83: I/O port 16-bit timer 0 output 1: Timer B0 Output 1
P84 TB1IN0 INT7	1	IO I I	Port 84: I/O port 16-bit timer 1 input 0:Timer B1 count/capture trigger Input 0 Interrupt Request Pin 7: Interrupt request pin with programmable rising edge / falling edge.
P85 TB1IN1 INT8	1	IO I I	Port 85: I/O port 16-bit timer 1 input 1:Timer B1 count/capture trigger Input 1 Interrupt Request Pin 8: Interrupt request on rising edge
P86 TB1OUT0	1	IO O	Port 86: I/O port 16-bit timer 1 output 0: Timer B1 Output 0
P87 TB1OUT1	1	IO O	Port 87: I/O port 16-bit timer 1 output 1: Timer B1 Output 1
P90 TXD0 RXD0	1	IO O I	Port 90: I/O port Serial Send Data 0 Serial Receive Data 0
P91 RXD0 TXD0	1	IO I O	Port 91: I/O port Serial Receive Data 0 Serial Send Data 0
P92 SCLK0 $\overline{\text{CTS0}}$	1	IO IO I	Port 92: I/O port Serial Clock I/O 0 Serial Data Send Enable 0 (Clear to Send)
P93 TXD1 RXD1	1	IO O I	Port 93: I/O port Serial Send Data 1 Serial Receive Data 1
P94 RXD1 TXD1	1	IO I O	Port 94: I/O port Serial Receive Data 1 Serial Send Data 1
P95 SCLK1 $\overline{\text{CTS1}}$	1	IO IO I	Port 95: I/O port Serial Clock I/O 1 Serial Data Send Enable 1 (Clear to Send)
P96 XT1	1	IO I	Port 96: I/O port Low-frequency oscillator connection pin
P97 XT2	1	IO O	Port 97: I/O port Low-frequency oscillator connection pin
PA0 TB2IN0 INT1	1	IO I I	Port A0: I/O port 16-bit timer 2 input 0:Timer B2 count/capture trigger Input 0 Interrupt Request Pin 1: Interrupt request pin with programmable rising edge / falling edge.
PA1 TB2IN1 INT2	1	IO I I	Port A1: I/O port 16-bit timer 2 input 1:Timer B2 count/capture trigger Input 1 Interrupt Request Pin 2: Interrupt request on rising edge
PA2 TB2OUT0	1	IO O	Port A2: I/O port 16-bit timer 2 output 0: Timer B2 Output 0
PA3 TB2OUT1	1	IO O	Port A3: I/O port 16-bit timer 2 output 1: Timer B2 Output 1

Table 1-1 Pin Names and Functions(3/3)

Pin Name	Pin Number	Input / Output	Functions
PB0-PB2	3	IO	Port B: I/O port that allows I/O to be selected at the bit level
AM0-1	2	I	Operation mode:Fixed to AM1 "1", AM0 "1". Single Boot mode:Fixed to AM1 "0", AM0 "1". Programmer mode:Fixed to AM1 "1", AM0 "0".
EMU0	1	O	Open pin
$\overline{\text{RESET}}$	1	I	Reset: initializes TMP91FU62. (with pull-up resistor)
AVCC	1		Power supply pin for AD converter
AVSS	1		GND pin for AD converter (0 V)
X1/X2	2	IO	High frequency oscillator connection pins
DVCC	3		Power supply pins (All DVCC pins should be connected with the power supply pin.)
DVSS	3		GND pins (0 V) (All DVSS pins should be connected with the GND (0V) pin.)

Note: All pins that have built-in pull-up resistors (other than the $\overline{\text{RESET}}$ pin) can be disconnected from the built-in pull-up resistor by software.

2. CPU

The TMP91FU62 incorporates a high-performance 16-bit CPU (The 900/L1-CPU). For CPU operation, see the "TLCS-900/L1 CPU".

The following describe the unique function of the CPU used in the TMP91FU62; these functions are not covered in the TLCS-900/L1 CPU section.

2.1 RESET

When resetting the TMP91FU62 microcontroller, ensure that the power supply voltage is within the operating voltage range, and that the internal high-frequency oscillator has stabilized. Then hold the $\overline{\text{RESET}}$ input to low level at least for 10 system clocks (1us at 20 MHz).

Thus, when turn on the switch, be set to the power supply voltage is within the operating voltage range, and that the internal high-frequency oscillator has stabilized. Then hold the $\overline{\text{RESET}}$ input to Low level at least for 10 system clocks.

It means that the system clock mode f_{SYS} is set to fc/2.

When the reset is accept, the CPU:

1. Sets as follows the program counter (PC) in accordance with the reset vector stored

   at address FFFF00H to FFFF02H:

   - PC (7:0) <- Value at FFFF00H address
   - PC (15:8) <- Value at FFFF01H address
   - PC (23:16) <- Value at FFFF02H address

2. Sets the stack pointer (XSP) to 100H.

3. Sets bits<IFF2:0> of the status register (SR) to 111 (Sets the interrupt level mask register to level 7).

4. Sets the <MAX> bit of the status register (SR) to 1 (MAX mode).

5. Clears bits<RFP2:0> of the status register (SR) to 000 (Sets the register bank to 0).

When reset is released, the CPU starts executing instructions in accordance with the program counter settings. CPU internal registers not mentioned above do not change when the reset is released.

When the reset is accepted, the CPU sets internal I/O, ports, and other pins as follows.

1. Initializes the internal I/O registers.

2. Sets the port pins, including the pins that also act as internal I/O, to general-purpose input or output

   port mode.

3. Sets ALE pin to high impedance.

Note 1: The CPU internal register (except to PC, SR, XSP in CPU) and internal RAM data do not change by resetting.
Note 2: It is necessary to re-set up a stack pointer XSP by the user program.

Figure 2-1 is a reset timing chart of the TMP91FU62.

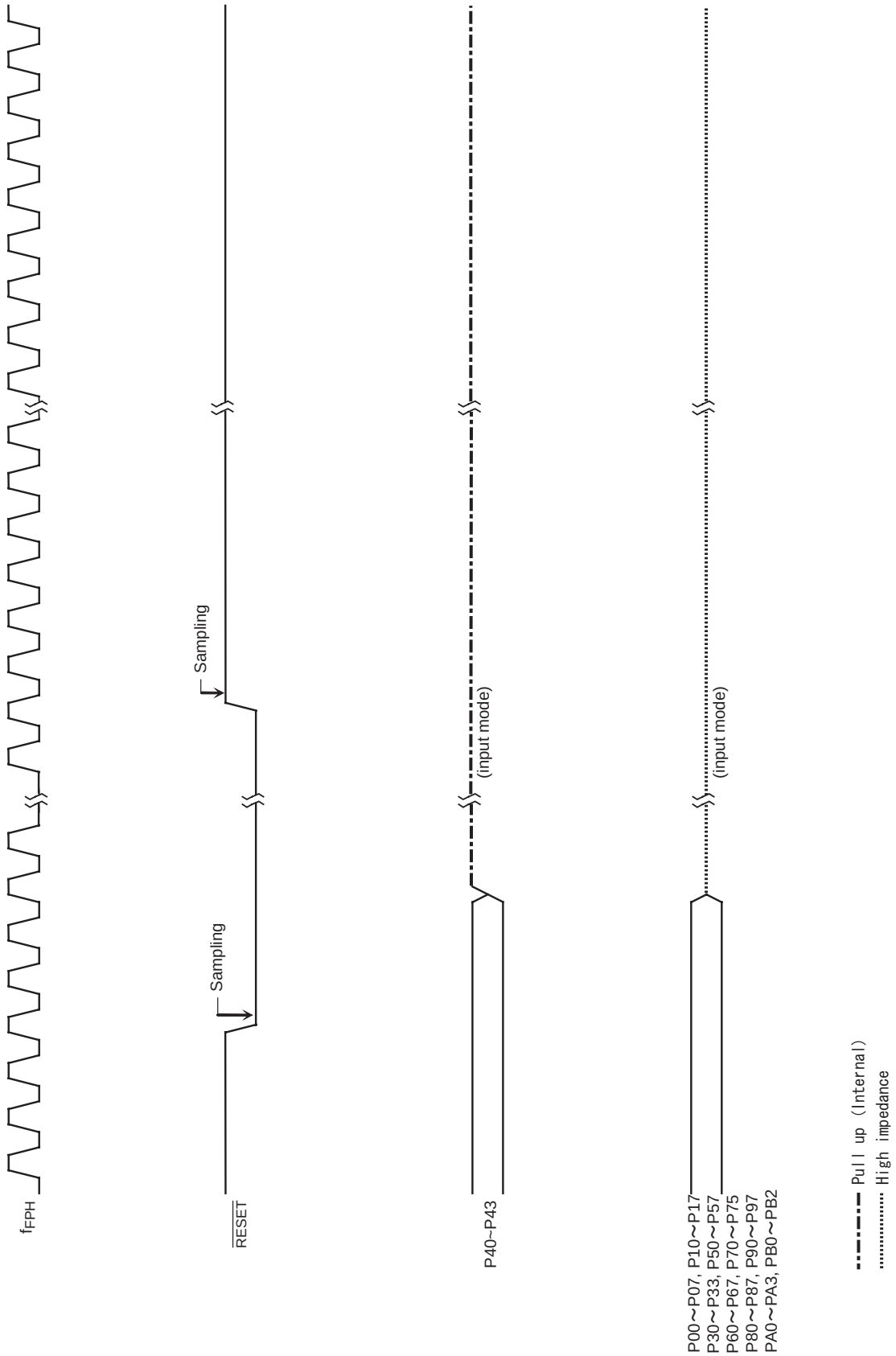


Figure 2-1 TMP91FU62 Reset Timing Chart

2.2 Memory Map

Figure 2-2 is a memory map of the TMP91FU62.

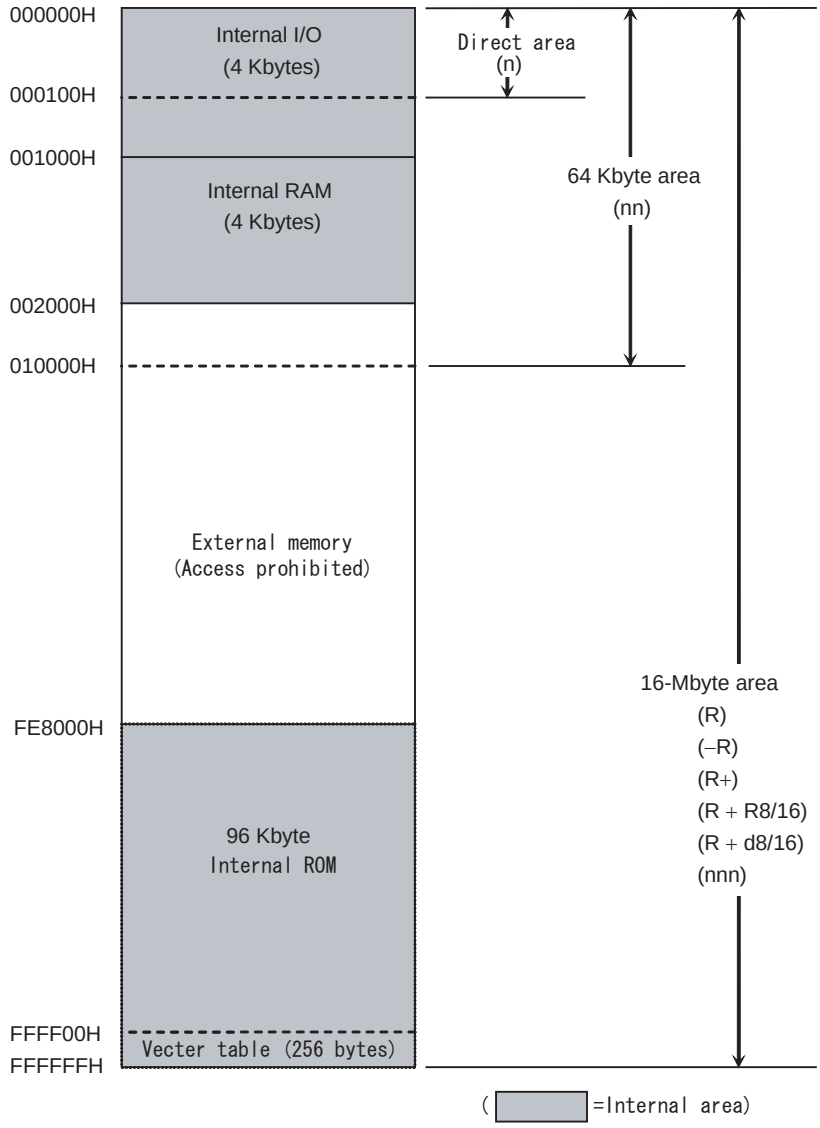


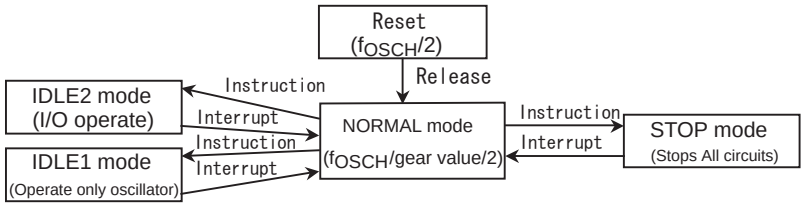
Figure 2-2 TMP91FU62 Memory Map

2.3 System Clock Function and Standby Control

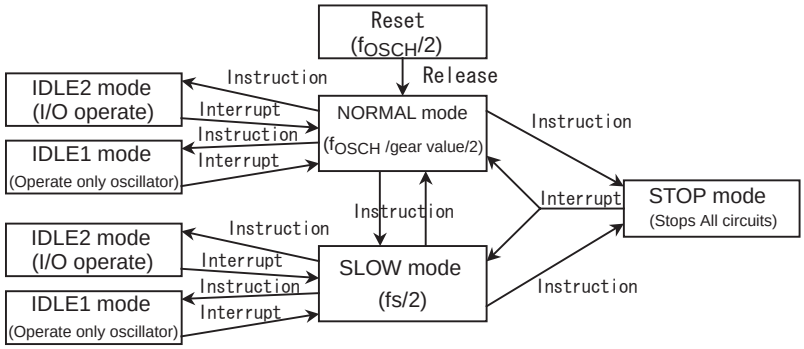
TMP91FU62 contains a clock gear, stand-by controller and noise-reduction circuit. It is used for low-noise systems.

The clock operating modes are as follows: (a) Single clock mode (X1 and X2 pins only), (b) Dual clock mode (X1,X2,XT1 and XT2 pins).

Figure 2-3 shows a transition figure.



(a) Single clock mode transition figure



(b) Dual clock mode transition figure

Figure 2-3 TMP91FU62 Clock Operating Mode

Note: The clock frequency input from the X1 and X2 pins is called f_{OSCH} and the clock frequency input from the XT1 and XT2 pins is called fs. The clock frequency selected by SYSCR1<SYSCK> is called f_{FPH}. The system clock $f_{SYS}$is defined as the divided clock of f_{FPH}, and one cycle of f_{SYS} is regret to as one state.

2.3.1 Block Diagram of System Clock

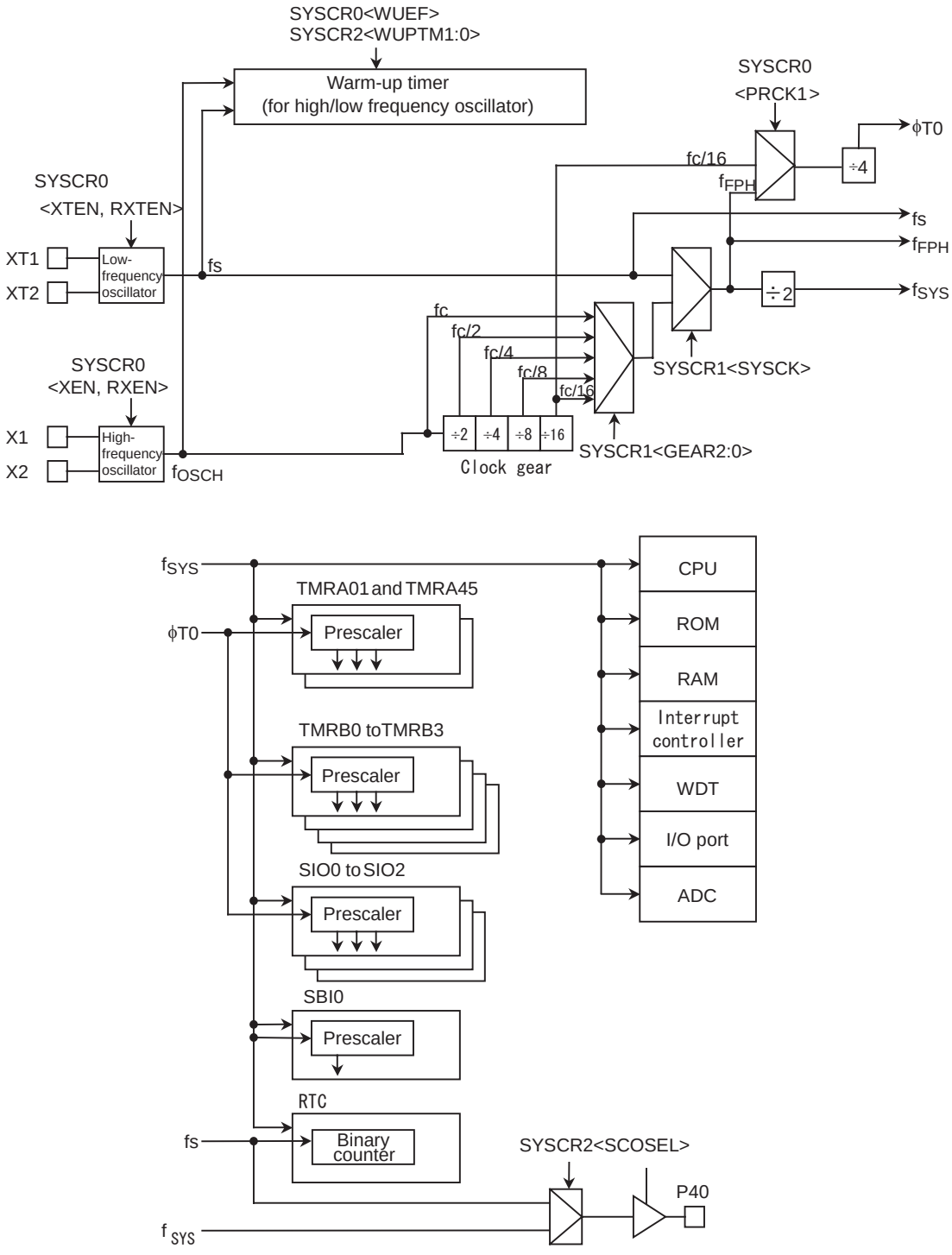


Figure 2-4 Block Diagram of System Clock

2.3.2 SFR

Table 2-1 SFR for System Clock

		7	6	5	4	3	2	1	0
SYSCR0 (00E0H)	Bit Symbol	XEN	XTEN	RXEN	RXTEN	RSYSCK	WUEF	PRCK1	–
	Read/Write	R/W							–
	After reset	1	0	1	0	0	0	0	–
	Function	High-frequency oscillator 0:Stop 1:Oscillation	Low-frequency oscillator 0:Stop 1:Oscillation	High-frequency oscillator (fc) after release of STOP mode 0:Stop 1:Oscillation	Low-frequency oscillator (fs) after release of STOP mode 0:Stop 1:Oscillation	Selects clock after release of STOP mode 0:fc 1:fs	Warm-up timer control 0 Write: Don't care 1 Write: Start warm-up 0 Read: End warm-up 1 Read: Do not end warm-up	Select pres-caler clock 0:f_{FPH} 1:fc/16	
SYSCR1 (00E1H)	Bit Symbol	–	–	–	–	SYSCK	GEAR2	GEAR1	GEAR0
	Read/Write	–	–	–	–	R/W			
	After reset	–	–	–	–	0	0	0	0
	Function	–	–	–	–	Select sys-tem clock 0: fc 1: fs	Select gear value of high frequency (fc) 000:fc 001:fc/2 010:fc/4 011:fc/8 100:fc/16 101:reserved 110:reserved 111:reserved		
SYSCR2 (00E2H)	Bit Symbol	–	SCOSEL	WUPTM1	WUPTM0	HALTM1	HALTM0	–	DRVE
	Read/Write	–	R/W					–	R/W
	After reset	–	0	1	0	1	1	–	0
	Function	–	Select SCOUT 0:fs 1:f_{SYS}	Select warm-up time for oscillator 00:2^{18}/inputted frequency 01:2^{8}/inputted frequency 10:2^{14}/inputted frequency 11:2^{16}/inputted frequency		HALT mode 00:reserved 01:STOP mode 10:IDLE1 mode 11:IDLE2 mode		–	Pin state control in STOP mode 0: I/O off 1: Remains the state before HALT

Note 1: "-" = Don't care

Note 2: SYSCR0<bit0>,SYSCR1<bit 7:4>,SYSCR2<bit7,bit1> are read as undefined value.

Note 3: As for the serial channels SIO0, SIO1 and SIO2, a baud rate generator is unavailable as an input clock of an I/O interface and a clock for a serial transfer if a prescaler clock is set to fc/16 when SYSCR0<PRCK1> is "1".

2.3.3 System Clock Controller

The system clock controller generates the system clock signal (f_{SYS}) for the CPU core and internal I/O.It contains two oscillation circuits and a clock gear circuit for high-frequency (fc) operation. The register SYSCR1<SYSCK> changes the system clock to either fc or fs, SYSCR0<XEN> and SYSCR0<XTEN> control enabling and disabling of each oscillator, and SYSCR1<GEAR2:0> sets the high-frequency clock gear to either 1, 2, 4, 8 or 16 (fc, fc/2, fc/4, fc/8 or fc/16). These functions can reduce the power consumption of the equipment in which the device is installed.

The combination of settings <XEN> = "1", <XTEN> = "0", <SYSCK> = "0" and <GEAR2:0> = "000" will cause the system clock (f_{SYS}) to be set to fc/2 (=fc x 1/2) after a Reset. For example, f_{SYS} is set to 8 MHz when the 16 MHz oscillator connected to the X1 and X2 pins.

(1) Switching from NORMAL mode to SLOW mode

When the resonator is connected to the X1 and X2 pins, or to the XT1 and XT2 pins, the warm-up timer can be used to change the operation frequency after stable oscillation has been attained.

The warm-up time can be selected using SYSCR2<WUPTM1:0>.

This warm-up timer can be programmed to start and stop as shown in the following examples 1 and 2.

Table 2-2 shows the warm-up time.

Note 1: When using an oscillator (other than a resonator) with stable oscillation, a warm-up timer is not needed.

Note 2: The warm-up timer is operated by an oscillation clock. Hence, there may be some variation in warm-up time.

Note 3: Note of using low-frequency oscillator

When connect low-frequency oscillator to ports 96 and 97, need below setting for cut consumption power.

(Case of resonators)

Set P9CR<P96C, P97C> = "11", P9<P96:97> = "00"

(Case of oscillator)

Set P9CR<P96C, P97C> = "11", P9<P96:97> = "10"

Table 2-2 Warm-up Times (when changing clock)

Select Warm-up Time SYSCR2<WUPTM1:0>	Change to NORMAL (fc)	Change to SLOW (fs)
01(2^8/frequency)	12.8[us]	7.8[ms]
10(2^{14}/frequency)	0.819[ms]	500[ms]
11(2^{16}/frequency)	3.277[ms]	2000[ms]
00(2^{18}/frequency)	13.107[ms]	8000[ms]

Note: At f_{OSCH}=20MHz, fs=32.768kHz

Example 1:

Changing from high frequency (fc) to low frequency (fs).

```
SYSCR0   EQU   00E0H
SYSCR1   EQU   00E1H
SYSCR2   EQU   00E2H
         LD    (SYSCR2),X-11--X-B   ;  Sets warm-up time to 2^16/fs.
         SET   6,(SYSCR0)           ;  Enables low-frequency oscillation.
         SET   2,(SYSCR0)           ;  Clears and starts warm-up timer.
WUP:     BIT   2,(SYSCR0)           ;
         JR    NZ,WUP               ;  Detects stopping of warm-up timer.
         SET   3,(SYSCR1)           ;  Changes fSYS from fc to fs.
         RES   7,(SYSCR0)           ;  Disables high-frequency oscillation.
```

Note: X: Don't care, -:No change

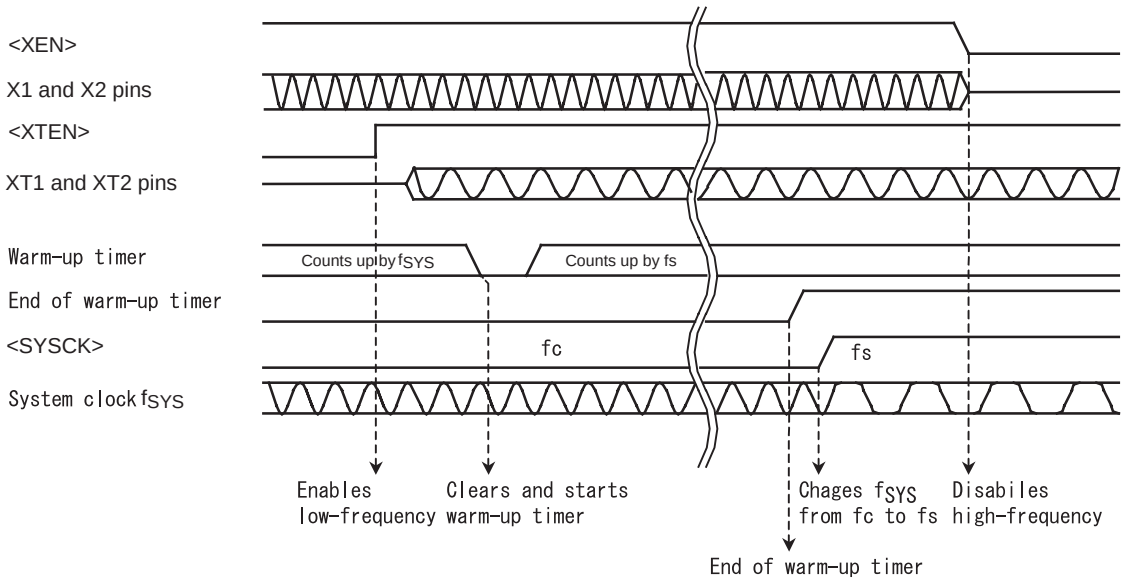


Figure 2-5 Changing from high frequency (fc) to low frequency (fs)

Example 2:

Changing from low frequency (fs) to high frequency (fc).

```
SYSCR0    EQU    00E0H
SYSCR1    EQU    00E1H
SYSCR2    EQU    00E2H
          LD     (SYSCR2),X-10--X-B   ;  Sets warm-up time to 2^14/fc.
          SET    7,(SYSCR0)           ;  Enables high-frequency oscillation.
          SET    2,(SYSCR0)           ;  Clears and starts warm-up timer.
WUP:      BIT    2,(SYSCR0)           ;
          JR     NZ,WUP               ;  Detects stopping of warm-up timer.
          RES    3,(SYSCR1)           ;  Changes f_SYS from fs to fc
          RES    6,(SYSCR0)           ;  Disables low-frequency oscillation.
```

Note: X: Don't care, -:No change

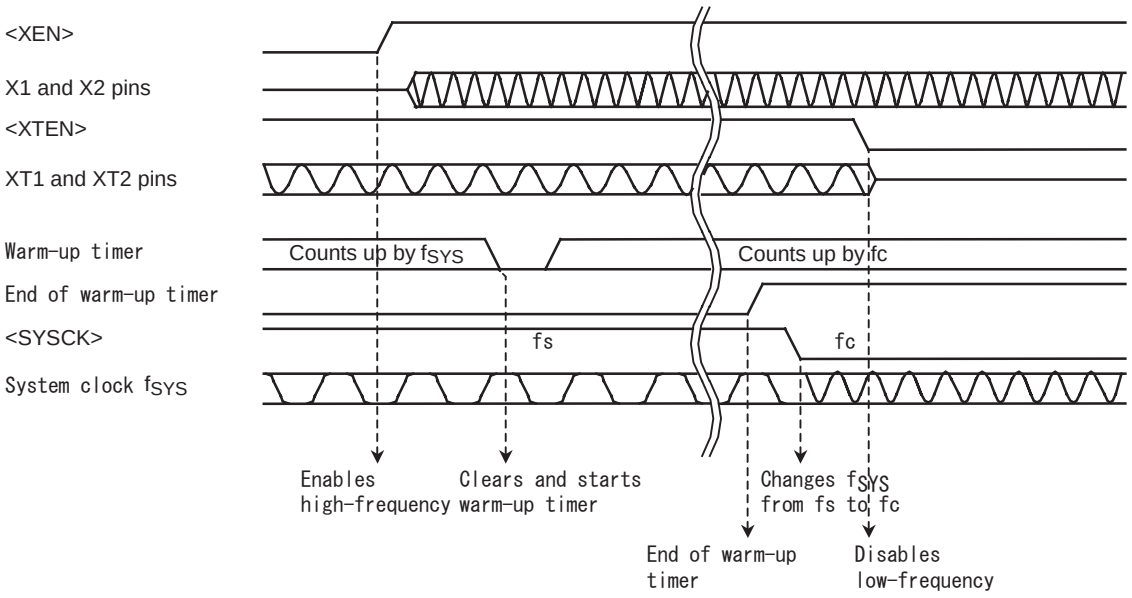


Figure 2-6 Changing from low frequency (fs) to high frequency (fc)

(2) Clock gear controller

When the high-frequency clock fc is selected by setting SYSCR1<SYSCK> = "0", f_{FPH} is set according to the contents of the clock gear select register SYSCR1<GEAR2:0> to either fc, fc/2, fc/4, fc/8 or fc/16. Using the clock gear to select a lower value of f_{FPH} reduces power consumption.

Below show example of changing clock gear.

Example 3:

Changing to a clock gear

```
SYSCR1  EQU   00E1H
        LD    (SYSCR1),XXXX0000B    ;  Changes fSYS to fc/2.
```

X:Don't care

(Clock gear changing)

To change the clock gear, write the register value to the SYSCR1<GEAR2:0> register. It is necessary the warm-up time until changing after writing the register value.

There is the possibility that the instruction next to the clock gear changing instruction is executed by the clock gear before changing. To execute the instruction next to the clock gear switching instruction by the clock gear after changing, input the dummy instruction as follows (instruction to execute the write cycle).

```
SYSCR1  EQU   00E1H
        LD    (SYSCR1),XXXX0000B    ;  Changes fSYS to fc/2.
        LD    (DUMMY),00H           ;  Dummy instruction
        Instruction to be executed after clock gear has changed.
```

(3)Internal clock output

The f_{SYS} or fs internal clock can be driven out from the P40/SCOUT pin.

The P40/SCOUT pin is configured as SCOUT (System clock output) by programming the port 4 registers as follows: P4CR<P40C> = "1" and P4FC<P40F> = "1". The output clock is selected through the SYSCR2<SCOSEL> bit.

Table 2-3 shows the pin states in each clocking mode when the P40/SCOUT pin is configured as SCOUT.

Table 2-3 SCOUT Output States

	NORMAL SLOW	HALT mode		
		IDLE2	IDLE1	STOP
<SCOSEL>="0"	The fs clock is driven out.			HOLD at either "1" or "0"
<SCOSEL>="1"	The f_{SYS} clock is driven out.			

2.3.4 Prescaler Clock Controller

For the internal I/O (TMRA01 and TMRA45, TMRB0 to TMRB3, SIO0 to SIO2, SBI0) there is a prescaler which can divide the clock.

The ϕT0 clock input to the prescaler is either the clock f_{FPH} divided by 2 or the clock fc/16 divided by 4. The setting of the SYSCR0<PRCK1> register determines which clock signal is input.

2.3.5 Runaway provision with SFR protection register

(Purpose)

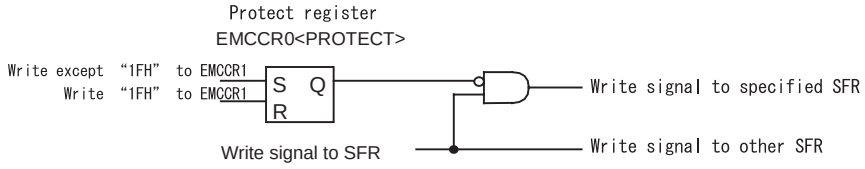
Provision in runaway of program by noise mixing.

Write operation to specified SFR is prohibited so that provision program in runaway prevents that it is it in the state which is fetch impossibility by stopping of clock, memory control register (CS/WAIT controller) is changed.

Specified SFR list

1. Clock gear (write enable only EMCCR1) SYSCR0, SYSCR1, SYSCR2

(Block diagram)



(Setting method)

If writing except "1FH" code to EMCCR1 register, it become protect ON. By this operation, write operation to specified SFR is disabling.

If writing "1FH" to EMCCR1 register, it become protect OFF. State of protect can to confirm by reading EMCCR0<PROTECT>.

Table 2-4 SFR for EMCCR

		7	6	5	4	3	2	1	0
EMCCR0 (00E3H)	Bit Symbol	PROTECT	–	–	–	–	–	–	–
	Read/Write	R	R/W						
	After reset	0	0	1	0	0	0	1	1
	Function	Protect flag 0: OFF 1: ON	Write "0".	Write "1".	Write "0".	Write "0".	Write "0".	Write "1".	Write "1".
EMCCR1 (00E4H)	Bit Symbol	Protect OFF by writing "1FH". Protect ON by writing except "1FH".							
	Read/Write								
	After reset								
	Function								

2.3.6 Standby Controller

(1)HALT modes

When the HALT instruction is executed, the operating mode switches to IDLE2, IDLE1 or STOP mode, depending on the contents of the SYSCR2<HALTM1:0> register.

The subsequent actions performed in each mode are as follows:

1. IDLE2: Only the CPU halts.

   The internal I/O is available to select operation during IDLE2 mode by setting the following register.

   Shows the registers of setting operation during IDLE2 mode.

Table 2-5 SFR Setting Operation during IDLE2 Mode

Internal I/O	SFR
TMRA01	TA01RUN<I2TA01>
TMRA45	TA45RUN<I2TA45>
TMRB0	TB0RUN<I2TB0>
TMRB1	TB1RUN<I2TB1>
TMRB2	TB2RUN<I2TB2>
TMRB3	TB3RUN<I2TB3>

Internal I/O	SFR
SIO0	SC0MOD1<I2S0>
SIO1	SC1MOD1<I2S1>
SIO2	SC2MOD1<I2S2>
SBI0	SBI0BR<I2SBI0>
AD	ADCCR2<I2AD>
WDT	WDMOD<I2WDT>

2. IDLE1: Only the oscillator and the RTC (Real time clock) continue to operate.
3. STOP: All internal circuits stop operating.

   The operation of each of the different HALT modes is described in Table 2-6.

Table 2-6 I/O Operation during HALT Modes

<table>
<tr><td colspan="2">HALT mode</td><td>IDLE2</td><td>IDLE1</td><td>STOP</td></tr>
<tr><td colspan="2">SYSCR2<HALTM1:0></td><td>11</td><td>10</td><td>01</td></tr>
<tr><td rowspan="8">Block</td><td>CPU</td><td colspan="3">Stop</td></tr>
<tr><td>I/O port</td><td colspan="2">Keep the state when the HALT instruction was executed.</td><td>See Table 2-9</td></tr>
<tr><td>TMRA,TMRB</td><td rowspan="5">Available to select operation block</td><td></td><td rowspan="6">Stop</td></tr>
<tr><td>RTC</td><td>Operate enable</td></tr>
<tr><td>SIO,SBI</td><td rowspan="4"></td></tr>
<tr><td>AD</td></tr>
<tr><td>WDT</td></tr>
<tr><td>Interrupt controller</td><td>Operate</td></tr>
</table>

(2)How to release the HALT mode

These halt states can be released by resetting or requesting an interrupt. The halt release sources are determined by the combination between the states of interrupt mask register <IFF2:0> and the HALT modes. The details for releasing the halt status are shown in Table 2-7.

Released by requesting an interrupt

The operating released from the HALT mode depends on the interrupt enabled status. When the interrupt request level set before executing the HALT instruction exceeds the value of interrupt mask register, the interrupt due to the source is processed after releasing the HALT mode, and CPU status executing an instruction that follows the HALT instruction. When the interrupt request level set before executing the HALT instruction is less than the value of the interrupt mask register, releasing the HALT mode is not executed. (In non-maskable interrupts, interrupt processing is processed after releasing the HALT mode regardless of the value of the mask register.) However only for INT0 and RTC interrupts, even if the interrupt request level set before executing the HALT instruction is less than the value of the interrupt mask register, releasing the HALT mode is executed. In this case, interrupt processing, and CPU starts executing the instruction next to the HALT instruction, but the interrupt request flag is held at "1".

Note: Usually, interrupts can release all halts status. However, the interrupts (INT0, INTRTC) which can release the HALT mode may not be able to do so if they are input during the period CPU is shifting to the HALT mode (for about 5 clocks of f_{FPH}) with IDLE1 or STOP mode (IDLE2 is not applicable to this case). (In this case, an interrupt request is kept on hold internally.) If another interrupt is generated after it has shifted to HALT mode completely, halt status can be released without difficulty. The priority of this interrupt is compared with that of the interrupt kept on hold internally, and the interrupt with higher priority is handled first followed by the other interrupt.

Releasing by resetting

Releasing all halt status is executed by resetting.

When the STOP mode is released by RESET, it is necessary enough resetting time (See Table 2-6)to set the operation of the oscillator to be stable.

When releasing the HALT mode by resetting, the internal RAM data keeps the state before the "HALT" instruction is executed. However the other settings contents are initialized. (Releasing due to interrupts keeps the state before the "HALT" instruction is executed.)

Table 2-7 Source of Halt State Clearance and Halt Clearance Operation

Status of Received Interrupt			Interrupt Enable (Interrupt level) ≥ (Interrupt mask)			Interrupt Disable (Interrupt level) < (Interrupt mask)		
HALT mode			IDLE2	IDLE1	STOP	IDLE2	IDLE1	STOP
Source of Halt state clearance	Interrupt	INTWDT	♦	×	×	-	-	-
		INT0(Note 1)	♦	♦	♦*1	O	O	O*1
		INTRTC	♦	♦	×	O	O	×
		INT1-INT8	♦(Note 2)	×	×	×	×	×
		INTTA0, INTTA1, INTTA4, INTTA5	♦	×	×	×	×	×
		INTTB00-30, INTTB01-31	♦	×	×	×	×	×
		INTTB0F0-3	♦	×	×	×	×	×
		INTRX0-INTRX2, INTTX0-INTTX2	♦	×	×	×	×	×
		INTSBI0	♦	×	×	×	×	×
		INTAD	♦	×	×	×	×	×
	RESET		Initialize LSI					

♦:After clearing the HALT mode, CPU starts interrupt processing.

O:After clearing the HALT mode, CPU resumes executing starting from instruction following the HALT instruction. (Interrupt routine don't execute.)

×:It can not be used to release the HALT mode.

- :The priority level (Interrupt request level) of non-maskable interrupts is fixed to 7, the highest priority level. There is not this combination type.

*1:Releasing the HALT mode is executed after passing the warm-up time.

Note 1: When the HALT mode is cleared by an INT0 interrupt of the level mode in the interrupt enabled status, hold high level until starting interrupt process. If low level was set before interrupt process is stared, interrupt process is not started correctly.

Note 2: If using external interrupt INT1 to INT8 in IDLE2 mode, set 16-bit timer RUN register TB0RUN<I2TB0>, TB1RUN<I2TB1>, TB2RUN<I2TB2>, TB3RUN<I2TB3> to "1".

Example:Clearing halt state

An INT0 interrupt clears the halt state when the device is in IDLE1 mode.

```
Address
8203H   LD     (IIMC), 00H          ; Selects INT0 interrupt rising edge.
8206H   LD     (INTE0AD), 06H       ; Sets INT0 interrupt level to 6.
8209H   EI     5                    ; Sets CPU interrupt level to 5.
820BH   LD     (SYSCR2), 28H        ; Sets HALT mode to IDLE1 mode.
820EH   HALT                        ; Halts CPU.

INT0                                          INT0 interrupt routine
                                                          RETI
820FH   LD     XX, XX
```

(3)Operation

1. IDLE2 mode

In IDLE2 mode only specific internal I/O operations, as designated by the IDLE2 setting register, can take place. Instruction execution by the CPU stops.

Figure 2-7 illustrates an example of the timing for clearance of the IDLE2 mode halt state by an interrupt.

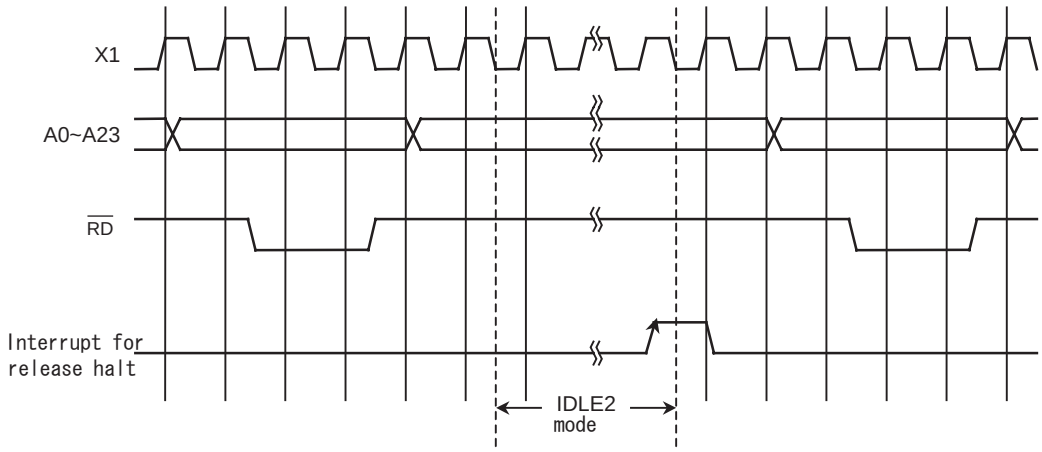


Figure 2-7 Timing Chart for IDLE2 Mode Halt State Cleared by Interrupt

2. IDLE1 mode

In IDLE1 mode, only the internal oscillator and the RTC continue to operate. The system clock in the MCU stops.

In the halt state, the interrupt request is sampled asynchronously with the system clock; however, clearance of the Halt state (e.g., restart of operation) is synchronous with it.

Figure 2-8 illustrates the timing for clearance of the IDLE1 mode halt state by an interrupt.

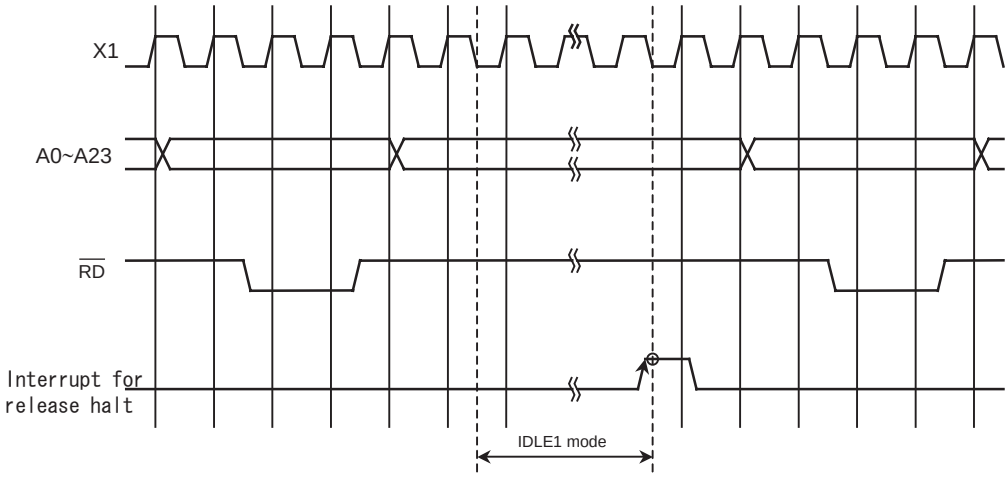


Figure 2-8 Timing Chart for IDLE1 Mode Halt State Cleared by Interrupt

3. STOP mode

When STOP mode is selected, all internal circuits stop, including the internal oscillator. Pin status in STOP mode depends on the settings in the SYSCR2<DRVE> register. Table 2-9 summarizes the state of these pins in STOP mode.

After STOP mode has been cleared, system clock output starts when the warm-up time has elapsed, in order to allow oscillation to stabilize. After STOP mode has been cleared, either NORMAL mode or SLOW mode can be selected using the SYSCR0<RSYSCK> register. Therefore, <RSYSCK>, <RXEN> and <RXTEN> must be set. See the sample warm-up times in Table 2-8.

Figure 2-9 illustrates the timing for clearance of the STOP mode halt state by an interrupt.

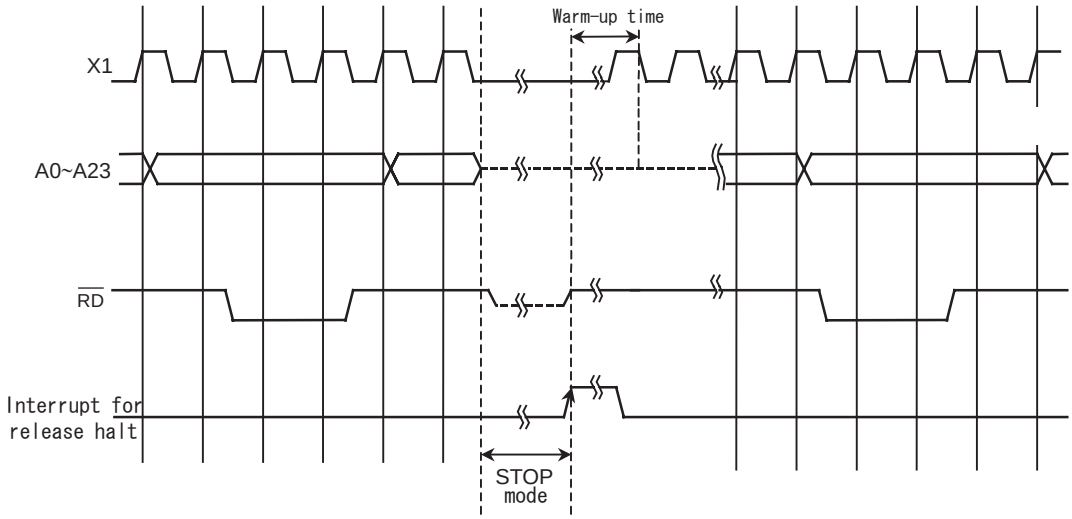


Figure 2-9 Timing Chart for STOP Mode Halt State Cleared by Interrupt

Table 2-8 Sample Warm-up Times after Clearance of STOP Mode

SYSCR0 <RSYSCK>	SYSCR2<WUPTM1:0>			
	01(2^8)	10(2^{14})	11(2^{16})	00(2^{18})
0(fc)	12.8us	0.819ms	3.277ms	13.107ms
1(fs)	7.8ms	500ms	2000ms	8000ms

Note: f_{OSCH}=20MHz, fs=32.768kHz

Example:

"The STOP mode is entered when the low-frequency operates, and high-frequency operates after releasing due to INT0.

```
Address
            SYSCR0   EQU   00E0H
            SYSCR1   EQU   00E1H
            SYSCR2   EQU   00E2H
8FFDH                LD    (SYSCR1), 08H          ;   fSYS = fs/2
9000H                LD    (SYSCR2), X–1001X1B    ;   Sets warm-up time to 2^14/fOSCH
9002H                LD    (SYSCR0), 011000––B    ;   Operates high frequency after released.

9005H                HALT
          INT0 pin input↓
                                                        Clears and starts
                                                          warm-up timer
                                                          (High frequency)
                                                              ↓
                                                             END
                                                              ↓
                                                        INT0 interrupt routine
9006H                LD    XX, XX                           RETI
          –: No change
```

Note: When different modes are used before and after STOP mode as the above mentioned, there is possible to release the HALT mode without changing the operation mode by acceptance of the halt release interrupt request during execution of "HALT" instruction (during 6 state). In the system which accepts the interrupts during execution "HALT" instruction, set the same operation mode before and after the STOP mode.

Table 2-9 Input/output Buffer State Table

Port Name	Input / Output	<DRVE>=0	<DRVE>=1
P00-07	input mode output mode	- -	- output
P10-17	input mode output mode	- -	- output
P30-33	input mode output mode	- -	- output
P40-43	input mode output mode	PU* PU*	PU* output
P50-57	input mode output mode analog input	- - -	- output -
P60-67	input mode output mode analog input	- - -	- output -
P70-74	input mode output mode	- -	input output
P75	input mode output mode	input -	input output
P80-87	input mode output mode	- -	- output
P90-97	input mode output mode	- -	- output
PA0-A3	input mode output mode	- -	- output
PB0-B2	input mode output mode	- -	- output
$\overline{\text{RESET}}$	input	input	input
AM0,AM1	input	input	input
X1	input	-	-
X2	output	"H" level output	"H" level output

- : Input for input mode / input pins is invalid; output mode / output pin is at high impedance.

input: Input gate in operation. Fix input voltage to "L" or "H" so that input pin stays constant.

output: Output state

PU*: Programmable pull-up pin. Input gate disable state. No through current even if the pin is set high impedance.

3. Interrupts

Interrupts are controlled by the CPU interrupt mask register SR<IFF2:0> and by the built-in interrupt controller.

The TMP91FU62 has a total of 48 interrupts divided into the following three types:

- Interrupts generated by CPU: 9 sources
  (Software interrupts, illegal instruction interrupt)
- Interrupts on external pins (INT0 to INT8): 9 sources
- Internal interrupts: 30 sources

A (fixed) individual interrupt vector number is assigned to each interrupt.

One of six (Variable) priority level can be assigned to each maskable interrupt.

The priority level of non-maskable interrupts are fixed at 7 as the highest level.

When an interrupt is generated, the interrupt controller sends the priority of that interrupt to the CPU. If multiple interrupts are generated simultaneously, the interrupt controller sends the interrupt with the highest priority to the CPU. (The highest priority is level 7 using for non-maskable interrupts.)

The CPU compares the priority level of the interrupt with the value of the CPU interrupt mask register <IFF2:0>. If the priority level of the interrupt is higher than the value of the interrupt mask register, the CPU accepts the interrupt.

The interrupt mask register <IFF2:0> value can be updated using the value of the EI instruction ("EI num" sets <IFF2:0> data to num).

For example, specifying "EI3" enables the maskable interrupts which priority level set in the interrupt controller is 3 or higher, and also non-maskable interrupts.

Operationally, the DI instruction (<IFF2:0> "7") is identical to the "EI7" instruction. DI instruction is used to disable maskable interrupts because of the priority level of maskable interrupts is 0 to 6. The EI instruction is valid immediately after execution.

In addition to the above general-purpose interrupt processing mode, TLCS-900/L1 has a micro DMA interrupt processing mode as well. The CPU can transfer the data (1/2/4 bytes) automatically in micro DMA mode, therefore this mode is used for speed-up interrupt processing, such as transferring data to the internal or external peripheral I/O. Moreover, TMP91FU62 has software start function for micro DMA processing request by the software not by the hardware interrupt.

Figure 3-1 shows the overall interrupt processing flow.

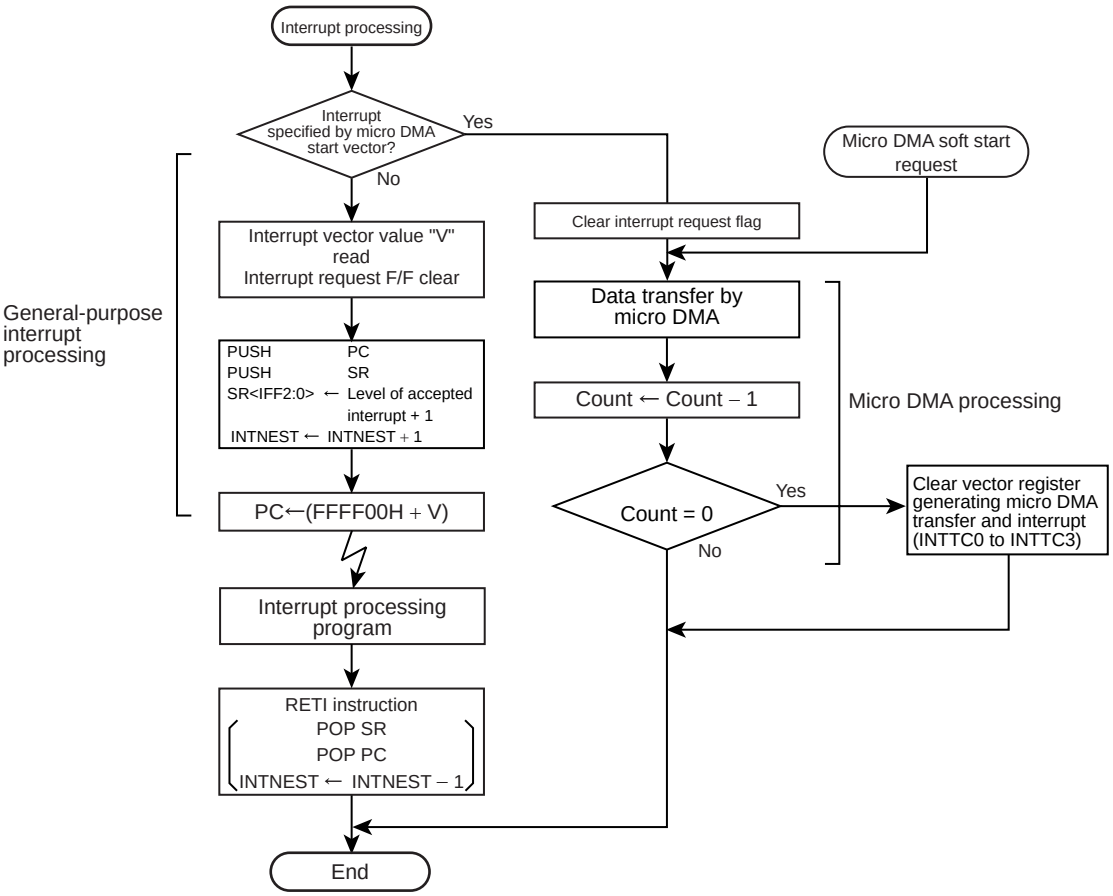


Figure 3-1 Overall Interrupt Processing Flow

3.1 General-purpose Interrupt Processing

When the CPU accepts an interrupt, it usually performs the following sequence of operations. That is also the same as TLCS-900/L and TLCS-900/H.

1. The CPU reads the interrupt vector from the interrupt controller.
   If the same level interrupts occur simultaneously, the interrupt controller generates an interrupt vector in accordance with the default priority and clears the interrupt request.
   (The default priority is already fixed for each interrupt. The smaller vector value has the higher priority level.)
2. The CPU pushes the value of program counter (PC) and status register (SR) onto the stack area (Indicated by XSP).
3. The CPU sets the value which is the priority level of the accepted interrupt plus 1 (+1) to the interrupt mask register <IFF2:0>. However, if the priority level of the accepted interrupt is 7, the register's value is set to 7.
4. The CPU increases the interrupt nesting counter INTNEST by 1 (+1).
5. The CPU jumps to the address indicated by the data at address "FFFF00H + Interrupt vector" and starts the interrupt processing routine.

The above processing time is 18 states (1.8 μs at 20 MHz) as the best case (16-bit data bus width and 0 waits).

When the CPU completed the interrupt processing, use the RETI instruction to return to the main routine. RETI restores the contents of program counter (PC) and status register (SR) from the stack and decreases the interrupt nesting counter INTNEST by 1 (–1).

Non-maskable interrupts cannot be disabled by a user program. Maskable interrupts, however, can be enabled or disabled by a user program. A program can set the priority level for each interrupt source. (A priority level setting of 0 or 7 will disable an interrupt request.)

If an interrupt request which has a priority level equal to or greater than the value of the CPU interrupt mask register <IFF2:0> comes out, the CPU accepts its interrupt. Then, the CPU interrupt mask register <IFF2:0> is set to the value of the priority level for the accepted interrupt plus 1 (+1).

Therefore, if an interrupt is generated with a higher level than the current interrupt during its processing, the CPU accepts the later interrupt and goes to the nesting status of interrupt processing.

Moreover, if the CPU receives another interrupt request while performing the said 1. to 5. processing steps of the current interrupt, the latest interrupt request is sampled immediately after execution of the first instruction of the current interrupt processing routine. Specifying DI as the start instruction disables maskable interrupt nesting.

A reset initializes the interrupt mask register <IFF2:0> to "111", disabling all maskable interrupts.

Table 3-1 shows the TMP91FU62 interrupt vectors and micro DMA start vectors. The address FFFF00H to FFFFFFH (256 bytes) is assigned for the interrupt vector area.

Table 3-1 TMP91FU62 Interrupt Vectors Table(1/2)

Default Priority	Type	Interrupt Source and Source of Micro DMA Request	Vector Value (V)	Vector Reference Address	Micro DMA Start Vector
1	Non-maskable	“Reset” or “SWI 0” instruction	0000H	FFFF00H	–
2		“SWI 1” instruction	0004H	FFFF04H	–
3		INTUNDEF: Illegal instruction or “SWI 2” instruction	0008H	FFFF08H	–
4		“SWI 3” instruction	000CH	FFFF0CH	–
5		“SWI 4” instruction	0010H	FFFF10H	–
6		“SWI 5” instruction	0014H	FFFF14H	–
7		“SWI 6” instruction	0018H	FFFF18H	–
8		“SWI 7” instruction	001CH	FFFF1CH	–
9		(Reserved)	0020H	FFFF20H	–
10		INTWD: Watchdog timer	0024H	FFFF24H	–
–	Maskable	Micro DMA (MDMA)	–	–	–
11		INT0: INT0 pin	0028H	FFFF28H	0AH
12		INT1: INT1 pin	002CH	FFFF2CH	0BH
13		INT2: INT2 pin	0030H	FFFF30H	0CH
14		INT3: INT3 pin	0034H	FFFF34H	0DH
15		INT4: INT4 pin	0038H	FFFF38H	0EH
16		INT5: INT5 pin	003CH	FFFF3CH	0FH
17		INT6: INT6 pin	0040H	FFFF40H	10H
18		INT7: INT7 pin	0044H	FFFF44H	11H
19		INT8: INT8 pin	0048H	FFFF48H	12H
20		(Reserved)	004CH	FFFF4CH	13H
21		(Reserved)	0050H	FFFF50H	14H
22		INTTA0: 8-bit timer 0	0054H	FFFF54H	15H
23		INTTA1: 8-bit timer 1	0058H	FFFF58H	16H
24		(Reserved)	005CH	FFFF5CH	17H
25		(Reserved)	0060H	FFFF60H	18H
26		INTTA4: 8-bit timer 4	0064H	FFFF64H	19H
27		INTTA5: 8-bit timer 5	0068H	FFFF68H	1AH
28		INTTB00: 16-bit timer 0 (TB0RG0)	006CH	FFFF6CH	1BH
29		INTTB01: 16-bit timer 0 (TB0RG1)	0070H	FFFF70H	1CH
30		INTTB10: 16-bit timer 1 (TB1RG0)	0074H	FFFF74H	1DH
31		INTTB11: 16-bit timer 1 (TB1RG1)	0078H	FFFF78H	1EH
32		INTTB20: 16-bit timer 2 (TB2RG0)	007CH	FFFF7CH	1FH
33		INTTB21: 16-bit timer 2 (TB2RG1)	0080H	FFFF80H	20H
34		INTTB30: 16-bit timer 3 (TB3RG0)	0084H	FFFF84H	21H
35		INTTB31: 16-bit timer 3 (TB3RG1)	0088H	FFFF88H	22H
36		(Reserved)	008CH	FFFF8CH	23H
37		(Reserved)	0090H	FFFF90H	24H

Table 3-1 TMP91FU62 Interrupt Vectors Table(2/2)

Default Priority	Type	Interrupt Source and Source of Micro DMA Request	Vector Value (V)	Vector Reference Address	Micro DMA Start Vector
38	Maskable	INTTBOF0: 16-bit timer 0 (Over flow)	0094H	FFFF94H	25H
39		INTTBOF1: 16-bit timer 1 (Over flow)	0098H	FFFF98H	26H
40		INTTBOF2: 16-bit timer 2 (Over flow)	009CH	FFFF9CH	27H
41		INTTBOF3: 16-bit timer 3 (Over flow)	00A0H	FFFFA0H	28H
42		(Reserved)	00A4H	FFFFA4H	29H
43		INTRX0:Serial reception (Channel 0)	00A8H	FFFFA8H	2AH
44		INTTX0:Serial transmission (Channel 0)	00ACH	FFFFACH	2BH
45		INTRX1:Serial reception (Channel 1)	00B0H	FFFFB0H	2CH
46		INTTX1:Serial transmission (Channel 1)	00B4H	FFFFB4H	2DH
47		INTRX2:Serial reception (Channel 2)	00B8H	FFFFB8H	2EH
48		INTTX2:Serial transmission (Channel 2)	00BCH	FFFFBCH	2FH
49		INTSBI0:Serial bus interface interrupt (Channel 0)	00C0H	FFFFC0H	30H
50		(Reserved)	00C4H	FFFFC4H	31H
51		INTRTC: Interrupt for special timer for CLOCK	00C8H	FFFFC8H	32H
52		INTAD: AD conversion end	00CCH	FFFFCCH	33H
53		INTTC0 Micro DMA end (Channel 0)	00D0H	FFFFD0H	–
54		INTTC1: Micro DMA end (Channel 1)	00D4H	FFFFD4H	–
55		INTTC2: Micro DMA end (Channel 2)	00D8H	FFFFD8H	–
56		INTTC3: Micro DMA end (Channel 3)	00DCH	FFFFDCH	–
		(Reserved) : (Reserved)	00E0H : 00FCH	FFFFE0H : FFFFFCH	– : –

Note: Micro DMA default priority: Micro DMA stands up prior to other maskable interrupt.

3.2 Micro DMA Processing

In addition to general-purpose interrupt processing, the TMP91FU62 supports a micro DMA function. Interrupt requests set by micro DMA perform micro DMA processing at the highest priority level (Level 6) among maskable interrupts, regardless of the priority level of the particular interrupt source. The micro DMA has 4 channels and is possible continuous transmission by specifying the described later burst mode.

The micro DMA has 4 channels and is possible continuous transmission by specifying the described later burst mode.

Because the micro DMA function has been implemented with the cooperative operation of CPU, when CPU goes to a standby mode (STOP, IDLE1 and IDLE2) by HALT instruction, the requirement of micro DMA will be ignored (Pending) and DMA transfer is started after release HALT.

3.2.1 Micro DMA Operation

When an interrupt request specified by the micro DMA start vector register is generated, the micro DMA triggers a micro DMA request to the CPU at interrupt priority level 6 and starts processing the request in spite of any interrupt source's level. The micro DMA is ignored on <IFF2:0> = "7".

The 4 micro DMA channels allow micro DMA processing to be set for up to 4 types of interrupts at any one time. When micro DMA is accepted, the interrupt request flip-flop assigned to that channel is cleared.

The data are automatically transferred once (1/2/4 bytes) from the transfer source address to the transfer destination address set in the control register, and the transfer counter is decreased by 1 (−1). If the decreased result is "0", the micro DMA transfer end interrupt (INTTC0 to INTTC3) passes from the CPU to the interrupt controller. In addition, the micro DMA start vector register DMAnV is cleared to 0, the next micro DMA is disabled and micro DMA processing completes. If the decreased result is other than "0", the micro DMA processing completes if it does not specify the described later burst mode. In this case, the micro DMA transfer end interrupt (INTTC0 to INTTC3) aren't generated.

If an interrupt request is triggered for the interrupt source in use during the interval between the clearing of the micro DMA start vector and the next setting, general-purpose interrupt processing executes at the interrupt level set. Therefore, if only using the interrupt for starting the micro DMA (Not using the interrupts as a general-purpose interrupt: Level 1 to 6), first set the interrupts level to 0 (Interrupt requests disabled).

If using micro DMA and general-purpose interrupts together, first set the level of the interrupt used to start micro DMA processing lower than all the other interrupt levels. (Note) In this case, the cause of general interrupt is limited to the edge interrupt.

The priority of the micro DMA transfer end interrupt (INTTC0 to INTTC3) is defined by the interrupt level and the default priority as the same as the other maskable interrupt.

If a micro DMA request is set for more than one channel at the same time, the priority is not based on the interrupt priority level but on the channel number. The smaller channel number has the higher priority (Channel 0 (High) > Channel 3 (Low)).

While the register for setting the transfer source/transfer destination addresses is a 32-bit control register, this register can only effectively output 24-bit addresses. Accordingly, micro DMA can access 16 Mbytes (The upper eight bits of the 32 bits are not valid).

Note: If the priority level of micro DMA is set higher than that of other interrupts, CPU operates as follows.

In case INTxxx interrupt is generated first and then INTyyy interrupt is generated between checking "Interrupt specified by micro DMA start vector" (in the Figure 3-1) and reading interrupt vector with setting below, the vector shifts to that of INTyyy at the time.
This is because the priority level of INTyyy is higher than that of INTxxx.
In the interrupt routine, CPU reads the vector of INTyyy because checking of micro DMA has been finished.
And INTyyy is generated regardless of transfer counter of micro DMA.
INTxxx: level 1 without micro DMA
INTyyy: level 6 with micro DMA

Three micro DMA transfer modes are supported: 1-byte transfer, 2-byte (One-word) transfer, and 4-byte transfer. After a transfer in any mode, the transfer source/destination addresses are increased, decreased, or remain unchanged.

This simplifies the transfer of data from I/O to memory, from memory to I/O, and from I/O to I/O. For details of the transfer modes, see" 3.2.4 Detailed Description of the Transfer Mode Register ".

As the transfer counter is a 16-bit counter, micro DMA processing can be set for up to 65536 times per interrupt source. (The micro DMA processing count is maximized when the transfer counter initial value is set to 0000H.)

Micro DMA processing can be started by the 42 interrupts shown in the micro DMA start vectors of Table 3-1 and by the micro DMA soft start, making a total of 43 interrupts.

Figure 3-2 shows the word transfer micro DMA cycle in transfer destination address INC mode (except for counter mode, the same as for other modes).

(The conditions for this cycle are based on an external 16-bit bus, 0 waits, transfer source/transfer destination addresses both even-numberd values).

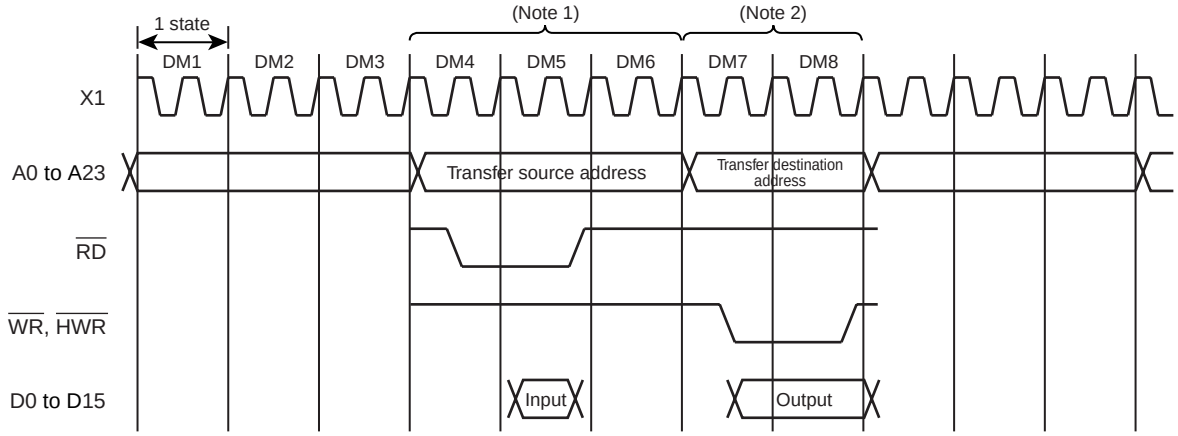


Figure 3-2 Timing for Micro DMA Cycle

States 1 to 3: Instruction fetch cycle (Gets next address code).
If 3 bytes and more instruction codes are inserted in the instruction queue buffer, this cycle becomes a dummy cycle.

States 4 to 5: Micro DMA read cycle

State 6: Dummy cycle (The address bus remains unchanged from state 5.)

States 7 to 8: Micro DMA write cycle

Note 1: If the source address area is an 8-bit bus, it is increased by two states.
If the source address area is a 16-bit bus and the address starts from an odd number, it is increased by two states.

Note 2: If the destination address area is an 8-bit bus, it is increased by two states.
If the destination address area is a 16-bit bus and the address starts from an odd number, it is increased by two states.

3.2.2 Soft Start Function

In addition to starting the micro DMA function by interrupts, TMP91FU62 includes a micro DMA software start function that starts micro DMA on the generation of the write cycle to the DMAR register.

Writing “1” to each bit of DMAR register causes micro DMA once (If write “0” to each bit, micro DMA doesn’t operate) At the end of transfer, the corresponding bit of the DMAR register is automatically cleared to “0”.

Only one-channel can be set once for micro DMA. (Do not write “1” to plural bits.)

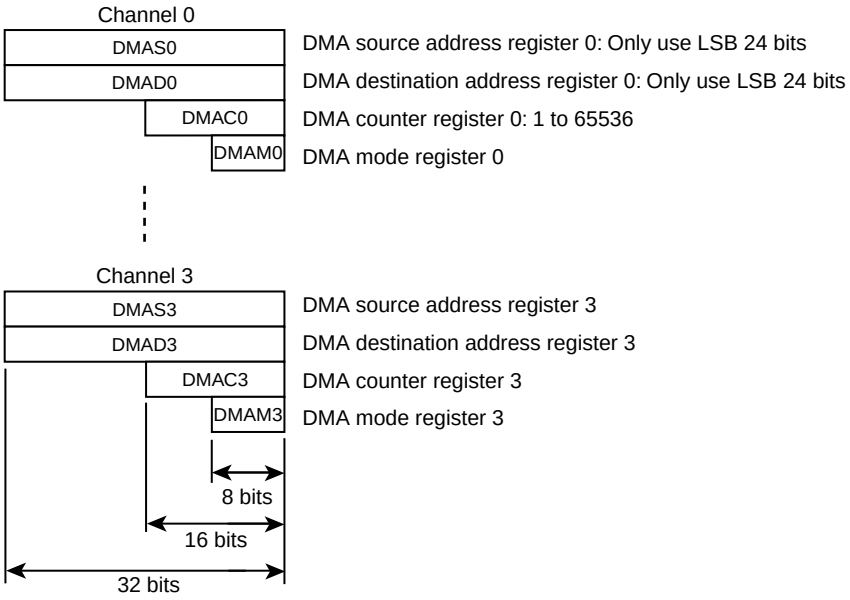
When writing again “1” to the DMAR register, check whether the bit is “0” before writing “1”. If read “1”, micro DMA transfer isn’t started yet.

When a burst is specified by DMAB register, data is continuously transferred until the value in the micro DMA transfer counter is “0” after start up of the micro DMA. If execute soft start during micro DMA transfer by interrupt source, micro DMA transfer counter doesn’t change. Don’t use Read-modify-write instruction to avoid writing to other bits by mistake.

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMAR	DMA Request Register	89H	–	–	–	–	DMAR3	DMAR2	DMAR1	DMAR0
		RMW instructions are prohib-ited.	–	–	–	–	R/W			
			–	–	–	–	0	0	0	0
							DMA request			

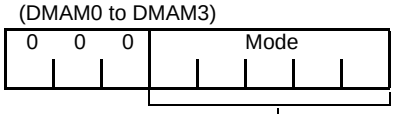
3.2.3 Transfer Control Registers

The transfer source address and the transfer destination address are set in the following registers in CPU. Data setting for these registers is done by an “LDC cr, r” instruction.



3.2.4 Detailed Description of the Transfer Mode Register

(DMAM0 to DMAM3)



Note: The upper three bit of data programmed to these registers must always be 0.

ZZ: 0 = Byte transfer, 1 = Word transfer, 2 = 4-byte transfer, 3 = Reserved

						Execution time
0	0	0	Z	Z	Transfer destination address INC mode ・・・・・・・・・・・・ I/O to memory (DMADn+) ← (DMASn) DMACn ← DMACn − 1 if DMACn = 0 then INTTC is generated	8 states (800 ns) @ byte/word transfer 12 states (1200 ns) @ 4-byte/word transfer
0	0	1	Z	Z	Transfer destination address DEC mode ・・・・・・・・・・・・ I/O to memory (DMADn-) ← (DMASn) DMACn ← DMACn − 1 if DMACn = 0 then INTTC is generated	8 states (800 ns) @ byte/word transfer 12 states (1200 ns) @ 4-byte/word transfer
0	1	0	Z	Z	Transfer source address INT mode ・・・・・・・・・・・・・・ memory to I/O (DMADn) ← (DMASn+) DMACn ← DMACn − 1 if DMACn = 0 then INTTC is generated	8 states (800 ns) @ byte/word transfer 12 states (1200 ns) @ 4-byte/word transfer
0	1	1	Z	Z	Transfer source address DEC mode ・・・・・・・・・・・・・・ memory to I/O (DMADn) ← (DMASn-) DMACn ← DMACn − 1 if DMACn = 0 then INTTC is generated	8 states (800 ns) @ byte/word transfer 12 states (1200 ns) @ 4-byte/word transfer
1	0	0	Z	Z	Address fixed mode ・・・・・・・・・・・・・・・・・・・・・・・・ I/O to I/O (DMADn) ← (DMASn) DMACn ← DMACn − 1 if DMACn = 0 then INTTC is generated	8 states (800 ns) @ byte/word transfer 12 states (1200 ns) @ 4-byte/word transfer
1	0	1	0	0	Counter mode for counting number of times interrupt is generated DMASn ← DMASn + 1 DMACn ← DMACn − 1 if DMACn = 0 then INTTC is generated	5 states (500 ns)

Note 1: “n” is the corresponding micro DMA channels 0 to 3.
DMADn+/DMASn+: Post-increment (Increment register value after transfer)
DMADn-/DMASn-: Post-decrement (Decrement register value after transfer)
The I/Os in the table mean fixed address and the memory means increment (INC) or decrement (DEC) addresses.

Note 2: Execution time is under the condition of:
16-bit bus width (Both transfer and destination address area)/0 waits/
fc = 20 MHz/selected high-frequency mode (fc × 1)

Note 3: Do not use an undefined code for the transfer mode register except for the defined codes listed in the above table.

3.3 Interrupt Controller Operation

The block diagram in Figure 3-3 shows the interrupt circuits. The left-hand side of the diagram shows the interrupt controller circuit. The right-hand side shows the CPU interrupt request signal circuit and the halt release circuit.

For interrupt controller there is an interrupt request flag (Consisting of a flip-flop), an interrupt priority setting register and a micro DMA start vector register. The interrupt request flag latches interrupt requests from the peripherals. The flag is cleared to 0 in the following cases:

- When reset occurs
- When the CPU reads the channel vector after accepted its interrupt
- When executing an instruction that clears the interrupt (Write DMA start vector to INTCLR register)
- When the CPU receives a micro DMA request (when micro DMA is set)
- When the micro DMA burst transfer is terminated

An interrupt priority can be set independently for each interrupt source by writing the priority to the interrupt priority setting register (e.g., INTE0AD or INTE56). 6 interrupt priorities levels (1 to 6) are provided. Setting an interrupt source's priority level to 0 (or 7) disables interrupt requests from that source. The priority of non-maskable interrupts (watchdog timer interrupts) is fixed at 7. If interrupt request with the same level are generated at the same time, the default priority is used to determine which interrupt request is accepted first.

The 3rd and 7th bits of the interrupt priority setting register indicate the state of the interrupt request flag and thus whether an interrupt request for a given channel has occurred.

The interrupt controller sends the interrupt request and its vector address to the CPU. The CPU compares the priority value <IFF2:0> in the status register by the interrupt request signal with the priority value set; if the latter is higher, the interrupt is accepted. Then the CPU sets a value higher than the priority value by 1 (+1) in the CPU SR<IFF2:0>. Interrupt request where the priority value equals or is higher than the set value are accepted simultaneously during the previous interrupt routine.

When interrupt processing is completed (after execution of the RETI instruction), the CPU restores the priority value saved in the stack before the interrupt was generated to the CPU SR<IFF2:0>.

The interrupt controller also has registers (4 channels) used to store the micro DMA start vector. Writing the start vector of the interrupt source for the micro DMA processing beforehand (see Table 3-1), enables the corresponding interrupt to be processed by micro DMA processing. The values must be set in the micro DMA parameter register (e.g., DMAS and DMAD) prior to the micro DMA processing.

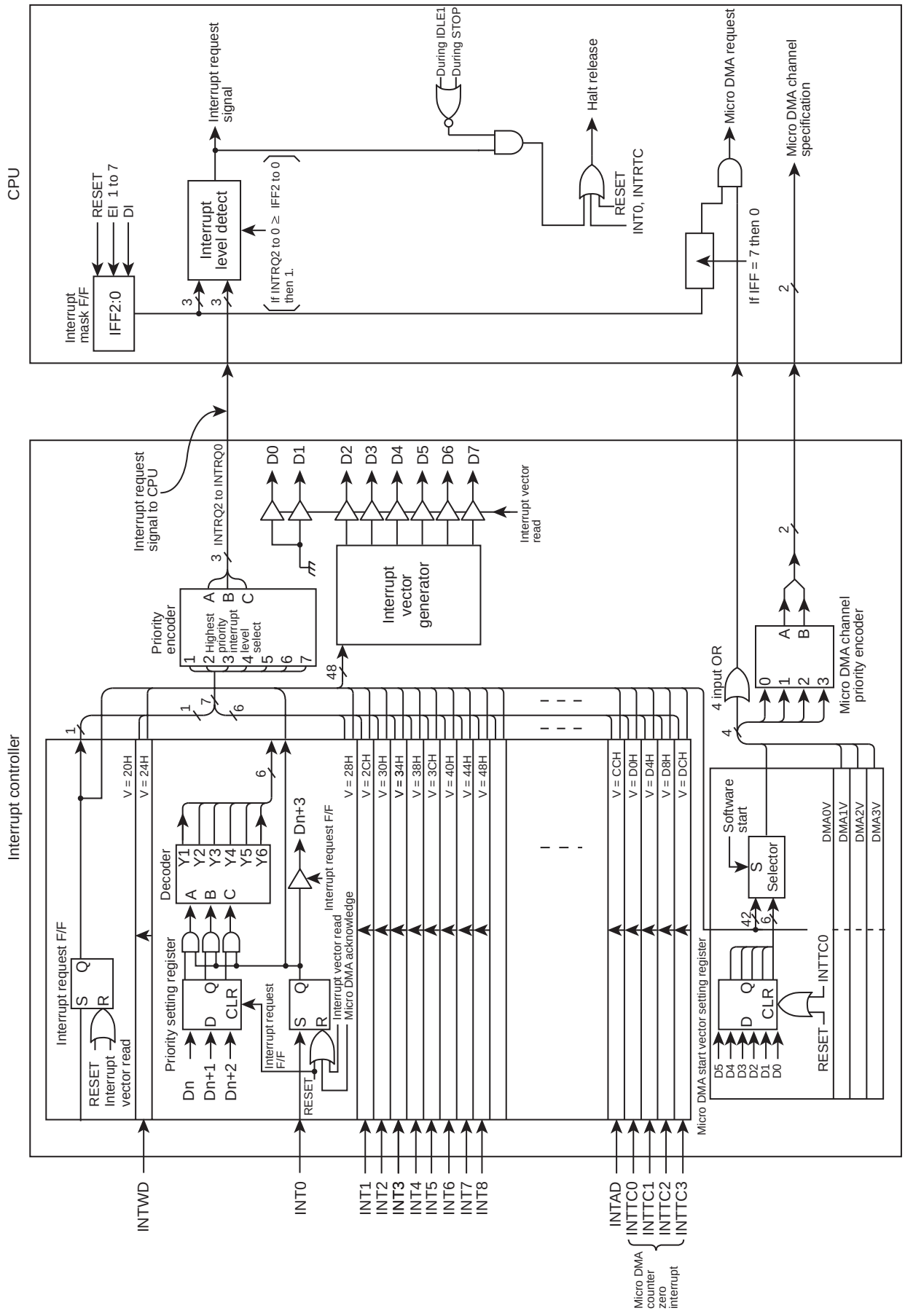


Figure 3-3 Block Diagram of Interrupt Controller

3.3.1 Interrupt Level Setting Registers

Interrupt Level Setting Registers

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTE0AD	INT0 & INTAD enable	90H	INTAD				INT0			
			IADC	IADM2	IADM1	IADM0	I0C	I0M2	I0M1	I0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE12	INT1 & INT2 enable	91H	INT2				INT1			
			I2C	I2M2	I2M1	I2M0	I1C	I1M2	I1M1	I1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE34	INT3 & INT4 enable	92H	INT4				INT3			
			I4C	I4M2	I4M1	I4M0	I3C	I3M2	I3M1	I3M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE56	INT5 & INT6 enable	93H	INT6				INT5			
			I6C	I6M2	I6M1	I6M0	I5C	I5M2	I5M1	I5M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTE78	INT7 & INT8 enable	94H	INT8				INT7			
			I8C	I8M2	I8M1	I8M0	I7C	I7M2	I7M1	I7M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETA01	INTTA0 & INTTA1 enable	96H	INTTA1(TMRA1)				INTTA0 (TMRA0)			
			ITA1C	ITA1M2	ITA1M1	ITA1M0	ITA0C	ITA0M2	ITA0M1	ITA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0

IxxxC
Interrupt request flag

IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Disables interrupt requests
0	0	1	Sets interrupt priority level to 1
0	1	0	Sets interrupt priority level to 2
0	1	1	Sets interrupt priority level to 3
1	0	0	Sets interrupt priority level to 4
1	0	1	Sets interrupt priority level to 5
1	1	0	Sets interrupt priority level to 6
1	1	1	Disables interrupt requests

Interrupt Level Setting Registers

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTETA45	INTTA4 & INTTA5 enable	98H	INTTA5 (TMRA5)				INTTA4 (TMRA4)			
			ITA5C	ITA5M2	ITA5M1	ITA5M0	ITA4C	ITA4M2	ITA4M1	ITA4M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETB0	Interrupt enable TMRB0	99H	INTTB01(TMRB0)				INTTB00(TMRB0)			
			ITB01C	ITB01M2	ITB01M1	ITB01M0	ITB00C	ITB00M2	ITB00M1	ITB00M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETB1	Interrupt enable TMRB1	9AH	INTTB11(TMRB1)				INTTB10(TMRB1)			
			ITB11C	ITB11M2	ITB11M1	ITB11M0	ITB10C	ITB10M2	ITB10M1	ITB10M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETB2	Interrupt enable TMRB2	9BH	INTTB21(TMRB2)				INTTB20(TMRB2)			
			ITB21C	ITB21M2	ITB21M1	ITB21M0	ITB20C	ITB20M2	ITB20M1	ITB20M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETB3	Interrupt enable TMRB3	9CH	INTTB31(TMRB3)				INTTB30(TMRB3)			
			ITB31C	ITB31M2	ITB31M1	ITB31M0	ITB30C	ITB30M2	ITB30M1	ITB30M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETB01V	Interrupt enable TMRB0/1 (Over flow)	9EH	INTTBOF1(TMRB1 Over flow)				INTTBOF0(TMRB0 Over flow)			
			ITF1C	ITF1M2	ITF1M1	ITF1M0	ITF0C	ITF0M2	ITF0M1	ITF0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0

IxxxC
Interrupt request flag

IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Disables interrupt requests
0	0	1	Sets interrupt priority level to 1
0	1	0	Sets interrupt priority level to 2
0	1	1	Sets interrupt priority level to 3
1	0	0	Sets interrupt priority level to 4
1	0	1	Sets interrupt priority level to 5
1	1	0	Sets interrupt priority level to 6
1	1	1	Disables interrupt requests

Interrupt Level Setting Registers

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTETB23V	Interrupt enable TMRB2/3 (Over flow)	9FH	INTTBOF3(TMRB3 Over flow)				INTTBOF2(TMRB2 Over flow)			
			ITF3C	ITF3M2	ITF3M1	ITF3M0	ITF2C	ITF2M2	ITF2M1	ITF2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTERTC	Interrupt enable INTRTC	A0H	INTRTC				–			
			IRTCC	IRTCM2	IRTCM1	IRTCM0	–	–	–	–
			R	R/W			–	–		
			0	0	0	0	–	–	–	–
INTES0	INTRX0 & INTTX0 enable	A1H	INTTX0				INTRX0			
			ITX0C	ITX0M2	ITX0M1	ITX0M0	IRX0C	IRX0M2	IRX0M1	IRX0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTES1	INTRX1 & INTTX1 enable	A2H	INTTX1				INTRX1			
			ITX1C	ITX1M2	ITX1M1	ITX1M0	IRX1C	IRX1M2	IRX1M1	IRX1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTES2	INTRX2 & INTTX2 enable	A3H	INTTX2				INTRX2			
			ITX2C	ITX2M2	ITX2M1	ITX2M0	IRX2C	IRX2M2	IRX2M1	IRX2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTESBI0	INTSBI0 enable	A4H	–				INTSBI0			
			–	–	–	–	ISBI0C	ISBI0M2	ISBI0M1	ISBI0M0
			–	–			R	R/W		
			–	–	–	–	0	0	0	0
INTETC01	INTTC0 & INTTC1 enable	A5H	INTTC1				INTTC0			
			ITC1C	ITC1M2	ITC1M1	ITC1M0	ITC0C	ITC0M2	ITC0M1	ITC0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
INTETC23	INTTC2 & INTTC3 enable	A6H	INTTC3				INTTC2			
			ITC3C	ITC3M2	ITC3M1	ITC3M0	ITC2C	ITC2M2	ITC2M1	ITC2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0

IxxxC
Interrupt request flag

IxxM2	IxxM1	IxxM0	Function (Write)
0	0	0	Disables interrupt requests
0	0	1	Sets interrupt priority level to 1
0	1	0	Sets interrupt priority level to 2
0	1	1	Sets interrupt priority level to 3
1	0	0	Sets interrupt priority level to 4
1	0	1	Sets interrupt priority level to 5
1	1	0	Sets interrupt priority level to 6
1	1	1	Disables interrupt requests

3.3.2 External Interrupt Control

External Interrupt Control Register (IIMC)

Symbol	Name	Address	7	6	5	4	3	2	1	0
IIMC	Interrupt input mode control	8CH RMW instructions are prohibited.	–	–	–	–	–	I0EDGE	I0LE	–
			W							
			0	0	0	0	0	0	0	0
			Always write “0”.	–	–	–	–	INT0 EDGE 0: Rising 1: Falling	INT0 mode 0: Edge 1: Level	–

INT0 setting

P7FC<P75F>	<IOLE>	<IOEDGE>	INT0
1	0	0	Rising edge interruption
1	0	1	Falling edge interruption
1	1	0	“H” level INT
1	1	1	“L” level INT

3.3.3 Interrupt Request Flag Clear Register

The interrupt request flag is cleared by writing the appropriate micro DMA start vector, as given in Table 3-1, to the register INTCLR.

For example, to clear the interrupt flag INT0, perform the following register operation after execution of the DI instruction.

INTCLR ← 0AH: Clears interrupt request flag INT0.

Interrupt Request Flag Clear Register (INTCLR)

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTCLR	Interrupt Clear Control	88H RMW instructions are prohibited.	–	–	CLRV5	CLRV4	CLRV3	CLRV2	CLRV1	CLRV0
			–	–	W					
			–	–	0	0	0	0	0	0
					Interrupt vector					

3.3.4 Micro DMA Start Vector Registers

This register assigns micro DMA processing to which interrupt source. The interrupt source with a micro DMA start vector that matches the vector set in this register is assigned as the micro DMA start source.

When the micro DMA transfer counter value reaches 0, the micro DMA transfer end interrupt corresponding to the channel is sent to the interrupt controller, the micro DMA start vector register is cleared, and the micro DMA start source for the channel is cleared. Therefore, to continue micro DMA processing, set the micro DMA start vector register again during the processing of the micro DMA transfer end interrupt.

If the same vector is set in the micro DMA start vector registers of more than one channel, the channel with the lowest number has a higher priority.

Accordingly, if the same vector is set in the micro DMA start vector registers of two channels, the interrupt generated in the channel with the lower number is executed until micro DMA transfer is complete. If the micro DMA start vector for this channel is not set again, the next micro DMA is started for the channel with the higher number. (Micro DMA chaining)

Micro DMA Start Vector Registers (DMAnV)

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMA0V	DMA0 Start Vector	80H	–	–	DMA0V5	DMA0V4	DMA0V3	DMA0V2	DMA0V1	DMA0V0
			–	–	R/W					
			–	–	0	0	0	0	0	0
					DMA0 start vector					
DMA1V	DMA1 Start Vector	81H	–	–	DMA1V5	DMA1V4	DMA1V3	DMA1V2	DMA1V1	DMA1V0
			–	–	R/W					
			–	–	0	0	0	0	0	0
					DMA1 start vector					
DMA2V	DMA2 Start Vector	82H	–	–	DMA2V5	DMA2V4	DMA2V3	DMA2V2	DMA2V1	DMA2V0
			–	–	R/W					
			–	–	0	0	0	0	0	0
					DMA2 start vector					
DMA3V	DMA3 Start Vector	83H	–	–	DMA3V5	DMA3V4	DMA3V3	DMA3V2	DMA3V1	DMA3V0
			–	–	R/W					
			–	–	0	0	0	0	0	0
					DMA3 start vector					

3.3.5 Micro DMA Burst Specification

Specifying the micro DMA burst continues the micro DMA transfer until the transfer counter register reaches 0 after micro DMA start. Setting a bit which corresponds to the micro DMA channel of the DMAB registers mentioned below to “1” specifies a burst.

If other interrupts (maskable/nonmaskable is not concerned) are generated during burst transfer, interrupt is executed after completed burst transfer.

Micro DMA Burst Request Registers (DMAR)

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMAR	DMA Software Request Register	89H RMW instructions are prohib-ited.	–	–	–	–	DMAR3	DMAR2	DMAR1	DMAR0
			–	–	–	–	R/W			
			–	–	–	–	0	0	0	0
			1: DMA software request							
DMAB	DMA Burst Register	8AH	–	–	–	–	DMAB3	DMAB2	DMAB1	DMAB0
			–	–	–	–	R/W			
			–	–	–	–	0	0	0	0
			1: DMA burst request							

3.3.6 Attention Point

The instruction execution unit and the bus interface unit of this CPU operate independently. Therefore, immediately before an interrupt is generated, if the CPU fetches an instruction that clears the corresponding interrupt request flag, the CPU may execute the instruction that clears the interrupt request flag (Note) between accepting and reading the interrupt vector. In this case, the CPU reads the default vector 0008H and reads the interrupt vector address FFFF08H.

To avoid the above problem, place instructions that clear interrupt request flags after a DI instruction. And in the case of setting an interrupt enable again by EI instruction after the execution of clearing instruction, execute EI instruction after clearing and more than 1-instructions (ex. “NOP” * 1 times). If executed EI instruction without waiting NOP instruction after execution of clearing instruction, interrupt will be enable before request flag is cleared.

In the case of changing the value of the interrupt mask register <IFF2:0> by execution of POP SR instruction, disable an interrupt by DI instruction before execution of POP SR instruction.

In addition, take care as the following 2 circuits are exceptional and demand special attention.

INT0 level mode	In level mode INT0 is not an edge-triggered interrupt. Hence, in level mode the interrupt request flip-flop for INT0 does not function. The peripheral interrupt request passes through the S input of the flip-flop and becomes the Q output. If the interrupt input mode is changed from edge mode to level mode, the interrupt request flag is cleared automatically.

	If the CPU enters the interrupt response sequence as a result of INT0 going from 0 to 1, INT0 must then be held at 1 until the interrupt response sequence has been completed. If INT0 is set to level mode so as to release a halt state, INT0 must be held at 1 from the time INT0 changes from 0 to 1 until the halt state is released. (Hence, it is necessary to ensure that input noise is not interpreted as a 0, causing INT0 to revert to 0 before the halt state has been released.)
	When the mode changes from level mode to edge mode, interrupt request flags which were set in level mode will not be cleared. Interrupt request flags must be cleared using the following sequence.
	DI
	LD (IIMC), 00H ; Switches interrupt input mode from level mode to edge mode.
	LD (INTCLR), 0AH ; Clears interrupt request flag.
	NOP ; Wait EI instruction
	EI

INTRXn	The interrupt request flip-flop can only be cleared by reset or by reading the serial channel receive buffer. It cannot be cleared by writing INTCLR register.

Note: The following instructions or pin input state changes are equivalent to instructions that clear the interrupt request flag.
INT0: Instructions which switch to level mode after an interrupt request has been generated in edge mode.
The pin input change from high to low after interrupt request has been generated in level mode. (H → L)

INTRXn: Instruction which reads the receive buffer.

4. Port Function

The TMP91FU62 features 69 bit settings which relate to the various I/O ports.

As well as general-purpose I/O port functionality, the port pins also have I/O functions which relate to the built-in CPU and internal I/Os. Table 4-1 lists the functions of each port pin. Table 4-1 lists the functions of each port pin. Table 4-2 lists I/O registers and their specifications.

Table 4-1 Port Functions (R: PU = with programmable pull-up resistor) (1/2)

Port Names	Pin Names	Number of Pins	Direction	R	Direction Setting Unit	Pin Names for Built-in Functions
Port0	P00 to P07	8	I/O	–	Bit	
Port1	P10 to P17	8	I/O	–	Bit	
Port3	P30	1	I/O	–	Bit	TB3IN0, INT3, SDA0
	P31	1	I/O	–	Bit	TB3IN1, INT4, SCL0
	P32	1	I/O	–	Bit	TB3OUT0
	P33	1	I/O	–	Bit	TB3OUT1
Port4	P40	1	I/O	PU	Bit	SCOUT
	P41	1	I/O	PU	Bit	TXD2, RXD2
	P42	1	I/O	PU	Bit	RXD2, TXD2
	P43	1	I/O	PU	Bit	SCLK2, $\overline{\text{CTS2}}$
Port5	P50	1	I/O	–	Bit	AN0
	P51	1	I/O	–	Bit	AN1
	P52	1	I/O	–	Bit	AN2
	P53	1	I/O	–	Bit	AN3
	P54	1	I/O	–	Bit	AN4
	P55	1	I/O	–	Bit	AN5
	P56	1	I/O	–	Bit	AN6
	P57	1	I/O	–	Bit	AN7
Port6	P60	1	I/O	–	Bit	AN8
	P61	1	I/O	–	Bit	AN9
	P62	1	I/O	–	Bit	AN10
	P63	1	I/O	–	Bit	AN11
	P64	1	I/O	–	Bit	AN12
	P65	1	I/O	–	Bit	AN13
	P66	1	I/O	–	Bit	AN14
	P67	1	I/O	–	Bit	AN15
Port7	P70	1	I/O	–	Bit	TA0IN
	P71	1	I/O	–	Bit	TA1OUT
	P72	1	I/O	–	Bit	
	P73	1	I/O	–	Bit	TA4IN
	P74	1	I/O	–	Bit	TA5OUT
	P75	1	I/O	–	Bit	INT0

Table 4-1 Port Functions (R: PU = with programmable pull-up resistor) (2/2)

Port Names	Pin Names	Number of Pins	Direction	R	Direction Setting Unit	Pin Names for Built-in Functions
Port8	P80	1	I/O	–	Bit	TB0IN0, INT5
	P81	1	I/O	–	Bit	TB0IN1, INT6
	P82	1	I/O	–	Bit	TB0OUT0
	P83	1	I/O	–	Bit	TB0OUT1
	P84	1	I/O	–	Bit	TB1IN0, INT7
	P85	1	I/O	–	Bit	TB1IN1, INT8
	P86	1	I/O	–	Bit	TB1OUT0
	P87	1	I/O	–	Bit	TB1OUT1
Port9	P90	1	I/O	–	Bit	TXD0, RXD0
	P91	1	I/O	–	Bit	RXD0, TXD0
	P92	1	I/O	–	Bit	SCLK0, $\overline{\text{CTS0}}$
	P93	1	I/O	–	Bit	TXD1, RXD1
	P94	1	I/O	–	Bit	RXD1, TXD1
	P95	1	I/O	–	Bit	SCLK1, $\overline{\text{CTS1}}$
	P96	1	I/O	–	Bit	XT1
	P97	1	I/O	–	Bit	XT2
PortA	PA0	1	I/O	–	Bit	TB2IN0, INT1
	PA1	1	I/O	–	Bit	TB2IN1, INT2
	PA2	1	I/O	–	Bit	TB2OUT0
	PA3	1	I/O	–	Bit	TB2OUT1
PortB	PB0	1	I/O	–	Bit	
	PB1	1	I/O	–	Bit	
	PB2	1	I/O	–	Bit	

Table 4-2 I/O Port Setting List(1/3)

Ports	Pin Names	Specifications	I/O Register Setting Values				
			Pn	PnCR	PnFC	PnFC2	ODE
Port0	P00 to P07	Input port	×	0	None	None	None
		Output port	×	1			
Port1	P10 to P17	Input port	×	0	None	None	None
		Output port	×	1			
Port3	P30 to P31	Input port	×	0	0	0	–
		Output port (CMOS output)	×	1	0	0	0
		Output port (open drain output)	×	1	0	0	1
	P32 to P33	Input port	×	0	0	None	None
		Output port	×	1	0		
	P30	TB3IN0 Input, INT3 Input	×	0	1	0	–
		SDA0 input/output (CMOS output)	×	1	0	1	0
		SDA0 input/output (open drain output)#1	×	1	0	1	1
	P31	TB3IN1 Input, INT4 Input	×	0	1	0	–
		SCL0 input/output (CMOS output)	×	1	0	1	0
		SCL0 input/output (open drain output)#2	×	1	0	1	1
	P32	TB3OUT0 output	×	1	1	None	None
	P33	TB3OUT1 output	×	1	1		
Port4	P40, P43	Input port (without pull up)	0	0	0	0	None
		Input port (with pull up)	1	0	0	0	
		Output port	×	1	0	0	
	P41	Input port (without pull up)	0	0	0	0	–
		Input port (with pull up)	1	0	0	0	–
		Output port (CMOS output)	×	1	0	0	0
		Output port (open drain output)	×	1	0	0	1
	P42	Input port (without pull up)	0	0	0	None	None
		Input port (with pull up)	1	0	0		
		Output port	×	1	0		
	P40	SCOUT output	×	1	0	1	None
	P41	TXD2 output (CMOS output)	×	1	0	1	0
		TXD2 output (open drain output)#2	×	1	0	1	1
	P42	RXD2 Input	×	0	0	None	None
	P43	SCLK2 Input	×	0	0	0	None
		SCLK2 output	×	1	0	1	
		$\overline{\text{CTS2}}$ Input	×	0	0	0	
Port5	P50 to P57	Input port	×	0	1	None	None
		Output port	×	1	0		
		AN0 to AN7 Input #2	×	0	0		
Port6	P60 to P67	Input port	×	0	1	None	None
		Output port	×	1	0		
		AN8 to AN15 Input #3	×	0	0		

Table 4-2 I/O Port Setting List(2/3)

Ports	Pin Names	Specifications	I/O Register Setting Values				
			Pn	PnCR	PnFC	PnFC2	ODE
Port7	P70 to P75	Input port	×	0	0	None	None
		Output port	×	1	0		
	P70	TA0IN Input	×	0	None		
	P71	TA1OUT output	×	1	1		
	P73	TA4IN Input	×	0	None		
	P74	TA5OUT output	×	1	1		
	P75	INT0 Input	×	0	1		
Port8	P80 to P87	Input port	×	0	0	None	None
		Output port	×	1	0		
	P80	TB0IN0, INT5 Input	×	0	1		
	P81	TB0IN1, INT6 Input	×	0	1		
	P82	TB0OUT0 output	×	1	1		
	P83	TB0OUT1 output	×	1	1		
	P84	TB1IN0, INT7 Input	×	0	1		
	P85	TB1IN1, INT8 Input	×	0	1		
	P86	TB1OUT0 output	×	1	1		
	P87	TB1OUT1 output	×	1	1		

Table 4-2 I/O Port Setting List(3/3)

Ports	Pin Names	Specifications	I/O Register Setting Values				
			Pn	PnCR	PnFC	PnFC2	ODE
Port9	P91 to P92, P94 to P95	Input port	×	0	0	None	None
		Output port	×	1	0		
	P90, P93	Input port	×	0	0		–
		Output port (CMOS output)	×	1	0		0
		Output port (open drain output)	×	1	0		1
	P90	TXD0 output (CMOS output)	×	1	1		0
		TXD0 output (open drain output)#2	×	1	1		1
	P91	RXD0 Input	×	0	None		None
	P92	SCLK0 Input	×	0	0		None
		SCLK0 output	×	1	1		
		$\overline{\text{CTS0}}$ Input	×	0	0		
	P93	TXD1 output (CMOS output)	×	1	1		0
		TXD1 output (open drain output)#2	×	1	1		1
	P94	RXD1 Input	×	0	None		None
	P95	SCLK1 Input	×	0	0		None
		SCLK1 output	×	1	1		
		$\overline{\text{CTS1}}$ Input	×	0	0		
	P96 to P97	Input port	×	0	1		None
		Output port	×	1	1		
		XT1 to XT2 #3	×	0	0		
PortA	PA0 to PA3	Input port	×	0	0	None	None
		Output port	×	1	0		
	PA0	TB2IN0 Input, INT1 Input	×	0	1		
	PA1	TB2IN1 Input, INT2 Input	×	0	1		
	PA2	TB2OUT0	×	1	1		
	PA3	TB2OUT1	×	1	1		
PortB	PB0 to PB2	Input port	×	0	None	None	None
		Output port	×	1			

#1 If using P30/P31/P41/P90/P93 as open-drain output in SDA0/SCL0/TXD2/TXD0/TXD1 output, please set ODE.
#2 If using P50 to P57,P60 to P67 as an analog input, please set ADCCR1<SAIN3:0>.
#3 If using P96 to P97 as XT1-XT2, please set SYSCR0.

Note: ×:Don't care

4.1 Port 0 (P00 to P07)

Port 0 is an 8-bit general-purpose I/O port. Each bit can be set individually for input or output using the control register P0CR. Reset operation initializes all bits of the control register P0CR to “0” and sets port 0 to input port.

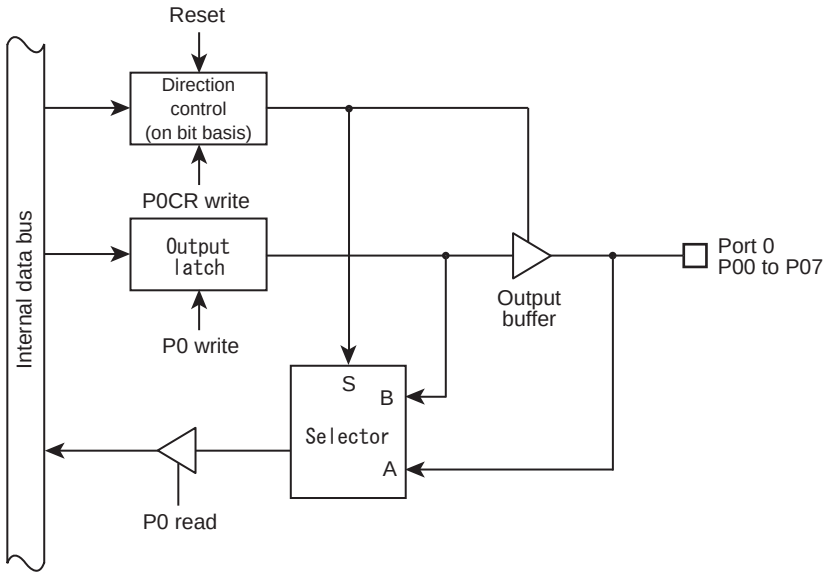


Figure 4-1 Port 0

Port 0 Register

		7	6	5	4	3	2	1	0
P0 (0000H)	Bit symbol	P07	P06	P05	P04	P03	P02	P01	P00
	Read/Write	R/W							
	After reset	Data from external port (Output latch register is undefined.)							

Port 0 Control Register (Read-modify-write instructions are prohibited.)

		7	6	5	4	3	2	1	0
P0CR (0002H)	Bit symbol	P07C	P06C	P05C	P04C	P03C	P02C	P01C	P00C
	Read/Write	W							
	After reset	0	0	0	0	0	0	0	0
	Function	0: Input 1: Output							

P0xC	P07 function	P06 function	P05 function	P04 function	P03 function	P02 function	P01 function	P00 function
0	input port	input port	input port	input port	input port	input port	input port	input port
1	output port	output port	output port	output port	output port	output port	output port	output port

Note: <P0xC> is bit X of each register P0CR.

4.2 Port 1 (P10 to P17)

Port 1 is an 8-bit general-purpose I/O port. Each bit can be set individually for input or output using the control register P1CR. Reset operation initializes all bits of output latch P1, the control register P1CR to “0” and sets port 1 to input port.

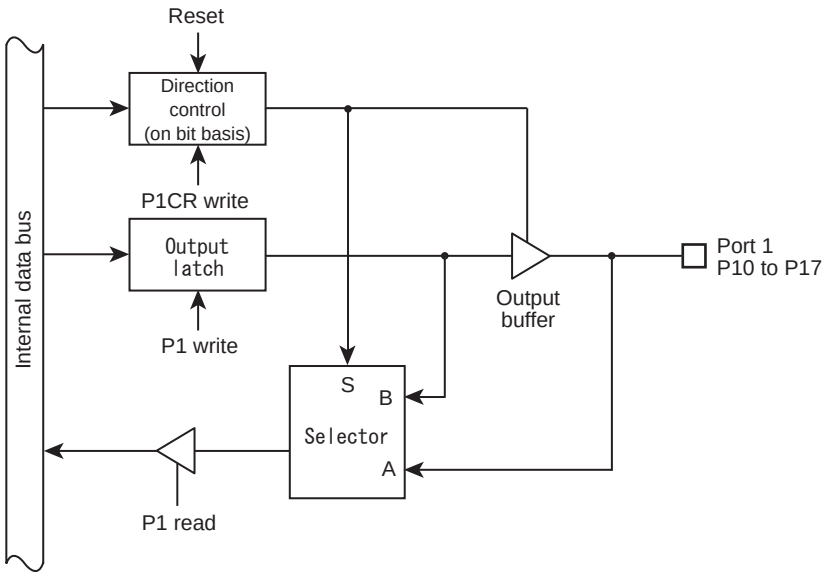


Figure 4-2 Port 1

Port 1 Register

P1 (0001H)

	7	6	5	4	3	2	1	0
Bit symbol	P17	P16	P15	P14	P13	P12	P11	P10
Read/Write	R/W							
After reset	Data from external port (Output latch register is cleared to “0”.)							

Port 1 Control Register (Read-modify-write instructions are prohibited.)

P1CR (0004H)

	7	6	5	4	3	2	1	0
Bit symbol	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

P1xC	P17 function	P16 function	P15 function	P14 function	P13 function	P12 function	P11 function	P10 function
0	input port	input port	input port	input port	input port	input port	input port	input port
1	output port	output port	output port	output port	output port	output port	output port	output port

Note:<P1XC> is bit X of each register P1CR.

4.3 Port3 (P30 to P33)

Port 3 is an 4-bit general-purpose I/O port. Reset operation initializes to input port. All bits of output latch register P3 are set to “1”.

There are the following functions in addition to an I/O port. This function enable each function by writing “1” to applicable bit of port 3 function register P3FC.

- The input function of external interrupt (INT3, INT4)
- The input function of 16-bit timer 3 (TB3IN0, TB3IN1)
- The output function of 16-bit timer 3 (TB3OUT0, TB3OUT1)
- The I/O function of serial bus interface 0 (SDA0, SCL0)

Reset operation initializes, P3CR,P3FC and P3FC2 to “0”, all bits are set to input port.

And Port 30 and 31 have a programmable open-drain function which can be controlled by the ODE register.

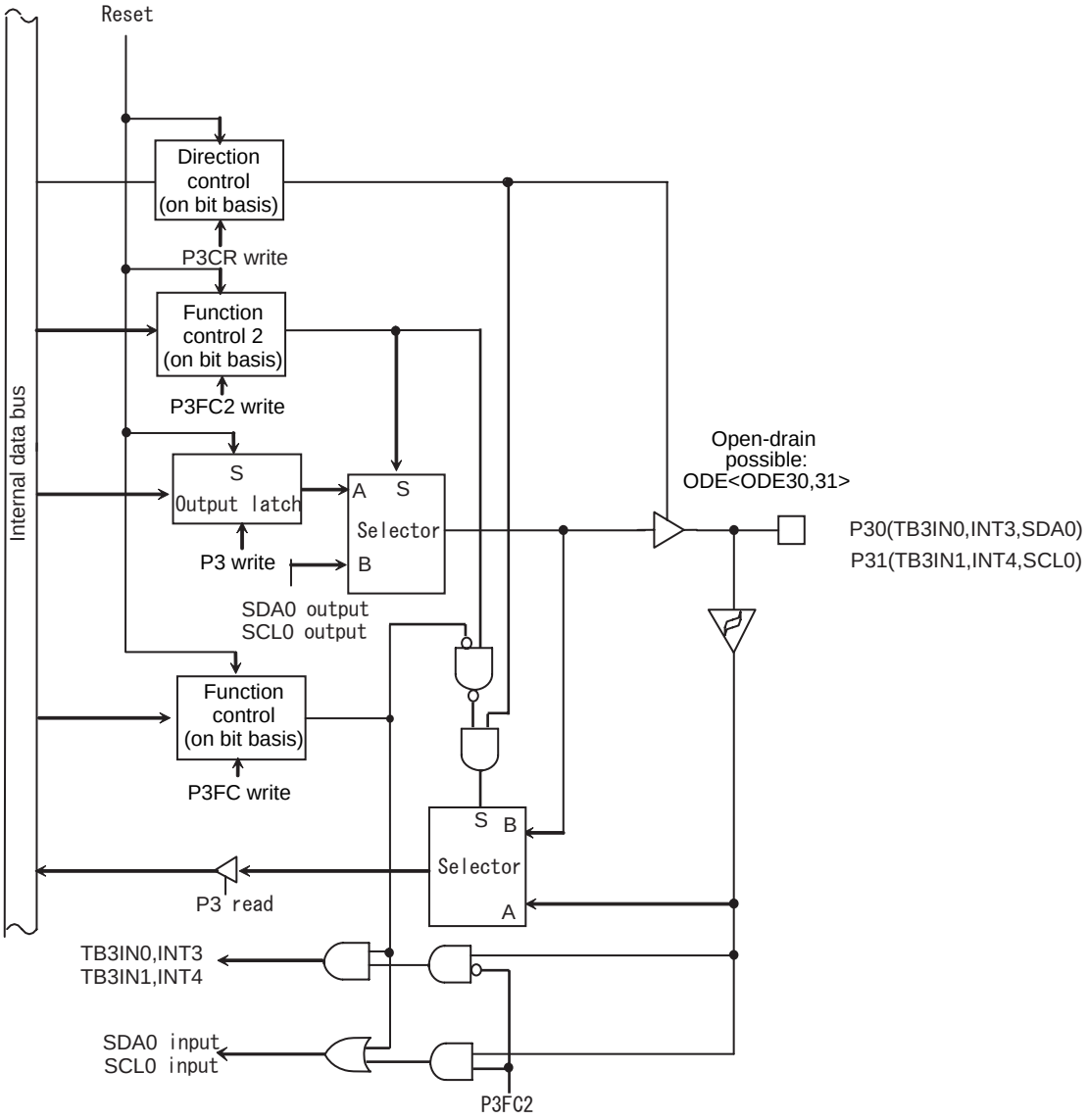


Figure 4-3 Port 30 and 31

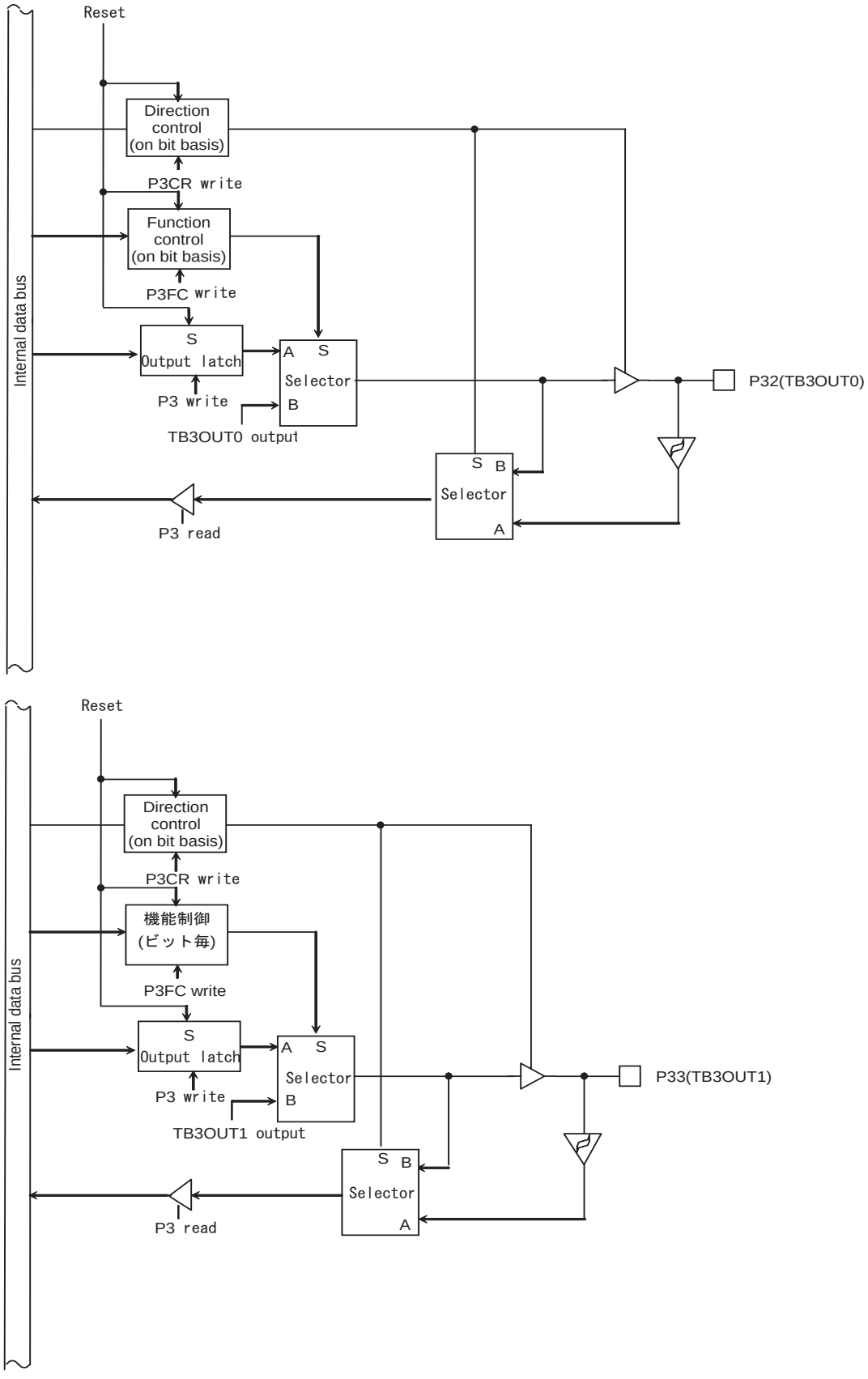


Figure 4-4 Port 32 and 33

Port 3 Register

P3 (000CH)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	–	P33	P32	P31	P30
Read/Write	–	–	–	–	R/W			
After reset	–	–	–	–	Data from external port (Output latch register is set to “1”.)			
Function	-				output mode			

Port 3 Control Register (Read-modify-write instructions are prohibited.)

P3CR (000EH)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	–	P33C	P32C	P31C	P30C
Read/Write	–	–	–	–	W			
After reset	–	–	–	–	0	0	0	0
Function	-				0:Input 1:Output			

Port 3 Function Register (Read-modify-write instructions are prohibited.)

P3FC (000FH)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	–	P33F	P32F	P31F	P30F
Read/Write	–	–	–	–	W			
After reset	–	–	–	–	0	0	0	0

Port 3 Function Register 2 (Read-modify-write instructions are prohibited.)

P3FC2 (000DH)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	–	–	–	P31F2	P30F2
Read/Write	–	–	–	–	–	–	W	
After reset	–	–	–	–	–	–	0	0

P3xF2	P3xF	P3xC	P33 function	P32 function	P31 function	P30 function
0	0	0	input port	input port	input port	input port
0	0	1	output port	output port	output port	output port
0	1	0	reserved	reserved	TB3IN1/INT4	TB3IN0/INT3
0	1	1	TB3OUT1	TB3OUT0	reserved	reserved
1	0	0	reserved	reserved	reserved	reserved
1	0	1	reserved	reserved	SCL0	SDA0
1	1	0	reserved	reserved	reserved	reserved
1	1	1	reserved	reserved	reserved	reserved

Note 1: <P3xF2>/<P3xF>/<P3xC> is bit X of each register P3FC2/P3FC/P3CR.

4.4 Port 4 (P40 to P43)

Port 4 is an 4-bit general-purpose I/O port. Reset operation initializes to input port, and connects a pull-up resistor. All bits of output latch register P4 are set to “1”.

There are the following functions in addition to an I/O port. This function enable each function by writing “1” to applicable bit of port 4 function register P4FC.

- The I/O function of the serial channel 2 (RXD2, TXD2, SCLK2/$\overline{\text{CTS2}}$)
- The output function of a system clock signal (SCOUT)

Reset operation initializes, P4CR,P4FC and P4FC2 to “0”, all bits are set to input port.

And Port 41 have a programmable open-drain function which can be controlled by the ODE register.

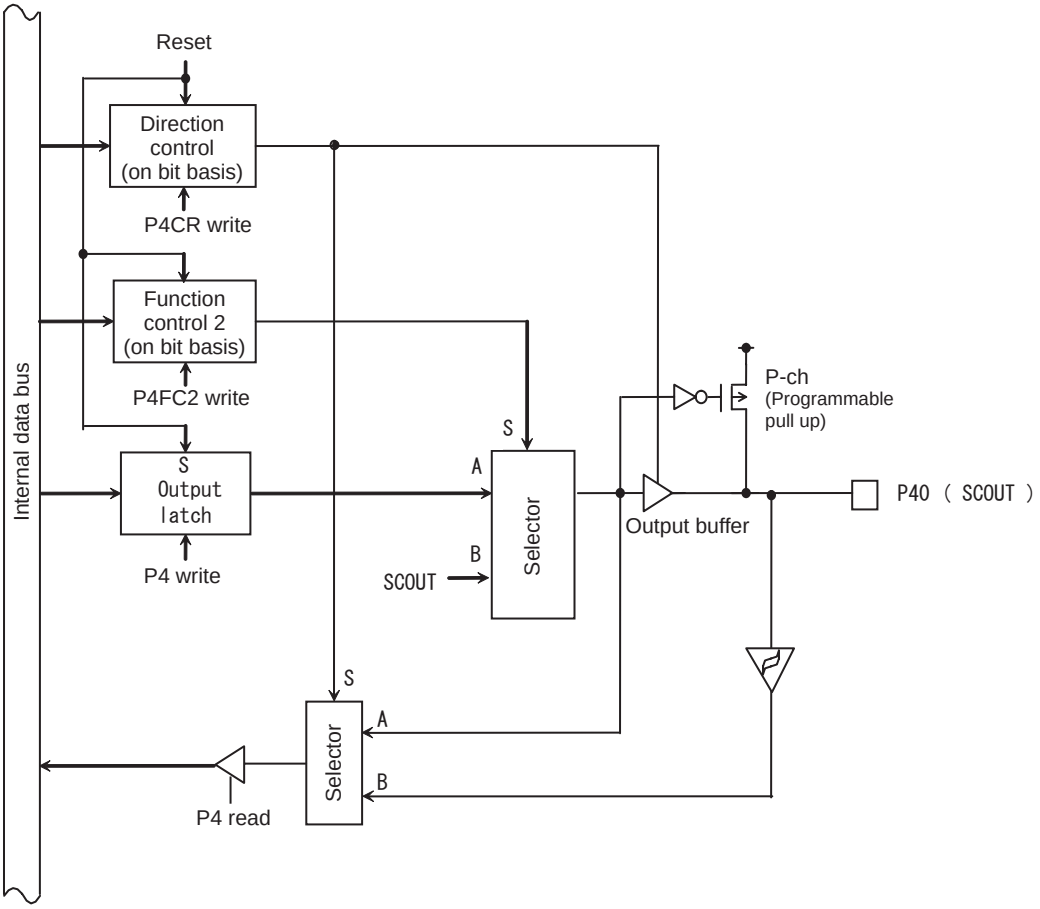


Figure 4-5 Port 40

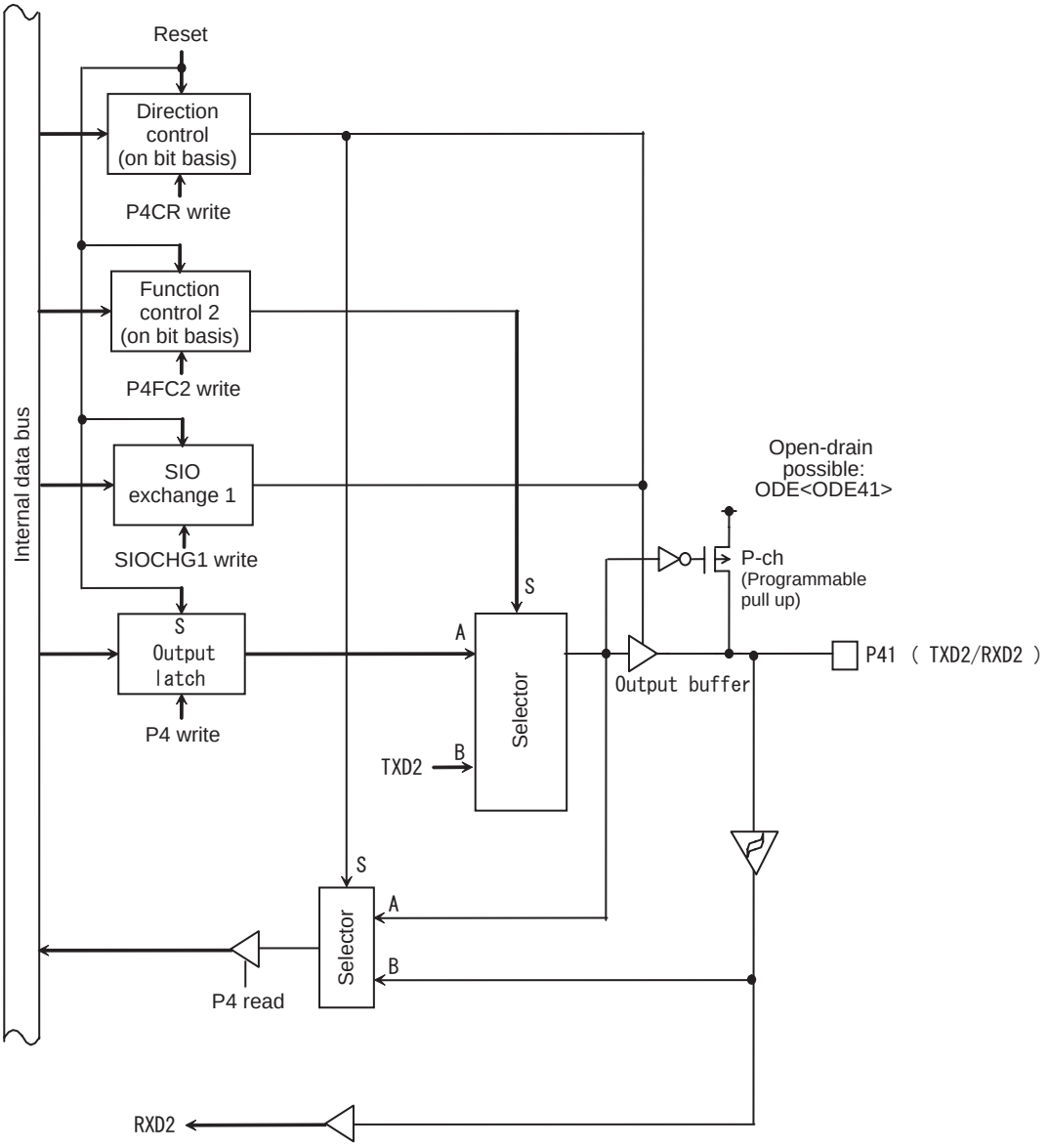


Figure 4-6 Port 41

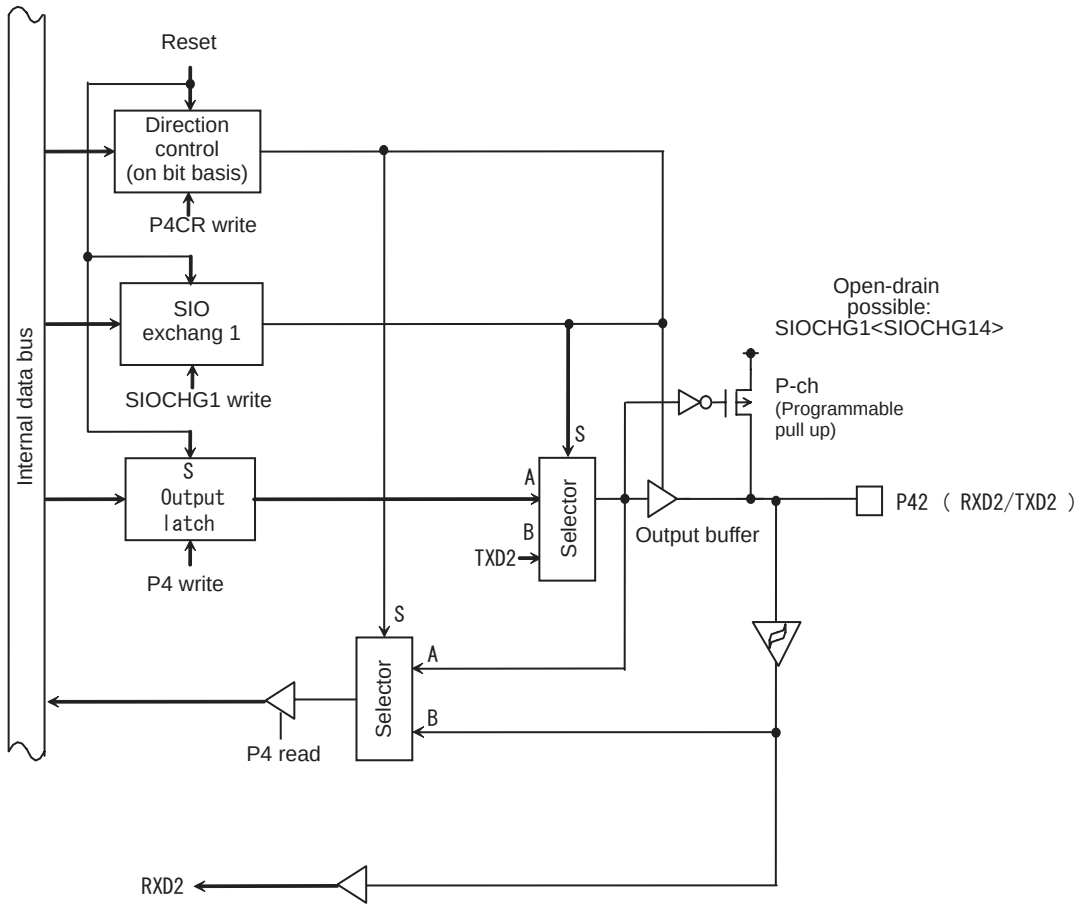


Figure 4-7 Port 42

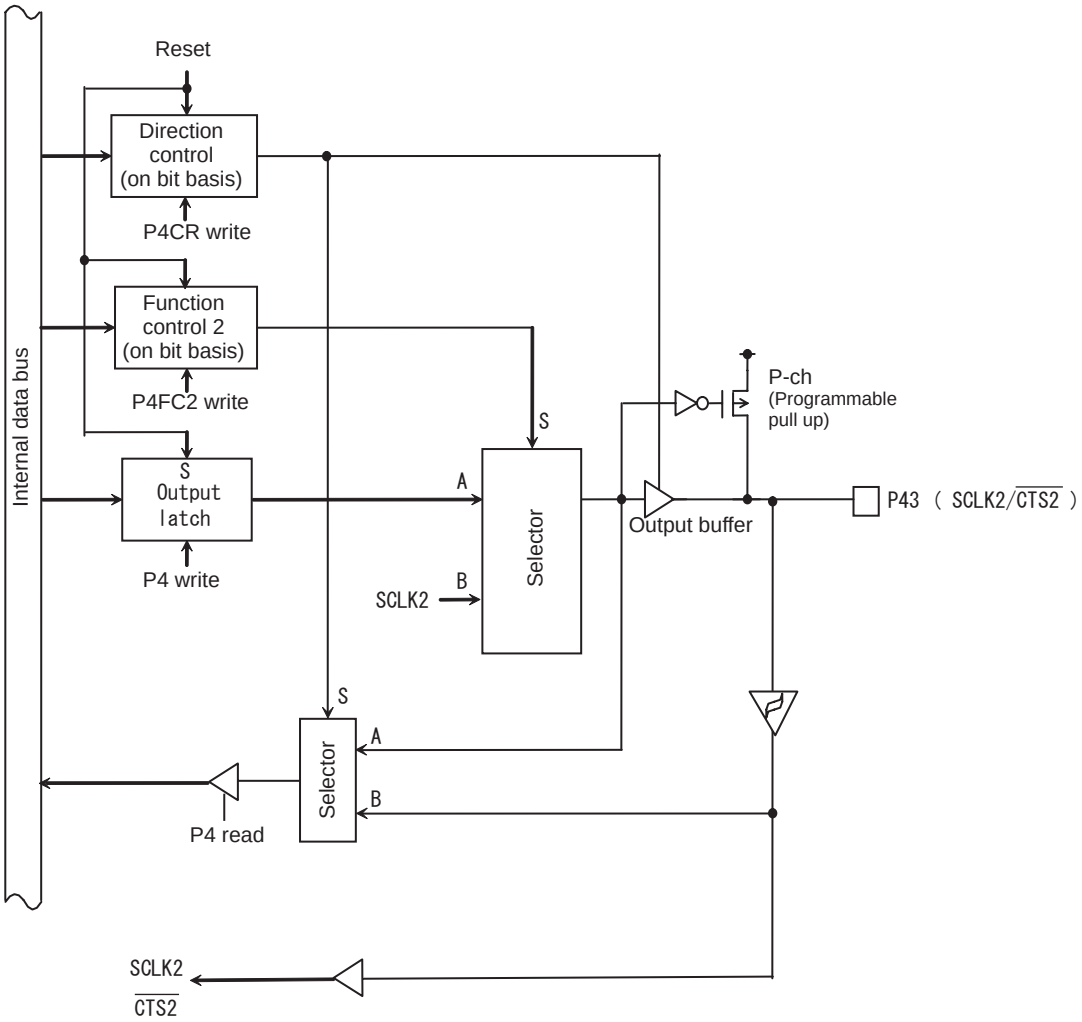


Figure 4-8 Port 43

Port 4 Register

P4 (0010H)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	–	P43	P42	P41	P40
Read/Write	–	–	–	–	R/W			
After reset	–	–	–	–	Data from external port (Output latch register is set to “1”.)			
Function					0 (Output latch register): Pull-up resistor OFF 1 (Output latch register): Pull-up resistor ON			

Port 4 Control Register (Read-modify-write instructions are prohibited.)

P4CR (0012H)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	–	P43C	P42C	P41C	P40C
Read/Write	–	–	–	–	W			
After reset	–	–	–	–	0	0	0	0
Function					0: Input 1: Output			

Port 4 Function Register 2 （Read-modify-write instructions are prohibited.)

P4FC2 (0011H)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	–	P43F2	–	P41F2	P40F2
Read/Write	–	–	–	–	W	–	W	
After reset	–	–	–	–	0	–	0	0

P4xF2	P4xC	P43 function	P42 function	P41 function	P40 function
0	0	input port (SCLK2/CTS2)	input port (RXD2)	input port	input port
0	1	output port	output port	output port	output port
1	0	reserved	reserved	reserved	reserved
1	1	SCLK2	reserved	TXD2	SCOUT

Note 1: <P4xF2>/<P4xC> is bit X of each register P4FC2/P4CR.

Note 2: When port 4 is used as input mode, P4 register controls internal pull-up resistor. Read-modify-write instruction is prohibited in input mode or I/O mode. Setting the internal pull-up resistor may be depended on the states of the input pin.

Note 3: When setting TXD2 pin to open-drain output, write “1” to bit2 of ODE register. P42/RXD2 pin does not have a register which changes Port/Function. For example, when it is also used as an input port, the input signal is inputted to SIO as serial receiving data.

4.5 Port 5 (P50 to P57)

Port 5 is an 8-bit general-purpose I/O port. By the reset action, it becomes Hi-Z and becomes analog input permission.All bits of output latch register P5 are set to “1”.

There are the following functions in addition to an I/O port.

- The input function of the Analog/Digital Converter (AN0 to AN7)

Reset operation initializes, P5CR,P5FC to “0”, all bits are set to input port.

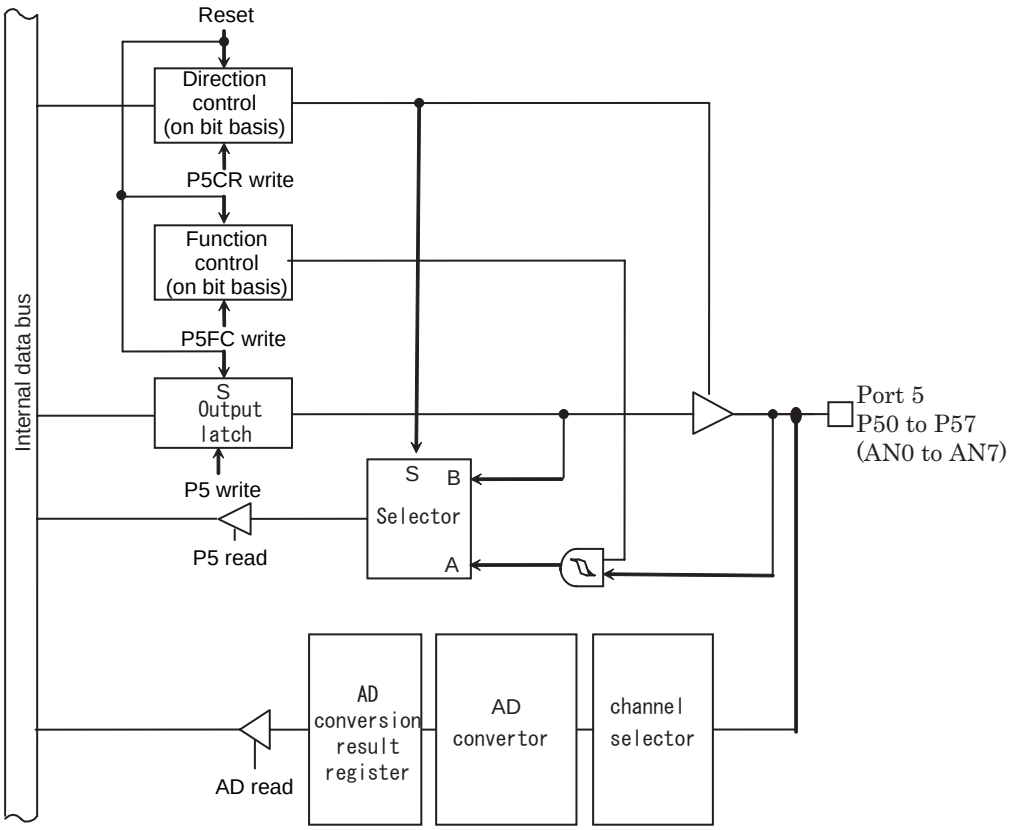


Figure 4-9 Port 5

Port 5 Register

P5 (0014H)

	7	6	5	4	3	2	1	0
Bit symbol	P57	P56	P55	P54	P53	P52	P51	P50
Read/Write	R/W							
After reset	Data from external port (Output latch register is set to “1”.)							

Port 5 Control Register (Read-modify-write instructions are prohibited.)

P5CR (0016H)

	7	6	5	4	3	2	1	0
Bit symbol	P57C	P56C	P55C	P54C	P53C	P52C	P51C	P50C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port 5 Function Register (Read-modify-write instructions are prohibited.)

P5FC (0017H)

	7	6	5	4	3	2	1	0
Bit symbol	P57F	P56F	P55F	P54F	P53F	P52F	P51F	P50F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	P57 input 0:disable 1:enable	P56 input 0:disable 1:enable	P55 input 0:disable 1:enable	P54 input 0:disable 1:enable	P53 input 0:disable 1:enable	P52 input 0:disable 1:enable	P51 input 0:disable 1:enable	P50 input 0:disable 1:enable

P5xF	P5xC	P57 function	P56 function	P55 function	P54 function	P53 function	P52 function	P51 function	P50 function
0	0	input disable	input disable	input disable	input disable	input disable	input disable	input disable	input disable
0	1	output port	output port	output port	output port	output port	output port	output port	output port
1	0	input enable	input enable	input enable	input enable	input enable	input enable	input enable	input enable
1	1	output port	output port	output port	output port	output port	output port	output port	output port

Note 1: <P5xF>/<P5xC> is bit X of each register P5FC/P5CR.
Note 2: The input channel selection of AD converter are set by AD converter mode register ADCCR1.

4.6 Port 6 (P60 to P67)

Port 6 is an 8-bit general-purpose I/O port. By the reset action, it becomes Hi-Z and becomes analog input permission.All bits of output latch register P6 are set to “1”.

There are the following functions in addition to an I/O port.

- The input function of the Analog/Digital Converter (AN8 to AN15)

Reset operation initializes, P6CR,P6FC to “0”, all bits are set to input port.

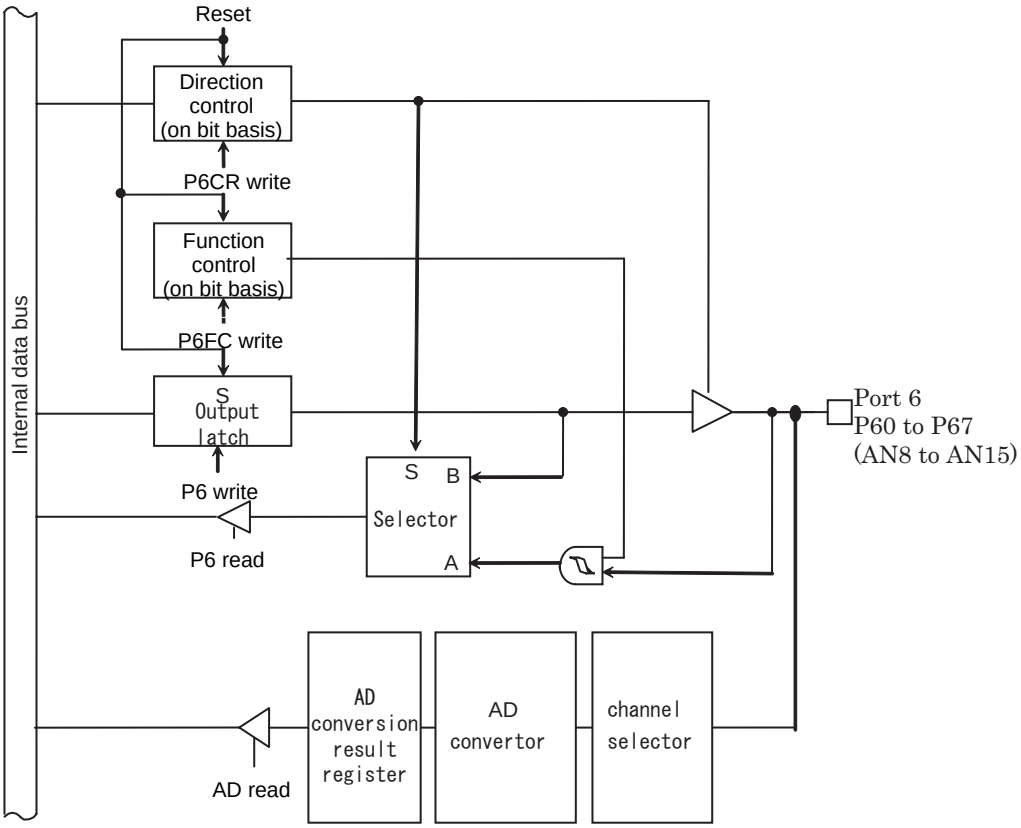


Figure 4-10 Port 6

Port 6 Register

P6 (0018H)

	7	6	5	4	3	2	1	0
Bit symbol	P67	P66	P65	P64	P63	P62	P61	P60
Read/Write	R/W							
After reset	Data from external port (Output latch register is set to “1”.)							

Port 6 Control Register (Read-modify-write instructions are prohibited.)

P6CR (001AH)

	7	6	5	4	3	2	1	0
Bit symbol	P67C	P66C	P65C	P64C	P63C	P62C	P61C	P60C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port 6 Function Register (Read-modify-write instructions are prohibited.)

P6FC (001BH)

	7	6	5	4	3	2	1	0
Bit symbol	P67F	P66F	P65F	P64F	P63F	P62F	P61F	P60F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	P67 input 0:disable 1:enable	P66 input 0:disable 1:enable	P65 input 0:disable 1:enable	P64 input 0:disable 1:enable	P63 input 0:disable 1:enable	P62 input 0:disable 1:enable	P61 input 0:disable 1:enable	P60 input 0:disable 1:enable

P6xF	P6xC	P67 function	P66 function	P65 function	P64 function	P63 function	P62 function	P61 function	P60 function
0	0	input disable	input disable	input disable	input disable	input disable	input disable	input disable	input disable
0	1	output port	output port	output port	output port	output port	output port	output port	output port
1	0	input enable	input enable	input enable	input enable	input enable	input enable	input enable	input enable
1	1	output port	output port	output port	output port	output port	output port	output port	output port

Note 1: <P6xF>/<P6xC> is bit X of each register P6FC/P6CR.
Note 2: The input channel selection of AD converter are set by AD converter mode register ADCCR1.

4.7 Port 7 (P70 to P75)

Port 7 is an 6-bit general-purpose I/O port. Reset operation initializes to input port. All bits of output latch register P7 are set to “1”.

There are the following functions in addition to an I/O port. This function enable each function by writing “1” to applicable bit of port 7 function register P7FC.

- The I/O function of 8-bit timer 01 (TA0IN,TA1OUT)
- The I/O function of 8-bit timer 45 (TA4IN,TA5OUT)
- The input function of external interrupt (INT0)

Reset operation initializes, P7CR and P7FC to “0”, all bits are set to input port.

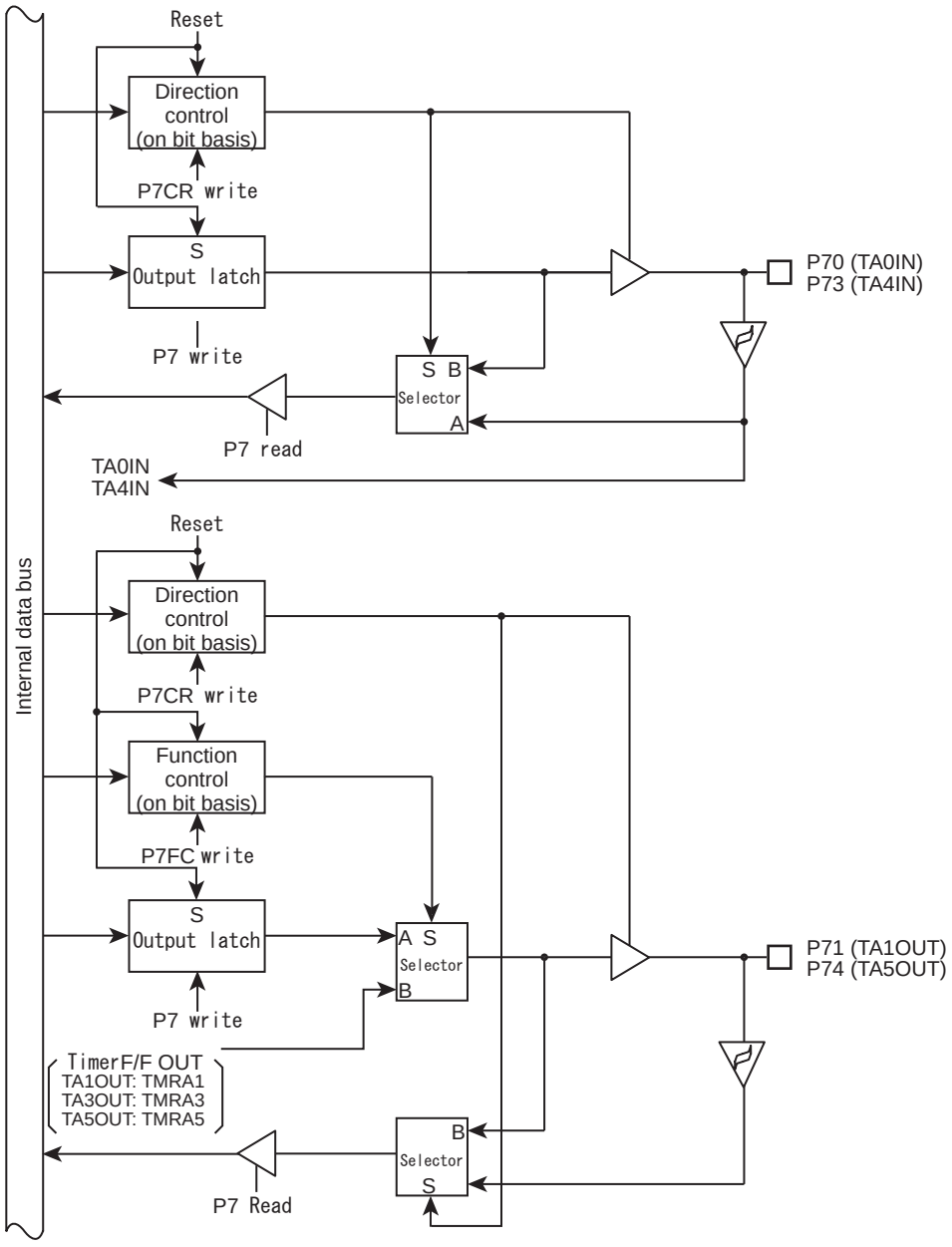


Figure 4-11 Port 70, 71, 73 and 74

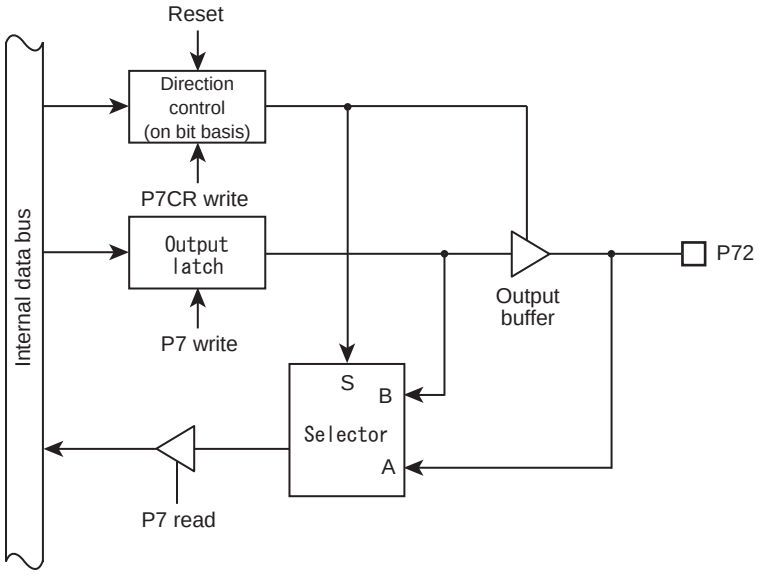


Figure 4-12 Port 72

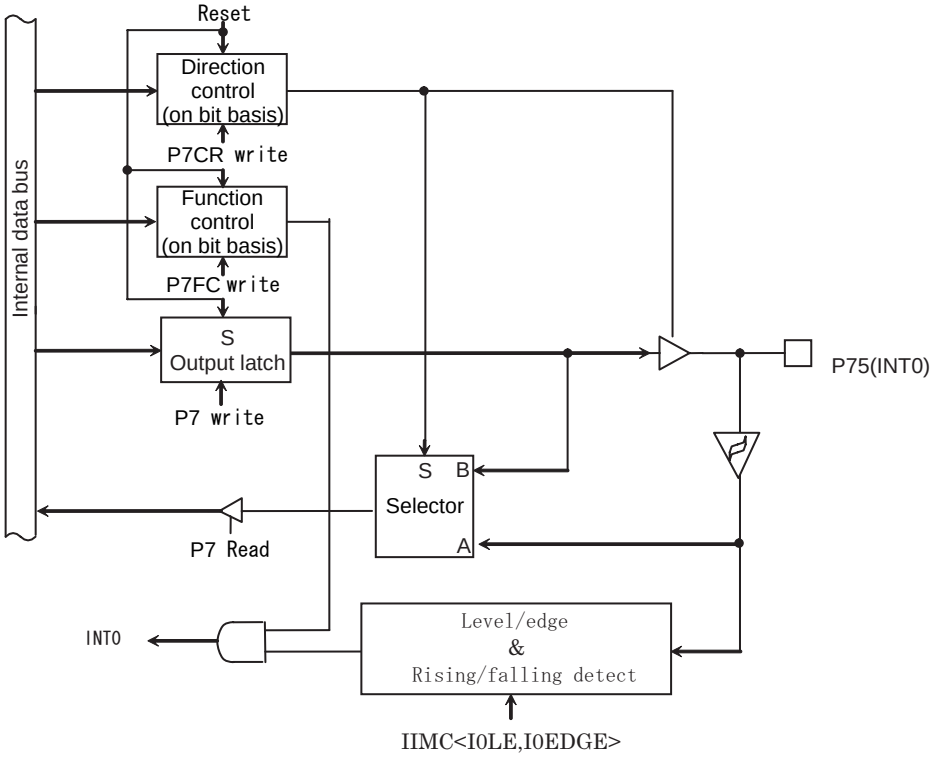


Figure 4-13 Port 75

Port 7 Register

P7 (001CH)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	P75	P74	P73	P72	P71	P70
Read/Write	–	–	R/W					
After reset	–	–	Data from external port (Output latch register is set to “1”.)					

Port 7 Control Register (Read-modify-write instructions are prohibited.)

P7CR (001EH)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	P75C	P74C	P73C	P72C	P71C	P70C
Read/Write	–	–	W					
After reset	–	–	0	0	0	0	0	0
Function			0: Input 1: Output					

Port 7 Function Register (Read-modify-write instructions are prohibited.)

P7FC (001FH)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	P75F	P74F	–	–	P71F	–
Read/Write	–	–	W		–	–	W	–
After reset	–	–	0	0	–	–	0	–
Function			0: port 1: INT0	0: port 1: TA5OUT			0: port 1: TA1OUT	

P75 INT0 setting

<P75F>	<IOLE>	<IOEDGE>	INT0
1	0	0	Rising edge detect INT
1	0	1	falling edge detect INT
1	1	0	H level INT
1	1	1	L level INT

P7xF	P7xC	P75 function	P74 function	P73 function	P72 function	P71 function	P70 function
0	0	input port	input port	input port (TA4IN)	input port	input port	input port (TA0IN)
0	1	output port	output port	output port	output port	output port	output port
1	0	INT0	reserved	reserved	reserved	reserved	reserved
1	1	reserved	TA5OUT	reserved	reserved	TA1OUT	reserved

Note 1: <P7xF>/<P7xC> is bit X of each register P7FC/P7CR.

Note 2: P70/TA0IN, P73/TA4IN pin dose not have a register changing PORT/FUNCTION. For example, when it is used as an input port, the input signal is inputted to 8bit Timer.

4.8 Port 8 (P80 to P87)

Port 8 is an 8-bit general-purpose I/O port. Reset operation initializes to input port. All bits of output latch register P8 are set to “1”.

There are the following functions in addition to an I/O port. This function enable each function by writing “1” to applicable bit of port 8 function register P8FC.

- The I/O function of 16-bit timer 0 (TB0IN0,TB0IN1,TB0OUT0,TB0OUT1)
- The I/O function of 16-bit timer 1 (TB1IN0,TB1IN1,TB1OUT0,TB1OUT1)
- The input function of external interrupt (INT5 to INT8)

Reset operation initializes, P8CR and P8FC to “0”, all bits are set to input port.

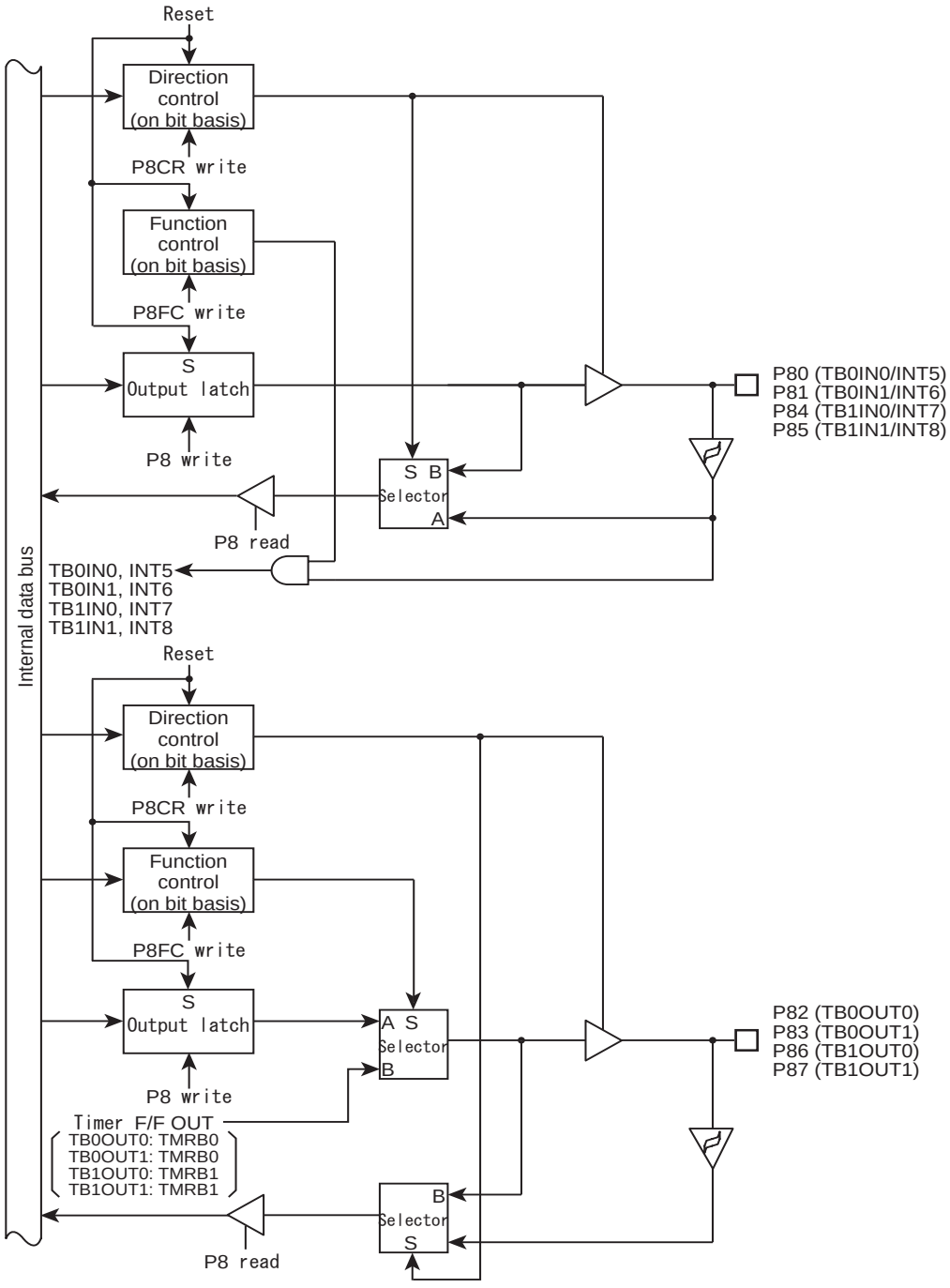


Figure 4-14 Port 8

Port 8 Register

P8 (0020H)

	7	6	5	4	3	2	1	0
Bit symbol	P87	P86	P85	P84	P83	P82	P81	P80
Read/Write	R/W							
After reset	Data from external port (Output latch register is set to “1”.)							

Port 8 Control Register (Read-modify-write instructions are prohibited.)

P8CR (0022H)

	7	6	5	4	3	2	1	0
Bit symbol	P87C	P86C	P85C	P84C	P83C	P82C	P81C	P80C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: Input 1: Output							

Port 8 Function Register (Read-modify-write instructions are prohibited.)

P8FC (0023H)

	7	6	5	4	3	2	1	0
Bit symbol	P87F	P86F	P85F	P84F	P83F	P82F	P81F	P80F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	0: port 1: TB1OUT1	0: port 1: TB1OUT0	0: port 1: TB1IN1, INT8	0: port 1: TB1IN0, INT7	0: port 1: TB0OUT1	0: port 1: TB0OUT0	0: port 1: TB0IN1, INT6	0: port 1: TB0IN0, INT5

P8xF	P8xC	P87 function	P86 function	P85 function	P84 function	P83 function	P82 function	P81 function	P80 function
0	0	input port	input port	input port	input port	input port	input port	input port	input port
0	1	output port	output port	output port	output port	output port	output port	output port	output port
1	0	reserved	reserved	TB1IN1/ INT8	TB1IN0/ INT7	reserved	reserved	TB0IN1/ INT6	TB0IN0/ INT5
1	1	TB1OUT1	TB1OUT0	reserved	reserved	TB0OUT1	TB0OUT0	reserved	reserved

Note: <P8xF>/<P8xC> is bit X of each register P8FC/P8CR.

4.9 Port 9 (P90 to P97)

- Port 90 to 95

  Port 90 to 95 are a 6-bit general-purpose I/O port. Reset operation initializes to input port. All bits of output latch register are set to “1”.

  In addition to functioning as a I/O port, port 90 to 95 can also function as I/O of SIO0, SIO1. This function enable each function by writing “1” to applicable bit of port 9 function register P9FC.

  Reset operation initializes P9CR and P9FC to “0”, all bits are set to input port.

- Port 96 to 97

  Port 96 to 97 are a 2-bit general-purpose I/O port. In case of output port, this is open drain output. Reset operation initializes output latch register and control register to “1”, and it is set to “High-Z” (High impedance).

  In addition to functioning as a I/O port, port 96 to 97 can also function as low-frequency oscillator connection pin (XT1 and XT2) during using low speed clock function. Therefore, dual clock function can use by setting of system clock control registers SYSCR0 and SYSCR1.

4.9.1 Port 90 (TXD0/RXD0), 93 (TXD1/RXD0)

In addition to functioning as a I/O port, Port 90 and 93 can also function as TXD output pin or RXD input pin of serial channel.

And Port 90 and 93 have a programmable open-drain function which can be controlled by the ODE register.

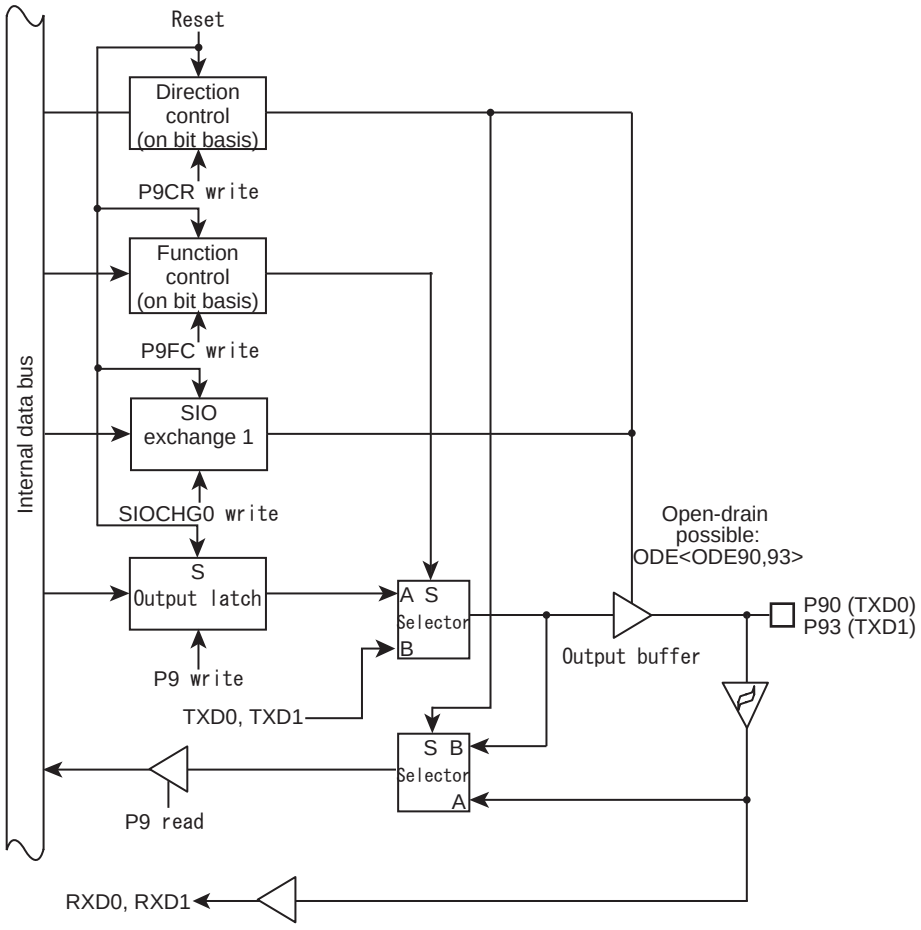


Figure 4-15 Port 90 and 93

4.9.2 Port 91 (RXD0/TXD0), 94 (RXD1/TXD1)

In addition to functioning as a I/O port, port 91 and 94 can also function as RXD input pin or TXD output pin of serial channel.

And Port 91 and 94 have a programmable open-drain function which can be controlled by the SIOCHG0 register.

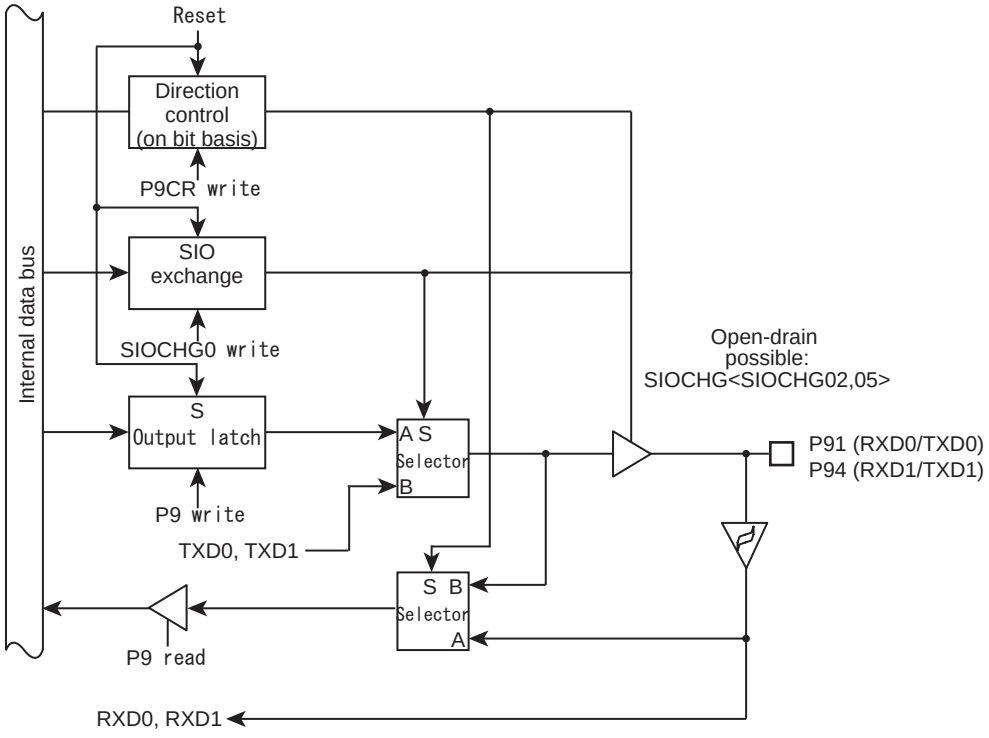


Figure 4-16 Port 91 and 94

4.9.3 Port 92($\overline{\text{CTS0}}$/SCLK0), 95 ($\overline{\text{CTS1}}$/SCLK1)

In addition to functioning as a I/O port, port 92 and 95 can also function as $\overline{\text{CTS}}$ input pin or SCLK I/O pin of serial channel.

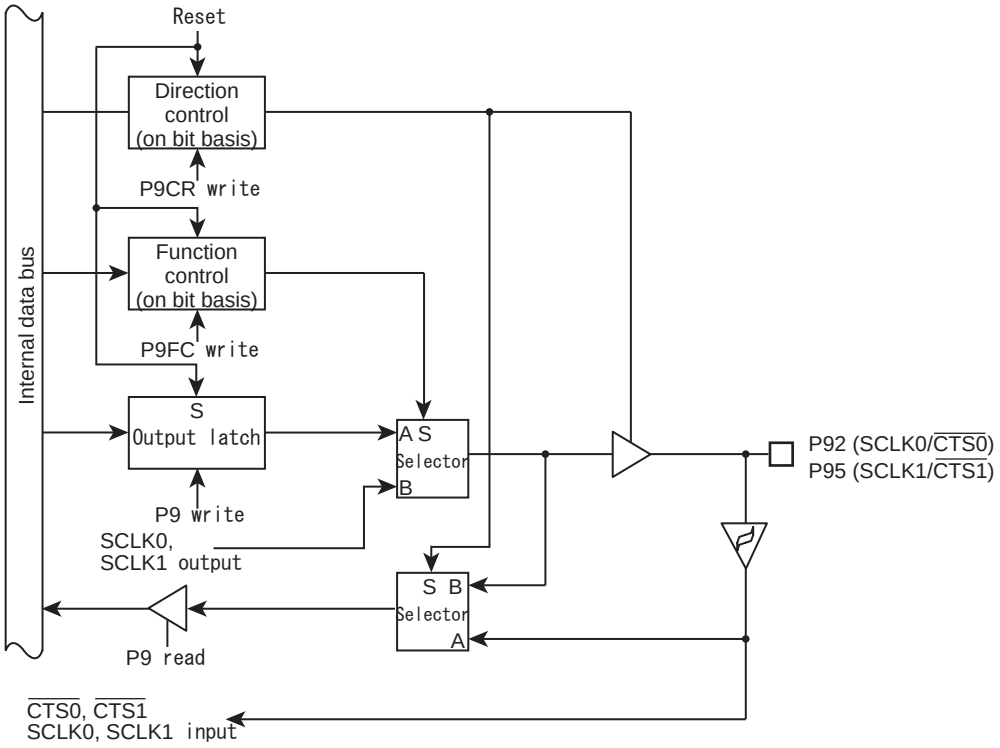


Figure 4-17 Port 92 and 95

4.9.4 Port 96 (XT1), 97 (XT2)

In addition to functioning as a I/O port, port 96 and 97 can also function as low frequency oscillator connection pins.

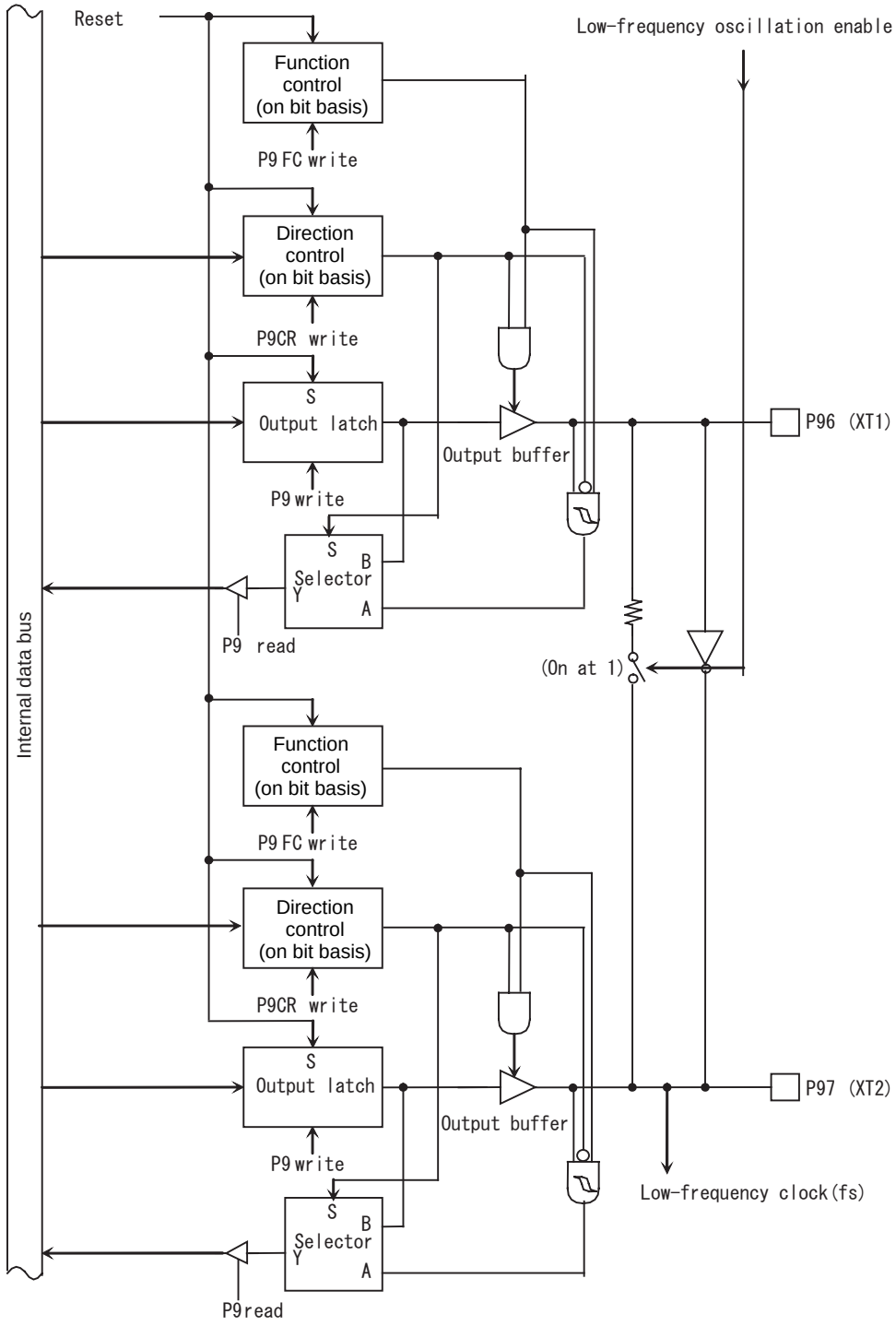


Figure 4-18 Port 96 and 97

Port 9 Register

P9 (0024H)

	7	6	5	4	3	2	1	0
Bit symbol	P97	P96	P95	P94	P93	P92	P91	P90
Read/Write	R/W							
After reset	Data from external port (Output latch register is set to “1”.)							

Port 9 Control Register (Read-modify-write instructions are prohibited.)

P9CR (0026H)

	7	6	5	4	3	2	1	0
Bit symbol	P97C	P96C	P95C	P94C	P93C	P92C	P91C	P90C
Read/Write	W							
After reset	1	1	0	0	0	0	0	0
Function	0: Input 1: Output							

Port 9 Function Register (Read-modify-write instructions are prohibited.)

P9FC (0027H)

	7	6	5	4	3	2	1	0
Bit Symbol	P97F	P96F	P95F	–	P93F	P92F	–	P90F
Read/Write	W			–	W		–	W
After reset	0	0	0	–	0	0	–	0
Function	Port 0: disable 1: enable	Port 0: disable 1: enable	0: port 1: SCLK1 output		0: port 1: TXD1 output	0: port 1: SCLK0 output		0: port 1: TXD0 output

P9xF	P9xC	P97 function	P96 function	P95 function	P94 function	P93 function	P92 function	P91 function	P90 function
0	0	XT2	XT1	input port (SCLK1/$\overline{\text{CTS1}}$)	input port (RXD1)	input port	input port (SCLK0/$\overline{\text{CTS0}}$)	input port (RXD0)	input port
0	1	reserved	reserved	output port	output port	output port	output port	output port	output port
1	0	input port	input port	reserved	reserved	reserved	reserved	reserved	reserved
1	1	output port	output port	SCLK1	reserved	TXD1	SCLK0	reserved	TXD0

Note 1: <P9xF>/<P9xC> is bit X of each register P9FC/P9CR.

Note 2: When setting TXD pin to open-drain output, write “1” to bit3 of ODE register (for TXD0 pin), or bit4 (for TXD1 pin). P91/RXD0 and P94/RXD1 pin does not have a register which changes Port/Function.
For example, when it is also used as an input port, the input signal is inputted to SIO as serial receiving data.

Note 3: Low frequency oscillation circuit
To connect a low frequency resonator to port 96 and 97, it is necessary to set a following procedure to reduce the consumption power supply.
(Case of resonator connection)
P9CR<P96C, P97C> = “11”, P9<P96:97> = “00”
(Case of external clock input)
P9CR<P96C, P97C> = “11”, P9<P96:97> = “10”

4.10 Port A (PA0 to PA3)

Port A is an 4-bit general-purpose I/O port. Reset operation initializes to input port. All bits of output latch register PA are set to “1”.

There are the following functions in addition to an I/O port. This function enable each function by writing “1” to applicable bit of port A function register PAFC.

- The I/O function of 16-bit timer 2 (TB2IN0,TB2IN1,TB2OUT0,TB2OUT1)
- The input function of external interrupt (INT1, INT2)

Reset operation initializes, PACR and PAFC to “0”, all bits are set to input port.

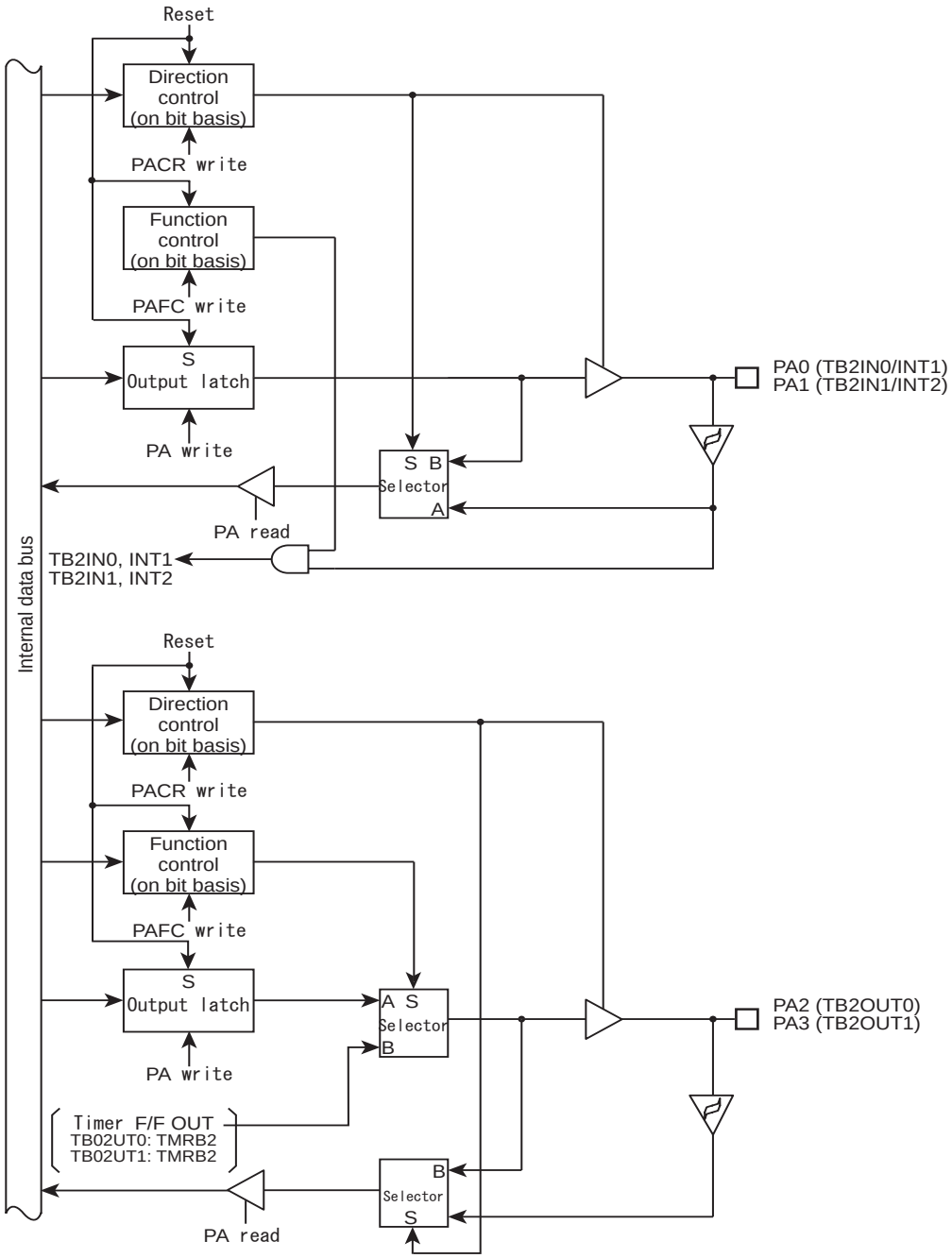


Figure 4-19 Port A

Port A Register

PA (0028H)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	–	PA3	PA2	PA1	PA0
Read/Write	–	–	–	–	R/W			
After reset	–	–	–	–	Data from external port (Output latch register is set to “1”.)			

Port A Control Register (Read-modify-write instructions are prohibited.)

PACR (002AH)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	–	PA3C	PA2C	PA1C	PA0C
Read/Write	–	–	–	–	W			
After reset	–	–	–	–	0	0	0	0
Function	–	–	–	–	0: Input 1: Output			

Port A Function Register (Read-modify-write instructions are prohibited.)

PAFC (002BH)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	–	PA3F	PA2F	PA1F	PA0F
Read/Write	–	–	–	–	W			
After reset	–	–	–	–	0	0	0	0
Function	–	–	–	–	0: port 1: TB2OUT1	0:port 1: TB2OUT0	0: port 1: TB2IN1, INT2	0: port 1: TB2IN0, INT1

PAxC	PAxF	PA3 function	PA2 function	PA1 function	PA0 function
0	0	input port	input port	input port	input port
0	1	output port	output port	output port	output port
1	0	reserved	reserved	TB2IN1/ INT2	TB2IN0/INT1
1	1	TB2OUT1	TB2OUT0	reserved	reserved

Note: <PAxF>/<PAxC> is bit X of each register PAFC/PACR.

4.11 Port B (PB0 to PB2)

Port B is an 3-bit general-purpose I/O port. Reset operation initializes to input port. All bits of output latch register PB are set to “1”.

Reset operation initializes, PBCR to “0”, all bits are set to input port.

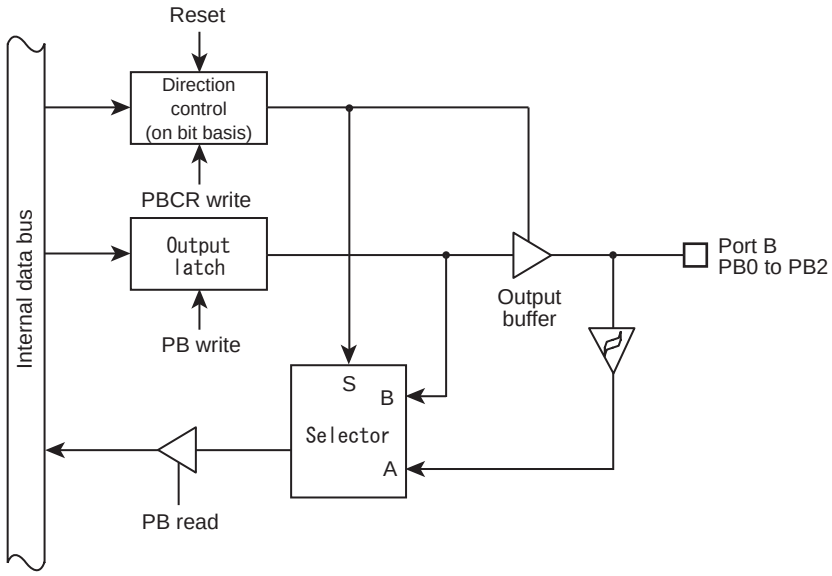


Figure 4-20 Port B0 to B2

Port B Register

PB (002CH)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	–	–	PB2	PB1	PB0
Read/Write	–	–	–	–	–	R/W		
After reset	–	–	–	–	–	Data from external port (Output latch register is set to “1”.)		

Port B Control Register (Read-modify-write instructions are prohibited.)

PBCR (002EH)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	–	–	PB2C	PB1C	PB0C
Read/Write	–	–	–	–	–	W		
After reset	–	–	–	–	–	0	0	0
Function	–	–	–	–	–	0: Input 1: Output		

PBxC	PB2 function	PB1 function	PB0 function
0	input port	input port	input port
1	output port	output port	output port

Note: <PBxC> is bit X of each register PBCR.

4.12 Open-drain Control

P30,P31,P41,P90,P93 can perform selection of an open-drain output per bit. Reset operation initializes all bits of the control register ODE to “0” and sets to CMOS output.

Open-drain Control Register

ODE (003FH)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	ODE93	ODE90	ODE41	ODE31	ODE30
Read/Write	–	–	–	R/W				
After reset	–	–	–	0	0	0	0	0
Function				0: CMOS output 1:Open drain output				

4.13 Serial pins switching / Open-drain output Control

TXD pin and RXD pin for a serial channel are interchangeable in P41, P42, P90, P91, P93 and P94.

Serial pins switching / Open-drain Control Register 0 (Read-modify-write instructions are prohibited.)

SIOCHG0 (0025H)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	SIOCHG05	SIOCHG04	SIOCHG03	SIOCHG02	SIOCHG01	SIOCHG00
Read/Write	–	–	W					
After reset	–	–	0	0	0	0	0	0
Function			P94 port 0: CMOS output 1: Open-drain output	0: Setting of P94C 1: TXD1	0: Setting of P93C and P93F 1: RXD1	P91 port 0: CMOS output 1: Open-drain output	0: Setting of P91C 1: TXD0	0: Setting of P90C and P90F 1: RXD0

SIOCHG02	SIOCHG01	SIOCHG00	P91	P90
0	0	0	Setting of P91C	Setting of P90C and P90F
0	1	1	TXD0 (CMOS output)	RXD0
1	1	1	TXD0 (Open-drain output)	RXD0

SIOCHG05	SIOCHG04	SIOCHG03	P94	P93
0	0	0	Setting of P94C	Setting of P93C and P93F
0	1	1	TXD1 (CMOS output)	RXD1
1	1	1	TXD1 (Open-drain output)	RXD1

Serial pins switching / Open-drain Control Register 1 (Read-modify-write instructions are prohibited.)

SIOCHG1 (0015H)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	SIOCHG14	–	SIOCHG12	SIOCHG11	–
Read/Write	–	–	–	W	–	W		–
After reset	–	–	–	0	–	0	0	–
Function				P42 port 0: CMOS output 1: Open-drain output		0: Setting of P42C 1: TXD2	0: Setting of P41C and P41F2 1: RXD2	

SIOCHG14	SIOCHG12	SIOCHG11	P42	P41
0	0	0	Setting of P42C	Setting of P41C and P41F2
0	1	1	TXD2 (CMOS output)	RXD2
1	1	1	TXD2 (Open-drain output)	RXD2

5. 8-Bit Timers (TMRA)

The TMP91FU62 features 4 channels (TMRA0, TMRA1, TMRA4, TMRA5) built-in 8-bit timers.

These timers are paired into 2 modules: TMRA01 and TMRA45. Each module consists of 2 channels and can operate in any of the following 4 operating modes.

- 8-bit interval timer mode
- 16-bit interval timer mode
- 8-bit programmable square wave pulse generation output mode (PPG – Variable duty cycle with variable period)
- 8-bit pulse width modulation output mode (PWM – Variable duty cycle with constant period)

Figure 5-1 to Figure 5-2 show block diagrams for TMRA01 and TMRA45.

Each channel consists of an 8-bit up counter, an 8-bit comparator and an 8-bit timer register. In addition, a timer flip-flop and a prescaler are provided for each pair of channels.

The operation mode and timer flip-flops are controlled by 5-byte registers SFRs (Special function registers).

Each of the three modules (TMRA01 and TMRA45) can be operated independently. All modules operate in the same manner; hence only the operation of TMRA01 is explained here.

Table 5-1 Registers and Pins for Each Module

Specification \ Module		TMRA01	TMRA45
External pin	Input pin for external clock	TA0IN (Shared with P70)	TA4IN (Shared with P73)
	Output pin for timer flip-flop	TA1OUT (Shared with P71)	TA5OUT (Shared with P74)
SFR (Address)	Timer run register	TA01RUN (0100H)	TA45RUN (0110H)
	Timer register	TA0REG (0102H) TA1REG (0103H)	TA4REG (0112H) TA5REG (0113H)
	Timer mode register	TA01MOD (0104H)	TA45MOD (0114H)
	Timer flip-flop control register	TA1FFCR (0105H)	TA5FFCR (0115H)

5.1 Block Diagrams

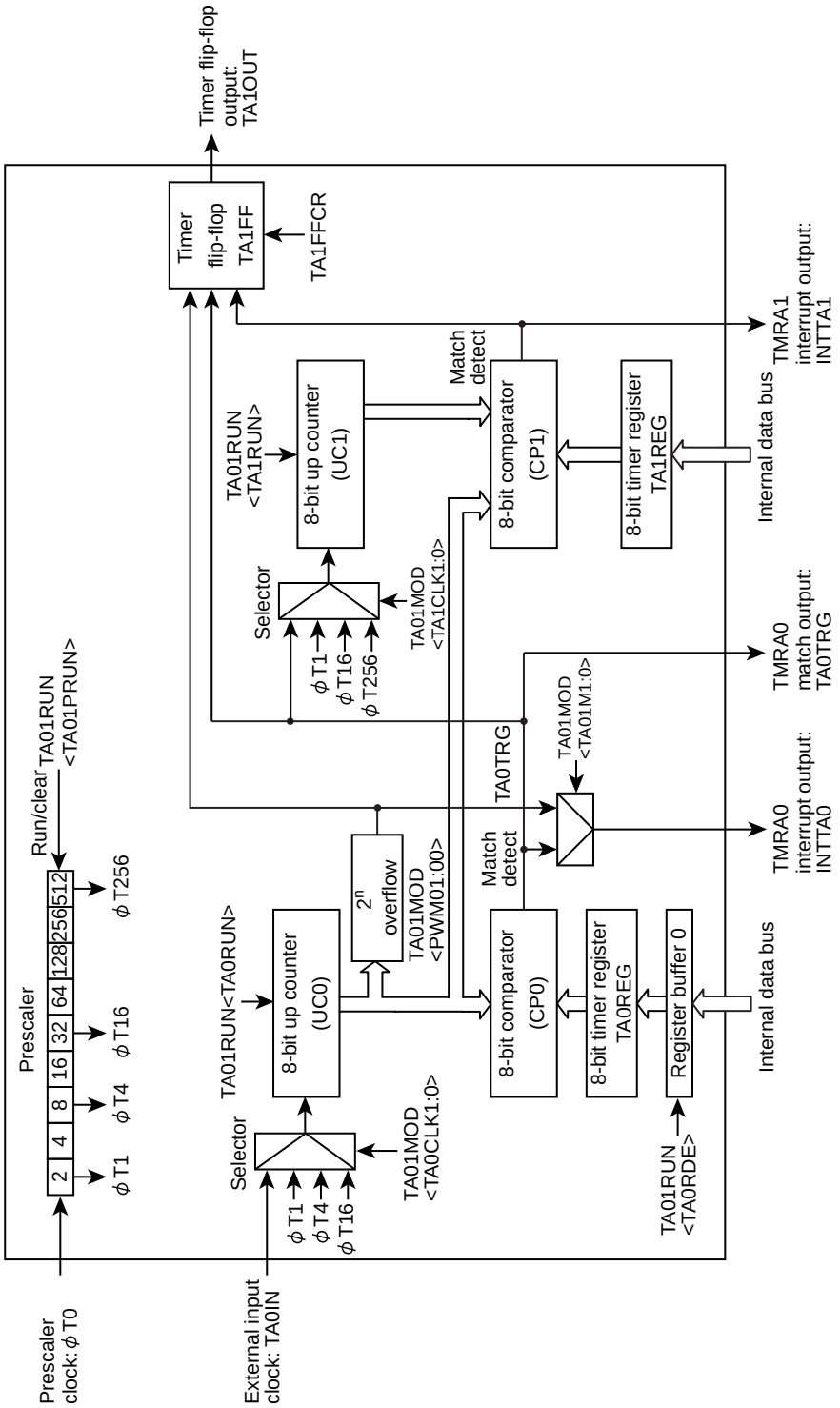


Figure 5-1 TMRA01 Block Diagram

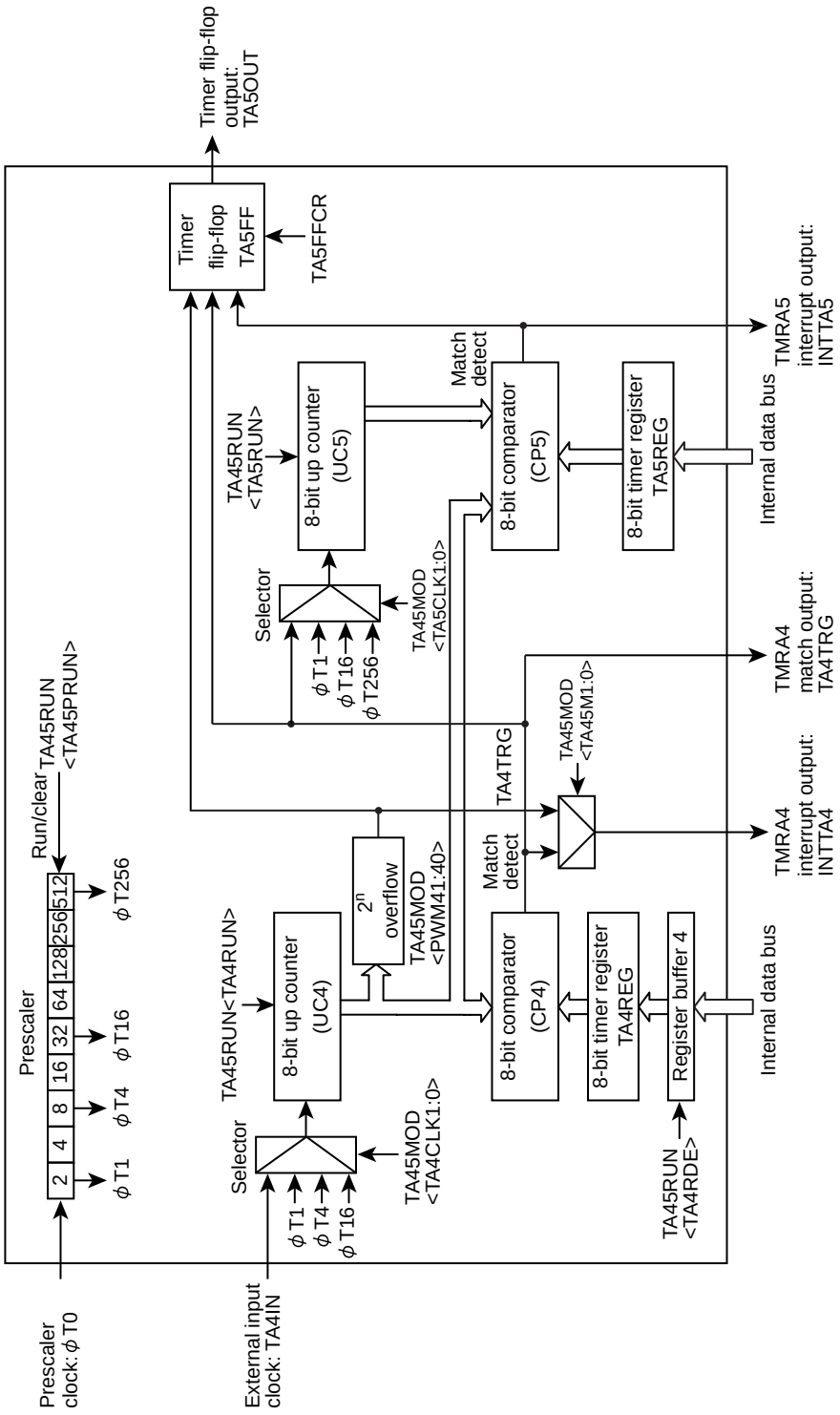


Figure 5-2 TMRA45 Block Diagram

5.2 Operation of Each Circuit

5.2.1 Prescalers

A 9-bit prescaler generates the input clock to TMRA01.

The "ϕT0" as the input clock to prescaler is a clock divided by 4 which is selected using the prescaler clock selection register SYSCR0<PRCK1>.

The prescaler's operation can be controlled using TA01RUN<TA01PRUN> in the timer control register. Setting <TA01PRUN> to "1" starts the count; setting <TA01PRUN> to "0" clears the prescaler to "0" and stops operation. Table 5-2 shows the various prescaler output clock resolutions.

Table 5-2 Prescaler Output Clock Resolution

@ fc = 20 MHz, fs = 32.768 kHz

System Clock Selection SYSCR1<SYSCK>	Gear Value SYSCR1<GEAR2:0>	Prescaler Clock Selection SYSCR0<PRCK1>	Prescaler Output Clock Resolution			
			ϕT1 (1/2)	ϕT4 (1/8)	ϕT16 (1/32)	ϕT256 (1/512)
1 (fs)	XXX	0 (1/1) f_{FPH}	2^3/fs (244 μs)	2^5/fs (977 μs)	2^7/fs (3.9 ms)	2^{11}/fs (62.5 ms)
0 (fc)	000 (fc)		2^3/fc (0.4 μs)	2^5/fc (1.6 μs)	2^7/fc(6.4 μs)	2^{11}/fc (102.4 μs)
	001 (fc/2)		2^4/fc (0.8 μs)	2^6/fc (3.2 μs)	2^8/fc (12.8 μs)	2^{12}/fc (204.8 μs)
	010 (fc/4)		2^5/fc (1.6 μs)	2^7/fc (6.4 μs)	2^9/fc (25.6 μs)	2^{13}/fc (409.6 μs)
	011 (fc/8)		2^6/fc (3.2 μs)	2^8/fc (12.8 μs)	2^{10}/fc (51.2 μs)	2^{14}/fc (819.2 μs)
	100 (fc/16)		2^7/fc (6.4 μs)	2^9/fc (25.6 μs)	2^{11}/fc (102.4 μs)	2^{15}/fc (1638.4 μs)
	XXX	1 (1/16) fc/16 CLOCK	2^7/fc (6.4 μs)	2^9/fc (25.6 μs)	2^{11}/fc (102.4 μs)	2^{15}/fc (1638.4 μs)

Note: xxx: Don't care

5.2.2 Up counters (UC0 and UC1)

These are 8-bit binary counters which count up the input clock pulses for the clock specified by TA01MOD.

The input clock for UC0 is selectable and can be either the external clock input via the TA0IN pin or one of the three internal clocks ϕT1, ϕT4, or ϕT16. The clock setting is specified by the value set in TA01MOD<TA01CLK1:0>.

The input clock for UC1 depends on the operation mode. In 16-bit timer mode, the overflow output from UC0 is used as the input clock. In any mode other than 16-bit timer mode, the input clock is selectable and can either be one of the internal clocks ϕT1, ϕT16 or ϕT256, or the comparator output (The match detection signal) from TMRA0.

For each interval timer the timer operation control register bits TA01RUN<TA0RUN> and TA01RUN<TA1RUN> can be used to stop and clear the up counters and to control their count. A reset clears both up counters, stopping the timers.

5.2.3 Timer registers (TA0REG and TA1REG)

These are 8-bit registers which can be used to set a time interval. When the value set in the timer register TA0REG or TA1REG matches the value in the corresponding up counter, the comparator match detect signal goes active. If the value set in the timer register is 00H, the signal goes active when the up counter overflows.

The TA0REG are double buffer structure, each of which makes a pair with register buffer.

The setting of the bit TA01RUN<TA0RDE> determines whether TA0REG's double buffer structure is enabled or disabled. It is disabled if <TA0RDE> = "0" and enabled if <TA0RDE> = "1".

When the double buffer is enabled, data is transferred from the register buffer to the timer register when a 2^n overflow occurs in PWM mode, or at the start of the PPG cycle in PPG mode. Hence the double buffer cannot be used in timer mode.

A reset initializes <TA0RDE> to "0", disabling the double buffer. To use the double buffer, write data to the timer register, set <TA0RDE> to "1", and write the following data to the register buffer. Figure 5-3 shows the configuration of TA0REG.

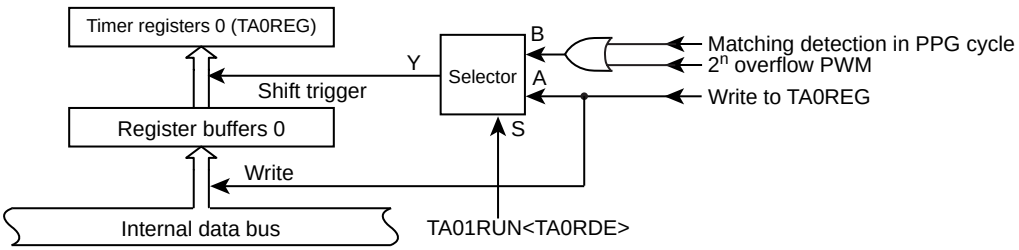


Figure 5-3 Configuration of TA0REG

Note: The same memory address is allocated to the timer register TA0REG and the register buffer 0. When <TA0RDE> = 0, the same value is written to the register buffer 0 and the timer register TA0REG; when <TA0RDE> = 1, only the register buffer 0 is written to.

5.2.4 Comparator (CP0 and CP1)

The comparator compares the value in an up counter with the value set in a timer register. If they match, the up counter is cleared to 0 and an interrupt signal (INTTA0 or INTTA1) is generated. If timer flip-flop inversion is enabled, the timer flip-flop is inverted at the same time.

Note: If a value smaller than the up-counter value is written to the timer register while the timer is counting up, this will cause the timer to overflow and an interrupt cannot be generated at the expected time. (The value in the timer register can be changed without any problem if the new value is larger than the up-counter value.) In 16-bit interval timer mode, be sure to write to both TA0REG and TA1REG in this order (16 bits in total), The compare circuit will not function if only the lower 8 bits are set.

5.2.5 Timer flip-flop (TA1FF)

The timer flip-flop (TA1FF) is a flip-flop inverted by the match detects signal (8-bit comparator output) of each interval timer.

Whether inversion is enabled or disabled is determined by the setting of the bit TA1FFCR<TA1FFIE> in the timer flip-flop control register.

A reset clears the value of TA1FF1 to “0”.

Writing “01” or “10” to TA1FFCR<TA1FFC1:0> sets TA1FF to 0 or 1. Writing “00” to these bits inverts the value of TA1FF (This is known as software inversion).

The TA1FF signal is output via the TA1OUT pin (Concurrent with P71). When this pin is used as the timer output, the timer flip-flop should be set beforehand using the port 7 function registers P7CR, P7FC.

The condition for TA1FF inversion varies with mode as shown below

8-bit interval timer mode	: UC0 matches TA0REG or UC1 matches TA1REG (Select either one of the two)
16-bit interval timer mode	: UC0 matches TA0REG or UC1 matches TA1REG
8 bit PWM mode	: UC0 matches TA0REG or a 2n overflow occurs
8 bit PPG mode	: UC0 matches TA0REG or UC0 matches TA1REG

Note: If an inversion by the match-detect signal and a setting change via the TMRA1 flip-flop control register occur simultaneously, the resultant operation varies depending on the situation, as shown below.

- If an inversion by the match-detect signal and an inversion via the register occur simultaneously, the flip-flop will be inverted only once.
- If an inversion by the match-detect signal and an attempt to set the flip-flop to 1 via the register occur simultaneously, the timer flip-flop will be set to 1.
- If an inversion by the match-detect signal and an attempt to clear the flip-flop to 0 via the register occur simultaneously the flip-flop will be cleared to 1.

Be sure to stop the timer before changing the flip-flop insertion setting.

If the setting is changed while the timer is counting, proper operation cannot be obtained.

5.3 SFR

TMRA01 Run Register

TA01RUN (0100H)

	7	6	5	4	3	2	1	0
Bit symbol	TA0RDE	–	–	–	I2TA01	TA01PRUN	TA1RUN	TA0RUN
Read/Write	R/W	–	–	–	R/W			
After Reset	0	–	–	–	0	0	0	0
Function	Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	TMRA01 prescaler	Up counter (UC1)	Up counter (UC0)
						0: Stop and clear 1: Run (count up)		

Count operation

TA01PRUN TA1RUN / TA0RUN	0	Stop and clear
	1	Run (Count up)

TA0REG double buffer control

TA0RDE	0	Disable
	1	Enable

Note: The values of bits 4 to 6 of TA01RUN are "1" when read.

TMRA45 Run Register

TA45RUN (0110H)

	7	6	5	4	3	2	1	0
Bit symbol	TA4RDE	–	–	–	I2TA45	TA45PRUN	TA5RUN	TA4RUN
Read/Write	R/W	–	–	–	R/W			
After Reset	0	–	–	–	0	0	0	0
Function	Double buffer 0: Disable 1: Enable				IDLE2 0: Stop 1: Operate	TMRA45 prescaler	Up counter (UC5)	Up counter (UC4)
						0: Stop and clear 1: Run (count up)		

Count operation

TA45PRUN TA5RUN / TA4RUN	0	Stop and clear
	1	Run (Count up)

TA4REG double buffer control

TA4RDE	0	Disable
	1	Enable

Note: The values of bits 4 to 6 of TA45RUN are "1" when read.

TMRA01 Mode Register

TA01MOD (0104H)

	7	6	5	4	3	2	1	0
Bit symbol	TA01M1	TA01M0	PWM01	PWM00	TA1CLK1	TA1CLK0	TA0CLK1	TA0CLK0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		Input clock for TMRA1 00: TA0TRG 01: ϕT1 10: ϕT16 11: ϕT256		Input clock for TMRA0 00: TA0IN pin 01: ϕT1 10: ϕT4 11: ϕT16	

TMRA0 input clock selection

<TA0CLK1:0>	00	TA0IN
	01	ϕT1
	10	ϕT4
	11	ϕT16

TMRA1 input clock selection

		TA01MOD<TA01M1:0> ≠ 01	TA01MOD<TA01M1:0> = 01
<TA1CLK1:0>	00	Comparator output from TMRA0	Overflow output from TMRA0 (16-bit timer mode)
	01	ϕT1	
	10	ϕT16	
	11	ϕT256	

PWM cycle selection

<PWM01:00>	00	Reserved
	01	2^6 × Clock source
	10	2^7 × Clock source
	11	2^8 × Clock source

TMRA01 operation mode selection

<TA01M1:0>	00	8-bit timers 2ch
	01	16-bit timer
	10	8-bit PPG
	11	8-bit PWM (TMRA0) + 8-bit timer (TMRA1)

TMRA45 Mode Register

TA45MOD (0114H)

	7	6	5	4	3	2	1	0
Bit symbol	TA45M1	TA45M0	PWM41	PWM40	TA5CLK1	TA5CLK0	TA4CLK1	TA4CLK0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		Input clock for TMRA5 00: TA4TRG 01: ϕT1 10: ϕT16 11: ϕT256		Input clock for TMRA4 00: TA4IN pin 01: ϕT1 10: ϕT4 11: ϕT16	

TMRA4 input clock selection

<TA4CLK1:0>	00	TA4IN
	01	ϕT1
	10	ϕT4
	11	ϕT16

TMRA5 input clock selection

		TA45MOD<TA45M1:0> ≠ 01	TA45MOD<TA45M1:0> = 01
<TA5CLK1:0>	00	Comparator output from TMRA4	Overflow output from TMRA4 (16-bit timer mode)
	01	ϕT1	
	10	ϕT16	
	11	ϕT256	

PWM cycle selection

<PWM41:40>	00	Reserved
	01	2^6 × Clock source
	10	2^7 × Clock source
	11	2^8 × Clock source

TMRA45 operation mode selection

<TA45M1:0>	00	8-bit timers 2ch
	01	16-bit timer
	10	8-bit PPG
	11	8-bit PWM (TMRA4) + 8-bit timer (TMRA5)

TMRA1 Flip-Flop Control Register

TA1FFCR (0105H)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	–	TA1FFC1	TA1FFC0	TA1FFIE	TA1FFIS
Read/Write	–	–	–	–	R/W		R/W	
After reset	–	–	–	–	1	1	0	0
Function					00: Invert TA1FF 01: Set TA1FF 10: Clear TA1FF 11: Don't care		TA1FF control for inversion 0: Disable 1: Enable	TA1FF inversion select 0: TMRA0 1:TMRA1

Inverse signal for timer flip-flop 1 (TA1FF) (Don't care except in 8-bit timer mode)

TA1FFIS	0	Inversion by TMRA0
	1	Inversion by TMRA1

Inversion of TA1FF

TA1FFIE	0	Disabled
	1	Enabled

Control of TA1FF

<TA1FFC1:0>	00	Inverts the value of TA1FF (Software inversion)
	01	Sets TA1FF to "1"
	10	Clears TA1FF to "0"
	11	Don't care

Note: The values of bits 4 to 7 of TA1FFCR are "1" when read.

TMRA5 Flip-Flop Control Register

TA5FFCR (0115H)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	–	TA5FFC1	TA5FFC0	TA5FFIE	TA5FFIS
Read/Write	–	–	–	–	R/W		R/W	
After reset	–	–	–	–	1	1	0	0
Function					00: Invert TA5FF 01: Set TA5FF 10: Clear TA5FF 11: Don't care		TA5FF control for inversion 0: Disable 1: Enable	TA5FF inversion select 0: TMRA4 1:TMRA5

Inverse signal for timer flip-flop 5 (TA5FF) (Don't care except in 8-bit timer mode)

TA5FFIS	0	Inversion by TMRA4
	1	Inversion by TMRA5

Inversion of TA5FF

TA5FFIE	0	Disabled
	1	Enabled

Control of TA5FF

<TA5FFC1:0>	00	Inverts the value of TA5FF (Software inversion)
	01	Sets TA5FF to "1"
	10	Clears TA5FF to "0"
	11	Don't care

Note: The values of bits 4 to 7 of TA5FFCR are "1" when read.

Timer Register

		7	6	5	4	3	2	1	0
TA0REG (0102H)	Bit symbol	–							
	Read/Write	W							
	After Reset	0							
TA1REG (0103H)	Bit symbol	–							
	Read/Write	W							
	After Reset	0							
TA4REG (0112H)	Bit symbol	–							
	Read/Write	W							
	After Reset	0							
TA5REG (0113H)	Bit symbol	–							
	Read/Write	W							
	After Reset	0							

5.4 Operation in Each Mode

5.4.1 8-bit timer mode

Both TMRA0 and TMRA1 can be used independently as 8-bit interval timers.

Set its function or counter data for TMRA0 and TMRA1 after stop these registers.

5.4.1.1 Generating interrupts at a fixed interval (Using TMRA1)

To generate interrupts at constant intervals using TMRA1 (INTTA1), first stop TMRA1 then set the operation mode, input clock and a cycle to TA01MOD and TA1REG register, respectively. Then, enable the interrupt INTTA1 and start TMRA1 counting.

Example: To generate an INTTA1 interrupt every 12 μs at fc = 20 MHz, set each register as follows:

* Clock state	System clock	: High frequency (fc)
	Prescaler clock	: f_{FPH}
	Clock gear	: 1 (fc)

	MSB 7	6	5	4	3	2	1	LSB 0	
TA01RUN	–	X	X	X	–	–	0	–	Stop TMRA1 and clear it to 0.
TA01MOD	0	0	X	X	0	1	X	X	Select 8-bit timer mode and select φT1 (0.4 μs at fc = 20 MHz) as the input clock.
TA1REG	0	0	0	1	1	1	1	0	Set TA1REG to 12 μs ÷ φT1 = 30 = 1EH
INTETA01	X	1	0	1	X	–	–	–	Enable INTTA1 and set it to level 5.
TA01RUN	–	X	X	X	–	1	1	–	Start TMRA1 counting.

Note: X: Don't care, –: No change

Select the input clock using Table 5-2.

Note: The input clocks for TMRA0 and TMRA1 are different from as follows.
TMRA0: TA0IN input, φT1, φT4 or φT16
TMRA1: Match output of TMRA0, φT1, φT16, φT256

5.4.1.2 Generating a 50% duty ratio square wave pulse

The state of the timer flip-flop (TA1FF) is inverted at constant intervals and its status output via the timer output pin (TA1OUT).

Example: To output a 2.4 μs square wave pulse from the TA1OUT pin at fc = 20 MHz, use the following procedure to make the appropriate register settings. This example uses TMRA1; however, either TMRA0 or TMRA1 may be used.

* Clock state	System clock	: High frequency (fc)
	Prescaler clock	: f_{FPH}
	Clock gear	: 1 (fc)

	MSB							LSB	
	7	6	5	4	3	2	1	0	
TA01RUN	–	X	X	X	–	–	0	–	Stop TMRA1 and clear it to 0.
TA01MOD	0	0	X	X	0	1	–	–	Select 8-bit timer mode and select φT1 (0.4 μs at fc = 20 MHz) as the input clock.
TA1REG	0	0	0	0	0	0	1	1	Set the timer register to 2.4 μs ÷ φT1 ÷ 2 = 03H
TA1FFCR	X	X	X	X	1	0	1	1	Clear TA1FF to “0” and set it to invert on the match detects signal from TMRA1.
P7CR	X	X	X	–	–	–	1	–	Set P71 to function as the TA1OUT pin.
P7FC	X	X	X	–	–	–	1	–	
TA01RUN	–	X	X	X	–	1	1	–	Start TMRA1 counting.

Note: X: Don’t care, –: No change

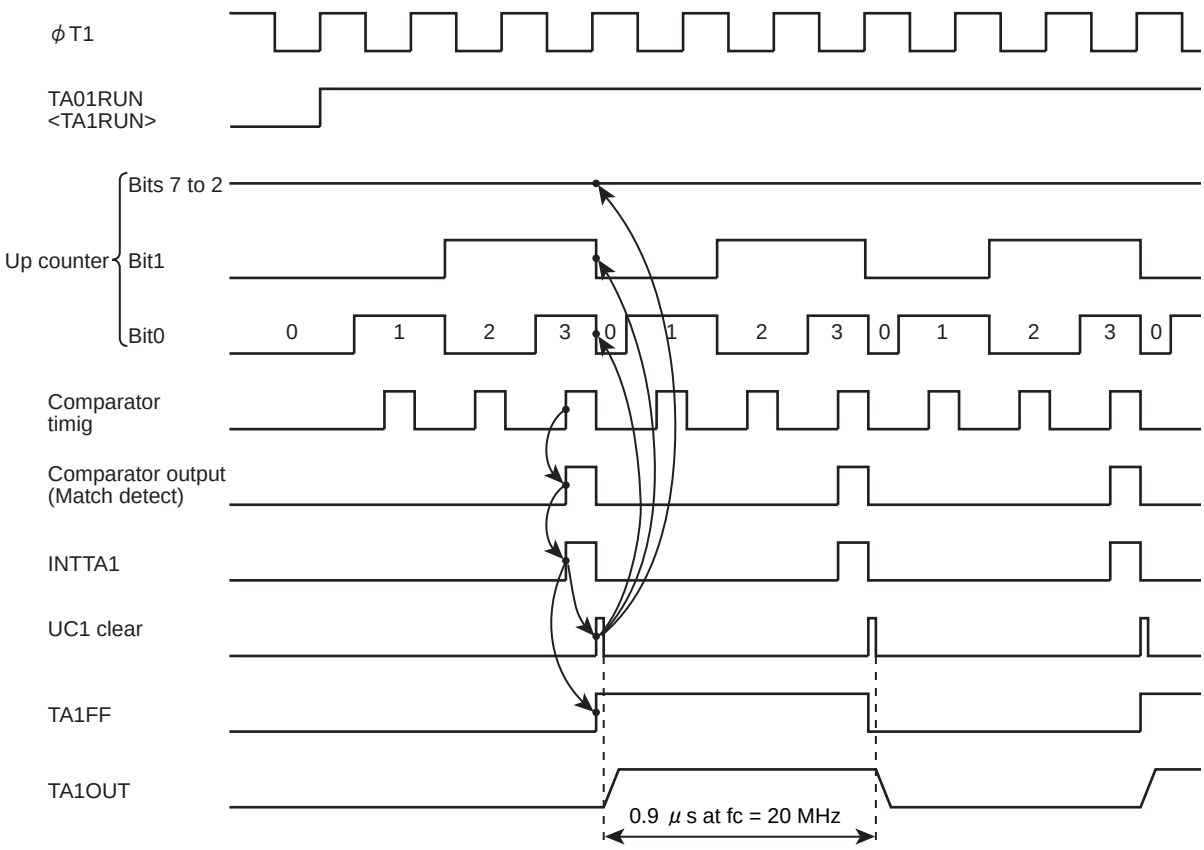


Figure 5-4 Square Wave Output Timing Chart (50% duty)

5.4.1.3 Making TMRA1 count up on the match signal from the TMRA0 comparator

Select 8-bit timer mode and set the comparator output from TMRA0 to be the input clock to TMRA1.

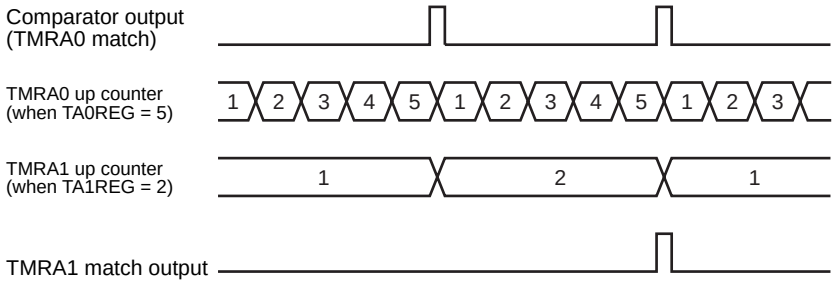


Figure 5-5 TMRA1 Count Up on Signal from TMRA0

5.4.2 16-bit timer mode

A 16-bit interval timer is configured by pairing the two 8-bit timers TMRA0 and TMRA1.

To make a 16-bit interval timer in which TMRA0 and TMRA1 are cascaded together, set TA01MOD<TA01M1:0> to 01.

In 16-bit timer mode, the overflow output from TMRA0 is used as the input clock for TMRA1, regardless of the value set in TA01MOD<TA1CLK1:0>. Table 5-2 shows the cycle of the input clock for TMRA0.

LSB 8-bit set to TA0REG and MSB 8-bit is for TA1REG. Please keep setting TA0REG first because setting data for TA0REG inhibit its compare function and setting data for TA1REG permit it.

Example: To generate an INTTA1 interrupt every 0.4 [s] at fc = 20 MHz, set the timer registers TA0REG and TA1REG as follows:

* Clock state	System clock	: High frequency (fc)
	Prescaler clock	: f_{FPH}
	Clock gear	: 1 (fc)

If ϕT16 (2^7/fc μs at fc = 20 MHz) is used as the input clock for counting, set the following value in the registers: 0.4 s/(2^7/fc) μs ≒ 62500 = F424H (e.g., set TA1REG to F4H and TA0REG to 24H). As a result, INTTA1 interrupt can be generated every 0.4 [s].

The comparator match signal is output from TMRA0 each time the up counter UC0 matches TA0REG, though the up counter UC0 is not cleared and also INTTA0 is not generated.

In the case of the TMRA1 comparator, the match detect signal is output on each comparator pulse on which the values in the up counter UC1 and TA1REG match.

When the match detect signal is output simultaneously from both the comparators TMRA0 and TMRA1, the up counters UC0 and UC1 are cleared to 0 and the interrupt INTTA1 is generated. Also, if inversion is enabled, the value of the timer flip-flop TA1FF is inverted.

Example: When TA1REG = 04H and TA0REG = 80H

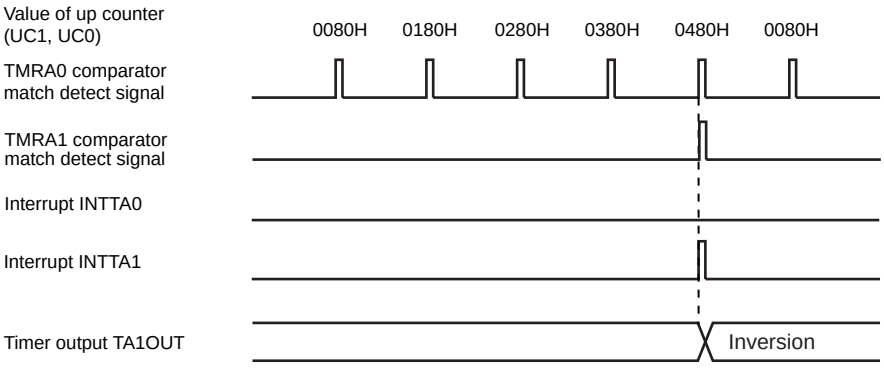


Figure 5-6 Timer Output by 16-Bit Timer Mode

5.4.3 8-bit PPG (Programmable pulse generation) output mode

Square wave pulses can be generated at any frequency and duty ratio by TMRA0. The output pulses may be active low or active high. In this mode TMRA1 cannot be used.

TMRA0 outputs pulses on the TA1OUT pin.

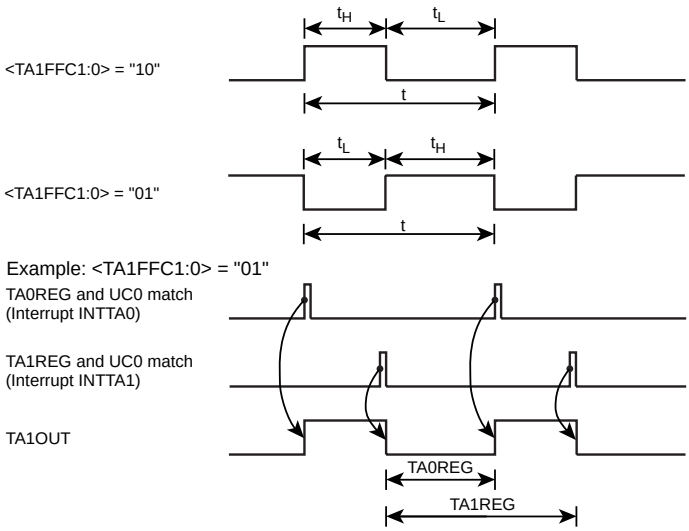


Figure 5-7 8-Bit PPG Output Waveforms

In this mode, a programmable square wave is generated by inverting the timer output each time the 8-bit up counter (UC0) matches the value in one of the timer registers TA0REG or TA1REG.

The value set in TA0REG must be smaller than the value set in TA1REG.

Although the up counter for TMRA1 (UC1) is not used in this mode, TA01RUN<TA1RUN> should be set to “1”, so that UC1 is set for counting.

Figure 5-8 shows a block diagram representing this mode.

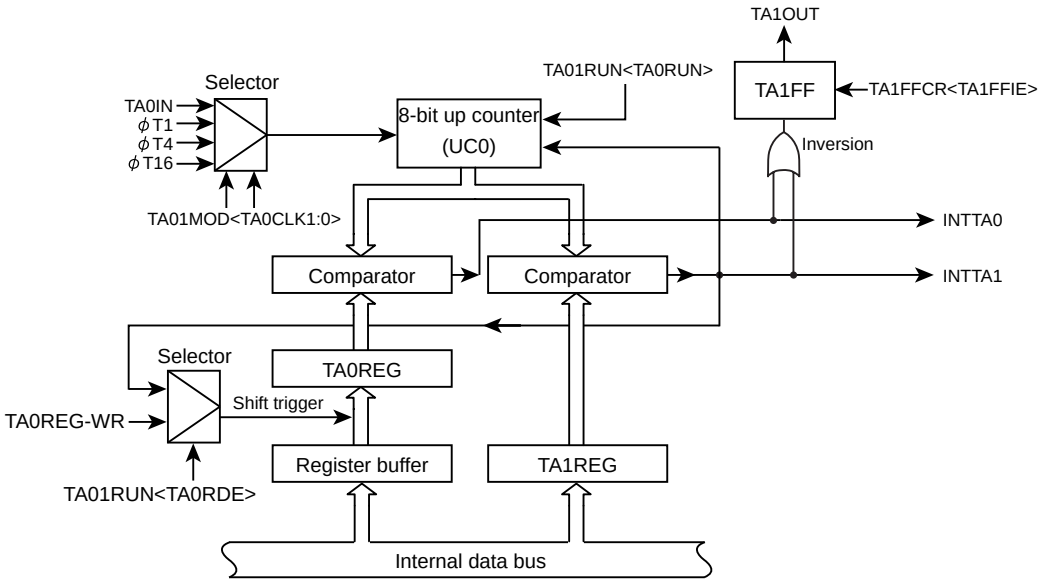


Figure 5-8 Block Diagram of 8-Bit PPG Output Mode

If the TA0REG double buffer is enabled in this mode, the value of the register buffer will be shifted into TA0REG each time TA1REG matches UC0.

Use of the double buffer facilitates the handling of low-duty waves (when duty is varied).

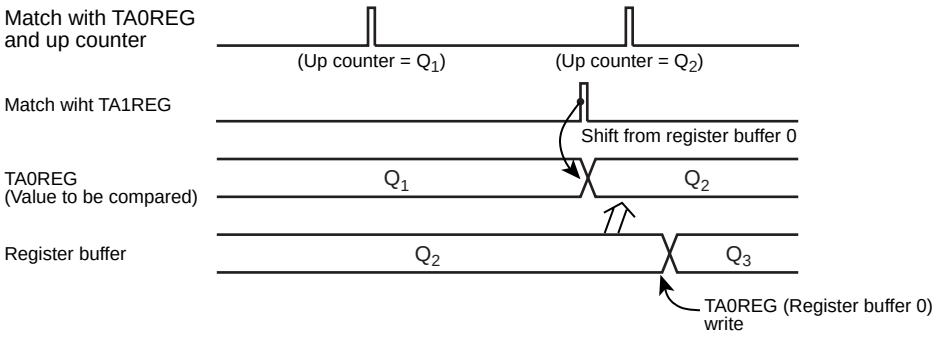


Figure 5-9 Operation of Register Buffer 0

Note: The values that can be set in TAxREG range from 01h to 00h (equivalent to 100h). If the maximum value 00h is set, the match-detect signal goes active when the up-counter overfolws.

Example: To generate 1/4-duty 50-kHz pulses (at fc = 20 MHz):

* Clock state	System clock	: High frequency (fc)
	Prescaler clock	: f_{FPH}
	Clock gear	: 1 (fc)

Calculate the value which should be set in the timer register.

To obtain a frequency of 50 kHz, the pulse cycle t should be: t = 1/50 kHz = 20 μs

$\phi T1 = 2^3/fc$ μs (at fc = 20 MHz);

20 μs/$(2^3/fc)$ μs = 50

Therefore set TA1REG to 50 (32H), and 50-kHz pulses can be obtained.

The duty is to be set to 1/4: t × 1/4 = 20 μs × 1/4 = 5 μs

5 μs/$(2^3/fc)$ μs ≒ 13

Therefore, set TA0REG = 13 = 0DH.

	7	6	5	4	3	2	1	0	
TA01RUN	0	X	X	X	–	–	0	0	Stop TMRA0 and TMRA01 and clear it to “0”.(Double buffer disable)
TA01MOD	1	0	X	X	X	X	0	1	Set the 8-bit PPG mode, and select ϕT1 as input clock.
TA0REG	0	0	0	0	1	1	0	1	Write 0DH.
TA1REG	0	0	1	1	0	0	1	0	Write 32H.
TA1FFCR	X	X	X	X	0	1	1	X	Set TA1FF, enabling both inversion and the double buffer. Writing “10” provides negative logic pulse.
P7CR	X	X	X	–	–	–	1	–	Set P71 as the TA1OUT pin.
P7FC	X	X	X	–	–	–	1	–	
TA01RUN	1	X	X	X	–	1	1	1	Start TMRA0 and TMRA01 counting.(Double buffer enable)

Note: X : Don't Care – : No change

5.4.4 8-bit PWM output mode

This mode is only valid for TMRA0. In this mode, a PWM pulse with the maximum resolution of 8 bits can be output.

When TMRA0 is used the PWM pulse is output on the TA1OUT pin. TMRA1 can also be used as an 8-bit timer.

The timer output is inverted when the up counter (UC0) matches the value set in the timer register TA0REG or when 2^n counter overflow occurs (n = 6, 7 or 8 as specified by TA01MOD<PWM01:00>). The up counter UC0 is cleared when 2^n counter overflow occurs.

The following conditions must be satisfied before this PWM mode can be used.

Value set in TA0REG < Value set for 2^n counter overflow

Value set in TA0REG ≠ 0

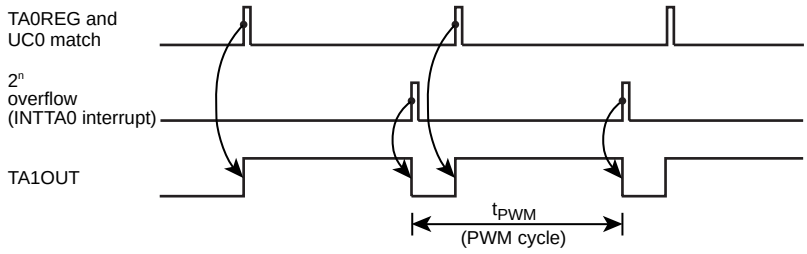


Figure 5-10 8-Bit PWM Waveforms

Figure 5-11 shows a block diagram representing this mode.

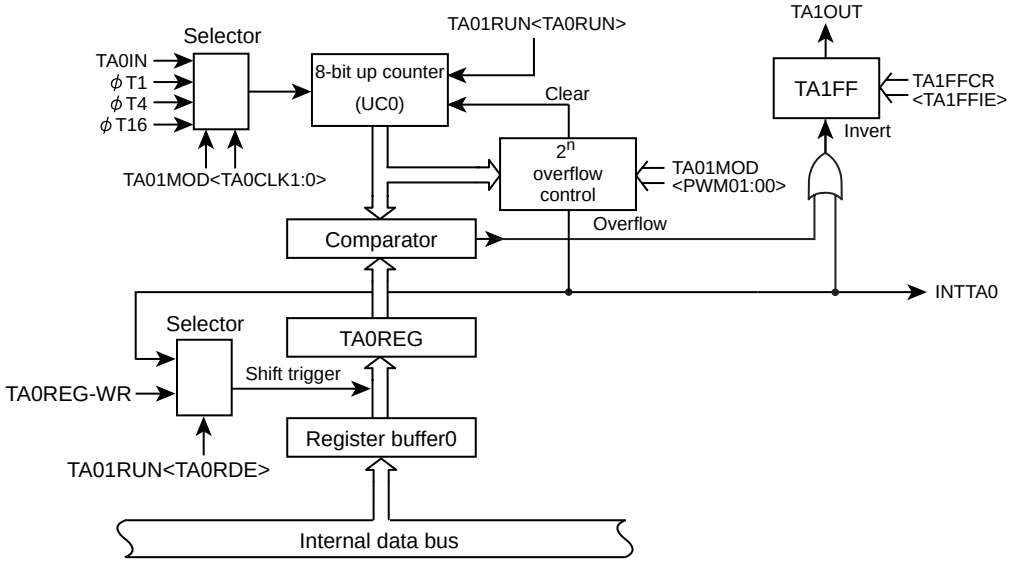


Figure 5-11 Block Diagram of 8-Bit PWM Mode

In this mode, the value of the register buffer will be shifted into TA0REG if 2^n overflow is detected when the TA0REG double buffer is enabled.

Use of the double buffer facilitates the handling of low duty ratio waves.

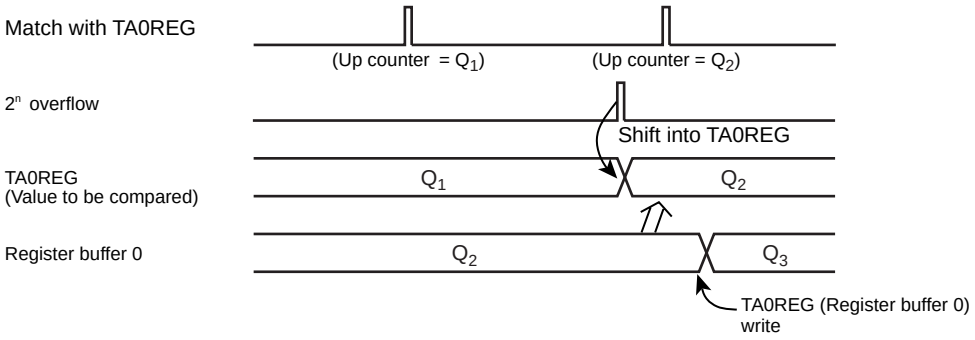
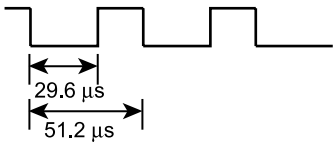


Figure 5-12 Operation of Register Buffer 0

Example: To output the following PWM waves on the TA1OUT pin at fc = 20 MHz:



* Clock state	System clock	: High frequency (fc)
	Prescaler clock	: f_{FPH}
	Clock gear	: 1 (fc)

To achieve a 51.2 μs PWM cycle by setting ϕT1 to 2^3/fc μs (at fc = 20 MHz):

51.2 μs/(2^3/fc) μs ≒ 128 = 2^n

Therefore n should be set to 7.

Since the low-level period is 29.6 μs when ϕT1 = 2^3/fc μs (at fc = 20 MHz), set the following value for TA0REG:

29.6 μs/(2^3/fc) μs ≒ 74 = 4AH

	MSB							LSB	
	7	6	5	4	3	2	1	0	
TA01RUN	–	X	X	X	–	–	–	0	Stop TMRA0 and clear it to 0.
TA01MOD	1	1	1	0	–	–	0	1	Select 8-bit PWM mode (Cycle: 2^7) and select ϕT1 as the input clock.
TA0REG	0	1	0	0	1	0	1	0	Write 4AH.
TA1FFCR	X	X	X	X	1	0	1	X	Clear TA1FF to 0, enable the inversion and double buffer.
P7CR	X	X	X	–	–	–	1	–	Set P71 and the TA1OUT pin.
P7FC	X	X	X	–	–	–	1	–	
TA01RUN	1	X	X	X	–	1	–	1	Start TMRA0 counting.

Note: X : Don't Care – : No change

Table 5-3 PWM Cycle @ fc = 20 MHz, fs = 32.768 kHz

Select System Clock SYSCR1 <SYSCK>	Gear Value SYSCR1 <GEAR2:0>	Select Prescaler Clock SYSCR0 <PRCK1>	PWM cycle								
			2^6			2^7			2^8		
			ϕT1	ϕT4	ϕT16	ϕT1	ϕT4	ϕT16	ϕT1	ϕT4	ϕT16
1 (fs)	XXX	0 (1/1) f_{FPH}	15.6 ms	62.5 ms	250 ms	31.3 ms	125 ms	500 ms	62.5 ms	250 ms	1000 ms
0 (fc)	000 (fc)		25.6 μs	102.4 μs	409.6 μs	51.2 μs	204.8 μs	819.2 μs	102.4 μs	409.6 μs	1638 μs
	001 (fc/2)		51.2 μs	204.8 μs	819.2 μs	102.4 μs	409.6 μs	1638 μs	204.8 μs	819.2 μs	3277 μs
	010 (fc/4)		102.4 μs	409.6 μs	1638 μs	204.8 μs	810.2 μs	3277 μs	409.6 μs	1638 μs	6554 μs
	011 (fc/8)		204.8 μs	819.2 μs	3277 μs	409.6 μs	1638 μs	6554 μs	819.2 μs	3277 μs	13107 μs
	100 (fc/16)		409.6 μs	1638 μs	6554 μs	819.2 μs	3277 μs	13107 μs	1638 μs	6554 μs	26214 μs
	XXX	1 (1/16) fc/16 clock	409.6 μs	1638 μs	6554 μs	819.2 μs	3277 μs	13107 μs	1638 μs	6554 μs	26214 μs

Note: xxx: Don't care

5.4.5 Settings for each mode

Table 5-4 shows the SFR settings for each mode.

Table 5-4 Timer Mode Setting Registers

Register Name	TA01MOD				TA1FFCR
<Bit Symbol>	<TA01M1:0>	<PWM01:00>	<TA1CLK1:0>	<TA0CLK1:0>	TA1FFIS
Function	Timer Mode	PWM Cycle	Upper Timer Input Clock	Lower Timer Input Clock	Timer F/F Invert Signal Select
8-bit timer × 2 channels	00	–	Lower timer match ϕT1, ϕT16, ϕT256 (00, 01, 10, 11)	External clock ϕT1 ϕT4, ϕT16 (00, 01, 10, 11)	0: Lower timer output 1: Upper timer output
16-bit timer mode	01	–	–	External clock ϕT1, ϕT4, ϕT16 (00, 01, 10, 11)	–
8-bit PPG × 1 channel	10	–	–	External clock ϕT1, ϕT4, ϕT16 (00, 01, 10, 11)	–
8-bit PWM × 1 channel	11	2^6, 2^7, 2^8 (01, 10, 11)	–	External clock ϕT1, ϕT4, ϕT16 (00, 01, 10, 11)	–
8-bit timer × 1 channel	11	–	ϕT1, ϕT16, ϕT256 (01, 10, 11)	–	Output disabled

Note: – : Don't care

6. 16-Bit Timer/Event Counters (TMRB)

The TMP91FU62 incorporates four multifunctional 16-bit timer/event counters (TMRB0, TMRB1, TMRB2, TMRB3) which have the following operation modes:

- 16-bit interval timer mode
- 16-bit event counter mode
- 16-bit programmable pulse generation (PPG) output mode

The capture function enables selection of the following modes:

- Frequency measurement mode
- Pulse width measurement mode
- Time differential measurement

Figure 6-1 show block diagrams for TMRB0, TMRB1, TMRB2 and TMRB3.

Each timer/event counter channel consists of a 16-bit up-counter, two 16-bit timer registers (one of them with a double-buffer structure), two 16-bit capture registers, two comparators, a capture input controller, two timer flip-flops and a timer flip-flop controller.

Each timer/event counter is controlled by an 11-byte SFR (special-function register).

Each of the four channels (TMRB0, TMRB1, TMRB2, TMRB3) can be used independently. Each channel features the same operations except for those described in Table 6-1. Hence, only the operation of TMRB0 is explained below.

Table 6-1 Registers and Pins for TMRB

Specification \ Channel		TMRB0	TMRB1	TMRB2	TMRB3
External pins	External clock/capture trigger input pins	TB0IN0 (also used as P80) TB0IN1 (also used as P81)	TB1IN0 (also used as P84) TB1IN1 (also used as P85)	TB2IN0 (also used as PA0) TB2IN1 (also used as PA1)	TB3IN0 (also used as P30) TB3IN1 (also used as P31)
	Timer flip-flop output pins	TB0OUT0 (also used as P82) TB0OUT1 (also used as P83)	TB1OUT0 (also used as P86) TB1OUT1 (also used as P87)	TB2OUT0 (also used as PA2) TB2OUT1 (also used as PA3)	TB3OUT0 (also used as P32) TB3OUT1 (also used as P33)
SFR (address)	Timer run register	TB0RUN (0180H)	TB1RUN (0190H)	TB2RUN (01A0H)	TB3RUN (01B0H)
	Timer mode register	TB0MOD (0182H)	TB1MOD (0192H)	TB2MOD (01A2H)	TB3MOD (01B2H)
	Timer flip-flop control register	TB0FFCR (0183H)	TB1FFCR (0193H)	TB2FFCR (01A3H)	TB3FFCR (01B3H)
	Timer registers	TB0RG0L (0188H) TB0RG0H (0189H) TB0RG1L (018AH) TB0RG1H (018BH)	TB1RG0L (0198H) TB1RG0H (0199H) TB1RG1L (019AH) TB1RG1H (019BH)	TB2RG0L (01A8H) TB2RG0H (01A9H) TB2RG1L (01AAH) TB2RG1H (01ABH)	TB3RG0L (01B8H) TB3RG0H (01B9H) TB3RG1L (01BAH) TB3RG1H (01BBH)
	Capture registers	TB0CP0L (018CH) TB0CP0H (018DH) TB0CP1L (018EH) TB0CP1H (018FH)	TB1CP0L (019CH) TB1CP0H (019DH) TB1CP1L (019EH) TB1CP1H (019FH)	TB2CP0L (01ACH) TB2CP0H (01ADH) TB2CP1L (01AEH) TB2CP1H (01AFH)	TB3CP0L (01BCH) TB3CP0H (01BDH) TB3CP1L (01BEH) TB3CP1H (01BFH)
capture of TMRA	Capture timing of TMRA	TA1OUT	TA1OUT	TA1OUT	Don't care

6.1 Block Diagrams

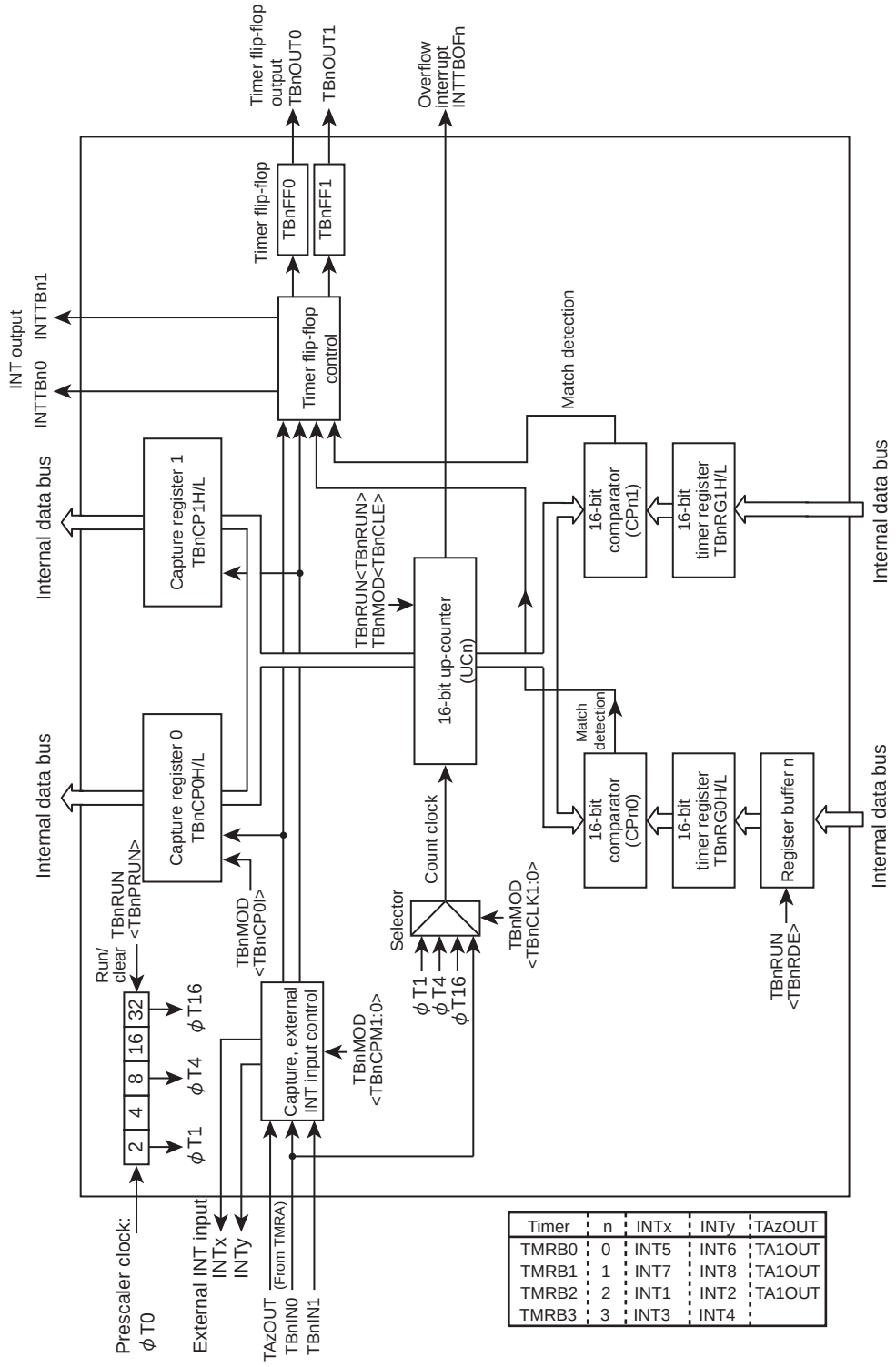


Figure 6-1 Block Diagrams of TMRB0 to TMRB3

6.2 Operation of Each Block

6.2.1 Prescaler

The 5-bit prescaler generates the source clock for TMRB0. The prescaler clock (ϕT0) is divided clock (divided by 4) from selected clock by the register SYSCR0<PRCK1> of clock gear.

This prescaler can be started or stopped using TB0RUN<TB0PRUN>. Counting starts when <TB0PRUN> is set to 1; the prescaler is cleared to 0 and stops operation when <TB0PRUN> is cleared to 0. Table 6-2 show prescaler output clock resolution.

Table 6-2 Prescaler Output Clock Resolution @fc = 20 MHz, fs = 32.768 kHz

System Clock SelectionSYSC1< SYSCK>	Clock Gear Value SYSCR1<GEAR2:0>	Prescaler Clock Selection <PRCK1>	Prescaler Output Clock Resolution		
			ϕT1 (1/2)	ϕT4 (1/8)	ϕT16 (1/32)
1 (fs)	XXX	0 (1/1) f_{FPH}	2^3/fs (244 μs)	2^5/fs (977 μs)	2^7/fs (3.9 ms)
0 (fc)	000 (fc)		2^3/fc (0.4 μs)	2^5/fc (1.6 μs)	2^7/fc(6.4 μs)
	001 (fc/2)		2^4/fc (0.8 μs)	2^6/fc (3.2 μs)	2^8/fc (12.8 μs)
	010 (fc/4)		2^5/fc (1.6 μs)	2^7/fc (6.4 μs)	2^9/fc (25.6 μs)
	011 (fc/8)		2^6/fc (3.2 μs)	2^8/fc (12.8 μs)	2^{10}/fc (51.2 μs)
	100 (fc/16)		2^7/fc (6.4 μs)	2^9/fc (25.6 μs)	2^{11}/fc (102.4 μs)
	XXX	1 (1/16) fc/16 clock	2^7/fc (6.4 μs)	2^9/fc (25.6 μs)	2^{11}/fc (102.4 μs)

Note: xxx: Don't care

6.2.2 Up counter (UC0)

UC0 is a 16-bit binary counter which counts up according to input from the clock specified by TB0MOD<TB0CLK1:0> register.

As the input clock, one of the prescaler internal clocks ϕT1, ϕT4 and ϕT16 or an external clock from TB0IN0 pin can be selected. Counting or stopping and clearing of the counter is controlled by timer operation control register TB0RUN<TB0RUN>.

When clearing is enabled, the up counter UC0 will be cleared to 0 each time its value matches the value in the timer register TB0RG1H/L. If clearing is disabled, the counter operates as a free-running counter. Clearing can be enabled or disabled by using TB0MOD<TB0CLE>.

A timer overflow interrupt (INTTBOF0) is generated when UC0 overflow occurs.

6.2.3 Timer registers (TB0RG0H/L, TB0RG1H/L)

These two 16-bit registers are used to set the interval time. When the value in the up counter UC0 matches the value set in this timer register, the comparator match detect signal will go active.

Setting data for both upper and lower timer registers is needed. For example, using 2-byte data transfer instruction or using 1-byte data transfer instruction twice for lower 8 bits and upper 8 bits in order. (The compare circuit will not operate if only the lower 8 bits are written. Be sure to write to both timer registers (16 bits) from the lower 8 bits followed by the upper 8 bits.)

The TB0RG0H/L timer register has a double-buffer structure, which is paired with register buffer 0. The value set in TB0RUN<TB0RDE> determines whether the double-buffer structure is enabled or disabled: it is disabled when <TB0RDE> = "0", and enabled when <TB0RDE> = "1".

When the double buffer is enabled, data is transferred from the register buffer 0 to the timer register when the values in the up counter (UC0) and the timer register TB0RG1H/L match.

The double buffer circuit incorporates two flags to indicate whether or not data is written to the lower 8 bits and the upper 8 bits of the register buffer, respectively. Only when both flags are set can data be transferred from the register buffer to the timer register by a match between the up-counter UC0 and the timer register TB0RG1. This data transfer is performed so long as 16-bit data is written in the register buffer regardless of the register buffer to the timer register unexpectedly as explained below.

For example, let us assume that an interrupt occurs when only the lower 8 bits (L1) of the register buffer data (H1L1) have been written and the interrupt routine includes writes to all 16 bits in the register buffer and a transfer of the data to the timer register. In this case, if the higher 8 bits (H1) are written after the interrupt routine is completed, only the flag for the higher 8 bits will be set, the flag for the lower 8 bits having been cleared in the interrupt routine. Therefore, even if a match occurs between UC0 and TB0RG1, no data transfer will be performed.

Then, in an attempt to set the next set of data (H2L2) in the register buffer, when the lower 8 bits (L2) are written, this will cause the flag for the lower 8 bits to be set as well as the flag for the higher 8 bits which has been set by writing the previous data (H1). If a match between UC0 and TB0RG1 occurs before the higher 8 bits (H2) are written, this will cause unexpected data (H1L2) to be sent to the timer register instead of the intended data (H2L2).

To avoid such transfer timing problems due to interrupts, the DI instruction (disable interrupts) and the EI (enable interrupts) can be executed before and after setting data in the register buffer, respectively.

After a reset, TB0RG0H/L and TB0RG1H/L are undefined. If the 16-bit timer is to be used after a reset, data should be written to it beforehand.

On a reset <TB0RDE> is initialized to "0", disabling the double buffer. To use the double buffer, write data to the timer register, set <TB0RDE> to "1", then write data to the register buffer 10 as shown below.

TB0RG0H/L and the register buffer 0 both have the same memory addresses (0188H and 0189H) allocated to them. If <TB0RDE> = "0", the value is written to both the timer register and the register buffer 0. If <TB0RDE> = "1", the value is written to the register buffer 0 only.

The addresses of the timer registers are as follows:

TMRB0	TB0RG0H/L		TB0RG1H/L	
	Upper 8 bits	Lower 8 bits	Upper 8 bits	Lower 8 bits
	000189H	000188H	00018BH	00018AH
TMRB1	TB1RG0H/L		TB1RG1H/L	
	Upper 8 bits	Lower 8 bits	Upper 8 bits	Lower 8 bits
	000199H	000198H	00019BH	00019AH
TMRB2	TB2RG0H/L		TB2RG1H/L	
	Upper 8 bits	Lower 8 bits	Upper 8 bits	Lower 8 bits
	0001A9H	0001A8H	0001ABH	0001AAH
TMRB3	TB3RG0H/L		TB3RG1H/L	
	Upper 8 bits	Lower 8 bits	Upper 8 bits	Lower 8 bits
	0001B9H	0001B8H	0001BBH	0001BAH

Note: The timer registers are write-only registers and thus cannot be read.

6.2.4 Capture registers (TB0CP0H/L, TB0CP1H/L)

These 16-bit registers are used to latch the values in the up counter (UC0).

Data in the capture registers should be read all 16 bits. For example, using a 2-byte data load instruction or two 1-byte data load instructions. The least significant byte is read first, followed by the most significant byte.

(during capture is read, capture operation is prohibited. In that case, the lower 8 bits should be read first, followed by the 8 bits.)

The addresses of the capture registers are as follows;

TMRB0	TB0CP0H/L		TB0CP1H/L	
	Upper 8 bits	Lower 8 bits	Upper 8 bits	Lower 8 bits
	00018DH	00018CH	00018FH	00018EH
TMRB1	TB1CP0H/L		TB1CP1H/L	
	Upper 8 bits	Lower 8 bits	Upper 8 bits	Lower 8 bits
	00019DH	00019CH	00019FH	00019EH
TMRB2	TB2CP0H/L		TB2CP1H/L	
	Upper 8 bits	Lower 8 bits	Upper 8 bits	Lower 8 bits
	0001ADH	0001ACH	0001AFH	0001AEH
TMRB3	TB3CP0H/L		TB3CP1H/L	
	Upper 8 bits	Lower 8 bits	Upper 8 bits	Lower 8 bits
	0001BDH	0001BCH	0001BFH	0001BEH

Note: The capture registers are read-only registers and thus cannot be written to.

6.2.5 Capture Input Control and External Interrupt Control

This circuit controls the timing to latch the value of up-counter UC0 into TB0CP0H/L and TB0CP1H/L, and generates external interrupt.The latch timing of capture register and selection of edge for external interrupt is controlled by TB0MOD<TB0CPM1:0>.

The value in the up-counter (UC0) can be loaded into a capture register by software. Whenever 0 is written to TB0MOD<TB0CP0I>, the current value in the up counter (UC0) is loaded into capture register TB0CP0H/L. It is necessary to keep the prescaler in RUN mode (e.g., TB0RUN<TB0PRUN> must be held at a value of 1).

6.2.6 Comparators (CP00, CP01)

CP10 and CP11 are 16-bit comparators which compare the value in the up counter UC0 with the value set in TB0RG0H/L or TB0RG1H/L respectively, in order to detect a match. If a match is detected, the comparator generates an interrupt (INTTB00 or INTTB01 respectively).

6.2.7 Timer flip-flops (TB0FF0, TB0FF1)

These flip-flops are inverted by the match detect signals from the comparators and the latch signals to the capture registers. Inversion can be enabled and disabled for each element using TB0FFCR<TB0C0T1, TB0E1T1, TB0E0T1>.

After a reset the value of TB0FF0 is undefined. If "00" is written to TB0FFCR <TB0FF0C1:0> or <TB0FF1C1:0>, TB0FF0 will be inverted. If "01" is written to the capture registers, the value of TB0FF0 will be set to "1". If "10" is written to the capture registers, the value of TB0FF0 will be set to "0".

Note: If an inversion by the match-detect signal and a setting change via the TB0FFCR register occurs simultaneously, the resultant operation varies depending on the situation, as shown below.

- If an inversion by the match-detect signal and an inversion via the register occur simultaneously, the flip-flop will be inverted only once.
- If an inversion by the match-detect signal and an attempt to set the flip-flop to 1 via the register occur simultaneously, the flip-flop will be set to 1.
- If an inversion by the match-detect signal and an attempt to clear the flip-flop to 0 via the register occur simultaneously, the flip-flop will be cleared to 0.

If an inversion by match-detect signal and inversion disable setting occur simultaneously, two case (it is inverted and it is not inverted) are occurred. Therefore, if changing inversion control (inversion enable/disable), stop timer operation beforehand.

The values of TB0FF0 and TB0FF1 can be output via the timer output pins TB0OUT0 (which is shared with P82 and TB0OUT1 (which is shared with P83). Timer output should be specified using the port P function register.

6.3 SFR

TMRB Run Register

		7	6	5	4	3	2	1	0
TB0RUN (0180H)	Bit symbol	TB0RDE	–	–	–	I2TB0	TB0PRUN	–	TB0RUN
	Read/Write	R/W	R/W	–	–	R/W	R/W	–	R/W
	After reset	0	0			0	0	–	0
	Function	Double Buffer 0: Disable 1: Enable	Always write 0.	Not in use		IDLE2 0: Stop 1: Operate	TMRB0 prescaler		UC0
							0: Stop and Clear 1: Run (count up)		
TB1RUN (0190H)	Bit symbol	TB1RDE	–	–	–	I2TB1	TB1PRUN	–	TB1RUN
	Read/Write	R/W	R/W	–	–	R/W	R/W	–	R/W
	After reset	0	0			0	0	–	0
	Function	Double Buffer 0: Disable 1: Enable	Always write 0.	Not in use		IDLE2 0: Stop 1: Operate	TMRB1 prescaler		UC1
							0: Stop and Clear 1: Run (count up)		
TB2RUN (01A0H)	Bit symbol	TB2RDE	–	–	–	I2TB2	TB2PRUN	–	TB2RUN
	Read/Write	R/W	R/W	–	–	R/W	R/W	–	R/W
	After reset	0	0			0	0	–	0
	Function	Double Buffer 0: Disable 1: Enable	Always write 0.	Not in use		IDLE2 0: Stop 1: Operate	TMRB2 prescaler		UC2
							0: Stop and Clear 1: Run (count up)		
TB3RUN (01B0H)	Bit symbol	TB3RDE	–	–	–	I2TB3	TB3PRUN	–	TB3RUN
	Read/Write	R/W	R/W	–	–	R/W	R/W	–	R/W
	After reset	0	0			0	0	–	0
	Function	Double Buffer 0: Disable 1: Enable	Always write 0.	Not in use		IDLE2 0: Stop 1: Operate	TMRB3 prescaler		UC3
							0: Stop and Clear 1: Run (count up)		

I2TB0, I2TB1, I2TB2, I2TB3: Operation of IDLE2 mode

TB0PRUN, TB1PRUN, TB2PRUN, TB3PRUN: Operation of prescaler

TB0RUN, TB1RUN, TB2RUN, TB3RUN: Operation of TMRB

Operation	
0	Stop and Clear
1	Count

Note: Bits 1, 4 and 5 of TB0RUN/TB1RUN/TB2RUN/TB3RUN are "1" when read.

TMRB Mode Register (Read-modify-write instructions are prohibited.) (1/2)

		7	6	5	4	3	2	1	0
TB0MOD (0182H)	Bit symbol	TB0CT1	TB0ET1	TB0CP0I	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
	Read/Write	R/W		W*	R/W				
	After reset	0	0	1	0	0	0	0	0
	Function	TB0FF1 inversion trigger 0: Trigger disable 1: Trigger enable		Software capture control 0: Software capture 1: Undefined	Capture timing 00: Disable INT5 occurs at rising edge 01: TB0IN0↑ TB0IN1↑ INT5 occurs at rising edge 10: TB0IN0↑ TB0IN0↓ INT5 occurs at falling edge 11: TA1OUT↑ TA1OUT↓ INT5 occurs at rising edge		Up counter control 0: Clear disable 1: Clear enable	TMRB0 input clock select 00: TB0IN0 pin input 01: ϕT1 10: ϕT4 11: ϕT16	
		Invert when UC0 is loaded into TB0CP1H/L	Invert when UC0 matches with TB0RG1H/L						
TB1MOD (0192H)	Bit symbol	TB1CT1	TB1ET1	TB1CP0I	TB1CPM1	TB1CPM0	TB1CLE	TB1CLK1	TB1CLK0
	Read/Write	R/W		W*	R/W				
	After reset	0	0	1	0	0	0	0	0
	Function	TB1FF1 inversion trigger 0: Trigger disable 1: Trigger enable		Software capture control 0: Software capture 1: Undefined	Capture timing 00: Disable INT7 occurs at rising edge 01: TB1IN0↑ TB1IN1↑ INT7 occurs at rising edge 10: TB1IN0↑ TB1IN0↓ INT7 occurs at falling edge 11: TA1OUT↑ TA1OUT↓ INT7 occurs at rising edge		Up counter control 0: Clear disable 1: Clear enable	TMRB1 input clock select 00: TB1IN0 pin input 01: ϕT1 10: ϕT4 11: ϕT16	
		Invert when UC1 is loaded into TB1CP1H/L	Invert when UC1 matches with TB1RG1H/L						
TB2MOD (01A2H)	Bit symbol	TB2CT1	TB2ET1	TB2CP0I	TB2CPM1	TB2CPM0	TB2CLE	TB2CLK1	TB2CLK0
	Read/Write	R/W		W*	R/W				
	After reset	0	0	1	0	0	0	0	0
	Function	TB2FF1 inversion trigger 0: Trigger disable 1: Trigger enable		Software capture control 0: Software capture 1: Undefined	Capture timing 00: Disable INT1 occurs at rising edge 01: TB2IN0↑ TB2IN1↑ INT1 occurs at rising edge 10: TB2IN0↑ TB2IN0↓ INT1 occurs at falling edge 11: TA1OUT↑ TA1OUT↓ INT1 occurs at rising edge		Up counter control 0: Clear disable 1: Clear enable	TMRB2 input clock select 00: TB2IN0 pin input 01: ϕT1 10: ϕT4 11: ϕT16	
		Invert when UC2 is loaded into TB2CP1H/L	Invert when UC2 matches with TB2RG1H/L						
TB3MOD (01B2H)	Bit symbol	TB3CT1	TB3ET1	TB3CP0I	TB3CPM1	TB3CPM0	TB3CLE	TB3CLK1	TB3CLK0
	Read/Write	R/W		W*	R/W				
	After reset	0	0	1	0	0	0	0	0
	Function	TB3FF1 inversion trigger 0: Trigger disable 1: Trigger enable		Software capture control 0: Software capture 1: Undefined	Capture timing 00: Disable INT3 occurs at rising edge 01: TB3IN0¦ TB3IN1¦ INT3 occurs at rising edge 10: TB3IN0¦ TB3IN0Ø INT3 occurs at falling edge 11: Don't care		Up counter control 0: Clear disable 1: Clear enable	TMRB3 input clock select 00: TB3IN0 pin input 01: ϕT1 10: ϕT4 11: ϕT16	
		Invert when UC3 is loaded into TB3CP1H/L	Invert when UC3 matches with TB3RG1H/L						

TMRB source clock

<TBnCLK1:0>	00	External input clock (TBnIN0 pin input)
	01	ϕT1
	10	ϕT4
	11	ϕT16

Up counter clear control (UCn)

<TBnCLE>	0	Disable to clear up counter
	1	Clear by match with TBnRG1H/L

Capture/Interrupt timing

		Capture control	INT5 control
<TB0CPM1:0>	00	Disable capture	INT generate at rising edge of TBnIN0
	01	Capture to TBnCP0H/L at rising edge of TBnIN0 Capture to TBnCP1H/L at rising edge of TBnIN1	
	10	Capture to TBnCP0H/L at rising edge of TBnIN0 Capture to TBnCP1H/L at falling edge of TBnIN0	INT generate at falling edge of TBnIN0
	11	Capture to TBnCP0H/L at rising edge of TA1OUT Capture to TBnCP1H/L at falling edge of TA1OUT	INT generate at rising edge of TBnIN0
		TMRB3: Don't care	

Software capture

<TBnCP0I>	0	Capture value of up counter to TBnCP0H/L.
	1	Undefined (Note 2)

Note 1: n=0,1,2,3

Note 2: As described above, whenever 0 is written to TBnMOD<TBnCP0I>, the current value in the up counter is loaded into capture register TBnCP0H/L. However, note that the current value in the up counter is also loaded into capture register TBnCP0H/L when 1 is written to TBnMOD<TBnCP0I> while this bit is holding 0.

TMRB Flip-Flop Control Register (Read-modify-write instructions are prohibited.) (1/2)

		7	6	5	4	3	2	1	0
TB0FFCR (0183H)	Bit symbol	TB0FF1C1	TB0FF1C0	TB0C1T1	TB0C0T1	TB0E1T1	TB0E0T1	TB0FF0C1	TB0FF0C0
	Read/Write	W*		R/W				W*	
	After reset	1	1	0	0	0	0	1	1
	Function	TB0FF1 control 00: Invert 01: Set 10: Clear 11: Don't care Note: Always read as 11.		TB0FF0 inversion trigger 0: Disable 1: Enable				TB0FF0 control 00: Invert 01: Set 10: Clear 11: Don't care Note: Always read as 11.	
				Invert when UC0 is loaded into TB0CP1H/L.	Invert when UC0 is loaded into TB0CP0H/L.	Invert when UC0 matches TB0RG1H/L.	Invert when UC0 matches TB0RG0H/L.		
TB1FFCR (0193H)	Bit symbol	TB1FF1C1	TB1FF1C0	TB1C1T1	TB1C0T1	TB1E1T1	TB1E0T1	TB1FF0C1	TB1FF0C0
	Read/Write	W*		R/W				W*	
	After reset	1	1	0	0	0	0	1	1
	Function	TB1FF1 control 00: Invert 01: Set 10: Clear 11: Don't care Note: Always read as 11.		TB1FF0 inversion trigger 0: Disable 1: Enable				TB1FF0 control 00: Invert 01: Set 10: Clear 11: Don't care Note: Always read as 11.	
				Invert when UC1 is loaded into TB1CP1H/L.	Invert when UC1 is loaded into TB1CP0H/L.	Invert when UC1 matches TB1RG1H/L.	Invert when UC1 matches TB1RG0H/L.		
TB2FFCR (01A3H)	Bit symbol	TB2FF1C1	TB2FF1C0	TB2C1T1	TB2C0T1	TB2E1T1	TB2E0T1	TB2FF0C1	TB2FF0C0
	Read/Write	W*		R/W				W*	
	After reset	1	1	0	0	0	0	1	1
	Function	TB2FF1 control 00: Invert 01: Set 10: Clear 11: Don't care Note: Always read as 11.		TB2FF0 inversion trigger 0: Disable 1: Enable				TB2FF0 control 00: Invert 01: Set 10: Clear 11: Don't care Note: Always read as 11.	
				Invert when UC2 is loaded into TB2CP1H/L.	Invert when UC2 is loaded into TB2CP0H/L.	Invert when UC2 matches TB2RG1H/L.	Invert when UC2 matches TB2RG0H/L.		
TB3FFCR (01B3H)	Bit symbol	TB3FF1C1	TB3FF1C0	TB3C1T1	TB3C0T1	TB3E1T1	TB3E0T1	TB3FF0C1	TB3FF0C0
	Read/Write	W*		R/W				W*	
	After reset	1	1	0	0	0	0	1	1
	Function	TB3FF1 control 00: Invert 01: Set 10: Clear 11: Don't care Note: Always read as 11.		TB3FF0 inversion trigger 0: Disable 1: Enable				TB3FF0 control 00: Invert 01: Set 10: Clear 11: Don't care Note: Always read as 11.	
				Invert when UC3 is loaded into TB3CP1H/L.	Invert when UC3 is loaded into TB3CP0H/L.	Invert when UC3 matches TB3RG1H/L.	Invert when UC3 matches TB3RG0H/L.		

<TBnFF0C1:0>Timer flip-flop (TBnFF0) control

<TBnFF0C1:0>	00	Invert TBnFF0.
	01	Set TBnFF0 to 1.
	10	Clear TBnFF0 to 0.
	11	Don't care

<TBnE0T1> TBnFF0 inversion when UCn matches TBnRG0H/L

<TBnE0T1>	0	Disable trigger (disable inversion).
	1	Enable trigger (enable inversion).

<TBnE1T1> TBnFF0 inversion when UCn matches TBnRG1H/L

<TBnE1T1>	0	Disable trigger (disable inversion).
	1	Enable trigger (enable inversion).

<TBnC0T1> TBnFF0 inversion when UCn is loaded into TBnCP0H/L

<TBnC0T1>	0	Disable trigger (disable inversion).
	1	Enable trigger (enable inversion).

<TBnC1T1> TBnFF0 inversion when UCn is loaded into TBnCP1H/L

<TBnC1T1>	0	Disable trigger (disable inversion).
	1	Enable trigger (enable inversion).

<TBnFF1C1:0>Timer flip-flop (TBnFF1) control

<TBnFF1C1:0>	00	Invert TBnFF1.
	01	Set TBnFF1 to 1.
	10	Clear TBnFF1 to 0.
	11	Don't care

Note: n=0,1,2,3

6.4 Operation in Each Mode

6.4.1 16-Bit Interval Timer Mode

Generating interrupts at fixed intervals

In this example the interrupt INTTB01 is set to be generated at fixed intervals. The interval time is set in the timer register TB0RG1H/L.

		7	6	5	4	3	2	1	0	
TB0RUN	←	0	0	X	X	–	0	X	0	Stop TMRB0.
INTETB0	←	X	1	0	0	X	0	0	0	Enable INTTB01 and set it to interrupt level 4. Disable INTTB00.
TB0FFCR	←	1	1	0	0	0	0	1	1	Disable trigger.
TB0MOD	←	0	0	1	0	0	1	*	*	Select internal clock for input and disable the capture function. (**=01, 10, 11)
TB0RG1	←	*	*	*	*	*	*	*	*	Set interval time (16 bits).
		*	*	*	*	*	*	*	*	
TB0RUN	←	0	0	X	X	–	1	X	1	Start TMRB0.

Note: X: Don't care, –: No change

6.4.2 16-Bit Event Counter Mode

If the external clock (TB0IN0 pin input) is selected as the input clock in 16-bit timer mode, the timer can be used as an event counter. The up-counter counts up on the rising edge of TB0IN0 input. To read the value of the counter, first perform software capture once, then read the captured value.

			6	5	4	3	2	1	0	
TB0RUN	←	0	0	X	X	–	0	X	0	Stop TMRB0.
P8CR	←	–	–	–	–	–	–	–	0	Set port to input mode.
P8FC	←	–	–	–	–	–	–	–	1	Set port to input mode.
INTETB0	←	X	1	0	0	X	0	0	0	Enable INTTB01 and set interrupt level 4. Disable INTTB00.
TB0FFCR	←	1	1	0	0	0	0	1	1	Disable trigger.
TB0MOD	←	0	0	1	0	0	1	0	0	Select TB0IN0 as the input clock.
TB0RG1	←	*	*	*	*	*	*	*	*	Set the number of counts (16 bits).
		*	*	*	*	*	*	*	*	
TB0RUN	←	0	0	X	X	–	1	X	1	Start TMRB0.

Note 1: X: Don't care, –: No change

Note 2: When the timer is used as an event counter, set the prescaler to run mode (TB0RUN<TB0PRUN> = 1).

6.4.3 16-Bit Programmable Pulse Generation (PPG) Output Mode

Square wave pulses can be generated at any frequency and duty ratio. The output pulse may be either active-Low or active-High.

In PPG mode a match between the value of the up-counter UC0 and either timer register TB0RG0 or TB0RG1 inverts the output value for timer flip-flop TB0FF0. The TB0FF0 output value is output on TB0OUT0. In this mode the following conditions must be satisfied.

(value set in TB0RG0) < (value set in TB0RG1)

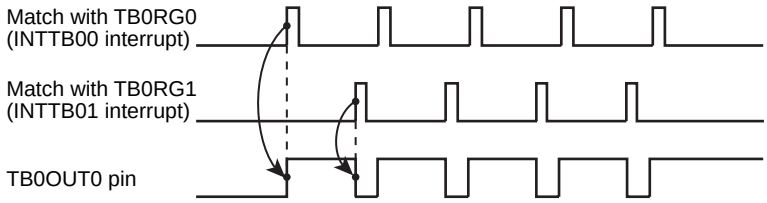


Figure 6-2 Programmable Pulse Generation (PPG) Output Waveforms

When the TB0RG0 double buffer is enabled in this mode, the value of register buffer 0 will be shifted into TB0RG0 when the up-counter value matches TB0RG1. This feature facilitates the handling of low-duty waves.

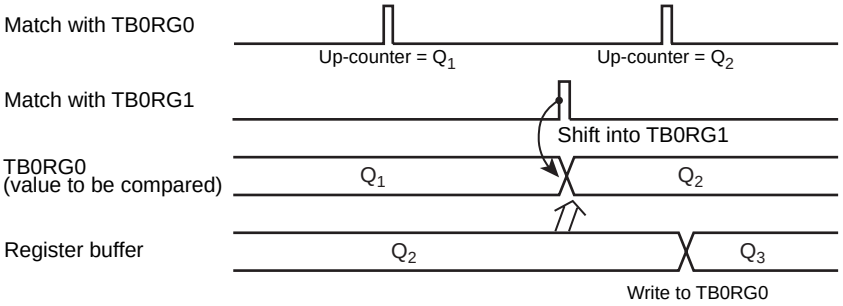


Figure 6-3 Operation of Register Buffer

Note: The values that can be set in TBxRGx range from 0001h to 0000h (equivalent to 10000h). If the maximum value 0000h is set, the match-detect signal goes active when the up-counter overflows.

The following block diagram illustrates this mode.

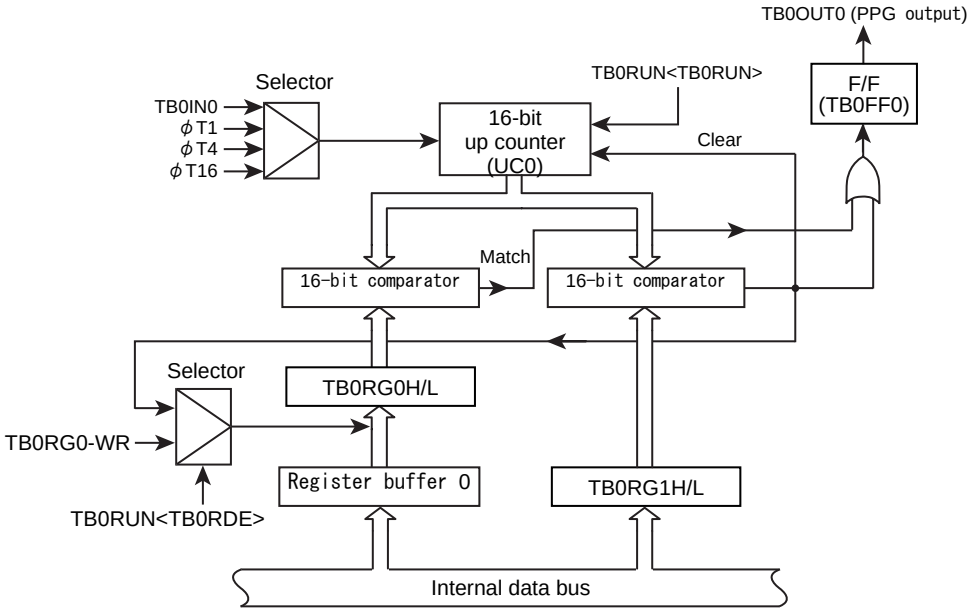


Figure 6-4 Block Diagram of 16-Bit PPG Mode

The following example shows how to set 16-bit PPG output mode:

		7	6	5	4	3	2	1	0	
TB0RUN	←	0	0	X	X	–	0	X	0	Disable the TB0RGH/L double buffer and stop TMRB0.
TB0RG0	←	*	*	*	*	*	*	*	*	Set the duty ratio.
		*	*	*	*	*	*	*	*	(16 bits)
TB0RG1	←	*	*	*	*	*	*	*	*	Set the frequency.
		*	*	*	*	*	*	*	*	(16 bits)
TB0RUN	←	1	0	X	X	–	0	X	0	Enable the TB0RG0H/L double buffer. (The duty and frequency are changed on an INTTB01 interrupt.)
TB0FFCR	←	1	1	0	0	1	1	1	0	Set the mode to invert TB0FF0 at the match with TB0RG0H/L, TB0RG1H/L. Clear TB0FF0 to “0”.
TB0MOD	←	0	0	1	0	0	1	*	*	Select prescaler output as input clock and disable the capture function.
							(**=01, 10, 11)			
P8CR	←	–	–	–	–	–	1	–	–	Set P82 to function as TB0OUT0.
P8FC	←	–	–	–	–	–	1	–	–	
TB0RUN	←	1	0	X	X	–	1	X	1	Start TMRB0.

Note: X: Don’t care, –: No change

6.4.4 Capture function examples

Used capture function, they can be applicable in many ways, for example:

1. One-shot pulse output from external trigger pulse
2. Frequency measurement
3. Pulse width measurement
4. Time difference measurement

6.4.4.1 One-shot pulse output from external trigger pulse

Set the up counter UC0 in free-running mode with the internal input clock, input the external trigger pulse from TB0IN0 pin, and load the value of up-counter into capture register TB0CP0H/L at the rise edge of the TB0IN0 pin.

When the interrupt INT5 is generated at the rise edge of TB0IN0 input, set the TB0CP0H/L value (c) plus a delay time (d) to TB0RG0H/L (= c + d), and set the above set value (c + d) plus a one-shot width (p) to TB0RG1H/L (= c + d + p). And, set "11" to timer flip-flop control register TB0FFCR<TB0E1T1, TB0E0T1>. Set to trigger enable for be inverted timer flip-flop TB0FF0 by UC0 matching with TB0RG0H/L and with TB0RG1H/L. When interrupt INTTB01 occurs, this inversion will be disabled after one-shot pulse is output.

The (c), (d) and (p) correspond to c, d and p Figure 6-5.

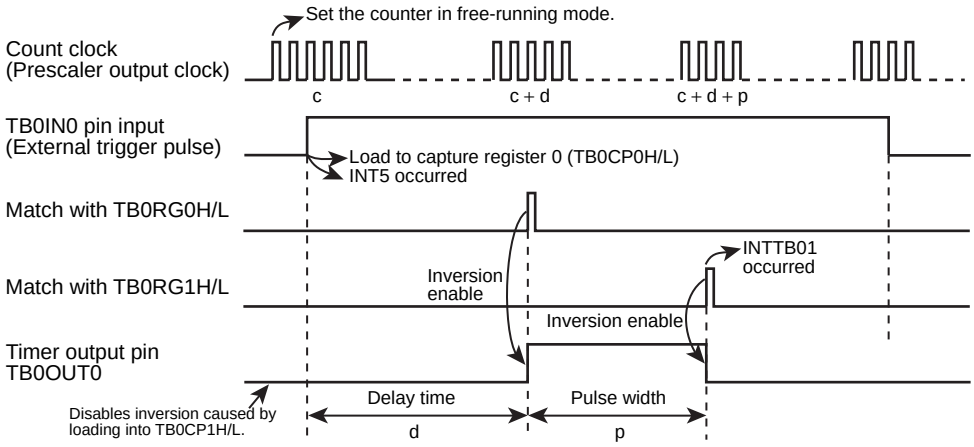


Figure 6-5 One-shot Pulse Output (with delay)

Example: To output a 2-ms one-shot pulse with a 3-ms delay to the external trigger pulse to the TB0IN0 pin.

					* Clock state				System clock:	High frequency (fc)
									Clock gear:	1 (fc)
									Prescaler clock:	f_{FPH}
TB0MOD	←	X	X	1	0	1	0	0	1	Set free running. Count with ϕT1. Load the up counter value into TB0CP0H/L at the rising edge of TB0IN0 pin input.
TB0FFCR	←	X	X	0	0	0	0	1	0	Clear TB0FF0 to 0. Disable inversion of TB0FF0.
P8CR	←	–	–	–	–	–	1	–	–	Set P82 to function as the TB0OUT0 pin.
P8FC	←	–	–	–	–	–	1	–	–	Set P80 to TB0IN0 input mode.
INTE56	←	X	–	–	–	X	1	0	0	Enable INT5.
INTETB0	←	X	0	0	0	X	0	0	0	Disable INTTB00 and INTTB01.
TB0RUN	←	–	0	X	X	–	1	X	1	Start TMRB0.
TB0RG0	←	TB0CP0 + 3 ms/ϕT1								
TB0RG1	←	TB0RG0 + 2 ms/ϕT1								
TB0FFCR	←	X	X	–	–	1	1	–	–	Enable TB0FF0 inversion when the up counter value match with TB0RG0H/L or TB0RG1H/L.
INTETB0	←	X	1	0	0	X	–	–	–	Enable INTTB01.
TB0FFCR	←	X	X	–	–	0	0	–	–	Disable inversion of TB0FF0 when the up counter value match with value of TB0RG0H/L or TB0RG1H/L.
INTETB0	←	X	0	0	0	X	–	–	–	Disable INTTB01.

Note: X: Don't care, –: No change

When delay time is unnecessary, invert timer flip-flop TB0FF0 when up-counter value is loaded into capture register (TB0CP0H/L), and set the TB0CP0H/L value (c) plus the one-shot pulse width (p) to TB0RG1H/L when the interrupt INT5 occurs. The TB0FF0 inversion should be enable when the up counter (UC10) value matches TB0RG1H/L, and disabled when generating the interrupt INTTB01.

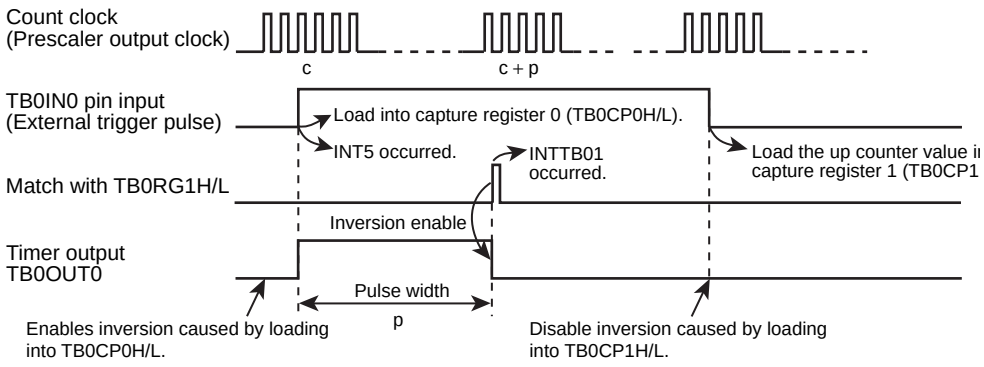


Figure 6-6 One-shot Pulse Output (without delay)

6.4.4.2 Frequency measurement

The frequency of the external clock can be measured in this mode. The clock is input through the TB0IN0 pin, and its frequency is measured by the 8-bit timers TMRA01 and the 16-bit timer/event counter (TMRB0). (TMRA01 is used to setting of measurement time by inversion TA1FF.)

The TB0IN0 pin input should be for the input clock of TMRB0. Set to TB0MOD <TB0CPM1:0> = “11”. The value of the up counter (UC10) is loaded into the capture register TB0CP0H/L at the rise edge of the timer flip-flop TA1FF of 8-bit timers (TMRA01), and into TB0CP1H/L at its fall edge.

The frequency is calculated by difference between the loaded values in TB0CP0H/L and TB0CP1H/L when the interrupt (INTTA0 or INTTA1) is generates by either 8-bit timer.

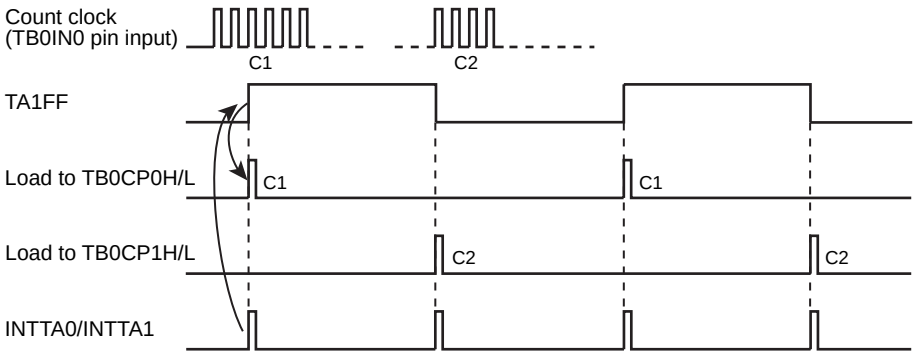


Figure 6-7 Frequency Measurement

For example, if the value for the level 1 width of TA1FF of the 8-bit timer is set to 0.5 s and the difference between the values in TB0CP0H/L and TB0CP1H/L is 100, the frequency is 100 ÷ 0.5 s = 200 Hz.

Note: The frequency in this example is calculated with 50 duty.

6.4.4.3 Pulse width measurement

This mode allows to measure the high-level width of an external pulse. While keeping the 16-bit timer/ event counter counting (Free running) with the internal clock input, external pulse is input through the TB0IN0 pin. Then the capture function is used to load the UC0 values into TB0CP0H/L and TB0CP1H/L at the rising edge and falling edge of the external trigger pulse respectively. The interrupt INT5 occurs at the falling edge of TB0IN0.

The pulse width is obtained from the difference between the values of TB0CP0H/L and TB0CP1H/L and the internal clock cycle.

For example, if the internal clock is 0.8 μs and the difference between TB0CP0H/L and TB0CP1H/L is 100, the pulse width will be 100 × 0.8 μs = 80 μs.

Additionally, the pulse width which is over the UC0 maximum count time specified by the clock source, can be measured by changing software.

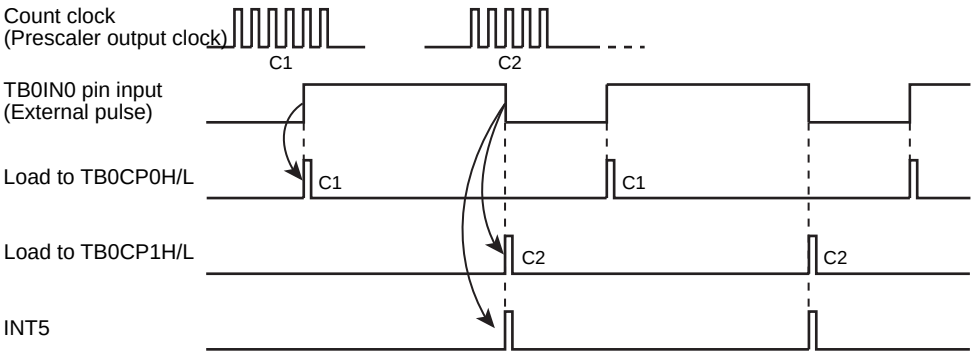


Figure 6-8 Pulse Width Measurement

Note: Only in this pulse width measuring mode (TB0MOD<TB0CPM1:0> = 10), external interrupt INT5 occurs at the falling edge of TB0IN0 pin input. In other modes, it occurs at the rising edge.

The width of low-level can be measured from the difference between the first C2 and the second C1 at the second INT5 interrupt.

6.4.4.4 Time Difference Measurement

This mode is used to measure the difference in time between the rising edges of external pulses input through TB0IN0 and TB0IN1.

Keep the 16-bit timer/event counter (TMRB0) counting (Free running) with the internal clock, and load the UC0 value into TB0CP0H/L at the rising edge of the input pulse to TB0IN0. Then the interrupt INT5 is generated.

Similarly, the UC0 value is loaded into TB0CP1H/L at the rising edge of the input pulse to TB0IN1, generating the interrupt INT6.

The time difference between these pulses can be obtained by multiplying the value subtracted TB0CP0H/L from TB0CP1H/L and the internal clock cycle together at which loading the up counter value into TB0CP0H/L and TB0CP1H/L has been done.

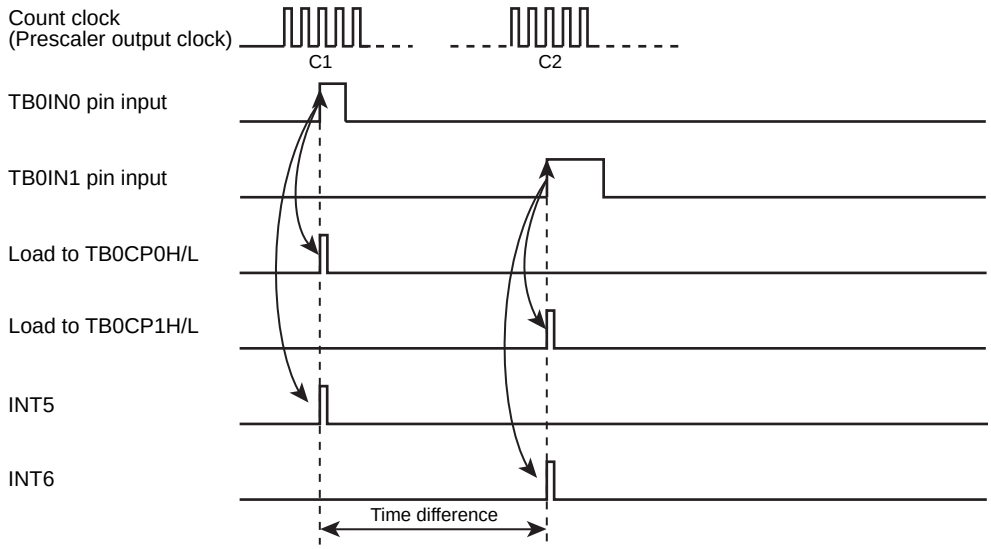


Figure 6-9 Time Difference Measurement

7. Serial Channels (SIO)

TMP91FU62 includes 3 serial I/O channels. For both channels either UART mode (Asynchronous transmission) or I/O interface mode (Synchronous transmission) can be selected.

1. I/O interface mode
   - Mode 0: For transmitting and receiving I/O data using the synchronizing signal SCLK for extending I/O.
2. UART mode
   - Mode 1: 7-bit data
   - Mode 1: 8-bit data
   - Mode 1: 9-bit data

In mode 1 and mode 2, a parity bit can be added. Mode 3 has a wakeup function for the master controller to start slave controllers via a serial link (A multi-controller system).

Figure 7-2 are block diagrams for each channel.

SIO is compounded mainly prescaler, serial clock generation circuit, receiving buffer and control circuit, transmission buffer and control circuit.

Both channels operate in the same function except for the following points; hence only the operation of channel 0 is explained below.

Table 7-1 Differences in Serial Channel Specifications

	SIO0	SIO1	SIO2
Pin name	TXD0, RXD0 (P90) RXD0, TXD0 (P91) $\overline{CTS0}$/SCLK0 (P92)	TXD1, RXD1 (P93) RXD1, TXD1 (P94) $\overline{CTS1}$/SCLK1 (P95)	TXD2, RXD2 (P41) RXD2, TXD2 (P42) $\overline{CTS2}$/SCLK2 (P43)

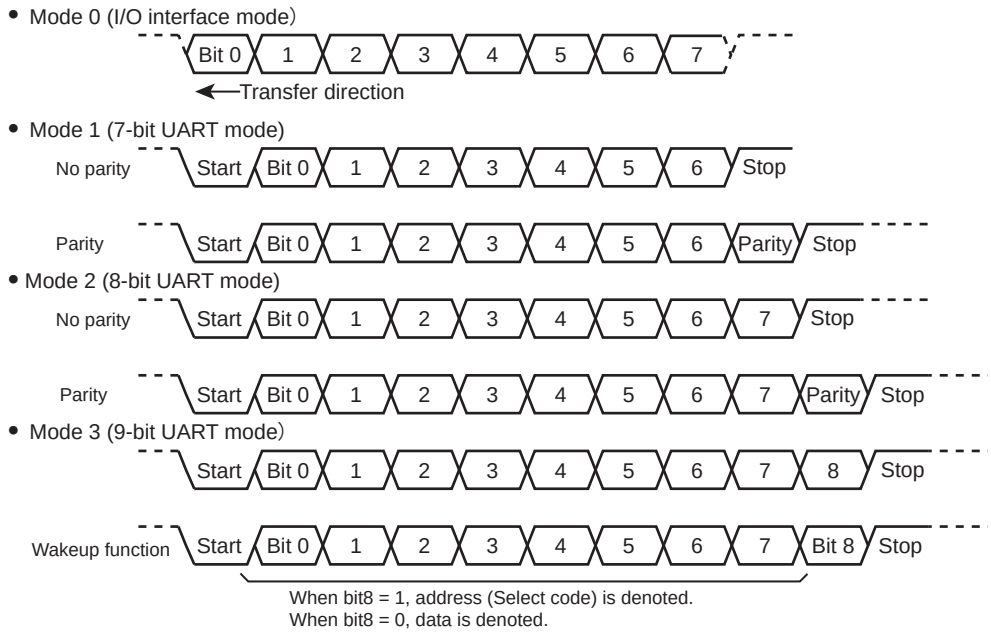


Figure 7-1 Data Formats

7.1 Block Diagrams

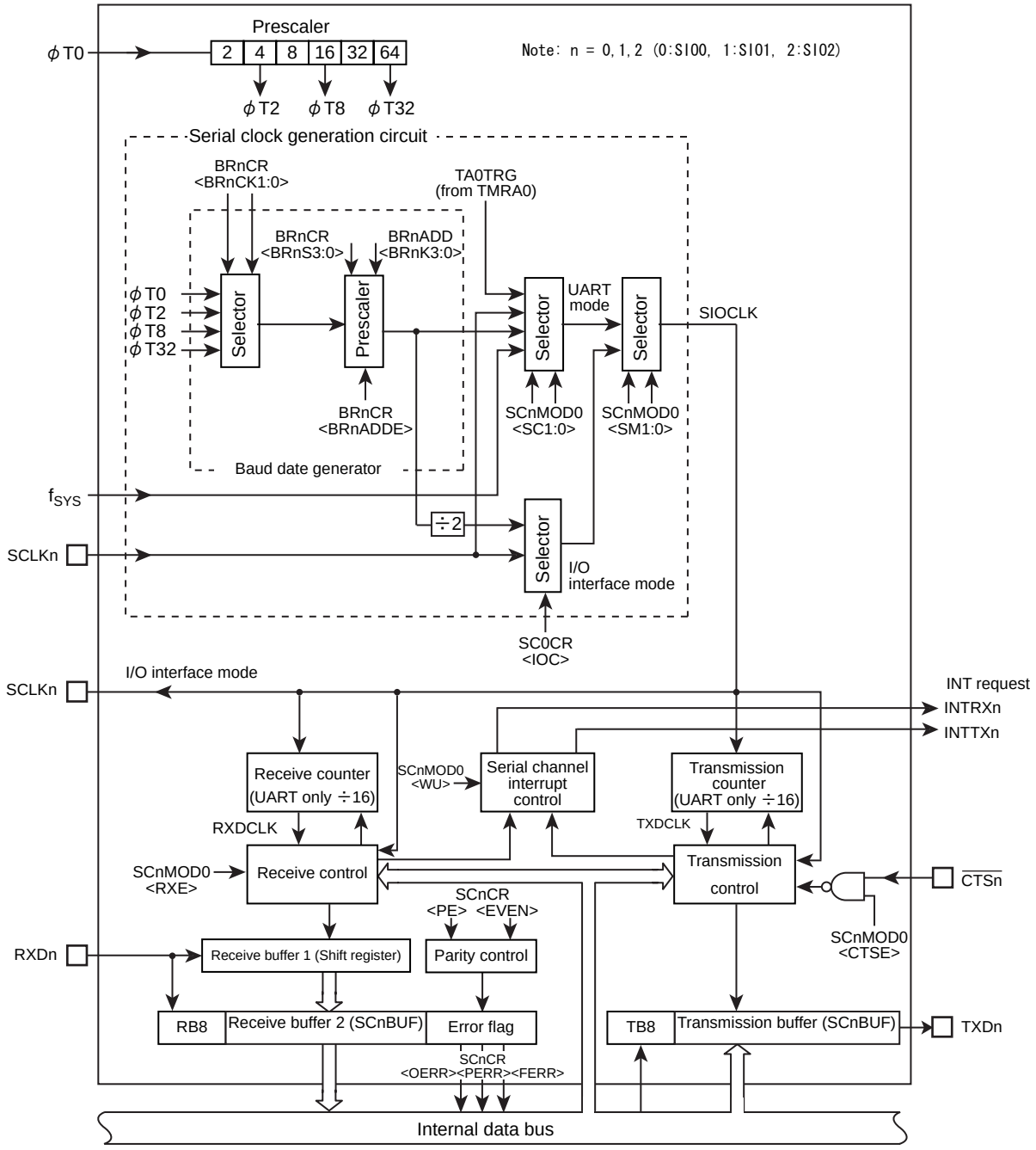


Figure 7-2 Block Diagram of the Serial Channel 0/1/2

7.2 Operation of Each Circuit

7.2.1 Prescaler

A 6-bit prescaler generates an operation clock for SIO0. The prescaler is acteve only when a baud rate generator is specified as a serial transfer clock. As an input clock of the prescaler, be sure to set SYSCR0<PRCK1> to “0” and then specify f_{FPH}. This clock is used for ϕT0 with being divided by 4.

Table 7-2 shows prescaler clock resolution into the baud rate generator.

Table 7-2 Prescaler Clock Resolution to Baud Rate Generator

Select System Clock <SYSCK>	Gear Value <GEAR2:0>	Select Prescaler Clock <PRCK1>	Prescaler Output Clock Resolution			
			ϕT0	ϕT2	ϕT8	ϕT32
1 (fs)	XXX	0 (1/1) f_{FPH}	2^2/fs	2^4/fs	2^6/fs	2^8/fs
0 (fc)	000 (fc)		2^2/fc	2^4/fc	2^6/fc	2^8/fc
	001 (fc/2)		2^3/fc	2^5/fc	2^7/fc	2^9/fc
	010 (fc/4)		2^4/fc	2^6/fc	2^8/fc	2^{10}/fc
	011 (fc/8)		2^5/fc	2^7/fc	2^9/fc	2^{11}/fc
	100 (fc/16)		2^6/fc	2^8/fc	2^{10}/fc	2^{12}/fc

The baud rate generator selects between 4 clock inputs: ϕT0, ϕT2, ϕT8, and ϕT32 among the prescaler outputs.

7.2.2 Baud rate generator

The baud rate generator is a circuit which generates transmission and receiving clocks which determine the transmission rate of the serial channels.

The input clock to the baud rate generator, ϕT0, ϕT2, ϕT8 or ϕT32, is generated by the 6-bit prescaler which is shared by the timers. One of these input clocks is selected using the BR0CR<BR0CK1:0> field in the baud rate generator control register.

The baud rate generator includes a frequency divider, which divides the frequency by 1, N + (16 – K)/16 or 16 values, determining the transmission rate. The transmission rate is determined by the settings of BR0CR<BR0ADDE><BR0S3:0> and BR0ADD<BR0K3:0>.

7.2.2.1 In UART mode

(1) When BR0CR<BR0ADDE> = 0

The settings BR0ADD<BR0K3:0> are ignored. The baud rate generator divides the selected prescaler clock by N, which is set in BR0CK<BR0S3:0>. (N = 1, 2, 3 ... 16)

(2) When BR0CR<BR0ADDE> = 1

The N + (16 – K)/16 division function is enabled. The baud rate generator divides the selected prescaler clock by N + (16 – K)/16 using the value of N set in BR0CR<BR0S3:0> (N = 2, 3 ... 15) and the value of K set in BR0ADD<BR0K3:0> (K = 1, 2, 3 ... 15)

Note: If N = 1 and N = 16, the N + (16 – K)/16 division function is disabled. Set BR0CR<BR0ADDE> to "0".

7.2.2.2 In I/O interface mode

The N + (16 – K)/16 division function is not available in I/O interface mode. Set BR0CR<BR0ADDE> to "0" before dividing by N.

The method for calculating the transmission rate when the baud rate generator is used is explained below.

(1) In UART mode

$$\text{Baud rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divider for baud rate generator}} \div 16$$

(2) In I/O interface mode

$$\text{Baud rate} = \frac{\text{Input clock of baud rate generator}}{\text{Frequency divider for baud rate generator}} \div 2$$

7.2.2.3 Integer divider (N divider)

For example, when the source clock frequency (fc) =19.6608 MHz, the input clock frequency = ϕT2 (fc/16), the frequency divider N (BR0CR<BR0S3:0>) = 8, and BR0CR<BR0ADDE> = 0, the baud rate in UART mode is as follows:

*Clock state	System clock:	High frequency (fc)
	Clock gear:	1 (fc)
	Prescaler clock:	f_{FPH}

$$\text{Baudrate} = \frac{fc/16}{8} \div 16$$

$$= 19.6608 \times 10^6 \div 16 \div 8 \div 16 = 9600 \text{ (bps)}$$

Note: The + (16 - K)/16 division function is disabled and setting BR0ADD<BR0K3:0> is invalid.

7.2.2.4 N + (16 - K)/16 divider (UART mode only)

Accordingly, when the source clock frequency (fc) = 15.9744 MHz, the input clock frequency = ϕT2, the frequency divider N (BR0CR<BR0S3:0>) = 6, K (BR0ADD<BR0K3:0>) = 8, and BR0CR<BR0ADDE> = 1, the baud rate in UART mode is as follows:

*Clock state	System clock:	High frequency (fc)
	Clock gear:	1 (fc)
	Prescaler clock:	f_{FPH}

$$\text{Baudrate} = \left(\frac{\text{fc}/16}{6 + \frac{(16-8)}{16}} \div 16 \right)$$

$$= 15.9744 \times 10^6 \div 16 \div \left(6 + \frac{8}{16}\right) \div 16 = 9600(\text{bps})$$

Table 7-3 show examples of UART mode transfer rates.

Additionally, the external clock input is available in the serial clock.

The method for calculating the baud rate is explained below:

- In UART mode
  Baud rate = External clock input frequency ÷ 16
  It is necessary to satisfy (External clock input cycle) ≥ $4/f_{SYS}$
- In I/O interface mode
  Baud rate = External clock input frequency
  It is necessary to satisfy (External clock input cycle) ≥ $16/f_{SYS}$

Table 7-3 UART Baud Rate Selection

(When baud rate generator is used and BR0CR<BR0ADDE>=0, SYSCR0<PRCK>=0) Unit (kbps)

fc [MHz]	Input Clock / Frequency Divider N	ϕT0 (fc/4)	ϕT2 (fc/16)	ϕT8 (fc/64)	ϕT32 (fc/256)
7.3728	1	115.200	28.800	7.200	1.800
↑	3	38.400	9.600	2.400	0.600
↑	6	19.200	4.800	1.200	0.300
↑	A	11.520	2.880	0.720	0.180
↑	C	9.600	2.400	0.600	0.150
↑	F	7.680	1.920	0.480	0.120
9.8304	1	153.600	38.400	9.600	2.400
↑	2	76.800	19.200	4.800	1.200
↑	4	38.400	9.600	2.400	0.600
↑	5	30.720	7.680	1.920	0.480
↑	8	19.200	4.800	1.200	0.300
↑	10	9.600	2.400	0.600	0.150

Note: Transmission rates in I/O interface mode are eight times faster than the values given above.

Timer out clock (TA0TRG) can be used for source clock of UART mode only.

Calculation method the frequency of TA0TRG

Frequency of TA0TRG = Baud rate × 16

Note: In case of I/O interface mode, prohibit to use TA0TRG for source clock.

7.2.3 Serial clock generation circuit

This circuit generates the basic clock for transmission and receiving data.

7.2.3.1 In I/O interface mode

In SCLK output mode with the setting SC0CR<IOC> = "0", the basic clock is generated by dividing the output of the baud rate generator by 2, as described previously.

In SCLK input mode with the setting SC0CR<IOC> = "1", the rising edge or falling edge will be detected according to the setting of the SC0CR<SCLKS> register to generate the basic clock.

7.2.3.2 In UART mode

The SC0MOD0<SC1:0> setting determines whether the baud rate generator clock, the internal system clock f_{SYS}, the match detect signal from timer TMRA0 or the external clock (SCLK0) is used to generate the basic clock SIOCLK.

7.2.4 Receiving counter

The receiving counter is a 4-bit binary counter used in UART mode which counts up the pulses of the SIOCLK clock. It takes 16 SIOCLK pulses to receive 1 bit of data; each data bit is sampled three times – on the 7th, 8th and 9th clock cycles.

The value of the data bit is determined from these three samples using the majority rule.

For example, if the data bit is sampled respectively as "1", "0" and "1" on 7th, 8th and 9th clock cycles, the received data bit is taken to be "1". A data bit sampled as "0", "0" and "1" is taken to be "0".

7.2.5 Receiving control

7.2.5.1 In I/O interface mode

In SCLK output mode with the setting SC0CR<IOC> = "0", the RXD0 signal is sampled on the rising or falling edge of the shift clock which is output on the SCLK0 pin, according to the SC0CR<SCLKS> setting.

In SCLK input mode with the setting SC0CR<IOC> = "1", the RXD0 signal is sampled on the rising or falling edge of the SCLK0 input, according to the SC0CR<SCLKS> setting.

7.2.5.2 In UART mode

The receiving control block has a circuit which detects a start bit using the majority rule. Received bits are sampled three times; when two or more out of three samples are "0", the bit is recognized as the start bit and the receiving operation commences.

The values of the data bits that are received are also determined using the majority rule.

7.2.6 Receiving buffers

To prevent overrun errors, the receiving buffers are arranged in a double-buffer structure.

Received data is stored one bit at a time in receiving buffer 1 (which is a shift register).

When 7 or 8 bits of data have been stored in receiving buffer 1, the stored data is transmitted to receiving buffer 2 (SC0BUF); this causes an INTRX0 interrupt to be generated. The CPU only reads receiving buffer 2 (SC0BUF). Even before the CPU reads receiving buffer 2 (SC0BUF), the received data can be stored in receiving buffer 1. However, unless receiving buffer 2 (SC0BUF) is read before all bits of the next data are received by receiving buffer 1, an overrun error occurs. If an overrun error occurs, the contents of receiving buffer 1 will be lost, although the contents of receiving buffer 2 and SC0CR<RB8> will be preserved.

SC0CR<RB8> is used to store either the parity bit – added in 8-bit UART mode – or the most significant bit (MSB) – in 9-bit UART mode.

In 9-bit UART mode the wakeup function for the slave controller is enabled by setting SC0MOD0<WU> to "1"; in this mode INTRX0 interrupts occur only when the value of SC0CR<RB8> is "1".

Note 1: The double buffer structure does not support SC0CR<RV08>.
Note 2: If the CPU reads receive buffer 2 while data is being transferred from receive buffer 1 to receive buffer 2, the data may not be read properly. To avoid this situation, a read of receive buffer 2 should be triggered by a receive interrupt.

7.2.7 Transmission counter

The transmission counter is a 4-bit binary counter which is used in UART mode and which, like the receiving counter, counts the SIOCLK clock pulses; a TXDCLK pulse is generated every 16 SIOCLK clock pulses.

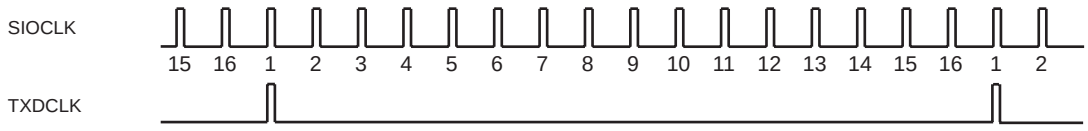


Figure 7-3 Generation of the Transmission Clock

7.2.8 Transmission controller

7.2.8.1 In I/O interface mode

In SCLK output mode with the setting SC0CR<IOC> = "0", the data in the transmission buffer is output one bit at a time to the TXD0 pin on the rising or falling edge of the shift clock which is output on the SCLK0 pin, according to the SC0CR<SCLKS> setting.

In SCLK input mode with the setting SC0CR<IOC> = "1", the data in the transmission buffer is output one bit at a time on the TXD0 pin on the rising or falling edge of the SCLK0 input, according to the SC0CR<SCLKS> setting.

7.2.8.2 In UART mode

When transmission data sent from the CPU is written to the transmission buffer, transmission starts on the rising edge of the next TXDCLK.

7.2.8.3 Handshake function

Use of $\overline{\text{CTS0}}$ pin allows data can be sent in units of one frame; thus, overrun errors can be avoided. The handshake function is enabled or disabled by the SC0MOD0<CTSE> setting.

When the $\overline{\text{CTS0}}$ pin goes high on completion of the current data send, data transmission is halted until the $\overline{\text{CTS0}}$ pin goes low again. However, the INTTX0 interrupt is generated, it requests the next data send to the CPU. The next data is written in the transmission buffer and data transmission is halted.

Though there is no $\overline{\text{RTS}}$ pin, a handshake function can be easily configured by setting any port assigned to be the $\overline{\text{RTS}}$ function. The $\overline{\text{RTS}}$ should be output "high" to request send data halt after data receive is completed by software in the $\overline{\text{RXD}}$ interrupt routine.

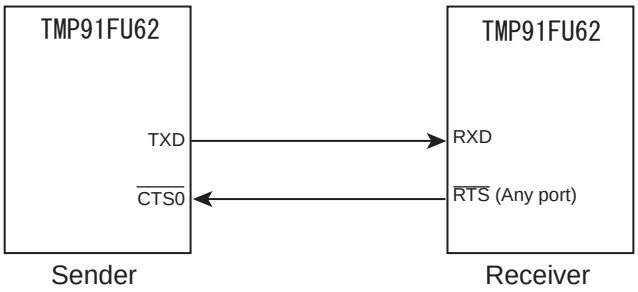
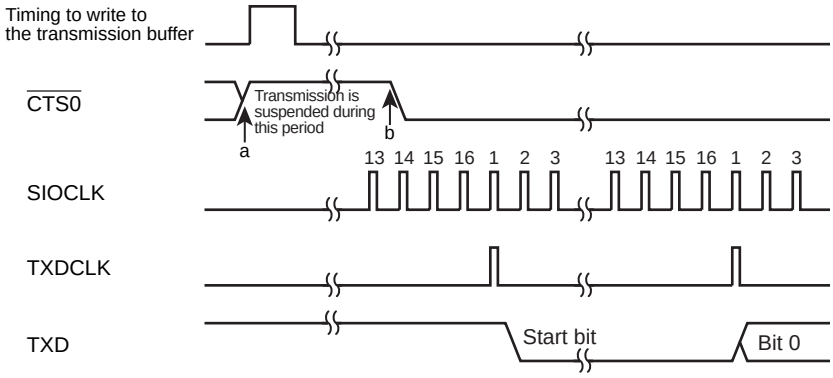


Figure 7-4 Handshake Function



Note 1: If the $\overline{\text{CTS0}}$ signal goes high during transmission, no more data will be sent after completion of the current transmission.

Note 2: Transmission starts on the first falling edge of the TXDCLK clock after the $\overline{\text{CTS0}}$ signal has fallen.

Figure 7-5 $\overline{\text{CTS0}}$ (Clear to send) Timing

7.2.9 Transmission buffer

The transmission buffer (SC0BUF) shifts out and sends the transmission data written from the CPU from the least significant bit (LSB) in order. When all the bits are shifted out, the transmission buffer becomes empty and generates an INTTX0 interrupt.

7.2.10 Parity control circuit

When SC0CR<PE> in the serial channel control register is set to "1", it is possible to transmit and receive data with parity. However, parity can be added only in 7-bit UART mode or 8-bit UART mode. The SC0CR<EVEN> field in the serial channel control register allows either even or odd parity to be selected.

In the case of transmission, parity is automatically generated when data is written to the transmission buffer SC0BUF. The data is transmitted after the parity bit has been stored in SC0BUF<TB7> in 7-bit UART mode or in SC0MOD0<TB8> in 8-bit UART mode. SC0CR<PE> and SC0CR<EVEN> must be set before the transmission data is written to the transmission buffer.

In the case of receiving, data is shifted into receiving buffer 1, and the parity is added after the data has been transmitted to receiving buffer 2 (SC0BUF), and then compared with SC0BUF<RB7> in 7-bit UART mode or with SC0CR<RB8> in 8-bit UART mode. If they are not equal, a parity error is generated and the SC0CR<PERR> flag is set.

7.2.11 Error flags

Three error flags are provided to increase the reliability of data reception.

7.2.11.1 Overrun error <OERR>

If all the bits of the next data item have been received in receiving buffer 1 while valid data still remains stored in receiving buffer 2 (SC0BUF), an overrun error is generated.

The below is a recommended flow when the overrun error is generated.

(INTRX interrupt routine)

1. Read receiving buffer
2. Read error flag
3. if <OERR> = 1
   then
   a. Set to disable receiving (Write "0" to SC0MOD0<RXE>)
   b. Wait to terminate current frame
   c. Read receiving buffer
   d. Read error flag
   e. Set to enable receiving (Write "1" to SC0MOD0<RXE>)
   f. Request to transmit again
4. Other

Note: Overrun errors are generated only with regard to receive buffer 2 (SC0BUF). Thus, if SC0CR<RB8> is not read, no overrun error will occur.

7.2.11.2 Parity error <PERR>

The parity generated for the data shifted into receiving buffer 2 (SC0BUF) is compared with the parity bit received via the RXD pin. If they are not equal, a parity error is generated.

Note: The parity error flag is cleared every time it is read. However, if a parity error is detected twice in succession and the parity error flag is read between the two parity errors, it may seem as if the flag had not been cleared. To avoid this situation, a read of the parity error flag should be triggered by a receive interrupt.

7.2.11.3 Framing error <FERR>

The stop bit for the received data is sampled three times around the center. If the majority of the samples are “0”, a framing error is generated.

7.2.12 Timing generation

7.2.12.1 In UART mode

Table 7-4 Receiving

Mode	9 Bits	8 Bits + Parity	8 Bits, 7 Bits + Parity, 7 Bits
Interrupt timing	Center of last bit (Bit8)	Center of last bit (Parity bit)	Center of stop bit
Framing error timing	Center of stop bit	Center of stop bit	Center of stop bit
Parity error timing	–	Center of last bit (Parity bit)	Center of stop bit
Overrun error timing	Center of last bit (Bit8)	Center of last bit (Parity bit)	Center of stop bit

Note 1: In 9 Bits and 8 Bits + Parity mode, interrupts coincide with the ninth bit pulse. Thus, when servicing the interrupt, it is necessary to wait for a 1-bit period (to allow the stop bit to be transferred) to allow checking for a framing error.

Note 2: The higher the transfer rate, the later than the middle receive interrupts and errors occur.

Table 7-5 Transmitting

Mode	9 Bits	8 Bits + Parity	8 Bits, 7 Bits + Parity, 7 Bits
Interrupt timing	Just before stop bit is transmitted	Just before stop bit is transmitted	Just before stop bit is transmitted

7.2.12.2 I/O interface

Transmission interrupt timing	SCLK output mode	Immediately after the last bit. (See Figure 7-8)
	SCLK input mode	Immediately after rise of last SCLK signal rising mode, or immediately after fall in falling mode. (See Figure 7-9)
Receiving interrupt timing	SCLK output mode	Timing used to transmit received data to receive buffer 2 (SC0BUF) (e.g., immediately after last SCLK). (See Figure 7-10)
	SCLK input mode	Timing used to transmit received data to receive buffer 2 (SC0BUF) (e.g., immediately after last SCLK). (See Figure 7-11)

7.3 SFR

Serial Control Register (Read-modify-write instructions are prohibited.)

		7	6	5	4	3	2	1	0
SC0CR (0201H)	Bit symbol	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
	Read/Write	R	R/W		R (Cleared to "0" when read)			R/W	
	After reset	Undefined	0	0	0	0	0	0	0
SC1CR (0209H) SC2CR (0211H)	Function	Received data bit8	Parity 0: Odd 1: Even	Parity addition 0: Disable 1: Enable	Overrun error flag 0: Undetect error 1: Detect error	Parity error flag 0: Undetect error 1: Detect error	Framing error flag 0: Undetect error 1: Detect error	Edge selection for SCLK pin (I/O mode) 0: SCLK↑ 1: SCLK↓	Edge selection for SCLK pin (I/O mode) 0: SCLK↑ 1: SCLK↓

Note1: As all error flags are cleared after reading, do not test only a single bit with a bit-testing instruction.
Note2: A baud rate generator SCnCR<IOC> = "0" is unavailable as an input clock for an I/O interface if a prescaler clock is set to fc/16 whenSYSCR0<PRCK1> is "1".
Note3: n =0, 1, 2.

Serial Mode Control Register 0

		7	6	5	4	3	2	1	0
SC0MOD0 (0202H)	Bit symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
SC1MOD0 (020AH) SC2MOD0 (0212H)	Function	Transmission data bit8	Handshake function 0: Disable 1: Enable	Receive function 0: Disable 1: Enable	Wakeup function 0: Disable 1: Enable	Serial transmission mode 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial transmission clock (UART) 00: Timer TA0TRG 01: Baud rate generator 10: Internal clock f_{SYS} 11: External clock (SCLK input)	

Note: SCLKpin and $\overline{CTS}$ pin

	SCLK pin	$\overline{CTS}$ pin
SIO0	SCLK0	$\overline{CTS0}$
SIO1	SCLK1	$\overline{CTS1}$
SIO2	SCLK2	$\overline{CTS2}$

Note2: A baud rate generator SCnMOD0<SC1:0> = "01" is unavailable as a serial transfer clock if a prescaler clock is set to fc/16 whenSYSCR0<PRCK1> is "1".
Note3: n =0, 1, 2.

Serial Mode Control Register 1

		7	6	5	4	3	2	1	0
SC0MOD1 (0205H) SC1MOD1 (020DH) SC2MOD1 (0215H)	Bit symbol	I2S0	FDPX0	–	–	–	–	–	–
	Read/Write	R/W	R/W	–	–	–	–	–	–
	After reset	0	0	–	–	–	–	–	–
	Bit symbol	I2S1	FDPX1	–	–	–	–	–	–
	Read/Write	R/W	R/W	–	–	–	–	–	–
	After reset	0	0	–	–	–	–	–	–
	Bit symbol	I2S2	FDPX2	–	–	–	–	–	–
	Read/Write	R/W	R/W	–	–	–	–	–	–
	After reset	0	0	–	–	–	–	–	–
	Function	IDLE2 0: Stop 1: Run	Duplex 0: Half 1: Full						

Baud Rate Generator Control

		7	6	5	4	3	2	1	0
BR0CR (0203H)	Bit symbol	–	BR0ADDE	BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
BR1CR (020BH)	Bit symbol	–	BR1ADDE	BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
BR2CR (0213H)	Bit symbol	–	BR2ADDE	BR2CK1	BR2CK0	BR2S3	BR2S2	BR2S1	BR2S0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Always write “0”.	+ (16 - K)/16 division 0: Disable 1: Enable	Input clock selection for baud rate generator 00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Setting of the divided frequency “N”			

		7	6	5	4	3	2	1	0
BR0ADD (0204H)	Bit symbol	–	–	–	–	BR0K3	BR0K2	BR0K1	BR0K0
	Read/Write	–	–	–	–	R/W			
	After reset	–	–	–	–	0	0	0	0
BR1ADD (020CH)	Bit symbol	–	–	–	–	BR1K3	BR1K2	BR1K1	BR1K0
	Read/Write	–	–	–	–	R/W			
	After reset	–	–	–	–	0	0	0	0
BR2ADD (0214H)	Bit symbol	–	–	–	–	BR2K3	BR2K2	BR2K1	BR2K0
	Read/Write	–	–	–	–	R/W			
	After reset	–	–	–	–	0	0	0	0
	Function					Sets frequency divisor “K” (Divided by N + (16 - K)/16)			

Baud rate generator frequency divisor setting

	BRnCR<BRnADDE> = 1		BRnCR<BRnADDE> = 0
BRnCR <BRnS3:0> / BRnADD <BRnK3:0>	0000(N=16) or 0001(N=1)	0010(N=2) to 1111(N=15)	0001(N=1)UART only to 1111(N=15) 0000(N=16)
0000	Disable	Disable	Divided by N
0001 (K = 1) to 1111 (K = 15)	Disable	Divided by N + (16 - K)/16	

Note: Availability of +(16 - K)/16 division function

N	UART mode	I/O mode
2 to 15	O	x
1, 16	x	x

The baud rate generator can be set “1” in UART mode and disable + (16 - K)/16 division function. Don’t use in I/O interface mode.

Note: Set BR1CR<BR1ADDE> to 1 after setting K (K = 1 to 15) to BR1ADD<BR1K3:0> when N+ (16 - K)/16 division function is used.

Note: n = 0,1,2

Serial Transmission/receiving Buffer Registers (Read-modify-write instructions are prohibited.)

	7	6	5	4	3	2	1	0	
SC0BUF (0200H)	TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(Transmission)
SC1BUF (0208H)	7	6	5	4	3	2	1	0	
SC2BUF (0210H)	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(Receiving)

7.4 Operation in Each Mode

7.4.1 Mode 0 (I/O interface mode)

This mode allows an increase in the number of I/O pins available for transmitting data to or receiving data from an external shift register.

This mode includes the SCLK output mode to output synchronous clock SCLK and SCLK input mode to input external synchronous clock SCLK.

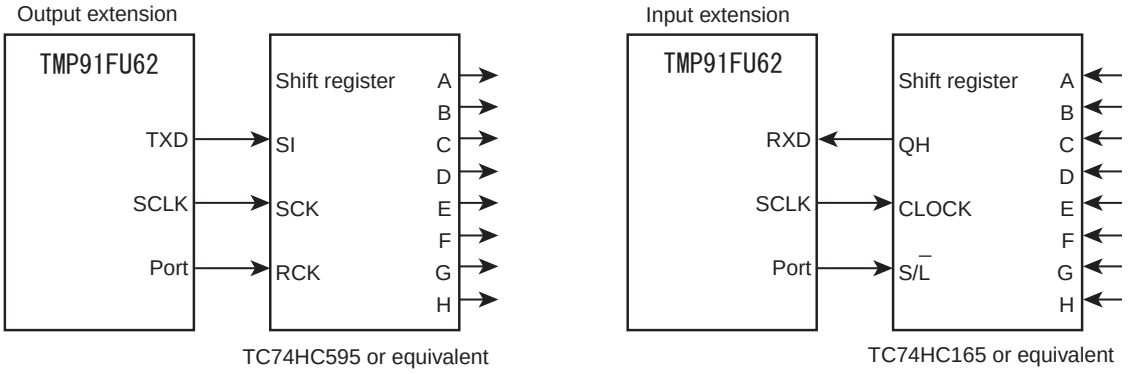


Figure 7-6 SCLK Output Mode Connection Example

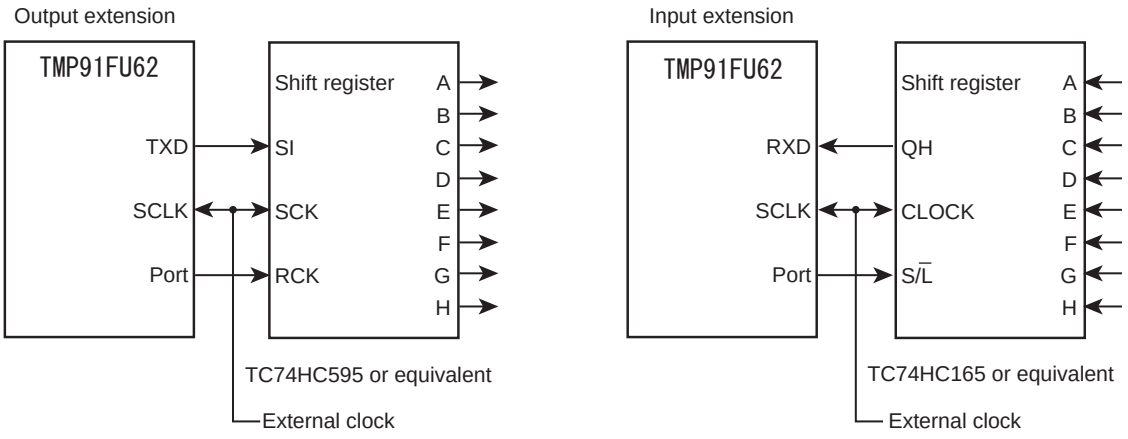


Figure 7-7 SCLK Input Mode Connection Example

7.4.1.1 Transmission

In SCLK output mode 8-bit data and a synchronous clock are output on the TXD0 and SCLK0 pins respectively each time the CPU writes the data to the transmission buffer. When all data is output, INTES0<ITX0C> will be set to generate the INTTX0 interrupt.

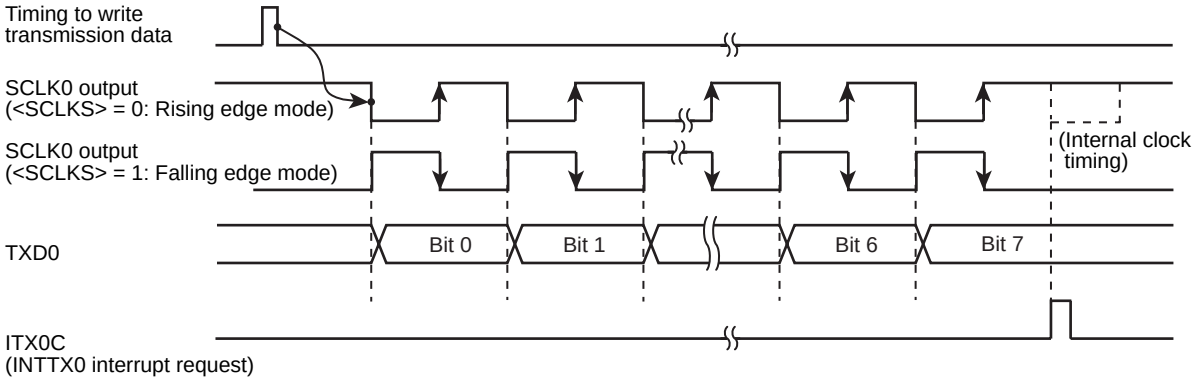


Figure 7-8 Transmitting Operation in I/O Interface Mode (SCLK output mode)

In SCLK input mode, 8-bit data is output on the TXD0 pin when the SCLK0 input becomes active after the data has been written to the transmission buffer by the CPU.

When all data is output, INTES0<ITX0C> will be set to generate INTTX0 interrupt.

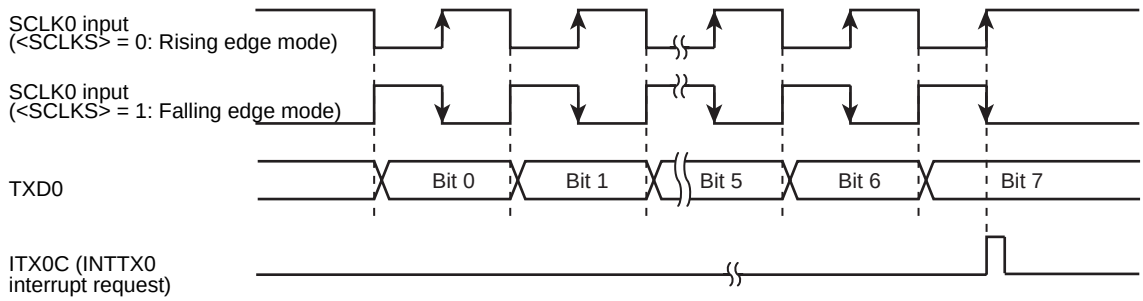


Figure 7-9 Transmitting Operation in I/O Interface Mode (SCLK input mode)

7.4.1.2 Receiving

In SCLK output mode, the synchronous clock is outputted from SCLK0 pin and the data is shifted to receiving buffer 1. This starts when the receive interrupt flag INTES0<IRX0C> is cleared by reading the received data. When 8-bit data are received, the data will be transmitted to receiving buffer 2 (SC0BUF according to the timing shown below) and INTES0<IRX0C> will be set to generate INTRX0 interrupt.

The outputting for the first SCLK0 starts by setting SC0MOD0<RXE> to "1".

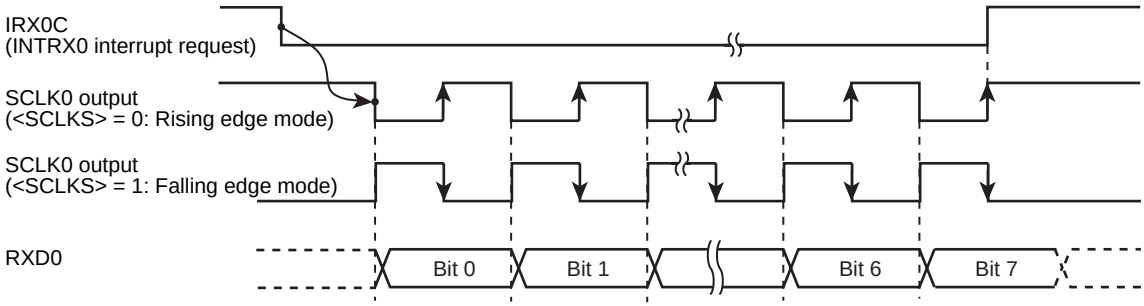


Figure 7-10 Receiving Operation in I/O Interface Mode (SCLK output mode)

In SCLK input mode, the data is shifted to receiving buffer 1 when the SCLK input becomes active after the receive interrupt flag INTES0<IRX0C> is cleared by reading the received data. When 8-bit data is received, the data will be shifted to receiving buffer 2 (SC0BUF according to the timing shown below) and INTES0<IRX0C> will be set again to be generate INTRX0 interrupt.

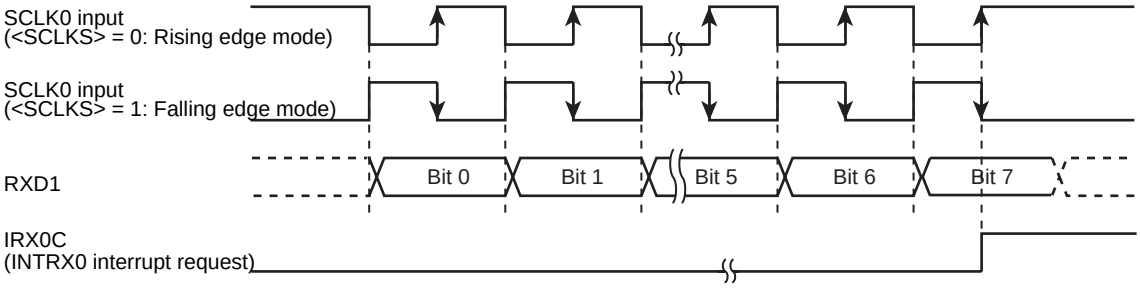


Figure 7-11 Receiving Operation in I/O Interface Mode (SCLK input mode)

Note: The system must be put in the receive enable state (SC0MOD0<RXE> = 1) before data can be received.

7.4.1.3 Transmission and receiving (Full duplex mode)

When the full duplex mode is used, set the level of receive interrupt to “0” and set enable the interrupt level (1 to 6) to the transmission interrupt. In the transmission interrupt program, the receiving operation should be done like the above example before setting the next transmission data.

Example: Channel 0, SCLK output
Baud rate = 9600 bps
fc = 14.7456 MHz

*Clock state		System clock:	High frequency (fc)
		Clock gear:	1 (fc)
		Prescaler clock:	f_{FPH}

	port setting								
	7	6	5	4	3	2	1	0	
INTES0	X	0	0	1	0	0	0	0	Set the INTTX0 level to 1. Set the INTRX0 level to 0.
SC0MOD0	–	–	–	–	0	0	–	–	Select I/O interface mode.
SC0MOD1	–	1	X	X	X	X	X	X	Select full duplex mode.
SC0CR	–	–	–	–	–	–	0	0	SCLK0 output mode, transmit on falling edge mode, receive on rising edge mode.
BR0CR	0	0	1	1	0	0	1	1	Baud rate = 9600 bps
SC0MOD0	–	–	1	–	–	–	–	–	Enable receiving
SC0BUF	*	*	*	*	*	*	*	*	Set the transmit data and start.

	7	6	5	4	3	2	1	0	
Acc SC0BUF									Read the receiving buffer.
SC0BUF	*	*	*	*	*	*	*	*	Set the next transmission data.

Note: X: Don't care, –: No change, *: Data

7.4.2 Mode 1 (7-bit UART mode)

7-bit UART mode is selected by setting serial channel mode register SC0MOD0<SM1:0> to “01”.

In this mode, a parity bit can be added. Use of a parity bit is enabled or disabled by the setting of the serial channel control register SC0CR<PE> bit; whether even parity or odd parity will be used is determined by the SC0CR<EVEN> setting when SC0CR<PE> is set to “1” (Enabled).

Example: When transmission data of the following format, the control registers should be set as described below. This explanation applies to channel 0.

Figure 7-12 7-bit UART mode

*Clock state System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: System clock

		7	6	5	4	3	2	1	0	
SC0MOD0	←	–	–	–	–	0	1	0	1	Select 7-bit UART mode.
SC0CR	←	–	1	1	–	–	–	–	–	Add even parity.
BR0CR	←	0	0	1	0	0	1	0	1	Set the transmission rate to 2400 bps.
INTES0	←	X	1	0	0	–	–	–	–	Enable the INTTX0 interrupt and set it to interrupt level 4.
SC0BUF	←	*	*	*	*	*	*	*	*	Set data for transmission.

Note: X: Don't care, –: No change, *: Data

7.4.3 Mode 2 (8-bit UART mode)

8-bit UART mode is selected by setting SC0MOD0<SM1:0> to “10”. In this mode, a parity bit can be added (Use of a parity bit is enabled or disabled by the setting of SC0CR<PE>); whether even parity or odd parity will be used is determined by the SC0CR<EVEN> setting when SC0CR<PE> is set to “1” (Enabled).

Example: When receiving data of the following format, the control registers should be set as described below.

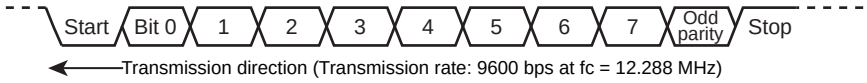


Figure 7-13 8-bit UART mode

*Clock state System clock: High frequency (fc)
Clock gear: 1 (fc)
Prescaler clock: System clock

		7	6	5	4	3	2	1	0	
SC0MOD0	←	–	–	1	–	1	0	0	1	Enable receiving in 8-bit UART mode.
SC0CR	←	–	0	1	–	–	–	–	–	Add odd parity.
BR0CR	←	0	0	0	1	0	1	0	1	Set the transmission rate to 9600 bps.
INTES0	←	–	–	–	–	X	1	0	0	Enable the INTTX0 interrupt and set it to interrupt level 4.

Note: X: Don't care, –: No change

```
Acc ← SC0CR AND 00011100          Check for errors.
if Acc ≠ 0 then ERROR
Acc ← SC0BUF                      Read the received data.
```

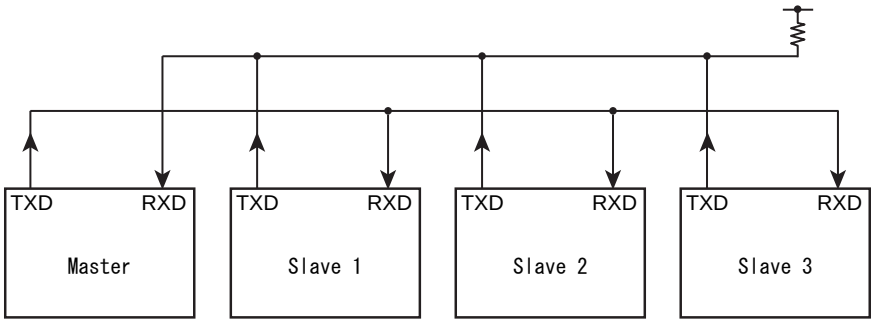
7.4.4 Mode 3 (9-bit UART mode)

9-bit UART mode is selected by setting SC0MOD0<SM1:0> to “11”. In this mode parity bit cannot be added.

In the case of transmission, the MSB (9th bit) is written to SC0MOD0<TB8>. In the case of receiving, it is stored in SC0CR<RB8>. When the buffer is written and read, the MSB is read or written first, before the rest of the SC0BUF data.

7.4.4.1 Wakeup function

In 9-bit UART mode, the wakeup function for slave controllers is enabled by setting SC0MOD0<WU> to “1”. The interrupt INTRX0 occurs only when <RB8> = “1”.

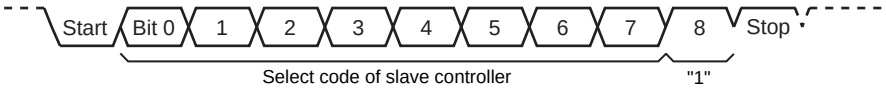


Note: The TXD pin of each slave controller must be in open-drain output mode.

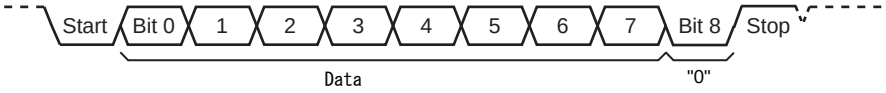
Figure 7-14 Serial Link Using Wakeup Function

7.4.4.2 Protocol

1. Select 9-bit UART mode on the master and slave controllers.
2. Set the SC0MOD0<WU> bit on each slave controller to “1” to enable data receiving.
3. The master controller transmits one-frame data including the 8-bit select code for the slave controllers. The MSB (Bit8) <TB8> is set to “1”.



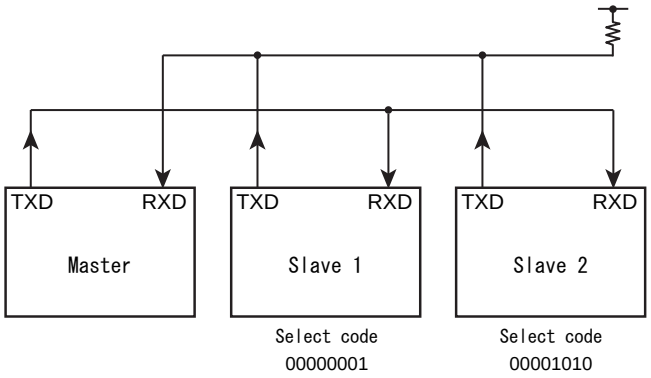
4. Each slave controller receives the above frame. Each controller checks the above select code against its own select code. The controller whose code matches clears its WU bit to “0”.
5. The master controller transmits data to the specified slave controller whose SC0MOD0<WU> bit is cleared to “0”. The MSB (Bit8) <TB8> is cleared to “0”.



6. The other slave controllers (whose <WU> bits remain at “1”) ignore the received data because their MSBs (Bit8 or <RB8>) are set to “0”, disabling INTRX0 interrupts.
   The slave controller (WU bit = “0”) can transmit data to the master controller, and it is possible to indicate the end of data receiving to the master controller by this transmission.

7.4.4.3 Example

To link two slave controllers serially with the master controller using the internal clock f_{SYS} as the transfer clock.



Main settings (except port setting)

Register	MSB								LSB	
		7	6	5	4	3	2	1	0	
INTES0	←	X	1	0	0	X	1	0	1	Enable the INTTX0 interrupt and set it to interrupt level 4. Enable the INTRX0 interrupt and set it to interrupt level 5.
SC0MOD0	←	1	0	1	0	1	1	1	0	Set f_{SYS} as the transmission clock for 9-bit UART mode.
SC0BUF	←	0	0	0	0	0	0	0	1	Set the select code for slave controller 1.

INTTX0 interrupt

Register	MSB								LSB	
		7	6	5	4	3	2	1	0	
SC0MOD0	←	0	–	–	–	–	–	–	–	Set TB8 to “0”.
SC0BUF	←	*	*	*	*	*	*	*	*	Set data for transmission.

Main settings (except port setting)

Register	MSB								LSB	
		7	6	5	4	3	2	1	0	
INTES0	←	X	1	0	1	X	1	1	0	Enable INTRX0 and INTTX0.
SC0MOD0	←	0	0	1	1	1	1	1	0	Set <WU> to “1” in 9-bit UART transmission mode using f_{SYS} as the transmission clock.

INTRX0 interrupt

Register	MSB								LSB	
		7	6	5	4	3	2	1	0	
Acc ← SC0BUF, if Acc = select code										
then SC0MOD0	←	–	–	–	0	–	–	–	–	Clear <WU> to “0”.

8. Serial Bus Interface (SBI)

The TMP91FU62 has a 1-channel serial bus interface which an I^2C bus mode. This circuit supports only I^2C bus mode (Multi master).

The serial bus interface is connected to an external device through SDA0 and SCL0 in the I^2C bus mode.

8.1 Configuration

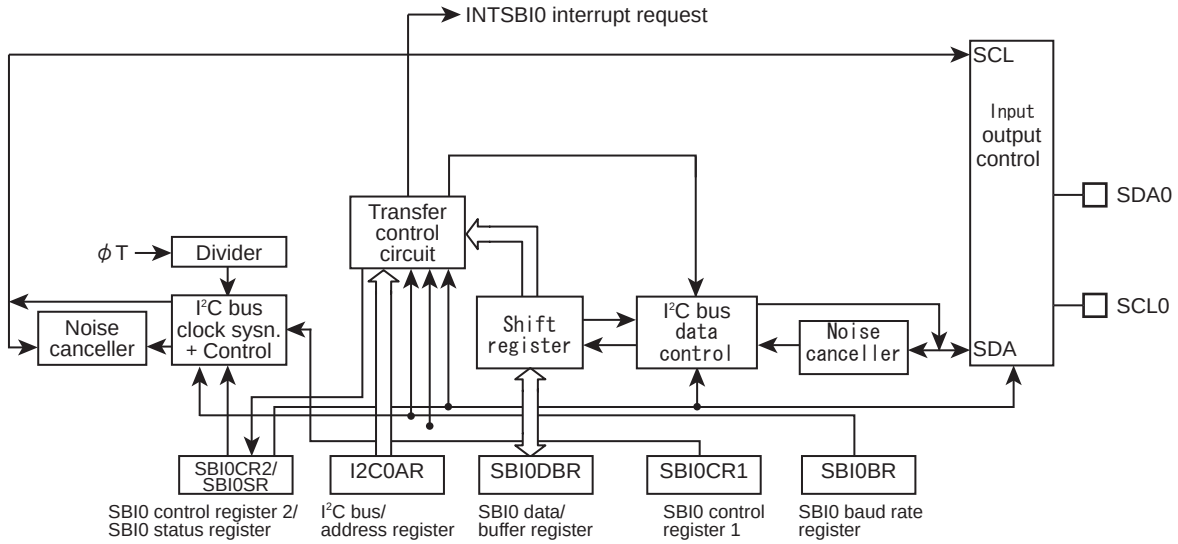


Figure 8-1 Serial Bus Interface (SBI)

8.2 Serial Bus Interface (SBI) Control

The following registers are used to control the serial bus interface and monitor the operation status.

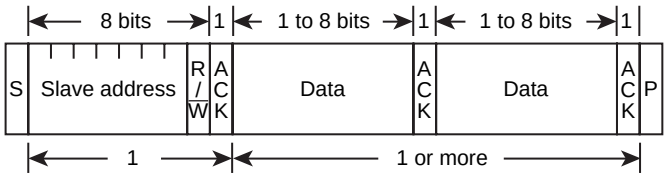
- Serial bus interface control register 0 (SBI0CR0)
- Serial bus interface control register 1 (SBI0CR1)
- Serial bus interface control register 2 (SBI0CR2)
- Serial bus interface data buffer register (SBI0DBR)
- I^2C bus address register (I2C0AR)
- Serial bus interface status register (SBI0SR)
- IDLE2 control register (SBI0BR)

8.3 Operation in I2C Bus Mode

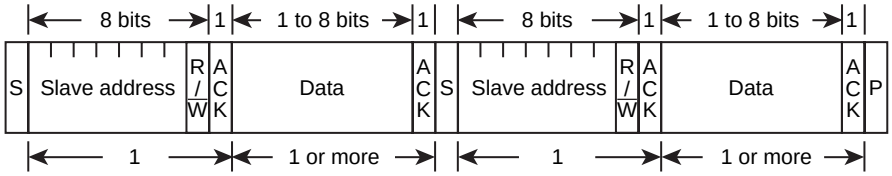
8.3.1 The Data Formats in the I2C Bus Mode

The data formats in the I2C bus mode is shown below.

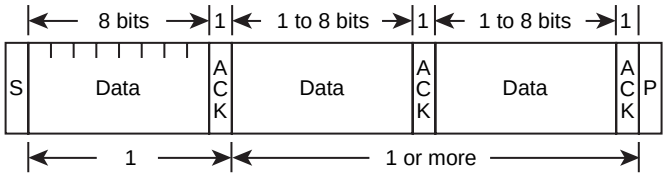
(a) Addressing format



(b) Addressing format (with restart)



(c) Free data format (Data transferred from master device to slave device)



S : Start condition
R/$\overline{W}$: Direction bit
ACK : Acknowledge bit
P : Stop condition

Figure 8-2 Data Format in the I2C Bus Mode

8.3.2 I^2C Bus Mode Control Register

The following registers are used to control and monitor the operation status when using the serial bus interface (SBI) in the I^2C bus mode.

Serial Bus Interface Control Register 0 (Read-modify-write instructions are prohibited.)

SBI0CR0 (0247H)

	7	6	5	4	3	2	1	0
Bit symbol	SBI0EN	–	–	–	–	–	–	–
Read/Write	R/W	R						
After reset	0	0	0	0	0	0	0	0
Function	SBI operation 0: disable 1: enable	Always read "0".						

Note <SBIEN>: When using SBI, <SBIEN> should be set "1" (SBI operation enable) before setting each register of SBI module.

Serial Bus Interface Control Register 1 (Read-modify-write instructions are prohibited.)

SBI0CR1 (0240H)

	7	6	5	4	3	2	1	0
Bit symbol	BC2	BC1	BC0	ACK	–	SCK2	SCK1	SCK0/ SWRMON
Read/Write	W			R/W	–	W		R/W
After reset	0	0	0	0	–	0	0	0/1
Function	Number of transferred bits (Note 1)			Acknowl-edge mode specification		Internal serial clock selection and software reset monitor (Note 2)		

Internal serial clock selection <SCK2:0> at write

SCK2:0	000	n = 4	– (Note3)	System clock: fc Clock gear: fc/1 fc =20 MHz (Internal SCL output) fscl = (f_{SYS}/2) / (2^n+36) [Hz]
	001	n = 5	73.53 kHz	
	010	n = 6	50.00 kHz	
	011	n = 7	30.49 kHz	
	100	n = 8	17.12 kHz	
	101	n = 9	9.12 kHz	
	110	n = 10	4.72 kHz	
	111	(Reserved)	(Reserved)	

Software reset state monitor <SWRMON> at read

SWRMON	0	During software reset
	1	Initial data

Acknowledge mode specification

ACK	0	Not generate clock pulse for acknowledge signal
	1	Generate clock pulse for acknowledge signal

Number of bits transferred

	<BC2:0>	<ACK> = 0		<ACK> = 1	
		Number of clock pulses	Bits	Number of clock pulses	Bits
BC2:0	000	8	8	9	8
	001	1	1	2	1
	010	2	2	3	2
	011	3	3	4	3
	100	4	4	5	4
	101	5	5	6	5
	110	6	6	7	6
	111	7	7	8	7

Note 1: For the frequency of the SCL line clock, see 8.3.3.3 “Serial clock”.

Note 2: Initial data of SCK0 is “0”,SWRMON is “1”.

Note 3: This I^2C bus circuit dose not support high-speed mode, it supports standard mode only. The fscl speed can be selected over 100 kbps by fc and <SCK2:0>, however it’s irregular operation.

Serial Bus Interface Control Register 2 (Read-modify-write instructions are prohibited.)

SBI0CR2 (0243H)

	7	6	5	4	3	2	1	0
Bit symbol	MST	TRX	BB	PIN	SBIM1	SBIM0	SWRST1	SWRST0
Read/Write	W				W		W	
After reset	0	0	0	1	0	0	0	0
Function	Master/slave selection	Transmitter/ receiver selection	Start/stop condition generation	Cancel INTSBI interrupt request	Serial bus interface operation mode selection		Software reset generate	

Software reset generate

SWRST1:0	10 ↓ 01	Write “10” and “01”, then an internal reset signal is generated

Serial bus interface operating mode selection (Note 2)

SBIM1:0	00	Port mode (Serial bus interface output disabled)
	01	(Reserved)
	10	I^2C bus mode
	11	(Reserved)

INTSBI interrupt request

PIN	0	–
	1	Cancel interrupt request

Start/stop condition generation

BB	0	Generates the stop condition
	1	Generates the start condition

Transmitter/receiver selection

TRX	0	Receiver
	1	Transmitter

Master/slave selection

MST	0	Slave
	1	Master

Note 1: Reading this register functions as SBI0SR register.

Note 2: Switch to port mode after confirming that the bus is free.

Switch a mode between I^2C bus mode and clocked-synchronous 8-bit SIO mode after confirming that input signals via port are high level.

Serial Bus Interface Status Register (Read-modify-write instructions are prohibited.)

SBI0SR (0243H)

	7	6	5	4	3	2	1	0
Bit symbol	MST	TRX	BB	PIN	AL	AAS	AD0	LRB
Read/Write	R							
After reset	0	0	0	1	0	0	0	0
Function	Master/slave status monitor	Transmitter/ receiver status monitor	I^2C bus status monitor	INTSBI interrupt request monitor	Arbitration lost detection monitor	Slave address match detection monitor	GENERAL CALL detection monitor	Last received bit monitor

Last received bit monitor

LRB	0	Last received bit was 0
	1	Last received bit was 1

GENERAL CALL detection monitor

AD0	0	Undetected
	1	GENERAL CALL detected

Slave address match detection monitor

AAS	0	Undetected
	1	Slave address match or GENERAL CALL detected

Arbitration lost detection monitor

AL	0	–
	1	Arbitration lost detected

INTSBI interrupt request monitor

PIN	0	Interrupt requested
	1	Interrupt canceled

I^2C bus status monitor

BB	0	Free
	1	Busy

Transmitter/receiver status monitor

TRX	0	Receiver
	1	Transmitter

Master/slave status monitor

MST	0	Slave
	1	Master

Note 1: Writing in this register functions as SBI0CR2.

Note 2: The initial data SBI0SR<PIN> is "1" if SBI operation is enable (SBI0CR0<SBI0EN> "1"). If SBI operation is disable (SBI0CR0<SBI0EN> "0"), the initial data of SBI0SR<PIN> is "0".

IDLE2 Control Register (Read-modify-write instructions are prohibited.)

SBI0BR (0244H)

	7	6	5	4	3	2	1	0
Bit symbol	–	I2SBI0	–	–	–	–	–	–
Read/Write	W	R/W	–	–	–	–	–	R/W
After reset	0	0	–	–	–	–	–	0
Function	Always write "0"	Operation in IDLE2 mode 0: Stop 1: Operate						Always write "0"

Serial Bus Interface Data Buffer Register (Read-modify-write instructions are prohibited.)

SBI0DBR (0241H)

	7	6	5	4	3	2	1	0
Bit symbol	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
Read/Write	R (Received)/W (Transfer)							
After reset	Undefined							

Note 1: When writing transmitted data, start from the MSB (bit7).Receiving data is placed from LSB (bit0).

Note 2: SBI0DBR can't be read the written data. Therefore read-modify-write instruction (e.g., "BIT" instruction) is prohibitted.

I^2C Bus Address Register (Read-modify-write instructions are prohibited.)

I2C0AR (0242H)

	7	6	5	4	3	2	1	0
Bit symbol	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ALS
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Slave address selection for when device is operating as slave device							Address recognition mode specification

Address recognition mode specification

ALS	0	Slave address recognition
	1	Non slave address recognition

8.3.3 Control in I^2C Bus Mode

8.3.3.1 Acknowledge mode specification

Set the SBI0CR1<ACK> to "1" for operation in the acknowledge mode. The TMP91FU62 generates an additional clock pulse for an acknowledge signal when operating in master mode. In the transmitter mode during the clock pulse cycle, the SDA pin is released in order to receive the acknowledge signal from the receiver. In the receiver mode during the clock pulse cycle, the SDA pin is set to the low in order to generate the acknowledge signal.

Clear the <ACK> to "0" for operation in the non-acknowledge mode, the TMP91FU62 does not generate a clock pulse for the acknowledge signal when operating in the master mode.

8.3.3.2 Number of transfer bits

The SBI0CR1<BC2:0> is used to select a number of bits for next transmitting and receiving data.

Since the <BC2:0> is cleared to "000" as a start condition, a slave address and direction bit transmission are always executed in 8 bits. Other than these, the <BC2:0> retains a specified value.

8.3.3.3 Serial clock

(1) Clock source

The SBI0CR1<SCK2:0> is used to select a maximum transfer frequency outputted on the SCL pin in master mode. Set the baud rates, which have been calculated according to the formula below, to meet the specifications of the I2C bus, such as the smallest pulse width of tLOW.

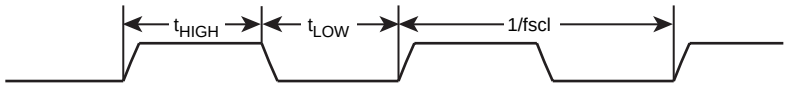


Figure 8-3 Clock Source

$t_{LOW} = (2^{n-1}+29)/f_{SBI}$

$t_{HIGH} = (2^{n-1}+7)/f_{SBI}$

$fscl = 1/(t_{LOW} + t_{HIGH}) = f_{SBI}/(2^n + 36)$

SBI0CR1<SCK2:0>	n
000	4
001	5
010	6
011	7
100	8
101	9
110	10

Note: f_{SBI} shows $f_{SYS}/2$

(2) Clock synchronization

In the I^2C bus mode, in order to wired-AND a bus, a master device which pulls down a clock line to low level, in the first place, invalidate a clock pulse of another master device which generates a high-level clock pulse. The master device with a high-level clock pulse needs to detect the situation and implement the following procedure.

The TMP91FU62 has a clock synchronization function for normal data transfer even when more than one master exists on the bus.

The example explains the clock synchronization procedures when two masters simultaneously exist on a bus.

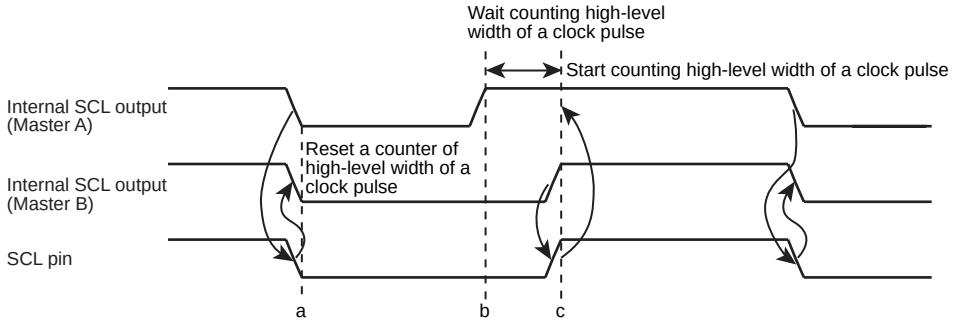


Figure 8-4 Clock Synchronization

As master A pulls down the internal SCL output to the low level at point “a”, the SCL line of the bus becomes the low level. After detecting this situation, master B resets a counter of high-level width of an own clock pulse and sets the internal SCL output to the low level.

Master A finishes counting low-level width of an own clock pulse at point “b” and sets the internal SCL output to the high level. Since master B holds the SCL line of the bus at the low level, master A waits for counting high-level width of an own clock pulse. After master B finishes counting low-level width of an own clock pulse at point “c” and master A detects the SCL line of the bus at the high level, and starts counting high level of an own clock pulse. The clock pulse on the bus is determined by the master device with the shortest high-level width and the master device with the longest low-level width from among those master devices connected to the bus.

8.3.3.4 Slave address and address recognition mode specification

When the TMP91FU62 is used as a slave device, set the slave address <SA6:0> and <ALS> to the I2C0AR. Clear the <ALS> to “0” for the address recognition mode.

8.3.3.5 Master/slave selection

Set the SBI0CR2<MST> to “1” for operating the TMP91FU62 as a master device. Clear the SBI0CR2<MST> to “0” for operation as a slave device. The <MST> is cleared to “0” by the hardware after a stop condition on the bus is detected or arbitration is lost.

8.3.3.6 Transmitter/receiver selection

Set the SBI0CR2<TRX> to "1" for operating the TMP91FU62 as a transmitter. Clear the <TRX> to "0" for operation as a receiver.

When data with an addressing format is transferred in slave mode, when a slave address with the same value that an I2C0AR or a GENERAL CALL is received (All 8-bit data are "0" after a start condition), the <TRX> is set to "1" by the hardware if the direction bit (R/$\overline{\mathbf{W}}$) sent from the master device is "1", and <TRX> is cleared to "0" by the hardware if direction bit is "0".

In the master mode, after an acknowledge signal is returned from the slave device, the <TRX> is cleared to "0" by the hardware if a transmitted direction bit is "1", and <TRX> is set to "1" by the hardware if direction is "0". When an acknowledge signal is not returned, the current condition is maintained.

The <TRX> is cleared to "0" by the hardware after a stop condition on the I^2C bus is detected or arbitration is lost.

8.3.3.7 Start/stop condition generation

When the SBI0SR<BB> is "0", slave address and direction bit which are set to SBI0DBR are output on a bus after generating a start condition by writing "1" to the SBI0CR2<MST, TRX, BB, PIN>. It is necessary to set transmitted data to the data buffer register (SBI0DBR) and set "1" to <ACK> beforehand.

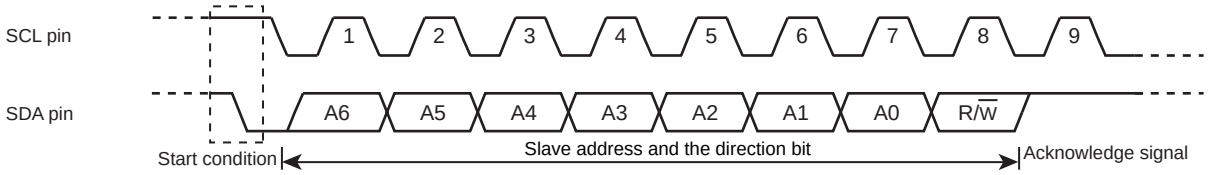


Figure 8-5 Start Condition Generation and Slave Address Generation

When the <BB> is "1", a sequence of generating a stop condition is started on the bus by writing "1" to the <MST, TRX, PIN>, and "0" to the <BB>. Do not modify the contents of <MST, TRX, BB, PIN> until a stop condition is generated on the bus.

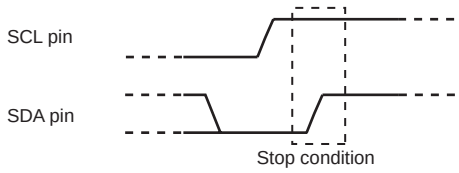


Figure 8-6 Stop Condition Generation

The state of the bus can be ascertained by reading the contents of SBI0SR<BB>. SBI0SR<BB> will be set to "1" if a start condition has been detected on the bus, and will be cleared to "0" if a stop condition has been detected.

8.3.3.8 Interrupt service requests and interrupt cancellation

When a serial bus interface interrupt request (INTSBI) occurs, the SBI0CR2<PIN> is cleared to “0”. During the time that the SBI0CR2<PIN> is “0”, the SCL line is pulled down to the low level.

The <PIN> is cleared to “0” when an 1 word of data is transmitted or received. Either writing/reading data to/from SBI0DBR sets the <PIN> to “1”.

The time from the <PIN> being set to “1” until the SCL line is released takes t_{LOW}.

In the address recognition mode (<ALS> = “0”), <PIN> is cleared to “0” when the received slave address is the same as the value set at the I2C0AR or when a GENERAL CALL is received (All 8-bit data are “0” after a start condition). Although SBI0CR2<PIN> can be set to “1” by the program, the <PIN> is not cleared to “0” when it is written “0”.

8.3.3.9 Serial bus interface operation mode selection

SBI0CR2<SBIM1:0> is used to specify the serial bus interface operation mode.

Set SBI0CR2<SBIM1:0> to “10” when the device is to be used in I^2C bus mode after confirming pin condition of serial bus interface to “H”.

Switch to port mode after confirming a bus is free.

8.3.3.10 Arbitration lost detection monitor

Since more than one master device can exist simultaneously on the bus in I^2C bus mode, a bus arbitration procedure has been implemented in order to guarantee the integrity of transferred data.

Data on the SDA line is used for I^2C bus arbitration.

The following shows an example of a bus arbitration procedure when two master devices exist simultaneously on the bus. Master A and master B output the same data until point “a”. After master A outputs “L” and master B, “H”, the SDA line of the bus is wired-AND and the SDA line is pulled down to the low level by master A. When the SCL line of the bus is pulled up at point b, the slave device reads the data on the SDA line, that is, data in master A. A data transmitted from master B becomes invalid. The state in master B is called “ARBITRATION LOST”. Master B device which loses arbitration releases the internal SDA output in order not to affect data transmitted from other masters with arbitration. When more than one master sends the same data at the first word, arbitration occurs continuously after the second word.

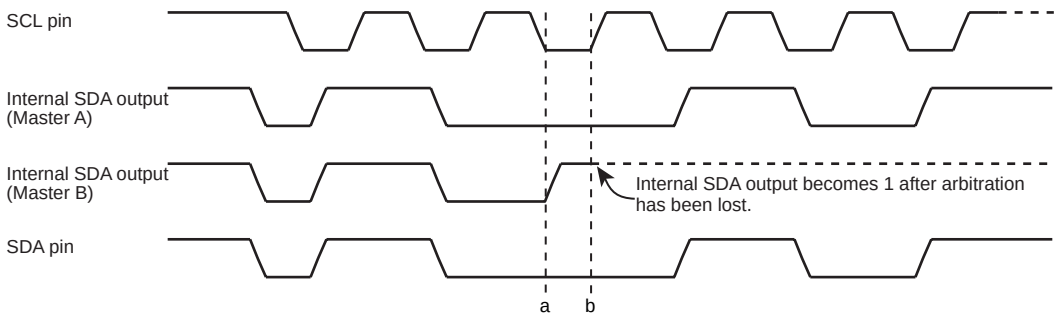


Figure 8-7 Arbitration Lost

The TMP91FU62 compares the levels on the bus’s SDA line with those of the internal SDA output on the rising edge of the SCL line. If the levels do not match, arbitration is lost and SBI0SR<AL> is set to “1”.

When SBI0SR<AL> is set to “1”, SBI0SR<MST, TRX> are cleared to “00” and the mode is switched to slave receiver mode. Thus, clock output is stopped in data transfer after setting <AL> = “1”.

SBI0SR<AL> is cleared to “0” when data is written to or read from SBI0DBR or when data is written to SBI0CR2.

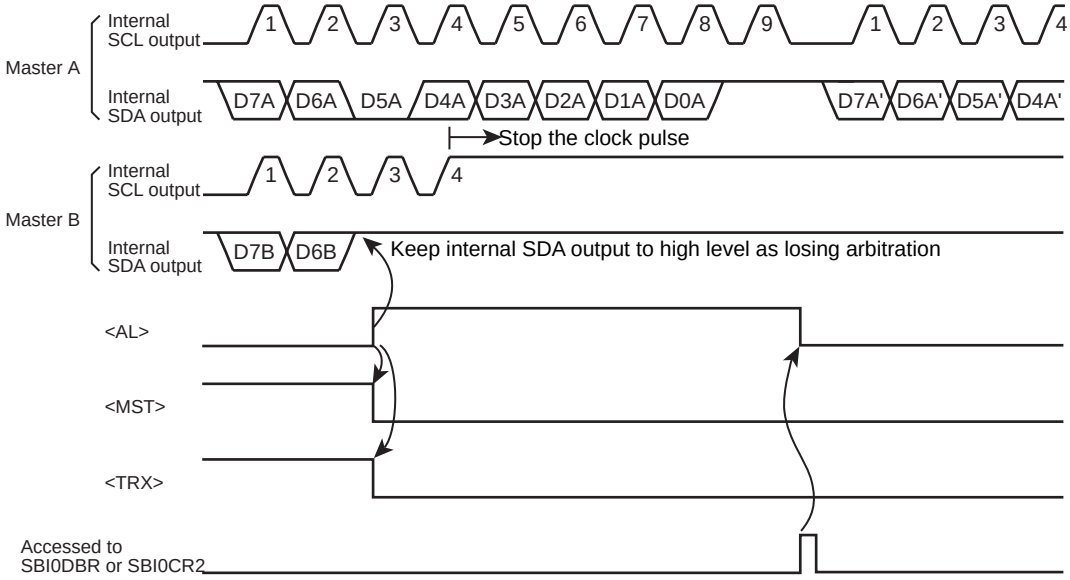


Figure 8-8 Example of when TMP91FU62 is a Master Device B (D7A = D7B, D6A = D6B)

8.3.3.11 Slave address match detection monitor

SBI0SR<AAS> is set to “1” in slave mode, in address recognition mode (e.g., when I2C0AR<ALS> = “0”), when a GENERAL CALL is received, or when a slave address matches the value set in I2C0AR. When I2C0AR<ALS> = “1”, SBI0SR<AAS> is set to “1” after the first word of data has been received. SBI0SR<AAS> is cleared to “0” when data is written to or read from the data buffer register SBI0DBR.

8.3.3.12 GENERAL CALL detection monitor

SBI0SR<AD0> is set to “1” in slave mode, when a GENERAL CALL is received (All 8-bit received data is “0” after a start condition). SBI0SR<AD0> is cleared to “0” when a start condition or stop condition is detected on the bus.

8.3.3.13 Last received bit monitor

The SDA line value stored at the rising edge of the SCL line is set to the SBI0SR<LRB>. In the acknowledge mode, immediately after an INTSBI interrupt request is generated, an acknowledge signal is read by reading the contents of the SBI0SR<LRB>.

8.3.3.14 Software reset function

The software reset function is used to initialize the SBI circuit, when SBI is locked by external noises, etc.

An internal reset signal pulse can be generated by setting SBI0CR2<SWRST1:0> to “10” and “01”. This initializes the SBI circuit internally. All control registers and status registers are initialized as well.

SBI0CR1<SWRMON> is automatically set to “1” after the SBI circuit has been initialized.

Note: If the software reset is executed, operation selection is reset, and its mode is set to port mode from I^2C mode.

8.3.3.15 Serial bus interface data buffer register (SBI0DBR)

The received data can be read and transferred data can be written by reading or writing the SBI0DBR.

In the master mode, after the start condition is generated the slave address and the direction bit are set in this register.

8.3.3.16 I^2CBUS address register (I2C0AR)

I2C0AR<SA6:0> is used to set the slave address when the TMP91FU62 functions as a slave device.

The slave address output from the master device is recognized by setting the I2C0AR<ALS> to "0". The data format is the addressing format. When the slave address is not recognized at the <ALS> = "1", the data format is the free data format.

8.3.3.17 Setting register for IDLE2 mode operation (SBI0BR0)

SBI0BR0<I2SBI0> is the register setting operation/stop during IDLE2 mode. Therefore, setting <I2SBI0> is necessary before the HALT instruction is executed.

8.3.4 Data Transfer in I^2C Bus Mode

8.3.4.1 Device initialization

Set the SBI0CR1<ACK, SCK2:0>, clear bits 2 to 0 and 4 in the SBI0CR1 to "0".

Set a slave address <SA6:0> and the <ALS> (<ALS> = "0" when an addressing format) to the I2C0AR.

For specifying the default setting to a slave receiver mode, clear "0" to the SBI0CR2<MST, TRX, BB>, set "1" to the <PIN>, "10" to the <SBIM1:0>, and write "0" to bit 1, 0.

		7	6	5	4	3	2	1	0	
SBI0CR1	←	X	X	X	0	X	0	0	0	Set acknowledge and SCL clock.
I2C0AR	←	X	X	X	X	X	X	X	0	Set slave address and address recognition mode.
SBI0CR2	←	0	0	0	1	1	0	0	0	Set to slave receiver mode.

Note: X: Don't care

8.3.4.2 Start condition and slave address generation

(1) Master mode

In the master mode, the start condition and the slave address are generated as follows.

Check a bus free status (when <BB> = “0”).

Set the SBI0CR1<ACK> to “1” (Acknowledge mode) and specify a slave address and a direction bit to be transmitted to the SBI0DBR.

When SBI0CR2<BB> = “0”, the start condition are generated by writing “1” to SBI0CR2<MST, TRX, BB, PIN>. Subsequently to the start condition, nine clocks are output from the SCL pin. While eight clocks are output, the slave address and the direction bit which are set to the SBI0DBR. At the 9th clock, the SDA line is released and the acknowledge signal is received from the slave device.

An INTSBI0 interrupt request occurs at the falling edge of the 9th clock. The <PIN> is cleared to “0”. In the master mode, the SCL pin is pulled down to the low level while <PIN> is “0”. When an interrupt request occurs, the <TRX> is changed according to the direction bit only when an acknowledge signal is returned from the slave device.

Setting in main routine

		7	6	5	4	3	2	1	0	
Reg	←	SBI0SR								
Reg	←	Reg. 0X20								
if Reg	≠0x00									Wait until bus is free.
Then										
SBI0CR1	←	X	X	X	1	X	0	0	0	Set to acknowledgement mode.
SBI0DBR	←	X	X	X	X	X	X	X	X	Set slave address and direction bit.
SBI0CR2	←	1	1	1	1	1	0	0	0	Generate start condition.

In INTSBI0 interrupt routine

INTCLR <-- 0x30 ; Clear the interrupt request

Process

End of interrupt

(2) Slave mode

In the slave mode, the start condition and the slave address are received.

After the start condition is received from the master device, while eight clocks are output from the SCL pin, the slave address and the direction bit which are output from the master device are received.

When a GENERAL CALL or the same address as the slave address set in I2C0AR is received, the SDA line is pulled down to the low level at the 9th clock, and the acknowledge signal is output.

An INTSBI0 interrupt request occurs on the falling edge of the 9th clock. The <PIN> is cleared to “0”. In slave mode the SCL line is pulled down to the low level while the <PIN> = “0”.

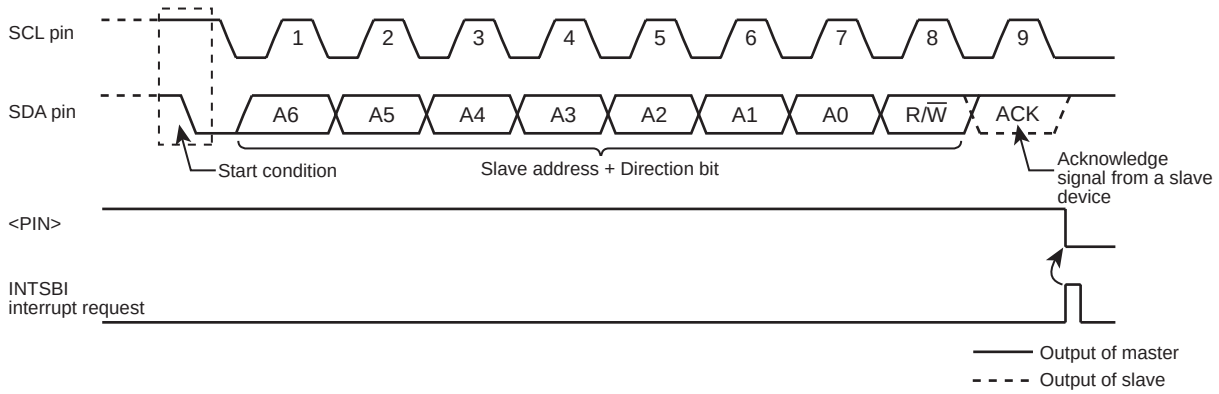


Figure 8-9 Start Condition Generation and Slave Address Transfer

8.3.4.3 1-word data transfer

Check the <MST> by the INTSBI0 interrupt process after the 1-word data transfer is completed, and determine whether the mode is a master or slave.

(1) If <MST> = “1” (Master mode)

Check the <TRX> and determine whether the mode is a transmitter or receiver.

(a) When the <TRX> = “1” (Transmitter mode)

Check the <LRB>. When <LRB> is “1”, a receiver does not request data. Implement the process to generate a stop condition (Refer to below) and terminate data transfer.

When the <LRB> is “0”, the receiver requests new data. When the next transmitted data is 8 bits, write the transmitted data to SBI0DBR. When the next transmitted data is other than 8 bits, set the <BC2:0> <ACK> and write the transmitted data to SBI0DBR. After written the data, <PIN> becomes “1”, a serial clock pulse is generated for transferring a new 1 word of data from the SCL pin, and then the 1-word data is transmitted. After the data is transmitted, an INTSBI interrupt request occurs. The <PIN> becomes “0” and the SCL line is pulled down to the low level. If the data to be transferred is more than one word in length, repeat the procedure from the <LRB> checking above.

```
if MST = 0
Then shift to the process when slave mode
if TRX = 0
Then shift to the process when receiver mode.
if LRB = 0
Then shift to the process that generates stop condition.
                  7  6  5  4  3  2  1  0
SBI0CR1     ←     0  0  0  1  X  X  X  X    Set the bit number of transmit and ACK.
SBI0DBR     ←     X  X  X  X  X  X  X  X    Write the transmit data.
End of interrupt
Note: X: Don’t care
```

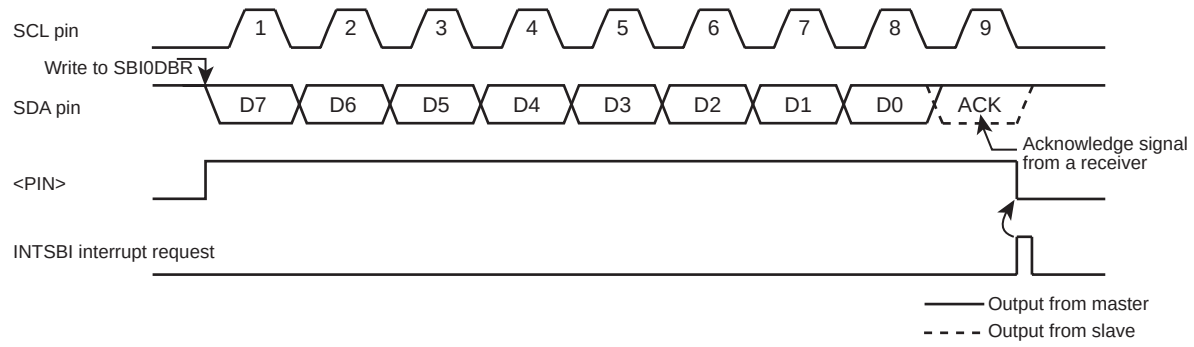


Figure 8-10 Example in which <BC2:0> = “000” and <ACK> = “1” in Transmitter Mode

(b) When the <TRX> is “0” (Receiver mode)

When the next transmitted data is other than 8 bits, set <BC2:0> <ACK> and read the received data from SBI0DBR to release the SCL line (Data which is read immediately after a slave address is sent is undefined). After the data is read, <PIN> becomes “1”. Serial clock pulse for transferring new 1 word of data is defined SCL and outputs “L” level from SDA pin with acknowledge timing.

An INTSBI0 interrupt request then occurs and the <PIN> becomes “0”, then the TMP91FU62 pulls down the SCL pin to the low level. The TMP91FU62 outputs a clock pulse for 1 word of data transfer and the acknowledge signal each time that received data is read from the SBI0DBR.

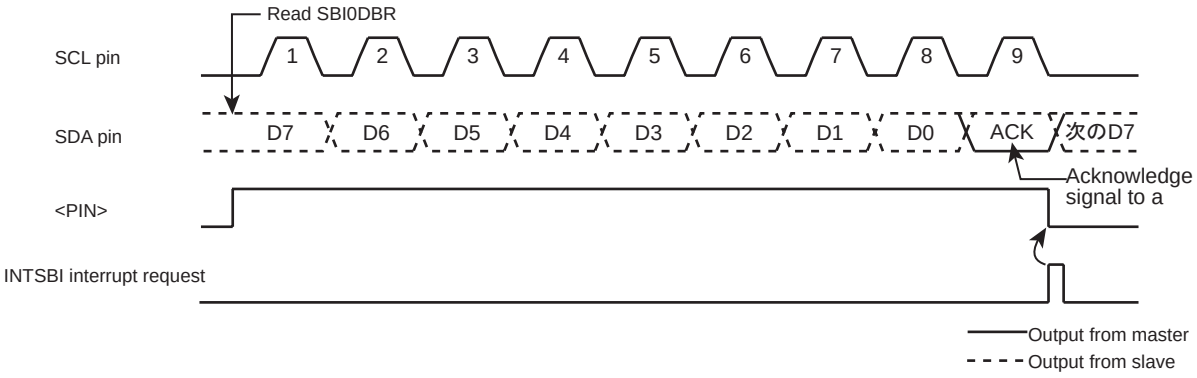


Figure 8-11 Example of when <BC2:0> = “000”, <ACK> = “1” in Receiver Mode

In order to terminate the transmission of data to a transmitter, clear <ACK> to “0” before reading data which is 1 word before the last data to be received. The last data word does not generate a clock pulse as the acknowledge signal. After the data has been transmitted and an interrupt request has been generated, set <BC2:0> to “001” and read the data. The TMP91FU62 generates a clock pulse for an 1-bit data transfer. Since the master device is a receiver, the SDA line on the bus remains high. The transmitter interprets the high signal as an ACK signal. The receiver indicates to the transmitter that data transfer is complete.

After the one data bit has been received and an interrupt request been generated, the TMP91FU62 generates a stop condition and terminates data transfer.

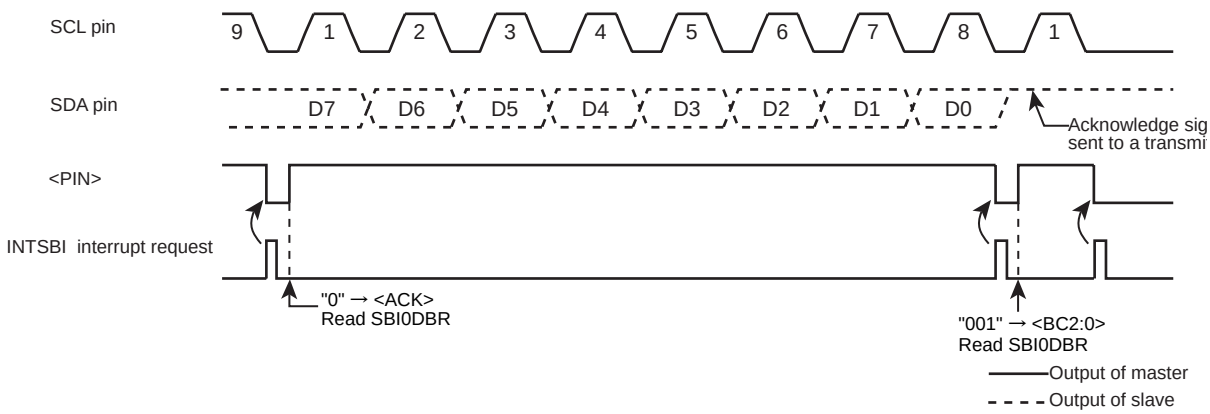


Figure 8-12 Termination of Data Transfer in Master Receiver Mode

Example: In case receive data N times

INTSBI0 interrupt (After transmitting data)

		7	6	5	4	3	2	1	0	
SBI0CR1	←	X	X	X	X	X	X	X	X	Set the bit number of receive data and ACK.
Reg.	←	SBI0DBR								Load the dummy data.

End of interrupt

INTSBI0 interrupt (Receive data of 1st to (N 2) th)

		7	6	5	4	3	2	1	0	
Reg.	←	SBI0DBR								Load the data of 1st to (N 2)th.

End of interrupt

INTSBI0 interrupt ((N 1) th Receive data)

		7	6	5	4	3	2	1	0	
SBI0CR1	←	X	X	X	0	0	X	X	X	Not generate acknowledge signal
Reg.	←	SBI0DBR								Load the data of (N 1)th

End of interrupt

INTSBI0 interrupt (Nth Receive data)

		7	6	5	4	3	2	1	0	
SBI0CR1	←	0	0	1	0	0	X	X	X	Generate the clock for 1bit transmit
Reg.	←	SBI0DBR								Receive the data of Nth.

End of interrupt

INTSBI0 interrupt (After receiving data)

The process of generating stop condition — Finish the transmit of data

End of interrupt

Note: X: Don't care

(2) If <MST> = 0 (Slave mode)

In the slave mode the TMP91FU62 operates either in normal slave mode or in slave mode after losing arbitration.

In the slave mode, an INTSBI0 interrupt request occurs when the TMP91FU62 receives a slave address or a GENERAL CALL from the master device, or when a GENERAL CALL is received and data transfer is complete, or after matching received address. In the master mode, the TMP91FU62 operates in a slave mode if it detects losing arbitration. An INTSBI0 interrupt request occurs when a word data transfer terminates after losing arbitration. When an INTSBI0 interrupt request occurs the <PIN> is cleared to "0" and the SCL pin is pulled down to the low level. Either reading/writing from/to the SBI0DBR or setting the <PIN> to "1" will release the SCL pin after taking t_{LOW} time.

Check the SBI0SR<AL>, <TRX>, <AAS>, and <AD0> and implements processes according to conditions listed in the next table.

Example: In case matching slave address in slave receive mode, direction bit is "1".

INTSBI0 interrupt

if TRX = 0

Then shift to other process

if AL = 1

Then shift to other process

if AAS = 0

Then shift to other process

		7	6	5	4	3	2	1	0	
SBI0CR1	←	X	X	X	1	X	X	X	X	Set the bit number of transmit.
SBI0DBR	←	X	X	X	X	X	X	X	X	Set the data of transmit.

Note: X: Don't care

Table 8-1 Operation in the Slave Mode

<TRX>	<AL>	<AAS>	<AD0>	Conditions	Process
1	1	1	0	The TMP91FU62 loses arbitration when transmitting a slave address and receives a slave address for which the value of the direction bit sent from another master is "1".	Set the number of bits a word in <BC2:0> and write the transmitted data to SBI0DBR.
	0	1	0	In slave receiver mode, the TMP91FU62 receives a slave address for which the value of the direction bit sent from the master is "1".	
		0	0	In slave transmitter mode, a single word of data is transmitted.	Check the <LRB> setting. If <LRB> is set to "1", set <PIN> to "1" since the receiver win no request the data which follows. Then, clear <TRX> to "0" to release the bus. If <LRB> is cleared to "0", set <BC2:0> to the number of bits in a word and write the transmitted data to SBI0DBR since the receiver requests next data.
0	1	1	1/0	The TMP91FU62 loses arbitration when transmitting a slave address and receives a slave address or GENERAL CALL for which the value of the direction bit sent from another master is "0".	Read the SBI0DBR for setting the <PIN> to "1" (Reading dummy data) or set the <PIN> to "1".
		0	0	The TMP91FU62 loses arbitration when transmitting a slave address or data and terminates word data transfer.	
	0	1	1/0	In slave receiver mode, the TMP91FU62 receives a slave address or GENERAL CALL for which the value of the direction bit sent from the master is "0".	
		0	1/0	In slave receiver mode, the TMP91FU62 terminates receiving word data.	Set <BC2:0> to the number of bits in a word and read the received data from SBI0DBR.

8.3.4.4 Stop condition generation

When SBI0SR<BB> = “1”, the sequence for generating a stop condition is started by writing “1” to SBI0CR2<MST, TRX, PIN> and “0” to SBI0CR2<BB>. Do not modify the contents of SBI0CR2<MST, TRX, PIN, BB> until a stop condition has been generated on the bus. When the bus’s SCL line has been pulled low by another device, the TMP91FU62 generates a stop condition when the other device has released the SCL line and SDA pin rising.

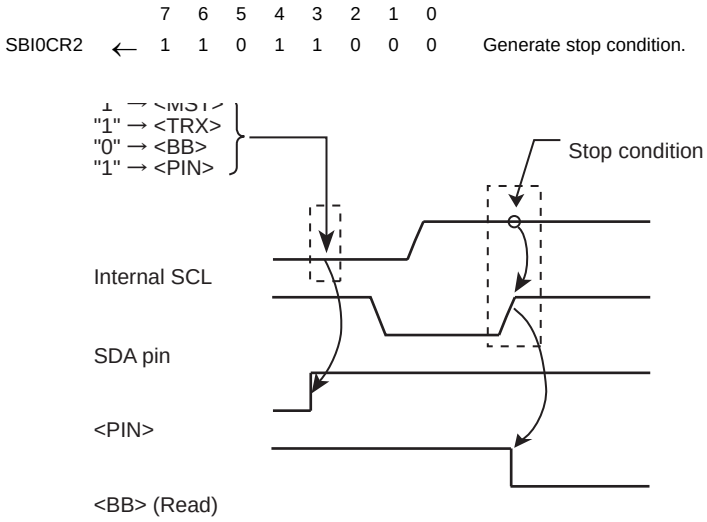


Figure 8-13 Stop Condition Generation (Single master)

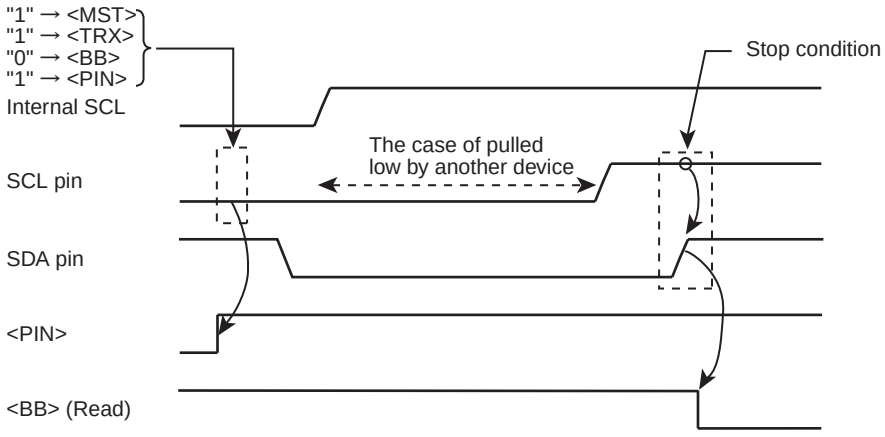


Figure 8-14 Condition Generation (Multi master)

8.3.4.5 Restart

Restart is used during data transfer between a master device and a slave to change the data transfer direction.

The following description explains how to restart when the TMP91FU62 is in Master mode.

Clear SBI0CR2<MST, TRX, BB> to “0” and set SBI0CR2<PIN> to “1” to release the bus. The SDA line remains High and the SCL pin is released. Since a stop condition has not been generated on the bus, other devices assume the bus to be in busy state.

And confirm SCL pin, that SCL pin is released and become bus-free state by SBI0SR<BB> = “0” or signal level “1” of SCL pin by sensing its port (change to input mode). Check the <LRB> until it becomes “1” to check that the SCL line on a bus is not pulled down to the low level by other devices. After confirming that the bus remains in a free state, generate a start condition using the procedure described in 8.3.4.2.

In order to satisfy the setup time requirements when restarting, take at least 4.7 μs of waiting time by software from the time of restarting to confirm that the bus is free until the time to generate the start condition.

		7	6	5	4	3	2	1	0	
SBI0CR2	←	0	0	0	1	1	0	0	0	Release the bus
if SBI0SR<BB> ≠ 0										Check if SCL pin is released.
Then										
if SBI0SR<LRB> ≠ 1										Check if SCL pin of other device is "L" level.
Then										
4.7us Wait										
SBI0CR1	←	0	0	0	1	0	X	X	X	Set acknowledgement mode.
SBI0DBR	←	X	X	X	X	X	X	X	X	Set the slave address and direction bit.
SBI0CR2	←	1	1	1	1	1	0	0	0	Generate start condition.

Note: X: Don’t care

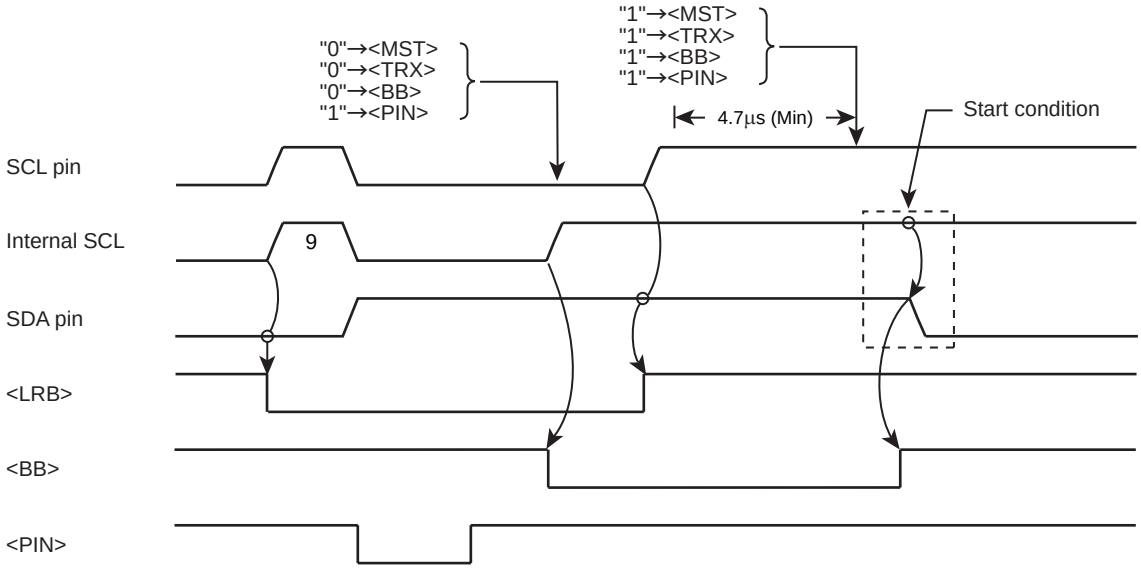


Figure 8-15 Timing Diagram for TMP91FU62 Restart

Note: Don't write <MST> "0", when <MST> "0" condition. (Cannot be restarted)

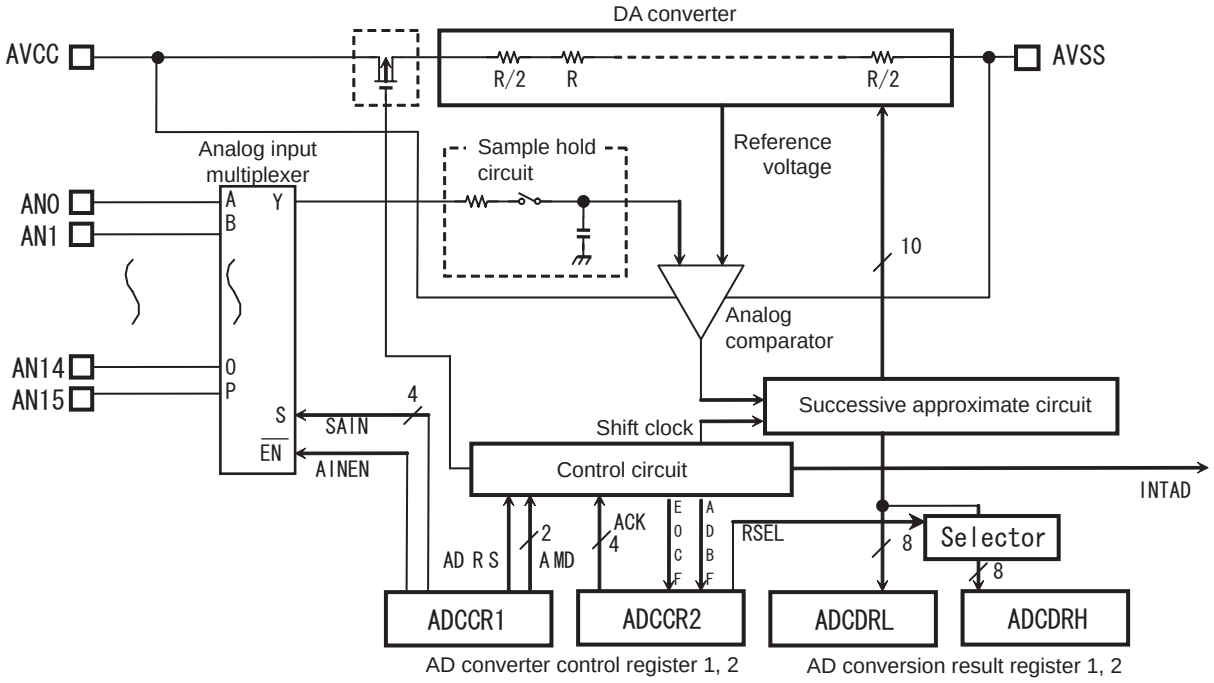
9. 10-bit AD Converter (ADC)

The TMP91FU62 have a 10-bit successive approximation type AD converter.

9.1 Configuration

The circuit configuration of the 10-bit AD converter is shown in Figure 9-1.

It consists of control register ADCCR1 and ADCCR2, converted value register ADCDRH and ADCDRL, a DA converter, a sample-hold circuit, a comparator, and a successive comparison circuit.



Note: Before using AD converter, set appropriate value to I/O port register combining a analog input port. For details, see the section on "I/O ports".

Figure 9-1 10-bit AD Converter

9.2 Register configuration

The AD converter consists of the following four registers:

1. AD converter control register 1 (ADCCR1)

   This register selects the analog channels and operation mode (single or repeat) in which to perform AD conversion and controls the AD converter as it starts operating.

2. AD converter control register 2 (ADCCR2)

   This register selects the AD conversion time and controls the connection of the DA converter (Ladder resistor network) and monitors the operating status of the AD converter.

3. AD converted value register (ADCDRH, ADCDRL)

   This register used to store the digital value after being converted by the AD converter.

AD Converter Control Register 1

ADCCR1 (02B0H)

	7	6	5	4	3	2	1	0
Bit symbol	ADRS	AMD		AINEN	SAIN			
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	AD conversion start 0: - 1: AD conversion start	AD operating mode 00: AD operation disable 01: single mode 10: Reserved 11: Repeat mode		Analog input control 0:disable 1:enable	Analog input channel select 0000: AN0 0001: AN1 0010: AN2 0011: AN3	 0100: AN4 0101: AN5 0110: AN6 0111: AN7	 1000: AN8 1001: AN9 1010: AN10 1011: AN11	 1100: AN12 1101: AN13 1110: AN14 1111: AN15

Note 1: Select analog input channel during AD converter stops (ADCCR2<ADBF> = “0”).

Note 2: When the analog input channel is all use disabling, the ADCCR1<AINEN> should be set to “0”.

Note 3: During conversion, Do not perform port output instruction to maintain a precision for all of the pins because analog input port use as general input port. And for port near to analog input, Do not input intense signaling of change.

Note 4: The ADCCR1<ADRS> is automatically cleared to “0” after starting conversion.

Note 5: Do not set ADCCR1<ADRS> newly again during AD conversion. Before setting ADCCR1<ADRS> newly again, check ADCDR2<EOCF> to see that the conversion is completed or wait until the interrupt signal (INTADC) is generated (e.g., interrupt handling routine).

Note 6: Starting of STOP mode, SLOW mode, and the IDLE1 mode initializes the AD control register 1 (ADCCR1) except for SAIN. Moreover, in the case of the IDLE2 mode, it controls by the <I2AD> bit of ADCCR2. Therefore, to use AD converter again, set the ADCCR1 newly after returning to NORMAL mode.

AD Converter Control Register 2 (Read-modify-write instructions are prohibited.)

ADCCR2 (02B1H)

	7	6	5	4	3	2	1	0
Bit symbol	EOCF	ADBF	RSEL	I2AD	ACK			
Read/Write	R		R/W					
After reset	0	0	0	0	1	1	0	0
Function	AD conversion end flag 0:Before or during conversion 1: Conversion completed	AD conversion BUSY flag 0: During stop of AD conversion 1: During AD conversion	Storing of an AD conversion result 0: 10bit mode 1: 8bit mode	IDLE2 control 0:Stop 1:Operation	AD conversion time select See" Table 9-1 ACK setting and Conversion time "			

Note 1: Starting of STOP mode, SLOW mode, and the IDLE1 mode initializes the AD control register 2 (ADCCR2) except for ACK and I2AD. Moreover, in the case of the IDLE2 mode, it controls by the <I2AD> bit of ADCCR2. Therefore, to use AD converter again, set the ADCCR2 newly after returning to NORMAL mode. Therefore, the AD conversion result should be read to ADCDRL more first than ADCDRH.

Note 2: The ADCCR2<EOCF> is cleared to “0” when reading the ADCDRH.

Note 3: The ADCCR2<ADBF> is set to “1” when AD conversion starts, and cleared to “0” when AD conversion finished.

Table 9-1 ACK setting and Conversion time

Condition / ACK	Conversion time	20MHz	16MHz	10 MHz	8MHz	4 MHz
0xxx	Reserved					
1000	Reserved					
1001	Reserved					
1010	78/fc	–	–	–	–	19.5 μs
1011	156/fc	–	–	15.6 μs	19.5 μs	39.0 μs
1100	312/fc	15.6 μs	19.5 μs	31.2 μs	39.0 μs	78.0 μs
1101	624/fc	31.2 μs	39.0 μs	62.4 μs	78.0 μs	156.0 μs
1110	1248/fc	62.4 μs	78.0 μs	124.8 μs	156.0 μs	–
1111	Reserved					

Note 1: Setting for "-" in the above table are inhibited. fc: High Frequency oscillation clock [Hz]

Note 2: Set conversion time setting should be kept more than the following time by Analog reference voltage.

– AVCC = 4.5 to 5.5 V 15.6 us and more

AD Converted value Register H (8-bit storing mode)

		7	6	5	4	3	2	1	0
ADCDRH (02B3H)	Bit symbol	AD09	AD08	AD07	AD06	AD05	AD04	AD03	AD02
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0

AD Converted value Register H (10-bit storing mode)

		7	6	5	4	3	2	1	0
ADCDRH (02B3H)	Bit symbol	–	–	–	–	–	–	AD09	AD08
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0

AD Converted value Register L

		7	6	5	4	3	2	1	0
ADCDRL (02B2H)	Bit symbol	AD07	AD06	AD05	AD04	AD03	AD02	AD01	AD00
	Read/Write	R							
	After reset	0	0	0	0	0	0	0	0

Note: At the time of 10-bit storing mode, if the bit 7 to 2 of ADCDRH is read, "0" will be read.

9.3 Function

9.3.1 Single mode

After setting ADCCR1<AMD> to “01” (single mode), set ADCCR1<ADRS> to “1”. AD conversion of the voltage at the analog input pin specified by ADCCR1<SAIN> is thereby started.

After completion of the AD conversion, the conversion result is stored in AD converted value registers (ADCDRH, ADCDRL) and at the same time ADCCR2<EOCF> is set to 1, the AD conversion finished interrupt (INTADC) is generated.

ADCCR1<ADRS> is automatically cleared after AD conversion has started. Do not set ADCCR1<ADRS> newly again (Restart) during AD conversion. Before setting ADCCR1<ADRS> newly again, check ADCCR2<EOCF> to see that the conversion is completed or wait until the interrupt signal (INTADC) is generated (e.g., interrupt handling routine).

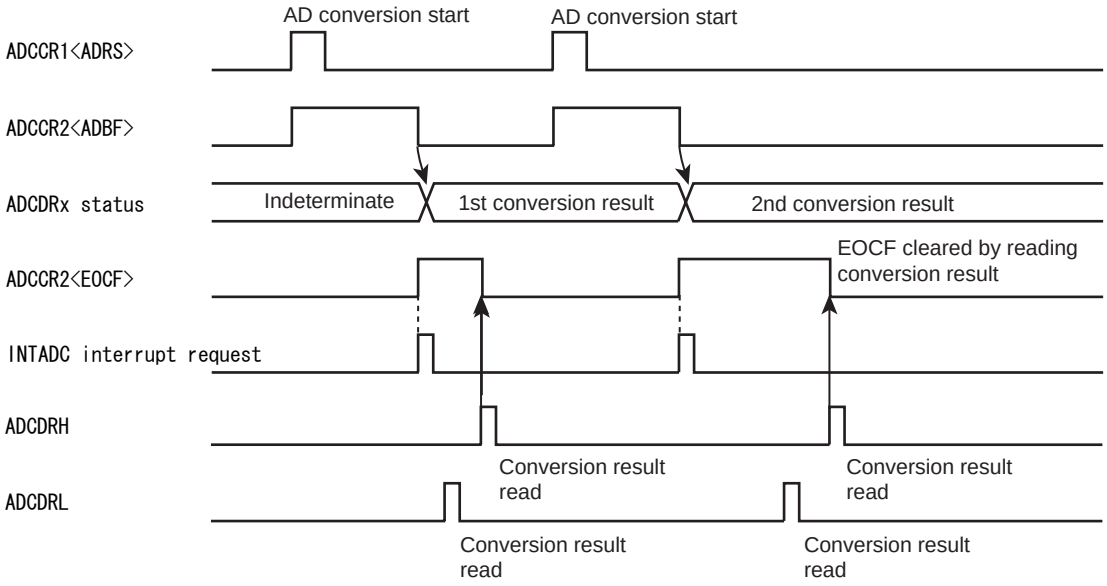


Figure 9-2 Single mode

9.3.2 Repeat Mode

AD conversion of the voltage at the analog input pin specified by ADCCR1<SAIN> is performed repeatedly. In this mode, AD conversion is started by setting ADCCR1<ADRS> to “1” after setting ADCCR1<AMD> to “11” (Repeat mode).

After completion of the AD conversion, the conversion result is stored in AD converted value registers (ADCDRL, ADCDRH) and at the same time ADCCR2<EOCF> is set to 1, the AD conversion finished interrupt (INTADC) is generated.

In repeat mode, each time one AD conversion is completed, the next AD conversion is started. To stop AD conversion, set ADCCR1<AMD> to “00” (Disable mode) by writing 0s. The AD convert operation is stopped immediately. The converted value at this time is not stored in the AD converted value register.

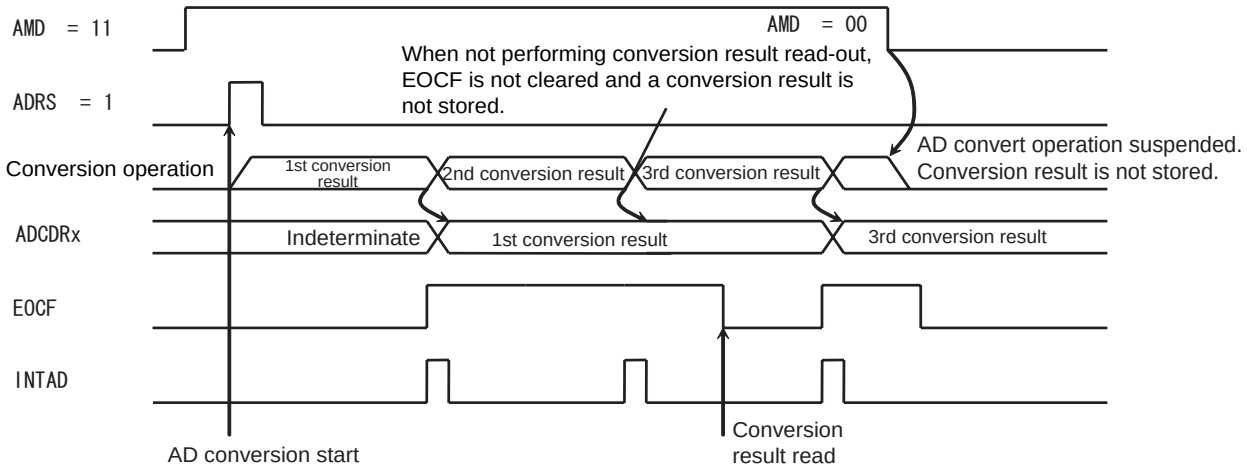


Figure 9-3 Repeat Mode

9.3.3 Register Setting

1. Set up the AD converter control register 1 (ADCCR1) as follows:
   - Choose the channel to AD convert using AD input channel select (SAIN).
   - Specify analog input enable for analog input control (AINDS).
   - Specify AMD for the AD converter control operation mode (ssingle or repeat mode).
2. Set up the AD converter control register 2 (ADCCR2) as follows:

   Set the AD conversion time using AD conversion time (ACK). For details on how to set the conversion time, refer to Table 9-1 and AD converter control register 2.
3. After setting up (1) and (2) above, set AD conversion start (ADRS) of AD converter control register 1 (ADCCR1) to “1”. If software start mode has been selected, AD conversion starts immediately.
4. After an elapse of the specified AD conversion time, the AD converted value is stored in AD converted value register (ADCDRH and ADCDRL) and the AD conversion finished flag (EOCF) of AD converter control register 2 (ADCCR2) is set to “1”, upon which time AD conversion interrupt INTADC is generated.
5. EOCF is cleared to “0” by a read of the conversion result. However, if reconverted before a register read, although EOCF is cleared the previous conversion result is retained until the next conversion is completed.

Example :After selecting the conversion time 19.5 μs at 16 MHz and the analog input channel AIN3 pin, perform AD conversion once. After checking EOCF, read the converted value, store the lower 2 bits in address 0009EH nd store the upper 8 bits in address 0009FH in RAM. The operation mode is single mode.

```
        LD      (ADCCR1) ,  00110011B       ; Select AIN3
        LD      (ADCCR2) ,  00001100B       ;Select conversion time(312/fc) and operation mode

        SET     (ADCCR1) . 7                ; ADRS = 1(AD conversion start)
SLOOP : TEST    (ADCCR2) . 7                ; EOCF= 1 ?
        JRS     T, SLOOP

        LD      A , (ADCDRL)                ; Read result data
        LD      (9EH) , A
        LD      A , (ADCDRH)                ; Read  result data
        LD      (9FH), A
```

9.4 IDLE1/STOP/SLOW Modes during AD Conversion

When standby mode (IDLE1,STOP or SLOW mode) is entered forcibly during AD conversion, the AD convert operation is suspended and the AD converter is initialized (ADCCR1 and ADCCR2 are initialized to initial value). Also, the conversion result is indeterminate. (Conversion results up to the previous operation are cleared, so be sure to read the conversion results before entering standby mode (IDLE1,STOP or SLOW mode).) When restored from standby mode (IDLE1,STOP or SLOW mode), AD conversion is not automatically restarted, so it is necessary to restart AD conversion. Note that since the analog reference voltage is automatically disconnected, there is no possibility of current flowing into the analog reference voltage.

Moreover, in the case of the IDLE2 mode, it controls by the <I2AD> bit of ADCCR2.

9.5 Analog Input Voltage and AD Conversion Result

The analog input voltage is corresponded to the 10-bit digital value converted by the AD as shown in Figure 9-4.

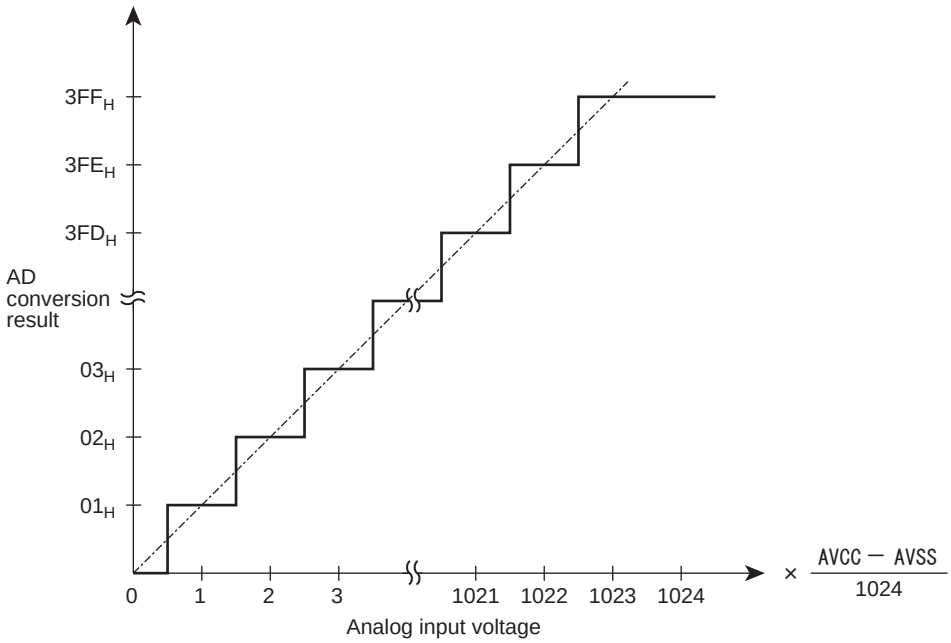


Figure 9-4 Analog Input Voltage and AD Conversion Result (Typ.)

9.6 Precautions about AD Converter

9.6.1 Analog input pin voltage range

Make sure the analog input pins (AN0 to AN15) are used at voltages within AVCC to AVSS. If any voltage outside this range is applied to one of the analog input pins, the converted value on that pin becomes uncertain. The other analog input pins also are affected by that.

9.6.2 Analog input shared pins

The analog input pins (AN0 to AN15) are shared with input/output ports. When using any of the analog inputs to execute AD conversion, do not execute input/output instructions for all other ports. This is necessary to prevent the accuracy of AD conversion from degrading. Not only these analog input shared pins, some other pins may also be affected by noise arising from input/output to and from adjacent pins.

9.6.3 Noise Countermeasure

The internal equivalent circuit of the analog input pins is shown in Figure 9-5. The higher the output impedance of the analog input source, more easily they are susceptible to noise. Therefore, make sure the output impedance of the signal source in your design is 5kΩ or less. Toshiba also recommends attaching a capacitor external to the chip.

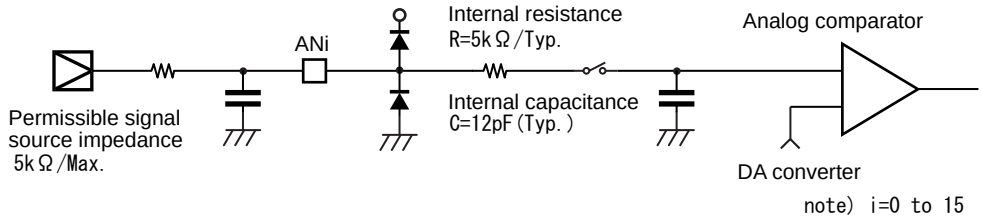


Figure 9-5 Analog Input Equivalent Circuit and Example of Input Pin Processing

10. Program Patch Logic

The TMP91FU62 has a program patch logic, which enables the user to fix the program code in the on-chip ROM without generating a new mask. Patch program must be read into on-chip RAM from external memory during the startup routine.

Up to six two-byte sequences, or banks (Twelve bytes in total) can be replaced with patch code. More significant code correction can be performed by replacing program code with single-byte instruction code which generates a software interrupt (SWI) to make a branch to a specified location in the on-chip RAM area.

The program patch logic only compares addresses in the on-chip ROM area; it cannot fix the program code in the on-chip peripheral, on-chip RAM and external ROM areas.

Each of six banks is independently programmable, and functionally equivalent. In the following sections, any references to bank0 also apply to other banks.

10.1 Block Diagram

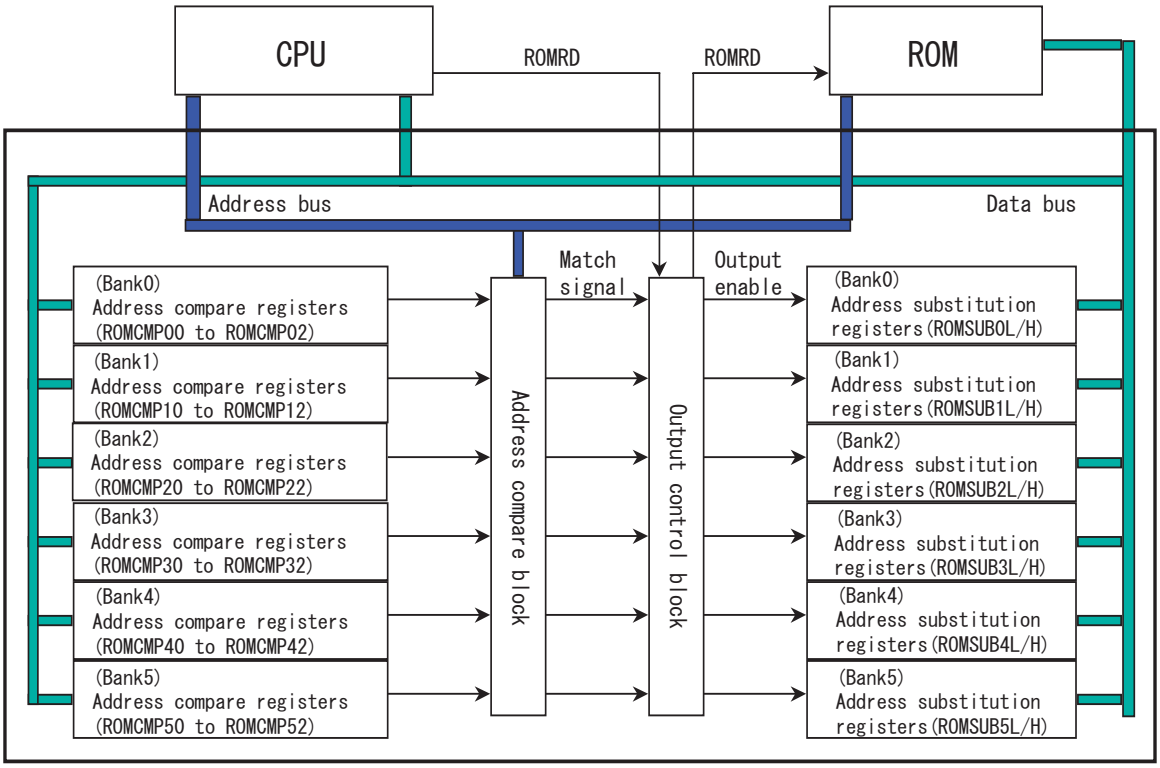


Figure 10-1 Program Patch Logic Diagram

10.2 SFR Descriptions

The program patch logic consists of six banks (0 to 5). Each bank is provided with three bytes of address compare registers (ROMCMPx0 to ROMCMPx2) and two bytes of address substitution registers (ROMSUBxL and ROMSUBxH).

Bank0 Address Compare Register 0

ROMCMP00 (0400H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	–
Read/Write	W							–
After reset	0	0	0	0	0	0	0	–
Function	Target ROM address (Lower 7 bits)							–

Bank0 Address Compare Register 1

ROMCMP01 (0401H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Middle 8 bits)							

Bank0 Address Compare Register 2

ROMCMP02 (0402H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Upper 8 bits)							

Bank0 Data Substitution Register L

ROMSUB0L (0404H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Lower 8 bits)							

Bank0 Data Substitution Register H

ROMSUB0H (0405H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Upper 8 bits)							

Note 1: The ROMCMP00/01/02, and ROMSUB0L/0H registers do not support read-modify-write operation.

Note 2: Bit0 of the Address Compare Register 0 is read as undefined.

Bank1 Address Compare Register 0

ROMCMP10 (0408H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	–
Read/Write	W							–
After reset	0	0	0	0	0	0	0	–
Function	Target ROM address (Lower 7 bits)							–

Bank1 Address Compare Register 1

ROMCMP11 (0409H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Middle 8 bits)							

Bank1 Address Compare Register 2

ROMCMP12 (040AH) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Upper 8 bits)							

Bank1 Data Substitution Register L

ROMSUB1L (040CH) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Lower 8 bits)							

Bank1 Data Substitution Register H

ROMSUB1H (040DH) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Upper 8 bits)							

Note 1: The ROMCMP10/11/12, and ROMSUB1L/1H registers do not support read-modify-write operation.

Note 2: Bit0 of the Address Compare Register 0 is read as undefined.

Bank2 Address Compare Register 0

ROMCMP20 (0410H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	–
Read/Write	W							–
After reset	0	0	0	0	0	0	0	–
Function	Target ROM address (Lower 7 bits)							–

Bank2 Address Compare Register 1

ROMCMP21 (0411H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Middle 8 bits)							

Bank2 Address Compare Register 2

ROMCMP22 (0412H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Upper 8 bits)							

Bank2 Data Substitution Register L

ROMSUB2L (0414H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Lower 8 bits)							

Bank2 Data Substitution Register H

ROMSUB2H (0415H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Upper 8 bits)							

Note 1: The ROMCMP20/21/22, and ROMSUB2L/2H registers do not support read-modify-write operation.

Note 2: Bit0 of the Address Compare Register 0 is read as undefined.

Bank3 Address Compare Register 0

ROMCMP30 (0418H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	–
Read/Write	W							–
After reset	0	0	0	0	0	0	0	–
Function	Target ROM address (Lower 7 bits)							–

Bank3 Address Compare Register 1

ROMCMP31 (0419H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Middle 8 bits)							

Bank3 Address Compare Register 2

ROMCMP32 (041AH) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Upper 8 bits)							

Bank3 Data Substitution Register L

ROMSUB3L (041CH) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Lower 8 bits)							

Bank3 Data Substitution Register H

ROMSUB3H (041DH) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Upper 8 bits)							

Note 1: The ROMCMP30/31/32, and ROMSUB3L/3H registers do not support read-modify-write operation.

Note 2: Bit0 of the Address Compare Register 0 is read as undefined.

Bank4 Address Compare Register 0

ROMCMP40 (0420H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	–
Read/Write	W							–
After reset	0	0	0	0	0	0	0	–
Function	Target ROM address (Lower 7 bits)							–

Bank4 Address Compare Register 1

ROMCMP41 (0421H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Middle 8 bits)							

Bank4 Address Compare Register 2

ROMCMP42 (0422H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Upper 8 bits)							

Bank4 Data Substitution Register L

ROMSUB4L (0424H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Lower 8 bits)							

Bank4 Data Substitution Register H

ROMSUB4H (0425H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Upper 8 bits)							

Note 1: The ROMCMP40/41/42, and ROMSUB4L/4H registers do not support read-modify-write operation.

Note 2: Bit0 of the Address Compare Register 0 is read as undefined.

Bank5 Address Compare Register 0

ROMCMP50 (0428H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	–
Read/Write	W							–
After reset	0	0	0	0	0	0	0	–
Function	Target ROM address (Lower 7 bits)							–

Bank5 Address Compare Register 1

ROMCMP51 (0429H) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Middle 8 bits)							

Bank5 Address Compare Register 2

ROMCMP52 (042AH) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Target ROM address (Upper 8 bits)							

Bank5 Data Substitution Register L

ROMSUB5L (042CH) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Lower 8 bits)							

Bank5 Data Substitution Register H

ROMSUB5H (042DH) RMW instructions are prohibited.

	7	6	5	4	3	2	1	0
Bit symbol	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Patch code (Upper 8 bits)							

Note 1: The ROMCMP50/51/52, and ROMSUB5L/5H registers do not support read-modify-write operation.

Note 2: Bit0 of the Address Compare Register 0 is read as undefined.

10.3 Operation

10.3.1 Replacing data

Two consecutive bytes of data can be replaced for each bank. A two-byte sequence to be replaced must start at an even address. If only a single byte at an even or odd address need be replaced, set the current masked ROM data in the other byte.

Correction procedure:

Load the address compare registers (ROMCMP00 to ROMCMP02) with the target address where ROM data need be replaced. Store 2-byte patch code in the ROMSUB0L and ROMSUB0H registers.

When the CPU address matches the value stored in the ROMCMP00 to ROMCMP02 registers, the program patch logic disables RD output to the masked ROM and drives out the code stored in the ROMSUB0L and ROMSUB0H to the internal bus. The CPU thus fetches the patch code.

The following shows some examples:

a. Replacing 00H at address FF1230H with AAH

	7	6	5	4	3	2	1	0	
ROMCMP00	0	0	1	1	0	0	0	0	Stores 30 in address compare register 0 for bank0.
ROMCMP01	0	0	0	1	0	0	1	0	Stores 12 in address compare register 1 for bank0.
ROMCMP02	1	1	1	1	1	1	1	1	Stores FF in address compare register 2 for bank0.
ROMSUB0L	1	0	1	0	1	0	1	0	Store AA in address substitution register low for bank0.
ROMSUB0H	0	0	0	1	0	0	0	1	Store 11 in address substitution register high for bank0.

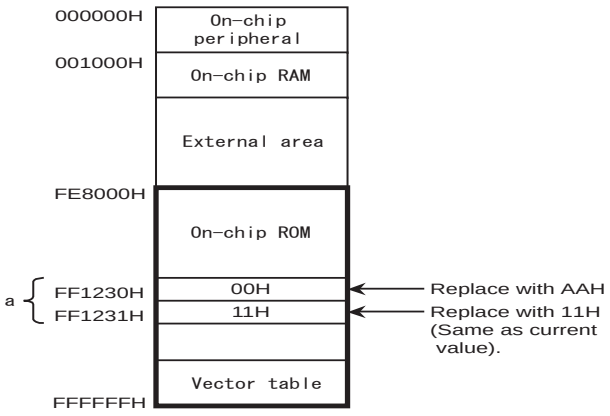


Figure 10-2 Example Patch Code Implementation

b. Replacing 33H at address FF1233H with BBH

	7	6	5	4	3	2	1	0	
ROMCMP00	0	0	1	1	0	0	1	0	Stores 32 in address compare register 0 for bank0.
ROMCMP01	0	0	0	1	0	0	1	0	Stores 12 in address compare register 1 for bank0.
ROMCMP02	1	1	1	1	1	1	1	1	Stores FF in address compare register 2 for bank0.
ROMSUB0L	0	0	1	0	0	0	1	0	Store 22 in address substitution register low for bank0.
ROMSUB0H	1	0	1	1	1	0	1	1	Store BB in address substitution register high for bank0.

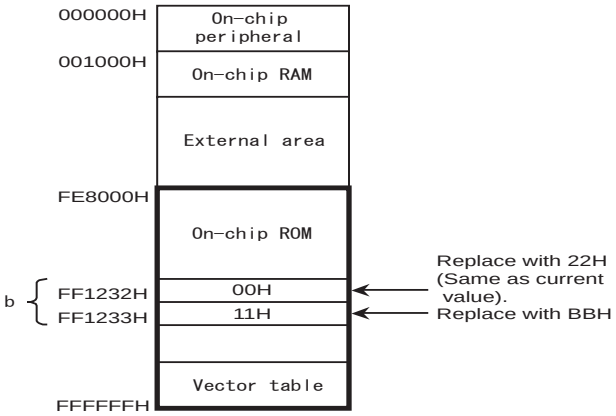


Figure 10-3 Example Patch Code Implementation

c. Replacing 44H at address FF1234H with CCH and 55H at address FF1235H with DDH

	7	6	5	4	3	2	1	0	
ROMCMP00	0	0	1	1	0	1	0	0	Stores 34 in address compare register 0 for bank0.
ROMCMP01	0	0	0	1	0	0	1	0	Stores 12 in address compare register 1 for bank0.
ROMCMP02	1	1	1	1	1	1	1	1	Stores FF in address compare register 2 for bank0.
ROMSUB0L	1	1	0	0	1	1	0	0	Store CC in address substitution register low for bank0.
ROMSUB0H	1	1	0	1	1	1	0	1	Store DD in address substitution register high for bank0.

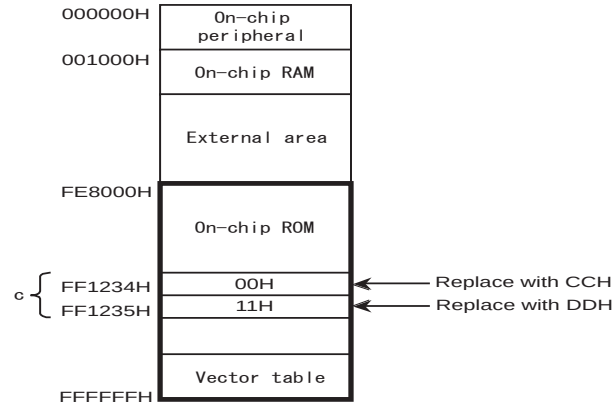


Figure 10-4 Example Patch Code Implementation

d. Replacing 77H at address FF1237H with EEH and 88H at address FF1238H with FFH (Requiring two banks)

	7	6	5	4	3	2	1	0	
ROMCMP00	0	0	1	1	0	1	1	0	Stores 36 in address compare register 0 for bank0.
ROMCMP01	0	0	0	1	0	0	1	0	Stores 12 in address compare register 1 for bank0.
ROMCMP02	1	1	1	1	1	1	1	1	Stores FF in address compare register 2 for bank0.
ROMSUB0L	0	1	1	0	0	1	1	0	Store 66 in address substitution register low for bank0.
ROMSUB0H	1	1	1	0	1	1	1	0	Store EE in address substitution register high for bank0.
ROMCMP10	0	0	1	1	1	0	0	0	Stores 38 in address compare register 0 for bank1.
ROMCMP11	0	0	0	1	0	0	1	0	Stores 12 in address compare register 1 for bank1.
ROMCMP12	1	1	1	1	1	1	1	1	Stores FF in address compare register 2 for bank1.
ROMSUB1L	1	1	1	1	1	1	1	1	Store FF in address substitution register low for bank1.
ROMSUB1H	1	0	0	1	1	0	0	1	Store 99 in address substitution register high for bank1.

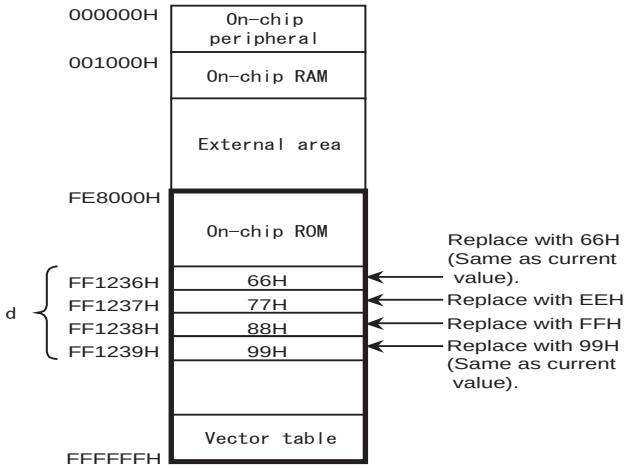


Figure 10-5 Example Patch Code Implementation

10.3.2 Using an interrupt to cause a branch

A wider range of program code can also be fixed using a software interrupt (SWI). With a patch code loaded into on-chip RAM, the program patch logic can be used to replace program code at a specified address with a single-byte SWI instruction, which causes a branch to the patch program.

Note that this method can only be used if the original masked ROM has been developed with on-chip RAM addresses specified as SWI vector addresses.

Correction procedure:

Load the address compare registers (ROMCMP00 to ROMCMP02) with the start address of the program code that is to be fixed. If it is an even address, store an SWI instruction code (e.g., SWI:F9H) in the ROMSUBL. If the start address is an odd address, store an SWI instruction code in the ROMSUBH and the current ROM data at the preceding even address in the ROMSUBL.

When the CPU address matches the value stored in the ROMCMP00 to ROMCMP02 registers, the program patch logic disables RD output to the masked ROM and drives out the SWI instruction code to the internal bus. Upon fetching the SWI code, the CPU makes a branch to the internal RAM area to execute the preloaded code.

At the end of the patch program executed from the internal RAM, the CPU directly rewrites the saved PC value so that it points to the address following the patch code, and then executes a RETI.

The following shows an example:

Example: Fixing a program within the range from FF5000H to FF507FH

Before developing the original masked ROM, set the SWI1 vector reference address to 001500H (on-chip RAM area).

Use the startup routine to load the patch code to on-chip RAM (001500H to 0015EFH). Store the start address (FF5000H) of the ROM area to be fixed in the ROMCMP00 to ROMCMP02. Store the SWI1 instruction code (F9H) in the ROMSUB0L and the current data at FF5001H (AAH) in the ROMSUB0H. When the CPU address matches the value stored in ROMCMP00 to ROMCMP02, the program patch logic replaces the ROM-based code at FF5000H with F9H. The CPU then executes the SWI1 instruction, which causes a branch to 001500H in the on-chip RAM area. After executing the patch program the CPU finally rewrites the saved PC value to FF5080H and executes a RETI.

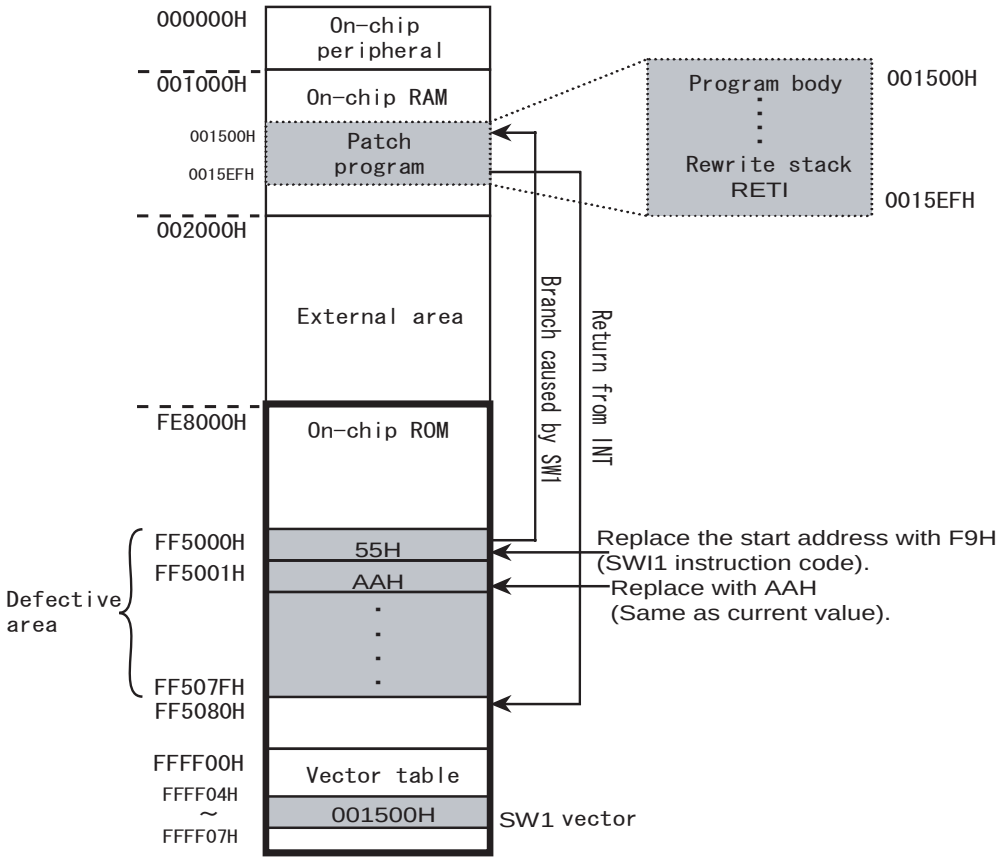


Figure 10-6 Example ROM Correction

11. Watchdog Timer (Runaway detection timer)

The TMP91FU62 features a watchdog timer for detecting runaway.

The watchdog timer (WDT) is used to return the CPU to normal state when it detects that the CPU has started to malfunction (Runaway) due to causes such as noise.

When the watchdog timer detects a malfunction, it generates a non-maskable interrupt INTWD to notify the CPU. Connecting the watchdog timer output to the reset pin internally forces a reset.(The level of external $\overline{\text{RESET}}$ pin is not changed)

11.1 Configuration

Figure 11-1 is a block diagram of he watchdog timer (WDT).

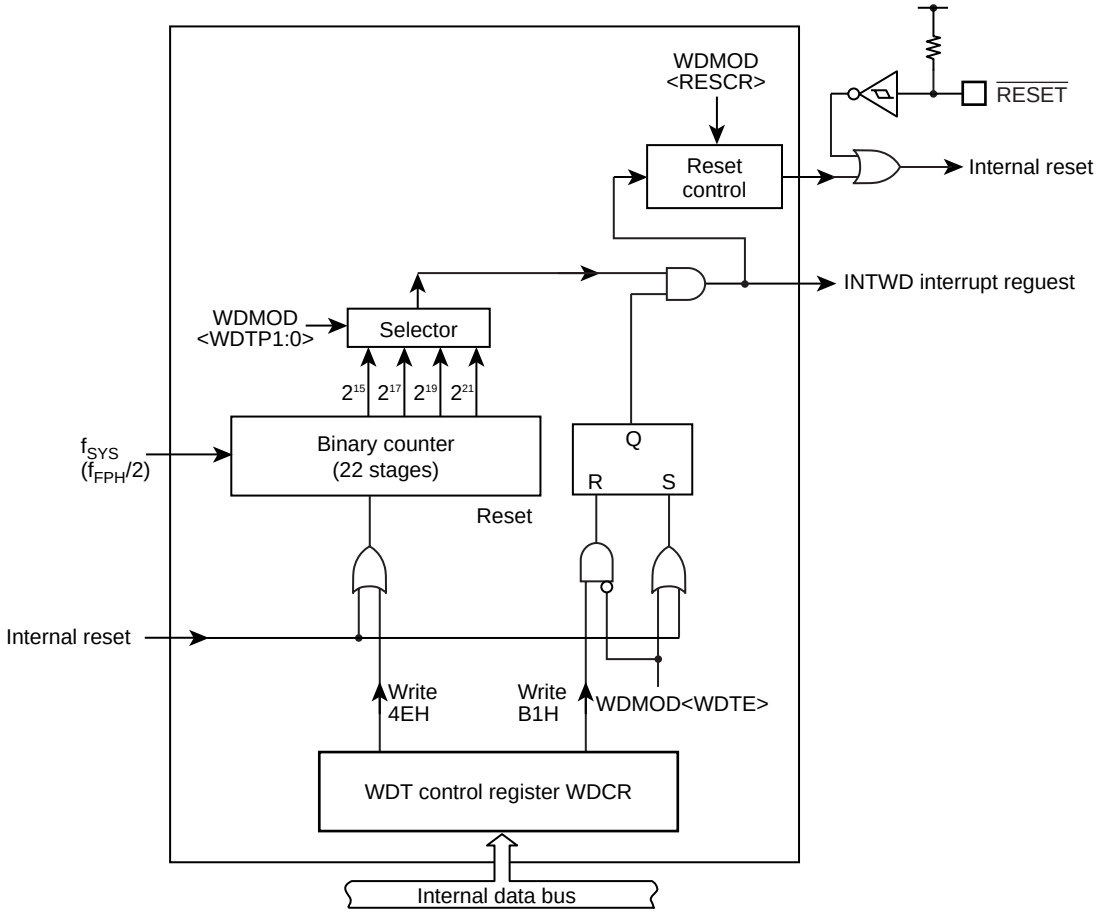


Figure 11-1 Block Diagram of Watchdog Timer

Note: It needs to care designing the total machine set, because watchdog timer can't operate completely by external noise.

11.2 Operation

The watchdog timer generates an INTWD interrupt when the detection time set in the WDMOD<WDTP1:0> has elapsed. The watchdog timer must be cleared “0” by software before an INTWD interrupt will be generated. If the CPU malfunctions (e.g., if runaway occurs) due to causes such as noise, but does not execute the instruction used to clear the binary counter, the binary counter will overflow and an INTWD interrupt will be generated. The CPU will detect malfunction (Runaway) due to the INTWD interrupt and in this case it is possible to return to the CPU to normal operation by means of an anti-malfunction program.

The watchdog timer works immediately after reset.

The watchdog timer does not operate in IDLE1 or STOP mode. When the device is in IDLE2 mode, the operation of WDT depends on the WDMOD<I2WDT> setting. Ensure that WDMOD<I2WDT> is set before the device enters IDLE2 mode.

The watchdog timer consists of a 22-stage binary counter which uses the system clock (f_{SYS}) as the input clock. The binary counter can output $f_{SYS}/2^{15}$, $f_{SYS}/2^{17}$, $f_{SYS}/2^{19}$ and $f_{SYS}/2^{21}$.

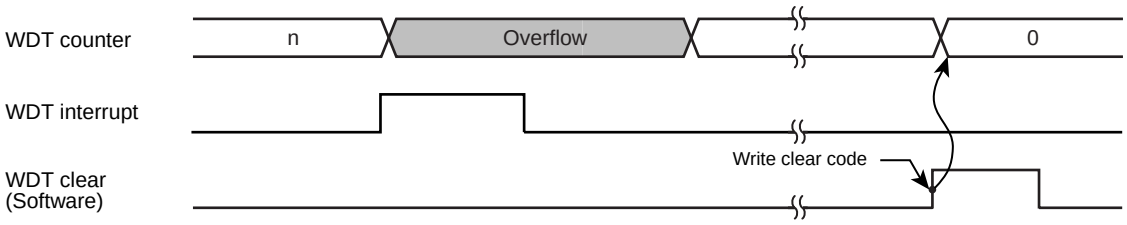


Figure 11-2 Normal Mode

The runaway is detected when an overflow occurs, and the watchdog timer can reset this device. In this case, the reset time will be between 22 and 29 states (51.2 μs at f_{OSCH} = 20 MHz) as shown in Figure 11-3. After a reset, the f_{SYS} clock (1 cycle = 1 state) is $f_{FPH}/2$, where f_{FPH} is generated by dividing the high-speed oscillator clock (f_{OSCH}) by sixteen through the clock gear function.

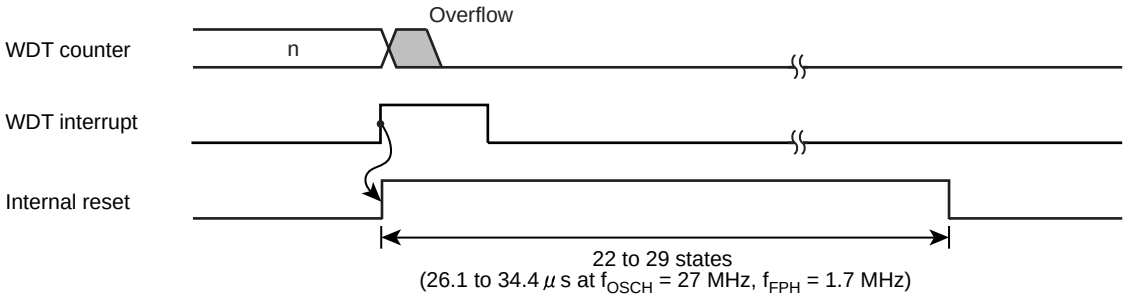


Figure 11-3 Reset Mode

11.3 Control Registers

The watchdog timer WDT is controlled by two control registers WDMOD and WDCR.

11.3.1 Watchdog timer mode register (WDMOD)

a. Setting the detection time for the watchdog timer in <WDTP1:0>

This 2-bit register is used for setting the watchdog timer interrupt time used when detecting runaway. After reset, this register is initialized to WDMOD<WDTP1:0> = "00"(2^{15}/ f_{SYS} [S]).

b. Watchdog timer enable/disable control register <WDTE>

After reset, WDMOD<WDTE> is initialized to "1", enabling the watchdog timer.

To disable the watchdog timer, it is necessary to set this bit to "0" and to write the disable code (B1H) to the watchdog timer control register WDCR. This makes it difficult for the watchdog timer to be disabled by runaway.

However, it is possible to return the watchdog timer from the disabled state to the enabled state merely by setting <WDTE> to "1".

c. Watchdog timer out reset connection <RESCR>

This register is used to connect the output of the watchdog timer with the internal RESET. Since WDMOD<RESCR> is initialized to "0" on reset, a reset by the watchdog timer will not be performed.

11.3.2 Watchdog timer control register (WDCR)

This register is used to disable and clear the binary counter for the watchdog timer.

- Disable control

The watchdog timer can be disabled by clearing WDMOD<WDTE> to "0" and then writing the disable code (B1H) to the WDCR register.

WDMOD	←	0	–	–	X	X	–	–	0	Clear WDMOD<WDTE> to "0".
WDCR	←	1	0	1	1	0	0	0	1	Write the disable code (B1H).

- Enable control

Set WDMOD<WDTE> to "1".

- Watchdog timer clear control

To clear the binary counter and cause counting to resume, write the clear code (4EH) to the WDCR register.

WDCR	←	0	1	0	0	1	1	1	0	Write the clear code (4EH).

Watchdog Timer Mode Register

WDMOD (0300H)

	7	6	5	4	3	2	1	0
Bit symbol	WDTE	WDTP1	WDTP0	–	–	I2WDT	RESCR	–
Read/Write	R/W	R/W		–	–	R/W		R/W
After reset	1	0	0	–	–	0	0	0
Function	WDT control 1: Enable	Select detecting time 00: $2^{15}/f_{SYS}$ 01: $2^{17}/f_{SYS}$ 10: $2^{19}/f_{SYS}$ 11: $2^{21}/f_{SYS}$				IDLE2 control	Reset control	Always write “0”.

Watchdog timer out control

RESCR	0	–
	1	Connect WDT out to a internal reset

IDLE2 control

I2WDT	0	Stop
	1	Operation

Watchdog timer detection time @fc = 20 MHz, fs = 32.768 kHz

SYSCR1 System Clock Selection <SYSCK>	SYSCR1 Gear Value <GEAR2:0>	Watchdog Timer Detection Time WDMOD<WDTP1:0>			
		00	01	10	11
1(fs)	xxx	2.0 s	8.0 s	32.0 s	128.0 s
0(fc)	000 (fc)	3.28 ms	13.11 ms	52.43 ms	209.72 ms
	001 (fc/2)	6.55 ms	26.21 ms	104.86 ms	419.43 ms
	010(fc/4)	13.11 ms	52.43 ms	209.72 ms	838.86 ms
	011 (fc/8)	26.21 ms	104.86 ms	419.43 ms	1677.72 ms
	100 (fc/16)	52.43 ms	209.72 ms	838.86 ms	3355.44 ms

Watchdog timer enable/disable control

WDTE	0	Disabled
	1	Enabled

Watchdog Timer Control Register

WDCR (0301H) RMW instructions are prohib-ited.

	7	6	5	4	3	2	1	0
Bit symbol	-							
Read/Write	W							
After reset	-							
Function	B1H: WDT disable code 4EH: WDT clear code							

Disable/clear WDT

B1H	Disable code
4EH	Clear code
Others	Don’t care

12. Special timer for CLOCK

The TMP91FU62 includes a timer that is used for a clock operation.

An interrupt (INTRTC) can be generated each 0.0625 [s] or 0.125 [s] or 0.25 [s] or 0.50 [s] by using a low frequency clock of 32.768 kHz. A clock function can be easily used.

In addition, INTRTC can return from each standby mode except STOP mode.

A special timer for CLOCK can operate in all modes in which a low-frequency oscillation is operated.

The special timer for CLOCK is controlled by the special timer for CLOCK control register (RTCCR) as shown in.

12.1 Configuration

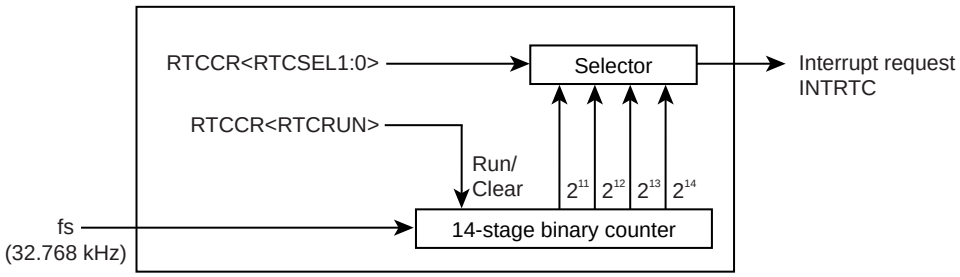


Figure 12-1 Block Diagram for Special Timer for CLOCK

Special Timer for CLOCK Control Register

RTCCR (0310H)

	7	6	5	4	3	2	1	0
Bit symbol	–	–	–	–	–	RTCSEL1	RTCSEL0	RTCRUN
Read/Write	R/W	–	–	–	–	R/W		R/W
After reset	0	–	–	–	–	0	0	0
Function	Always write “0”.	–	–	–	–	00: 2^{14}/fs 01: 2^{13}/fs 10: 2^{12}/fs 11: 2^{11}/fs		0: Stop & clear 1: Count

Counting operation

<RTCRUN>	0	Stop & clear
	1	Count

Interrupt generation cycle (fs = 32.768 kHz)

<RTCSEL1:0>	00	0.50 s
	11	0.25 s
	10	0.125 s
	11	0.0625 s

13. Flash Memory

The TMP91FU62 incorporates flash memory that can be electrically erased and programmed using a single 5V power supply.

The flash memory is programmed and erased using JEDEC-standard commands. After a program or erase command is input, the corresponding operation is automatically performed internally. Erase operations can be performed by the entire chip (chip erase) or on a sector basis (sector erase).

The configuration and operations of the flash memory are described below.

13.1 Features

Power supply voltage for program/erase operations
- Vcc = 4.75 to 5.25 V
  (T_{OPR} = -10 to 40 °C, fc = 4 to 20MHz)

Configuration
- 48K × 16 bits (96 k bytes)

Functions
- Single-word programming
- Chip erase
- Sector erase
- Data polling / Toggle bit

Sector size
- 8Kbytes × 12

Mode control
- JEDEC-standard commands

Programming method
- On-board programming
- Parallel programmer

Security
- Write protection
- Read protection

13.2 Block Diagram

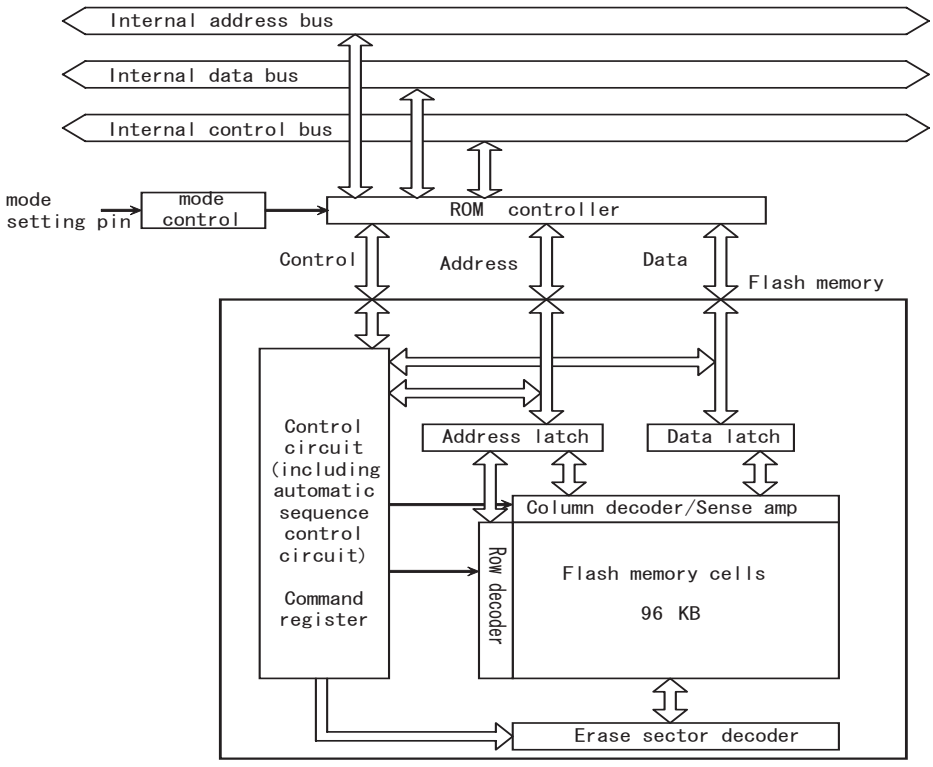


Figure 13-1 Block Diagram of Flash Memory Unit

13.3 Operation Modes

13.3.1 Overview

The following three types of operation modes are available to control program/erase operations on the flash memory.

Table 13-1 Description of Operation Modes

Operation Mode Name		Description
Single Chip mode		After reset release, the device starts up from the internal flash memory. Single Chip mode is further divided into two modes: “Normal mode” is a mode in which user application programs are executed, and “User Boot mode” is used to program the flash memory on-board. The means of switching between these two modes can be set by the user as desired. For example, it can be set so that Port 00 = '1' selects Normal mode and Port 00 = '0' selects User Boot mode. The user must include a routine to handle mode switching in a user application program.
	Normal mode	In this mode, the device starts up from a user application program.
	User Boot mode	In this mode, the flash memory can be programmed by a user-specified method.
Single Boot mode		After reset release, the device starts up from the internal boot ROM (mask ROM). The boot ROM includes an algorithm which allows a program for programming/erasing the flash memory on-board via a serial port to be transferred to the device's internal RAM. The transferred program is then executed in the internal RAM so that the flash memory can be programmed/erased by receiving data from an external host and issuing program/erase commands.
Programmer mode		This mode enables the internal flash memory to be programmed/erased using a general-purpose programmer. For programmers that can be used, please contact your local Toshiba sales representative.

Of the modes listed in Table 13-1, the internal flash memory can be programmed in User Boot mode, Single Boot mode and Programmer mode.

The mode in which the flash memory can be programmed/erased while mounted on the user board is defined as the on-board programming mode. Of the modes listed above, Single Boot mode and User Boot mode are classified as on-board programming modes. Single Boot mode supports Toshiba's proprietary programming/erase method using serial I/O. User Boot mode (within Single Chip mode) allows the flash memory to be programmed/erased by a user-specified method.

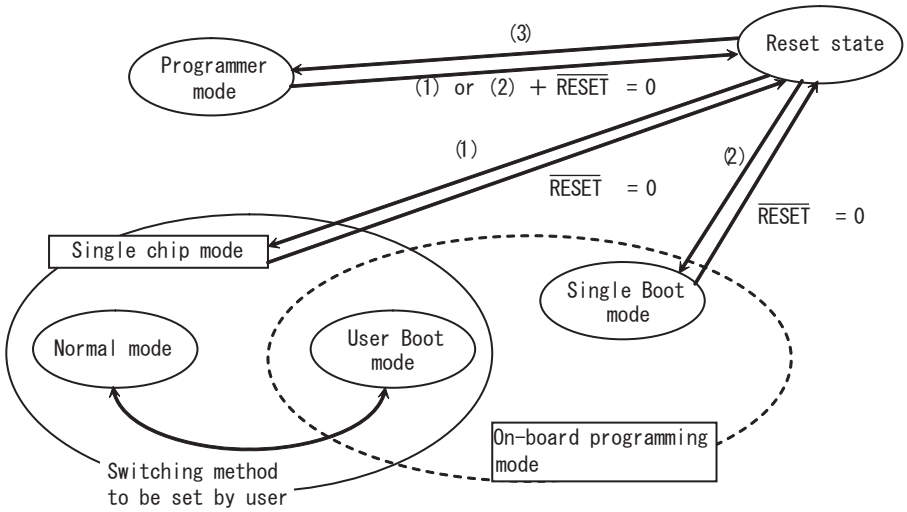
Programmer mode is provided with a read protect function which prohibits reading of ROM data. By enabling the read protect function upon completion of programming, the user can protect ROM data from being read by third parties.

The operation mode Single Chip mode, Single Boot mode or Programmer mode is determined during reset by externally setting the input levels on the AM0, AM1 and $\overline{BOOT}$ (EMU0) pins.

Except in Programmer mode which is entered with $\overline{RESET}$ held at "0", the CPU will start operating in the selected mode after the reset state is released. Once the operation mode has been set, make sure that the input levels on the mode setting pins are not changed during operation.Table 13-2 shows how to set each operation mode, and Figure 13-2 shows a mode transition diagram.

Table 13-2 Operation Mode Pin Settings

	Operation Mode	Input pins		
		$\overline{RESET}$	AM1	AM0
(1)	Single Chip mode (Normal or User Boot mode)	rising edge	1	1
(2)	Single Boot mode		0	1
(3)	Programmer mode	0	1	0



Note: Numbers in ( ) correspond to the operation mode pin settings shown in Table 13-2.

Figure 13-2 Mode Transition Diagram

13.3.2 Reset Operation

To reset the device, hold the $\overline{RESET}$ input at "0" for at least 10 system clocks while the power supply voltage is within the rated operating voltage range and the internal high-frequency oscillator is oscillating stably.

For details, refer to "Reset of CPU".

13.3.3 Memory Map for Each Operation Mode

In this product, the memory map varies with operation mode. The memory map and sector address ranges for each operation mode are shown below.

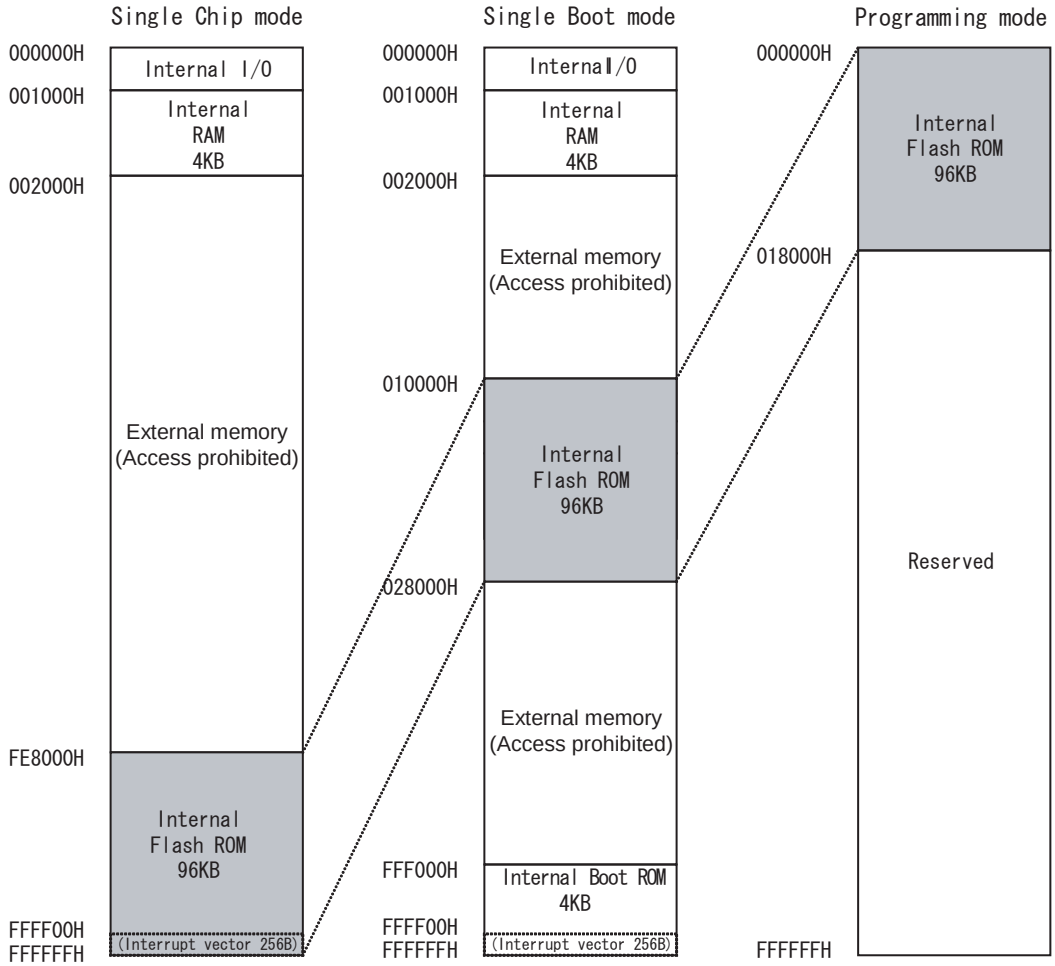


Figure 13-3 Memory Map for Each Operation Mode

Table 13-3 Sector Address Ranges for Each Operation Mode

	Single Chip Mode	Single Boot Mode
Sector-0	FE8000H to FE9FFFH	10000H to 11FFFH
Sector-1	FEA000H to FEBFFFH	12000H to 13FFFH
Sector-2	FEC000H to FEDFFFH	14000H to 15FFFH
Sector-3	FEE000H to FEFFFFH	16000H to 17FFFH
Sector-4	FF0000H to FF1FFFH	18000H to 19FFFH
Sector-5	FF2000H to FF3FFFH	1A000H to 1BFFFH
Sector-6	FF4000H to FF5FFFH	1C000H to 1DFFFH
Sector-7	FF6000H to FF7FFFH	1E000H to 1FFFFH
Sector-8	FF8000H to FF9FFFH	20000H to 21FFFH
Sector-9	FFA000H to FFBFFFH	22000H to 23FFFH
Sector-10	FFC000H to FFDFFFH	24000H to 25FFFH
Sector-11	FFE000H to FFFFFFH	26000H to 27FFFH

13.4 Single Boot Mode

In Single Boot mode, the internal boot ROM (mask ROM) is activated to transfer a program/erase routine (user-created boot program) from an external source into the internal RAM. This program/erase routine is then used to program/erase the flash memory. In this mode, the internal boot ROM is mapped into an area containing the interrupt vector table, in which the boot ROM program is executed. The flash memory is mapped into an address space different from the one into which the boot ROM is mapped (See Figure 13-3).

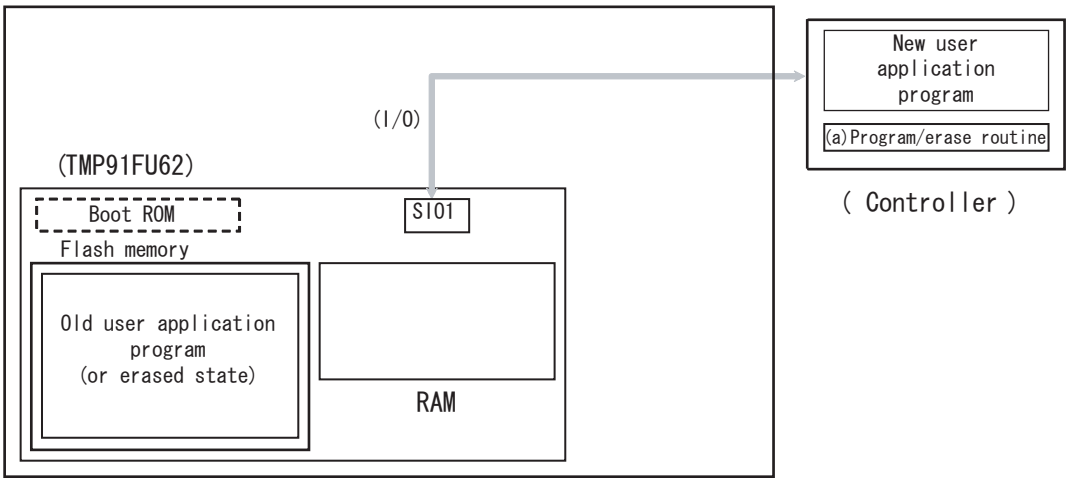
The device's SIO (SIO1) and the controller are connected to transfer the program/erase routine from the controller to the device's internal RAM. This program/erase routine is then executed to program/erase the flash memory. The program/erase routine is executed by sending commands and write data from the controller. The communications protocol between the device and the controller is described later in this manual. Before the program/erase routine can be transferred to the RAM, user password verification is performed to ensure the security of user ROM data. If the password is not verified correctly, the RAM transfer operation cannot be performed. In Single Boot mode, disable interrupts and use the interrupt request flags to check for an interrupt request.

Note: Do not change to another operation mode in the program/erase routine.

13.4.1 Using the program/erase algorithm in the internal boot ROM

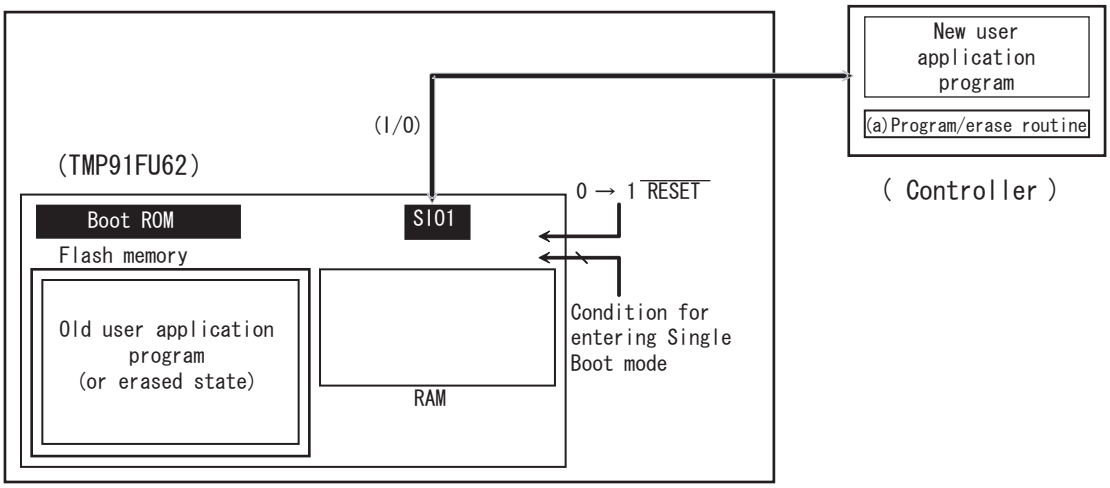
(Step-1)Environment setup

Since the program/erase routine and write data are transferred via SIO (SIO1), connect the device's SIO (SIO1) and the controller on the board. The user must prepare the program/erase routine (a) on the controller.



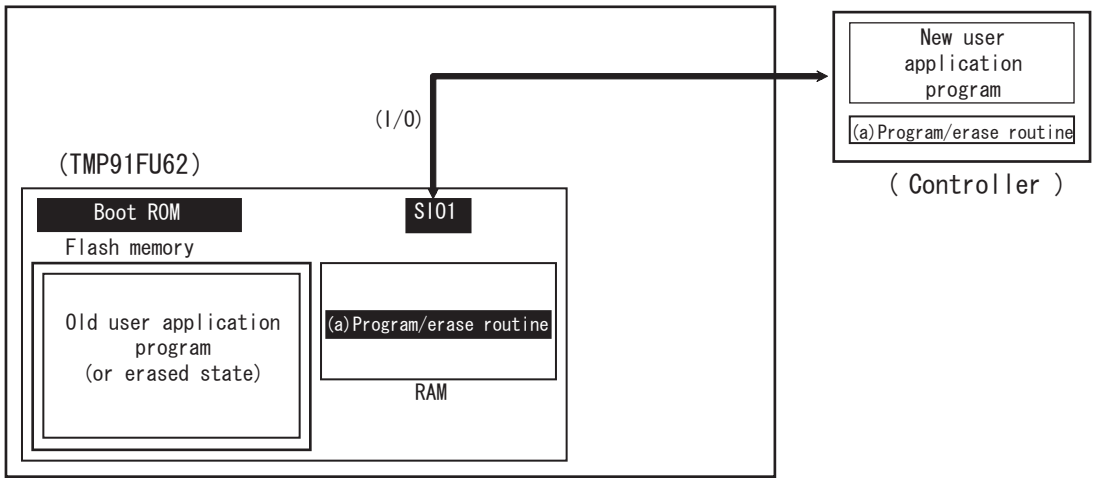
(Step-2) Starting up the internal boot ROM

Release the reset with the relevant input pins set for entering Single Boot mode. When the internal boot ROM starts up, the program/erase routine (a) is transferred from the controller to the internal RAM via SIO according to the communications procedure for Single Boot mode. Before this can be carried out, the password entered by the user is verified against the password written in the user application program. (If the flash memory has been erased, 12 bytes of “0xFF” are used as the password.)



(Step-3) Copying the program/erase routine to the RAM

After password verification is completed, the boot ROM copies the program/erase routine (a) from the controller to the RAM using serial communications. The program/erase routine must be stored within the RAM address range of 001000H to 001DFFH.

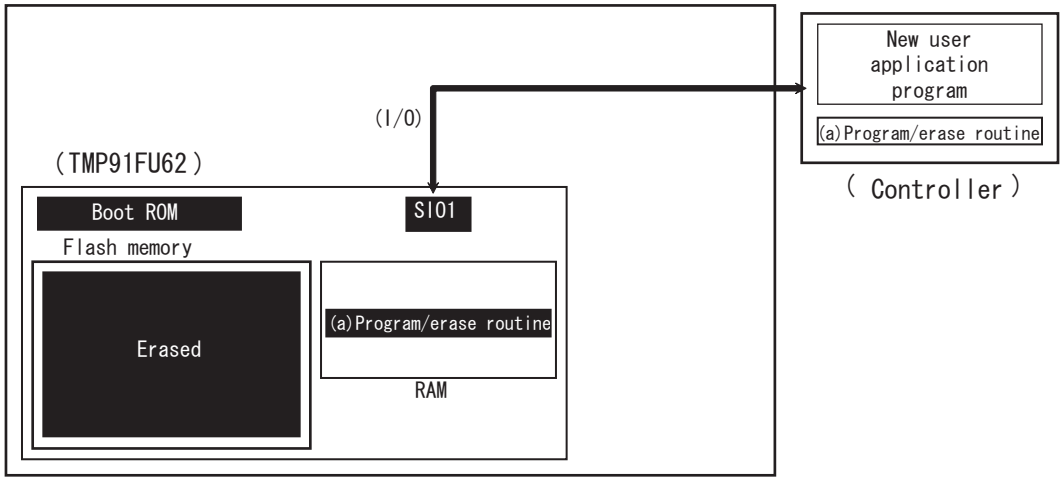


(Step-4) Executing the program/erase routine in the RAM

Control jumps to the program/erase routine (a) in the RAM. If necessary, the old user application program is erased (sector erase or chip erase).

Note 1: The boot ROM is provided with an erase command, which enables the entire chip to be erased from the controller without using the program/erase routine.

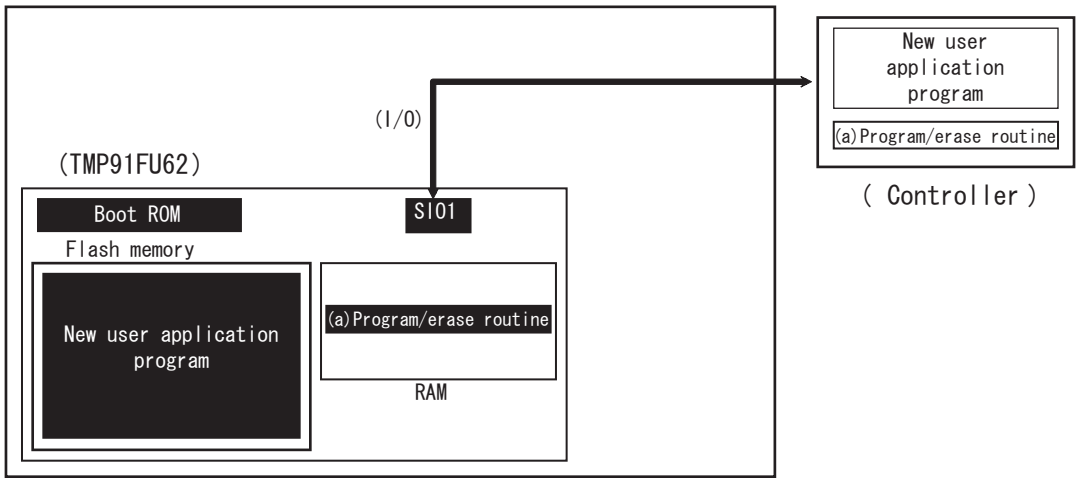
Note 2: If it is necessary to erase data on a sector basis, incorporate the necessary code in the program/erase routine.



(Step-5) Copying the new user application program

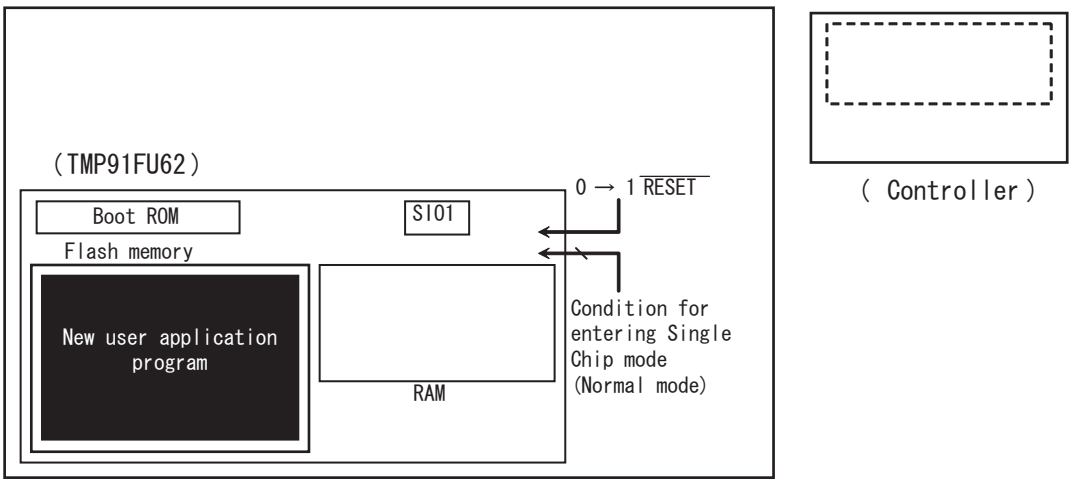
The program/erase routine (a) loads the new user application program from the controller into the erased area of the flash memory.

In the example below, the new user application program is transferred under the same communications conditions as those used for transferring the program/erase routine. However, after the program/erase routine has been transferred, this routine can be used to change the transfer settings (data bus and transfer source). Configure the board hardware and program/erase routine as desired.



(Step-6) Executing the new user application program

After the programming operation has been completed, turn off the power to the board and remove the cable connecting the device and the controller. Then, turn on the power again and start up the device in Single Chip mode to execute the new user application program.



13.4.2 Connection Examples for Single Boot Mode

In Single Boot mode the flash memory is programmed by serial transfer. Therefore, on-board programming is performed by connecting the device's SIO (SIO1) and the controller (programming tool) and sending commands from the controller to the device. Figure 13-4 shows an example of connection between the target board and a programming controller. Figure 13-5 shows an example of connection between the target board and an RS232C board.

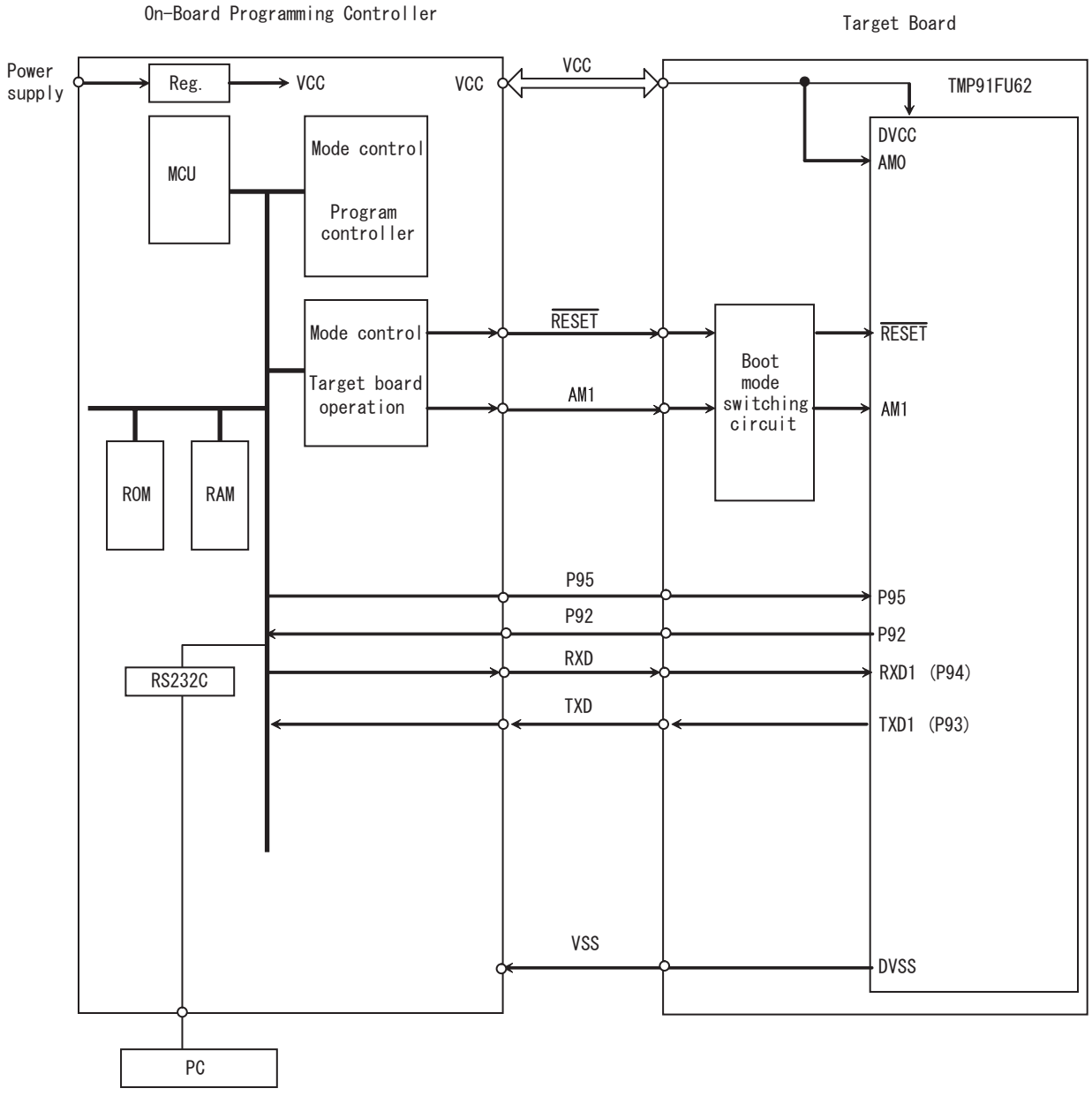


Figure 13-4 Example of Connection with an External Controller in Single Boot Mode

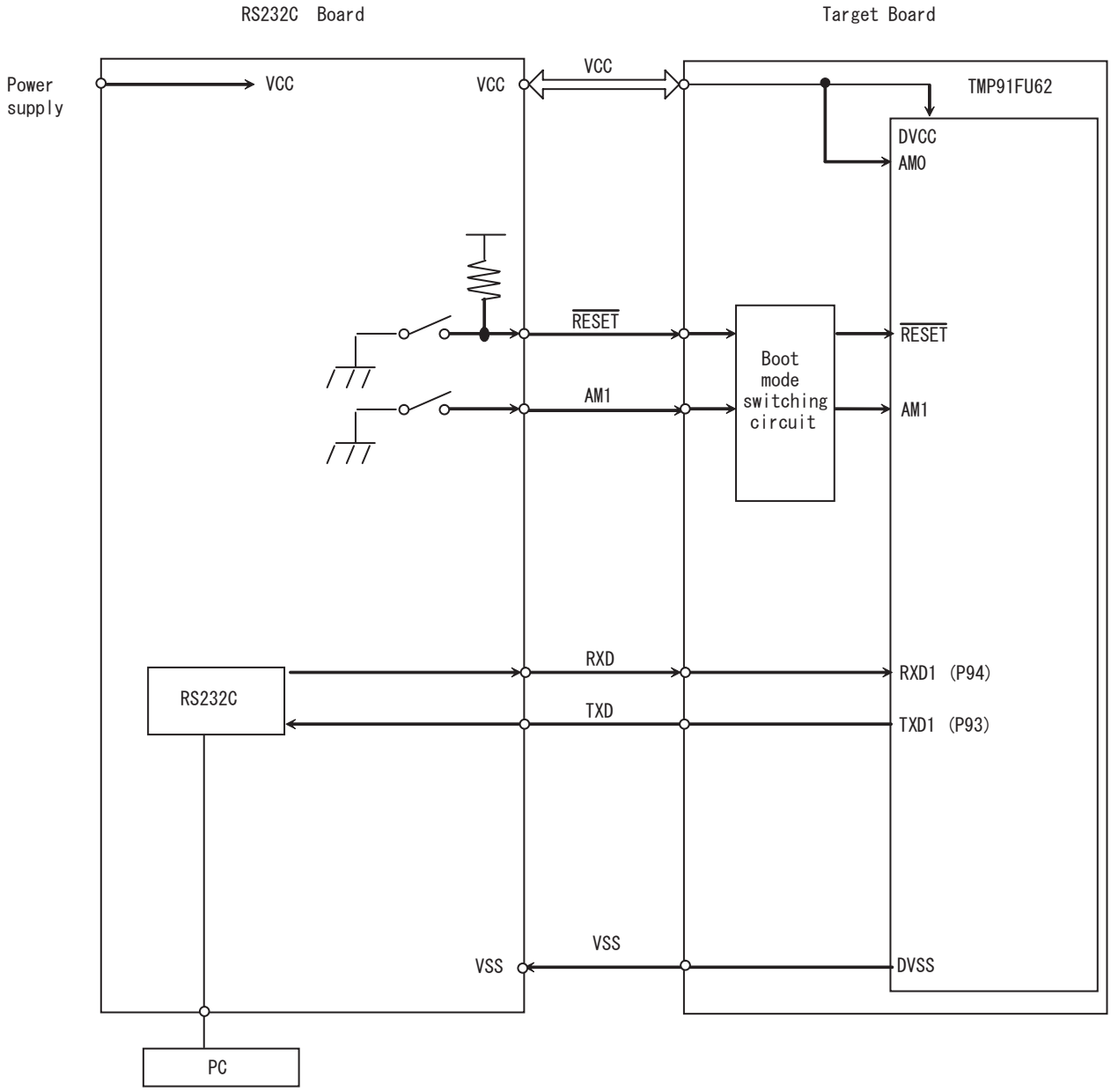


Figure 13-5 Example of Connection with an RS232C Board in Single Boot Mode

13.4.3 Mode Setting

To perform on-board programming, the device must be started up in Single Boot mode by setting the input pins as shown below.

AM0 = 1

AM1 = 0

$\overline{\text{RESET}}$= 0 → 1

Set the AM0 and AM1 pins as shown above with the $\overline{\text{RESET}}$ pin held at “0”. Then, setting the $\overline{\text{RESET}}$ pin to “1” will start up the device in Single Boot mode.

13.4.4 Memory Maps

Figure 13-6 shows a comparison of the memory map for Normal mode (in Single Chip mode) and the memory map for Single Boot mode. In Single Boot mode, the flash memory is mapped to addresses 10000H to 27FFFH (physical addresses) and the boot ROM (mask ROM) is mapped to addresses FFF000H to FFFFFFH.

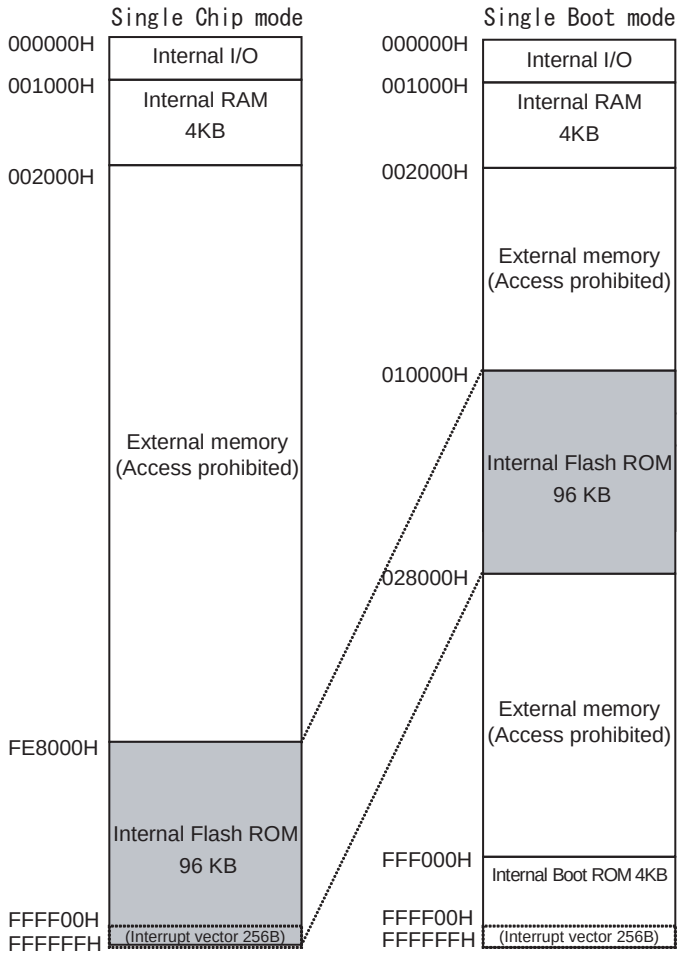


Figure 13-6 Comparison of Memory Maps

13.4.5 Interface Specifications

The SIO communications format in Single Boot mode is shown below. The device supports the UART (asynchronous communications) serial operation mode.

To perform on-board programming, the same communications format must also be set on the programming controller's side.

UART (asynchronous) communications

- Communications channel : SIO channel 1(For the pins be used, see Table 13-4)
- Serial transfer mode : UART (asynchronous communications) mode
- Data length : 8 bits
- Parity bit : None
- STOP bit : 1 bit
- Baud rate : See Table 13-5, Table 13-6

Table 13-4 Pin Connections

Pins		UART
Power supply pins	DVCC	O
	DVSS	O
Mode setting pins	AM1,AM0	O
Reset pin	$\overline{\text{RESET}}$	O
Communications pins	TXD1	O
	RXD1	O

Note: Unused pins are in the initial state after reset release.

Table 13-5 Baud Rate Table

SIO	Transfer Rate (bps)				
UART	115200	57600	38400	19200	9600

Table 13-6 Correspondence between Operating Frequency and Baud Rate in Single Boot Mode

Reference Baud Rate (bps)		9600		19200		38400		57600		115200	
Reference Frequency (MHz)	Supported Range (MHz)	Baud Rate (bps)	Error (%)	Baud Rate (bps)	Error (%)	Baud Rate (bps)	Error (%)	Baud Rate (bps)	Error (%)	Baud Rate (bps)	Error (%)
8	7.87 to 8.14	9615	+0.16	-	-	-	-	-	-	-	-
10	9.69 to 10.02	9766	+1.73	19531	+1.73	39063	+1.73	-	-	-	-
11.0592	10.90 to 11.28	9600	0	19200	0	-	-	-	-	-	-
12.288	12.11 to 12.53	9600	0	19200	0	38400	0	-	-	-	-
14.7456	14.53 to 15.04	9600	0	19200	0	38400	0	57600	0	115200	0
16	15.74 to 16.29	9615	+0.16	19231	+0.16	-	-	-	-	-	-
18.4320	18.16 to 18.80	9600	0	19200	0	-	-	57600	0	-	-
20	19.37 to 20.05	9766	+1.73	19531	+1.73	39063	+1.73	-	-	-	-

Reference frequency:

The frequency of the high-speed oscillation circuit that can be used in Single Boot mode.

To program the flash memory using Single Boot mode, one of the reference frequencies must be selected as a high-speed clock.

Supported Range:

The range of clock frequencies that are detected as each reference frequency. It may not be possible to perform Single Boot operations at clock frequencies outside of the supported range.

Note: To automatically detect the reference frequency (microcontroller clock frequency), the transfer baud rate error of the flash memory programming controller and the oscillation frequency error must be within −1.5%, +2% in total.

13.4.6 Data Transfer Formats

Table 13-7 to Table 13-12 show the operation command data and the data transfer format for each operation mode.

Table 13-7 Operation Command Data

Operation Command Data	Operation Mode
10H	RAM Transfer
20H	Flash Memory SUM
30H	Product Information Read
40H	Flash Memory Chip Erase
60H	Flash Memory Protect Set

Table 13-8 Transfer Format of Single Boot Program [RAM Transfer]

	Transfer Byte Number	Transfer Data from Controller to Device	Baud Rate	Transfer Data from Device to Controller
BOOT ROM	1st byte	Baud rate setting UART 86H	Desired baud rate#1	-
	2nd byte	-		ACK response to baud rate setting Normal (baud rate OK) >UART 86H (If the desired baud rate cannot be set, operation is terminated.)
	3rd byte	Operation command data (10H)		-
	4th byte	-		ACK response to operation command#2 Normal 10H Error x1H Protection applied#3 x6H Communications error x8H
	5th byte to 16th byte	PASSWORD data (12 bytes) (027EF4H to 027EFFH)		-
	17th byte	CHECKSUM value for 5th to 16th bytes		-
	18th byte	-		ACK response to CHECKSUM value#2 Normal 10H Error 11H Communications error 18H
	19th byte	RAM storage start address 31 to 24 #4		-
	20th byte	RAM storage start address 23 to 16#4		-
	21th byte	RAM storage start address 15 to 8#4		-
	22th byte	RAM storage start address 7 to 0#4		-
	23th byte	RAM storage byte count 15 to 8#4		-
	24th byte	RAM storage byte count 7 to 0#4		-
	25th byte	CHECKSUM value for 19th to 24th bytes #4		-
	26th byte	-		ACK response to CHECKSUM value#2 Normal 10H Error 11H Communications error 18H
	27th byte to (m)th byte	RAM storage data		-
	(m+1)th byte	CHECKSUM value for 27th to m'th bytes		-
	(m+2)th byte	-		ACK response to CHECKSUM value#2 Normal 10H Error 11H Communications error 18H
RAM	(m+3)th byte	-		JUMP to RAM storage start address

#1 For the desired baud rate setting, see Table 13-6.
#2 After sending an error response, the device waits for operation command data (3rd byte).
#3 When read protection or write protection is applied, the device aborts the received operation command and waits for the next operation command data (3rd byte).
#4 The data to be transferred in the 19th to 25th bytes should be programmed within the RAM address range of 001000H to 001DFFH (3.5 Kbytes).

Table 13-9 Transfer Format of Single Boot Program [Flash Memory SUM]

	Transfer Byte Number	Transfer Data from Controller to Device	Baud Rate	Transfer Data from Device to Controller
BOOT ROM	1st byte	Baud rate setting UART 86H	Desired baud rate#1	-
	2nd byte	-		ACK response to baud rate setting Normal (baud rate OK) >UART 86H (If the desired baud rate cannot be set, operation is terminated.)
	3rd byte	Operation command data (20H)		-
	4th byte	-		ACK response to CHECKSUM value#2 Normal 20H Error x1H Communications error x8H
	5th byte	-		SUM (upper)
	6th byte	-		SUM (lower)
	7th byte	-		CHECKSUM value for 5th and 6th bytes
	8th byte	(Wait for the next operation command data)		-

#1 For the desired baud rate setting, see Table 13-6.
#2 After sending an error response, the device waits for operation command data (3rd byte).

Table 13-10 Transfer Format of Single Boot Program [Product Information Read](1/2)

	Transfer Byte Number	Transfer Data from Controller to Device	Baud Rate	Transfer Data from Device to Controller
BOOT ROM	1st byte	Baud rate setting UART 86H	Desired baud rate[#1]	-
	2nd byte	-		ACK response to baud rate setting Normal (baud rate OK) >UART 86H (If the desired baud rate cannot be set, operation is terminated.)
	3rd byte	Operation command data (30H)		-
	4th byte	-		ACK response to operation command[#2] Normal 30H Error x1H Communications x8H
	5th byte	-		Flash memory data (address 027EF0H)
	6th byte	-		Flash memory data (address 027EF1H)
	7th byte	-		Flash memory data (address 027EF2H)
	8th byte	-		Flash memory data (address 027EF3H)
	9th byte to 20th byte	-		Part number (ASCII code, 12 bytes) 'TMP91FU62_ _ _ ' (from 9th byte)
	21th byte to 24th byte	-		Password comparison start address (4 bytes) F4H, 7EH, 02H, 00H (from 21st byte)
	25th byte to 28th byte	-		RAM start address (4 bytes) 00H, 10H, 00H, 00H (from 25th byte)
	29th byte to 32th byte	-		RAM (user area) end address (4 bytes) FFH, 1DH, 00H, 00H (from 29th byte)
	33th byte to 36th byte	-		RAM end address (4 bytes) FFH, 1FH, 00H, 00H (from 33rd byte)
	37th byte to 40th byte	-		Dummy data (4 bytes) 00H,00H,00H,00H (from 37th byte)
	41th byte to 44th byte	-		Dummy data (4 bytes) 00H, 00H, 00H, 00H (from 41st byte)
	45th byte to 46th byte	-		FUSE information (2 bytes from 45th byte) Read protection/Write protection 1) Applied/Applied : 00H, 00H 2) Not applied/Applied : 01H, 00H 3) Applied/Not applied : 02H, 00H 4) Not applied/Not applied : 03H, 00H
	47th byte to 50th byte	-		Flash memory start address (4 bytes) 00H, 00H, 01H, 00H (from 47th byte)
	51th byte to 54th byte	-		Flash memory end address (4 bytes) FFH, 7FH, 02H, 00H (from 51st byte)
	55th byte to 56th byte	-		Number of sectors in flash memory (2 bytes) 0CH, 00H (from 55th byte)
	57th byte to 60th byte	-		Start address of flash memory sectors of the same size (4 bytes) 00H, 00H, 01H, 00H (from 57th byte)

Table 13-10 Transfer Format of Single Boot Program [Product Information Read](2/2)

	Transfer Byte Number	Transfer Data from Controller to Device	Baud Rate	Transfer Data from Device to Controller
	61th byte to 64th byte	-		Size (in half words) of flash memory sectors of the same size (4 bytes) 00H, 10H, 00H, 00H (from 61st byte)
	65th byte	-		Number of flash memory sectors of the same size (1 byte) 0CH
	66th byte	-		CHECKSUM value for 5th to 65th bytes
	67th byte	(Wait for the next operation command data)		-

#1 For the desired baud rate setting, see Table 13-6.
#2 After sending an error response, the device waits for operation command data (3rd byte).

Table 13-11 Transfer Format of Single Boot Program [Flash Memory Chip Erase]

	Transfer Byte Number	Transfer Data from Controller to Device	Baud Rate	Transfer Data from Device to Controller
BOOT ROM	1st byte	Baud rate setting UART 86H	Desired baud rate[#1]	-
	2nd byte	-		ACK response to baud rate setting Normal (baud rate OK) >UART 86H (If the desired baud rate cannot be set, operation is terminated.)
	3rd byte	Operation command data (40H)		-
	4th byte	-		ACK response to operation command[#2] Normal 40H Error x1H Communications x8H
	5th byte	Erase Enable command data (54H)		-
	6th byte	-		ACK response to operation command[#2] Normal 54H Error x1H Communications x8H
	7th byte	-		ACK response to Erase command Normal 4FH Error 4CH
	8th byte	-		ACK response Normal 5DH Error 60H
	9th byte	(Wait for the next operation command data)		-

#1 For the desired baud rate setting, see Table 13-6.
#2 After sending an error response, the device waits for operation command data (3rd byte).

Table 13-12 Transfer Format of Single Boot Program [Flash Memory Protect Set]

	Transfer Byte Number	Transfer Data from Controller to Device	Baud Rate	Transfer Data from Device to Controller
BOOT ROM	1st byte	Baud rate setting UART 86H	Desired baud rate#1	-
	2nd byte	-		ACK response to baud rate setting Normal (baud rate OK) >UART 86H (If the desired baud rate cannot be set, operation is terminated.)
	3rd byte	Operation command data (60H)		-
	4th byte	-		ACK response to operation command#2 Normal 60H Error x1H Communications x8H
	5th byte to 16th byte	Password data (12 bytes) (027EF4H to 027EFFH)		-
	17th byte	CHECKSUM value for 5th to 16th bytes		-
	18th byte	-		ACK response to checksum value#2 Normal 60H Error 61H Communications 68H
	19th byte	-		ACK response to Protect Set command Normal 6FH Error 6CH
	20th byte	-		ACK response Normal 31H Error 34H
	21th byte	(Wait for the next operation command data)		-

#1 For the desired baud rate setting, see Table 13-6.

#2 After sending an error response, the device waits for operation command data (3rd byte).

13.4.7 Boot Program

When the device starts up in Single Boot mode, the boot program is activated.

The following explains the commands that are used in the boot program to communicate with the controller when the device starts up in Single Boot mode. Use this information for creating a controller for using Single Boot mode or for building a user boot environment.

1. RAM Transfer command

   In RAM transfer, data is transferred from the controller and stored in the device's internal RAM. When the transfer completes normally, the boot program will start running the transferred user program. Up to 3.5 Kbytes of data can be transferred as a user program. (This limit is implemented in the boot program to protect the stack pointer area.) The user program starts executing from the RAM storage start address.

   This RAM transfer function enables a user-created program/erase routine to be executed, allowing the user to implement their own on-board programming method. To perform on-board programming with a user program, the flash memory command sequences (see section 13.6) must be used. After the RAM Transfer command has been completed, the entire internal RAM area can be used.

   If read protection or write protection is applied on the device or a password error occurs, this command will not be executed.

2. Flash Memory SUM command

   This command calculates the SUM of 96 Kbytes of data in the flash memory and returns the result. There is no operation command available to the boot program for reading data from the entire area of the flash memory. Instead, this Flash Memory SUM command can be used. Reading the SUM value enables revision management of the application program.

3. Product Information Read command

   This command returns the information about the device including its part number and memory details stored in the flash memory at addresses 027EF0H to 027EF3H. This command can also be used for revision management of the application program.

4. Flash Memory Chip Erase command

   This command erases all the sectors in the flash memory. If read protection or write protection is applied on the device, all the sectors in the flash memory are erased and the read protection or write protection is cleared.

   Since this command is also used to restore the operation of the boot program when the password is forgotten, it does not include password verification.

5. Flash Memory Protect Set command

   This command sets both read protection and write protection on the device. However, if a password error occurs, this command will not be executed.

   When read protection is set, the flash memory cannot be read in Programmer mode. When write protection is set, the flash memory cannot be written in Programmer mode.

13.4.8 RAM Transfer Command

See Table 13-8.

1. From the controller to the device

   The data in the 1st byte is used to determine the baud rate. The 1st byte is transferred with receive operation disabled (SC1MOD0<RXE> = 0). (The baud rate is determined using an internal timer.)

   To communicate in UART mode

   Send the value 86H from the controller to the target board using UART settings at the desired baud rate. If the serial operation mode is determined as UART, the device checks to see whether or not the desired baud rate can be set. If the device determines that the desired baud rate cannot be set, operation is terminated and no communications can be established.

2. From the device to the controller

   The data in the 2nd byte is the ACK response returned by the device for the serial operation mode setting data sent in the 1st byte. If the data in the 1st byte is found to signify UART and the desired baud rate can be set, the device returns 86H.

   Baud rate determination

   The device determines whether or not the desired baud rate can be set. If it is found that the baud rate can be set, the boot program rewrites the BR1CR and BR1ADD values and returns 86H. If it is found that the desired baud rate cannot be set, operation is terminated and no data is returned. The controller sets a time-out time (5 seconds) after it has finished sending the 1st byte. If the controller does not receive the response (86H) normally within the time-out time, it should be considered that the device is unable to communicate. Receive operation is enabled (SC1MOD0<RXE> = 1) before 86H is written to the transmission buffer.

3. From the controller to the device

   The data in the 3rd byte is operation command data. In this case, the RAM Transfer command data (10H) is sent from the controller to the device.

4. From the device to the controller

   The data in the 4th byte is the ACK response to the operation command data in the 3rd byte. First, the device checks to see if the received data in the 3rd byte contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) x8H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined (They are the upper four bits of the immediately preceding operation command data).

   Next, if the data received in the 3rd byte corresponds to one of the operation commands given in Table 13-7, the device echoes back the received data (ACK response for normal reception). In the case of the RAM Transfer command, if read or write protection is not applied, 10H is echoed back and then execution branches to the RAM transfer processing routine. If protection is applied, the device returns the corresponding ACK response data (bit 2/1) x6H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)

   After branching to the RAM transfer processing routine, the device checks the data in the password area. For details, see " 13.4.15 Password ".

   If the data in the 3rd byte does not correspond to any operation command, the device returns the ACK response data for operation command error (bit0) x1H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)

5. From the controller to the device

   The 5th to 16th bytes contain password data (12 bytes). The data in the 5th to 16th bytes is verified against the data at addresses 027EF4H to 027EFFH in the flash memory, respectively.

6. From the controller to the device

   The 17th byte contains CHECKSUM data. The CHECKSUM data sent by the controller is the two's complement of the lower 8-bit value obtained by summing the data in the 5th to 16th bytes by unsigned 8-bit addition (ignoring any overflow). For details on CHECKSUM, see " 13.4.17 How to Calculate CHECKSUM ".

7. From the device to the controller

   The data in the 18th byte is the ACK response data to the 5th to 17th bytes (ACK response to the CHECKSUM value). The device first checks to see whether the data received in the 5th to 17th bytes contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) 18H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are the upper four bits of the immediately preceding operation command data, so the value of these bits is “1”.

   Next, the device checks the CHECKSUM data in the 17th byte. This check is made to see if the lower 8-bit value obtained by summing the data in the 5th to 17th bytes by unsigned 8-bit addition (ignoring any overflow) is 00H. If the value is not 00H, the device returns the ACK response data for CHECKSUM error (bit 0) 11H and waits for the next operation command data (3rd byte).

   Finally, the device examines the result of password verification. If all the data in the 5th to 16th bytes is not verified correctly, the device returns the ACK response data for password error (bit 0) 11H and waits for the next operation command data (3rd byte).

   If no error is found in all the above checks, the device returns the ACK response data for normal reception 10 H.

8. From the controller to the device

   The data in the 19th to 22nd bytes indicates the RAM start address for storing block transfer data. The 19th byte corresponds to address bits 31 to 24, the 20th byte to address bits 23 to 16, the 21st byte to address bits 15 to 8, and the 22nd byte to address bits 7 to 0.

9. From the controller to the device

   The data in the 23rd and 24th bytes indicates the number of bytes to be transferred. The 23rd byte corresponds to bits 15 to 8 of the transfer byte count and the 24th byte corresponds to bits 7 to 0.

10. From the controller to the device

    The data in the 25th byte is CHECKSUM data. The CHECKSUM data sent by the controller is the two's complement of the lower 8-bit value obtained by summing the data in the 19th to 24th bytes by unsigned 8-bit addition (ignoring any overflow). For details on CHECKSUM, see " 13.4.17 How to Calculate CHECKSUM ".

    Note: The data in the 19th to 25th bytes should be placed within addresses 001000H to 001DFFH (3.5Kbytes) in the internal RAM.

11. From the device to the controller

The data in the 26th byte is the ACK response data to the data in the 19th to 25th bytes (ACK response to the CHECKSUM value).

The device first checks to see whether the data received in the 19th to 25th bytes contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) 18H and waits for the next operation command (3rd byte). The upper four bits of the ACK response data are the upper four bits of the immediately preceding operation command data, so the value of these bits is “1”.

Next, the device checks the CHECKSUM data in the 25th byte. This check is made to see if the lower 8-bit value obtained by summing the data in the 19th to 25th bytes by unsigned 8-bit addition (ignoring any overflow) is 00H. If the value is not 00H, the device returns the ACK response data for CHECKSUM error (bit 0) 11H and waits for the next operation command data (3rd byte).

12. From the controller to the device

The data in the 27th to m'th bytes is the data to be stored in the RAM. This data is written to the RAM starting at the address specified in the 19th to 22nd bytes. The number of bytes to be written is specified in the 23rd and 24th bytes.

13. From the controller to the device

The data in the (m+1)th byte is CHECKSUM data. The CHECKSUM data sent by the controller is the two's complement of the lower 8-bit value obtained by summing the data in the 27th to m'th bytes by unsigned 8-bit addition (ignoring any overflow). For details on CHECKSUM, see " 13.4.17 How to Calculate CHECKSUM ".

14. From the device to the controller

The data in the (m+2)th byte is the ACK response data to the 27th to (m+1)th bytes (ACK response to the CHECKSUM value).

The device first checks to see whether the data in the 27th to (m+1)th byte contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) 18H and waits for the next operation command (3rd byte). The upper four bits of the ACK response are the upper four bits of the immediately preceding operation command data, so the value of these bits is “1”.

Next, the device checks the CHECKSUM data in the (m+1)th byte. This check is made to see if the lower 8-bit value obtained by summing the data in the 27th to (m+1)th bytes by unsigned 8-bit addition (ignoring any overflow) is 00H. If the value is not 00H, the device returns the ACK response data for CHECKSUM error (bit 0) 11H and waits for the next operation command data (3rd byte).

If no error is found in all the above checks, the device returns the ACK response data for normal reception 10H.

15. From the device to the controller

If the ACK response data in the (m+2)th byte is 10H (normal reception), the boot program then jumps to the RAM start address specified in the 19th to 22nd bytes.

13.4.9 Flash Memory SUM command

See Table 13-9.

1. The data in the 1st and 2nd bytes is the same as in the case of the RAM Transfer command.
2. From the controller to the device

   The data in the 3rd byte is operation command data. The Flash Memory SUM command data (20H) is sent here.
3. From the device to the controller

   The data in the 4th byte is the ACK response data to the operation command data in the 3rd byte.

   The device first checks to see if the data in the 3rd byte contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) x8H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)

   Then, if the data in the 3rd byte corresponds to one of the operation command values given in Table 13-7, the device echoes back the received data (ACK response for normal reception). In this case, 20H is echoed back and execution then branches to the flash memory SUM processing routine. If the data in the 3rd byte does not correspond to any operation command, the device returns the ACK response data for operation command error (bit 0) x1H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)
4. From the device to the controller

   The data in the 5th and 6th bytes is the upper and lower data of the SUM value, respectively. For details on SUM, see " 13.4.16 How to Calculate SUM ".
5. From the device to the controller

   The data in the 7th byte is CHECKSUM data. This is the two's complement of the lower 8-bit value obtained by summing the data in the 5th and 6th bytes by unsigned 8-bit addition (ignoring any overflow).
6. From the controller to the device

   The data in the 8th byte is the next operation command data.

13.4.10Product Information Read command

See Table 13-10.

1. The data in the 1st and 2nd bytes is the same as in the case of the RAM Transfer command.
2. From the controller to the device

   The data in the 3rd byte is operation command data. The Product Information Read command data (30H) is sent here.
3. From the device to the controller

   The data in the 4th byte is the ACK response data to the operation command data in the 3rd byte.

   The device first checks to see if the data in the 3rd byte contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) x8H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)

   Then, if the data in the 3rd byte corresponds to one of the operation command values given in Table 13-7, the device echoes back the received data (ACK response for normal reception). In this case, 30H is returned and execution then branches to the product information read processing routine. If the data in the 3rd byte does not correspond to any operation command, the device returns the ACK response data for operation command error (bit 0) x1H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)
4. From the device to the controller

   The data in the 5th to 8th bytes is the data stored at addresses 027EF0H to 027EF3H in the flash memory. By writing the ID information of software at these addresses, the version of the software can be managed. (For example, 0002H can indicate that the software is now in version 2.)
5. From the device to the controller

   The data in the 9th to 20th bytes denotes the part number of the device. 'TMP91FU62_ _ _' is sent in ASCII code starting from the 9th byte.

   Note: An underscore ('_') indicates a space.
6. From the device to the controller

   The data in the 21st to 24th bytes is the password comparison start address. F4H, 7EH, 02H and 00H are sent starting from the 21st byte.
7. From the device to the controller

   The data in the 25th to 28th bytes is the RAM start address. 00H, 10H, 00H and 00H are sent starting from the 25th byte.
8. From the device to the controller

   The data in the 29th to 32nd bytes is the RAM (user area) end address. FFH, 1DH, 00H and 00H are sent starting from the 29th byte.
9. From the device to the controller

   The data in the 33rd to 36th bytes is the RAM end address. FFH, 1FH, 00H and 00H are sent starting from the 33rd byte.
10. From the device to the controller

    The data in the 37th to 44th bytes is dummy data.
11. From the device to the controller

    The data in the 45th and 46th bytes contains the protection status and sector division information of the flash memory.

    >Bit 0 indicates the read protection status.

    0: Read protection is applied.

1: Read protection is not applied.

>Bit 1 indicates the write protection status.

0: Write protection is applied.

1: Write protection is not applied.

>Bit 2 indicates whether or not the flash memory is divided into sectors.

0: The flash memory is divided into sectors.

1: The flash memory is not divided into sectors.

>Bits 3 to 15 are sent as “0”.

12. From the device to the controller

    The data in the 47th to 50th bytes is the flash memory start address. 00H, 00H, 01H and 00H are sent starting from the 47th byte.

13. From the device to the controller

    The data in the 51st to 54th bytes is the flash memory end address. FFH, 7FH, 02H and 00H are sent starting from the 51st byte.

14. From the device to the controller

    The data in the 55th and 56th bytes indicates the number of sectors in the flash memory. 0CH and 00H are sent starting from the 55th byte.

15. From the device to the controller

    The data in the 57th to 65th bytes contains sector information of the flash memory. Sector information is comprised of the start address (starting from the flash memory start address), sector size and number of consecutive sectors of the same size. Note that the sector size is represented in word units.

    The data in the 57th to 65th bytes indicates 8 Kbytes of sectors (sector 0 to sector 11).

    For the data to be transferred, see Table 13-10.

16. From the device to the controller

    The data in the 66th byte is CHECKSUM data. This is the two's complement of the lower 8-bit value obtained by summing the data in the 5th to 65th bytes by unsigned 8-bit addition (ignoring any overflow).

17. From the controller to the device

    The data in the 67th byte is the next operation command data.

13.4.11 Flash Memory Chip Erase Command

See Table 13-11.

1. The data in the 1st and 2nd bytes is the same as in the case of the RAM Transfer command.
2. From the controller to the device

   The data in the 3rd byte is operation command data. The Flash Memory Chip Erase command data (40H) is sent here.
3. From the device to the controller

   The data in the 4th byte is the ACK response data to the operation command data in the 3rd byte.

   The device first checks to see if the data in the 3rd byte contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) x8H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)

   Then, if the data in the 3rd byte corresponds to one of the operation command values given in Table 13-7, the device echoes back the received data (ACK response for normal reception). In this case, 40H is echoed back. If the data in the 3rd byte does not correspond to any operation command, the device returns the ACK response data for operation command error (bit 0) x1H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)
4. From the controller to the device

   The data in the 5th byte is Erase Enable command data (54H).
5. From the device to the controller

   The data in the 6th byte is the ACK response data to the Erase Enable command data in the 5th byte.

   The device first checks to see if the data in the 5th byte contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) x8H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined (They are the upper four bits of the immediately preceding operation command data.)

   Then, if the data in the 5th byte corresponds to the Erase Enable command data, the device echoes back the received data (ACK response for normal reception). In this case, 54H is echoed back and execution jumps to the flash memory chip erase processing routine. If the data in the 5th byte does not correspond to the Erase Enable command data, the device returns the ACK response data for operation command error (bit 0) x1H and waits for the next operation command (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)
6. From the device to the controller

   The data in the 7th byte indicates whether or not the erase operation has completed successfully. If the erase operation has completed successfully, the device returns the end code (4FH). If an erase error has occurred, the device returns the error code (4CH).
7. From the device to the controller

   The data in the 8th byte is ACK response data. If the erase operation has completed successfully, the device returns the ACK response for erase completion (5DH). If an erase error has occurred, the device returns the ACK response for erase error (60H).
8. From the controller to the device

   The data in the 9th byte is the next operation command data.

13.4.12Flash Memory Protect Set command

See Table 13-12.

1. The data in the 1st and 2nd bytes is the same as in the case of the RAM Transfer command.
2. From the controller to the device

   The data in the 3rd byte is operation command data. The Flash Memory Protect Set command data (60H) is sent here.
3. From the device to the controller

   The data in the 4th byte is the ACK response data to the operation command data in the 3rd byte.

   The device first checks to see if the data in the 3rd byte contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) x8H and waits for the next operation command data. The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)

   Then, if the data in the 3rd byte corresponds to one of the operation command data values given in Table 13-7, the device echoes back the received data (ACK response for normal reception). In this case, 60H is echoed back and execution branches to the flash memory protect set processing routine.

   After branching to this routine, the data in the password area is checked. For details, see " 13.4.15 Password ". If the data in the 3rd byte does not correspond to any operation command, the device returns the ACK response data for operation command error (bit 0) x1H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are undefined. (They are the upper four bits of the immediately preceding operation command data.)
4. From the controller to the device

   The data in the 5th to 16th bytes is password data (12 bytes). The data in the 5th byte is verified against the data at address 027EF4H in the flash memory and the data in the 6th byte against the data at address 027EF5H. In this manner, the received data is verified consecutively against the data at the specified address in the flash memory. The data in the 16th byte is verified against the data at address 027EFFH in the flash memory.
5. From the controller to the device

   The data in the 17th byte is CHECKSUM data. The CHECKSUM data sent by the controller is the two's complement of the lower 8-bit value obtained by summing the data in 5th to 16th bytes by unsigned 8-bit addition (ignoring any overflow). For details on CHECKSUM, see " 13.4.17 How to Calculate CHECKSUM ".
6. From the device to the controller

   The data in the 18th byte is the ACK response data to the data in the 5th to 17th bytes (ACK response to the CHECKSUM value).

   The device first checks to see whether the data in the 5th to 17th bytes contains any error. If a receive error is found, the device returns the ACK response data for communications error (bit 3) 68H and waits for the next operation command data (3rd byte). The upper four bits of the ACK response data are the upper four bits of the immediately preceding operation command data, so the value of these bits is “6”.

   Then, the device checks the CHECKSUM data in the 17th byte. This check is made to see if the lower 8 bits of the value obtained by summing the data in the 5th to 17th bytes by unsigned 8-bit addition (ignoring any overflow) is 00H. If the value is not 00H, the device returns the ACK response data for CHECKSUM error (bit 0) 61H and waits for the next operation command data (3rd byte).

   Finally, the device examines the result of password verification. If all the data in the 5th to 16th bytes is not verified correctly, the device returns the ACK response data for password error (bit 0) 61H and waits for the next operation command data (3rd byte).

   If no error is found in the above checks, the device returns the ACK response data for normal reception 60H.

7. From the device to the controller

   The data in the 19th byte indicates whether or not the protect set operation has completed successfully. If the operation has completed successfully, the device returns the end code (6FH). If an error has occurred, the device returns the error code (6CH).

8. From the device to the controller

   The data in the 20th byte is ACK response data. If the protect set operation has completed successfully, the device returns the ACK response data for normal completion (31H). If an error has occurred, the device returns the ACK response data for error (34H).

9. From the device to the controller

   The data in the 21st byte is the next operation command data.

13.4.13ACK Response Data

The boot program notifies the controller of its processing status by sending various response data. Table 13-13 to Table 13-18 show the ACK response data returned for each type of received data. The upper four bits of ACK response data are a direct reflection of the upper four bits of the immediately preceding operation command data. Bit 3 indicates a receive error and bit 0 indicates an operation command error, CHECKSUM error or password error.

Table 13-13 ACK Response Data to Serial Operation Mode Setting Data

Transfer Data	Meaning
86H	The device can communicate in UART mode. (Note)

Note: If the desired baud rate cannot be set, the device returns no data and terminates operation.

Table 13-14 ACK Response Data to Operation Command Data

Transfer data	Meaning
x8H (Note)	A receive error occurred in the operation command data.
x6H (Note)	Terminated receive operation due to protection setting.
x1H (Note)	Undefined operation command data was received normally.
10H	Received the RAM Transfer command.
20H	Received the Flash Memory SUM command.
30H	Received the Product Information Read command.
40H	Received the Flash Memory Chip Erase command.
60H	Received the Flash Memory Protect Set command.

Note: The upper four bits are a direct reflection of the upper four bits of the immediately preceding operation command data.

Table 13-15 ACK Response data to CHECKSUM Data for RAM Transfer Command

Transfer data	Meaning
18H	A receive error occurred.
11H	A CHECKSUM error or password error occurred.
10H	Received the correct CHECKSUM value.

Table 13-16 ACK Response Data to Flash Memory Chip Erase Operation

Transfer data	Meaning
54H	Received the Erase Enable command.
4FH	Completed erase operation.
4CH	An erase error occurred.
5DH (Note)	Reconfirmation of erase operation
60H (Note)	Reconfirmation of erase error

Note: These codes are returned for reconfirmation of communications.

Table 13-17 ACK Response Data to CHECKSUM Data for Flash Memory Protect Set Command

Transfer data	Meaning
68H	A receive error occurred.
61H	A CHECKSUM or password error occurred.
60H	Received the correct CHECKSUM value.

Table 13-18 ACK Response Data to Flash Memory Protect Set Operation

Transfer data	Meaning
6FH	Completed the protect (read/write) set operation.
6CH	A protect (read/write) set error occurred.
31H (Note)	Reconfirmation of protect (read/write) set operation
34H (Note)	Reconfirmation of protect (read/write) set error

Note: These codes are returned for reconfirmation of communications.

13.4.14 Determining Serial Operation Mode

To communicate in UART mode, the controller should transmit the data value 86H as the first byte at the desired baud rate. Figure 13-7 shows the waveform of this operation.

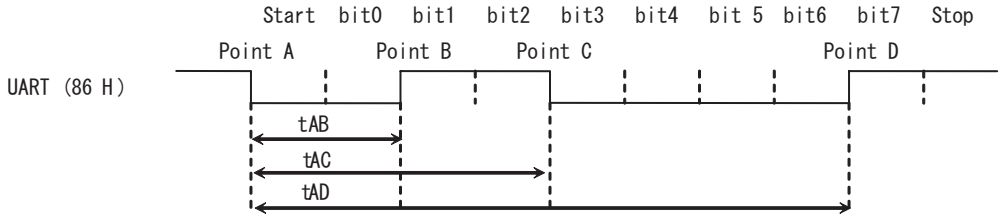


Figure 13-7 Data for Determining Serial Operation Mode

The boot program receives the first byte (86H) after reset release not as serial communications data. Instead, the boot program uses the first byte to determine the baud rate. The baud rate is determined by the output periods of tAB, tAC and tAD as shown in Figure 13-7 using the procedure shown in Figure 13-8.

The CPU monitors the level of the receive pin. Upon detecting a level change, the CPU captures the timer value to determine the baud rate.

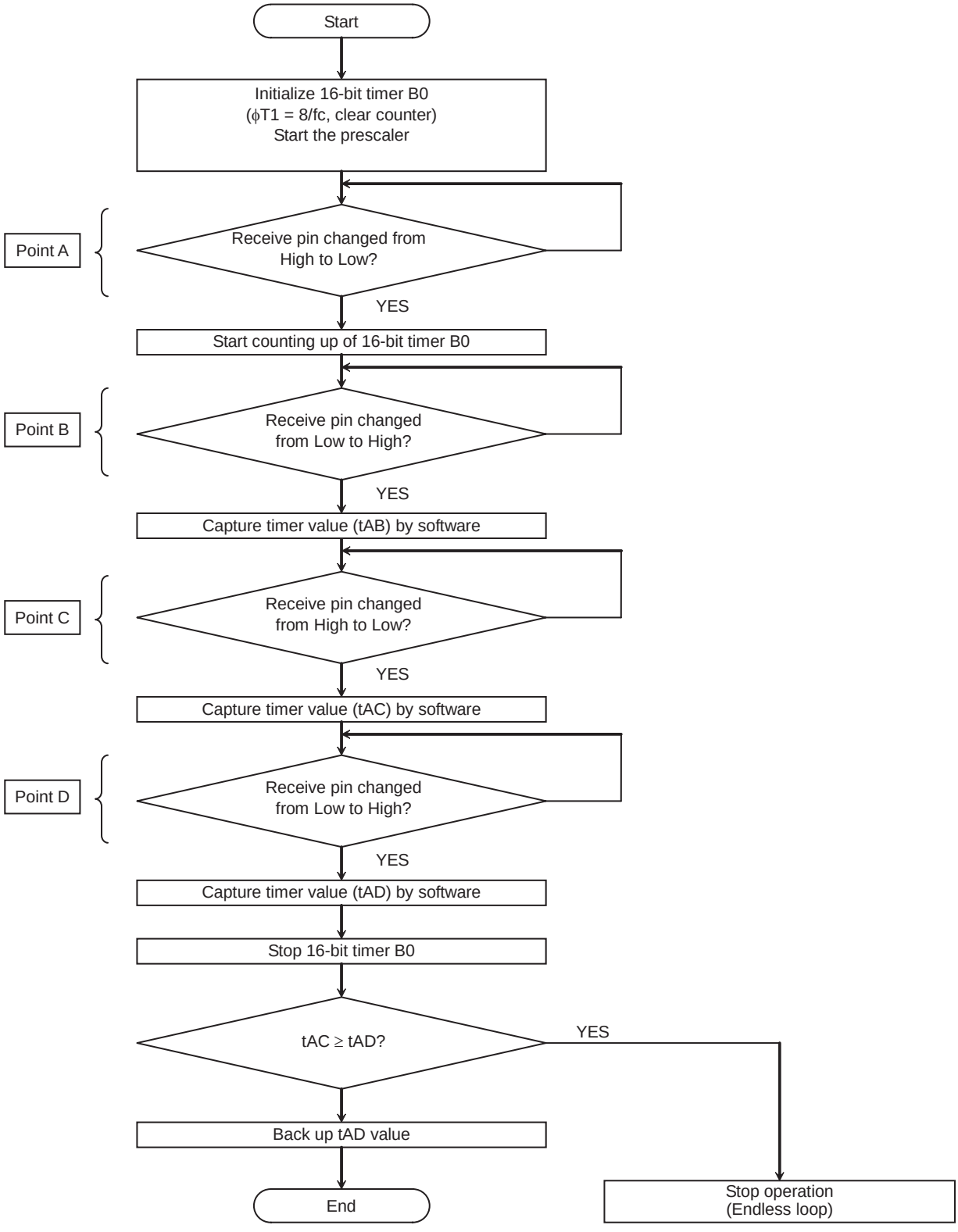


Figure 13-8 Flowchart for Serial Operation Mode Receive Operation

13.4.15Password

When the RAM Transfer command (10H) or the Flash Memory Protect Set command (60H) is received as operation command data, password verification is performed. First, the device echoes back the operation command data (10H to 60H) and checks the data (12 bytes) in the password area (addresses 027EF4H to 027EFFH).

Then, the device verifies the password data received in the 5th to 16th bytes against the data in the password area as shown in Table 13-19.

Unless all the 12 bytes are verified correctly, a password error will occur.

A password error will also occur if all the 12 bytes of password data contain the same value. Only exception is when all the 12 bytes are “FFH” and verified correctly and the reset vector area (addresses 027F00H to 027F02H) is all “FFH”. In this case, a blank device will be assumed and no password error will occur.

If a password error has occurred, the device returns the ACK response data for password error in the 18th byte.

Table 13-19 Password Verification Table

Receive data	Data to be verified against
5th byte	Data at address 027EF4H
6th byte	Data at address 027EF5H
7th byte	Data at address 027EF6H
8th byte	Data at address 027EF7H
9th byte	Data at address 027EF8H
10th byte	Data at address 027EF9H
11th byte	Data at address 027EFAH
12th byte	Data at address 027EFBH
13th byte	Data at address 027EFCH
14th byte	Data at address 027EFDH
15th byte	Data at address 027EFEH
16th byte	Data at address 027EFFH

Example of data that cannot be specified as a password

For blank products (Note)

The password of a blank product must be all “FFH” (FFH, FFH, FFH, FFH, FFH, FFH, FFH, FFH, FFH, FFH, FFH, FFH).

Note:A blank product is a product in which all the bytes in the password area (addresses 02FEF4H to 02FEFFH) and the reset vector area (addresses 02FF00H to 02FF02H) are “FFH”.

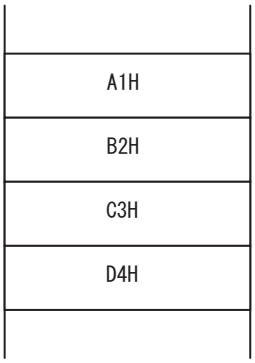
For programmed products

The same 12 consecutive bytes cannot be specified as a password.
The table below shows password error examples.

Programmed product	1	2	3	4	5	6	7	8	9	10	11	12	Note
Error example 1	FFH	FFH	FFH	FFH	FFH	FFH	FFH	FFH	FFH	FFH	FFH	FFH	ALL"FF"
Error example 2	00H	00H	00H	00H	00H	00H	00H	00H	00H	00H	00H	00H	ALL"00"
Error example 3	5AH	5AH	5AH	5AH	5AH	5AH	5AH	5AH	5AH	5AH	5AH	5AH	ALL"5A"

13.4.16How to Calculate SUM

SUM is calculated by summing the values of all data read from the flash memory by unsigned 8-bit addition and is returned as a word (16-bit) value. The resulting SUM value is sent to the controller in order of upper 8 bits and lower 8 bits. All the 96 Kbytes of data in the flash memory are included in the calculation of SUM. When the Flash Memory SUM command is executed, SUM is calculated in this way.



When SUM is calculated from the four data entries shown to the left, the result is as follows:

A1H+ B2H + C3H + D4H = 02EAH
SUM upper 8 bits: 02H
SUM lower 8 bits: EAH

Thus, the SUM value is sent to the controller in order of 02H and EAH.

13.4.17How to Calculate CHECKSUM

CHECKSUM is calculated by taking the two's complement of the lower 8-bit value obtained by summing the values of received data by unsigned 8-bit addition (ignoring any overflow). When the Flash Memory SUM command or the Product Information Read command is executed, CHECKSUM is calculated in this way. The controller should also use this CHECKSUM calculation method for sending CHECKSUM values.

Example: Calculating CHECKSUM for the Flash Memory SUM command

When the upper 8-bit data of SUM is E5H and the lower 8-bit data is F6H, CHECKSUM is calculated as shown below.

First, the upper 8 bits and lower 8 bits of the SUM value are added by unsigned operation.

E5H+F6H = 1DBH

Then, the two's complement of the lower 8 bits of this result is obtained as shown below. The resulting CHECKSUM value (25H) is sent to the controller.

0-DBH = 25H

13.5 User Boot Mode (in Single Chip Mode)

User Boot mode, which is a sub mode of Single Chip mode, enables a user-created flash memory program/erase routine to be used. To do so, the operation mode of Single Chip mode must be changed from Normal mode for executing a user application program to User Boot mode for programming/erasing the flash memory.

For example, the reset processing routine of a user application program may include a routine for selecting Normal mode or User Boot mode upon entering Single Chip mode. Any mode-selecting condition may be set using the device's I/O to suit the user system.

To program/erase the flash memory in User Boot mode, a program/erase routine must be incorporated in the user application program in advance. Since the processor cannot read data from the internal flash memory while it is being programmed or erased, the program/erase routine must be executed from the outside of the flash memory. While the flash memory is being programmed/erased in User Boot mode, interrupts must be disabled.

The pages that follow explain the procedure for programming the flash memory using two example cases. In one case the program/erase routine is stored in the internal flash memory (1-A); in the other the program/erase routine is transferred from an external source (1-B).

13.5.1 (1-A) Program/Erase Procedure Example 1

When the program/erase routine is stored in the internal flash memory

(Step-1)Environment setup

First, the condition (e.g. pin status) for entering User Boot mode must be set and the I/O bus for transferring data must be determined. Then, the device's peripheral circuitry must be designed and a corresponding program must be written. Before mounting the device on the board, it is necessary to write the following four routines into one of the sectors in the flash memory.

(a)Mode select routine:

Selects Normal mode or User Boot mode.

(b)Program/erase routine:

Loads program/erase data from an external source and programs/erases the flash memory.

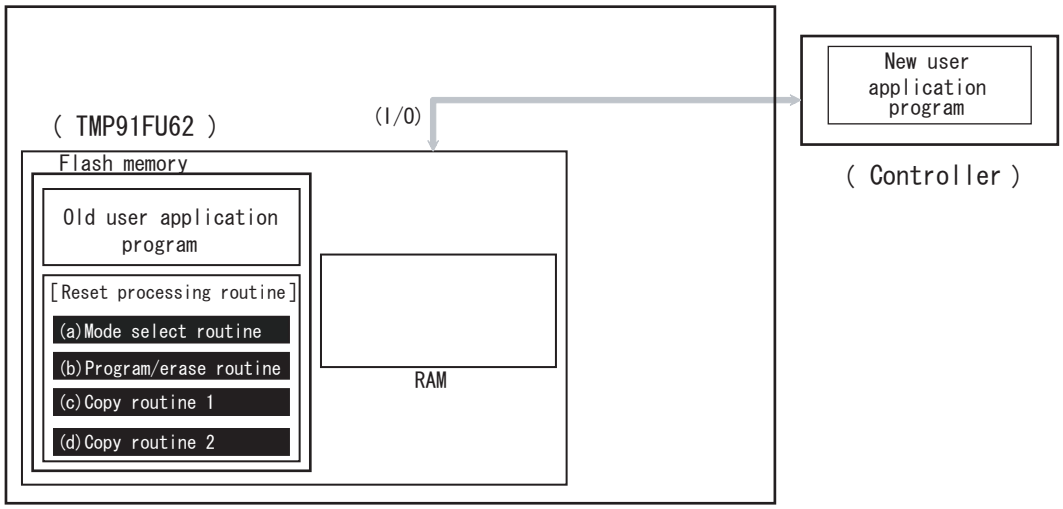
(c)Copy routine 1:

Copies routines (a) to (d) into the internal RAM or external memory.

(d)Copy routine 2:

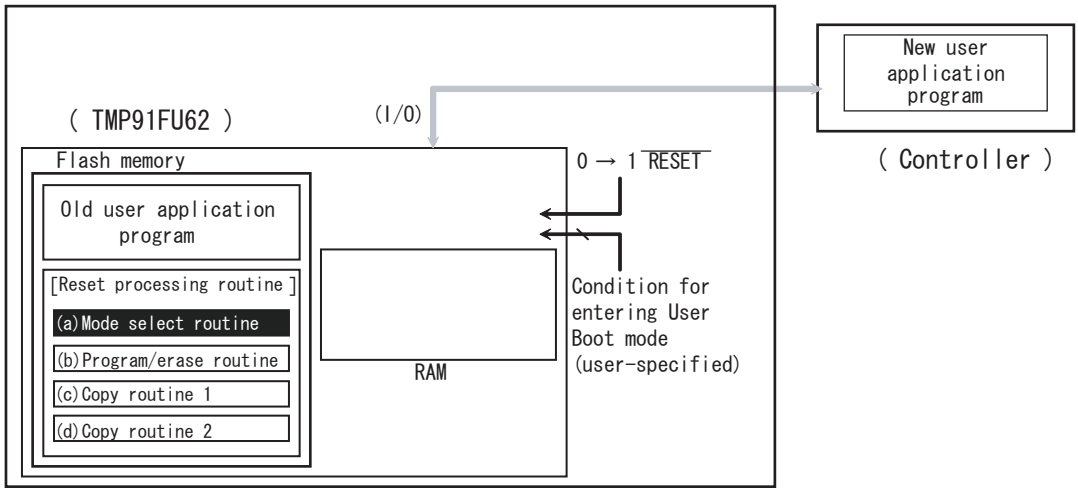
Copies routines (a) to (d) from the internal RAM or external memory into the flash memory.

Note: The above (d) is a routine for reconstructing the program/erase routine on the flash memory. If the entire flash memory is always programmed and the program/erase routine is included in the new user application program, this copy routine is not needed.



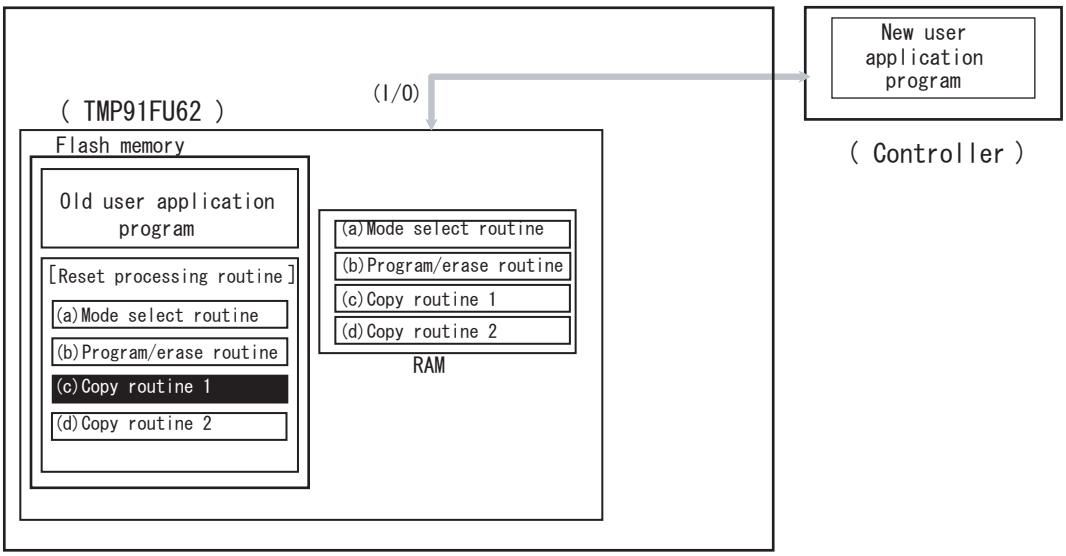
(Step-2) Entering User Boot mode (using the reset processing)

After reset release, the reset processing program determines whether or not the device should enter User Boot mode. If the condition for entering User Boot mode is true, User Boot mode is entered to program/erase the flash memory.



(Step-3) Copying the program/erase routine

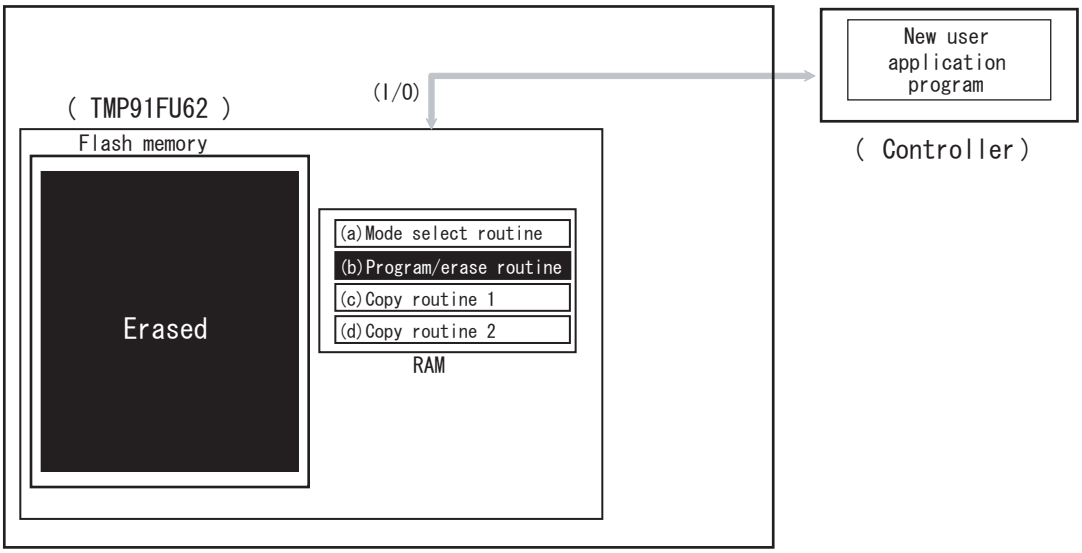
After the device has entered User Boot mode, the copy routine 1 (c) copies the routines (a) to (d) into the internal RAM or external memory (The routines are copied into the internal RAM here.)



(Step-4) Erasing the flash memory by the program/erase routine

Control jumps to the program/erase routine in the RAM and the old user program area is erased (sector erase or chip erase). (In this case, the flash memory erase command is issued from the RAM.)

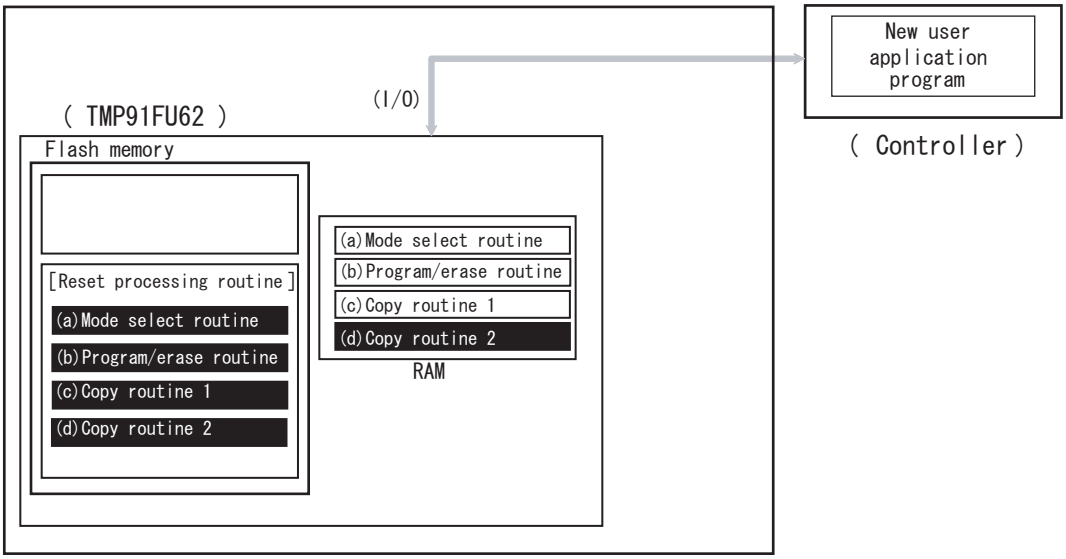
Note: If data is erased on a sector basis and the routines (a) to (d) are left in the flash memory, only the program/ erase routine (b) need be copied into the RAM.



(Step-5) Restoring the user boot program in the flash memory

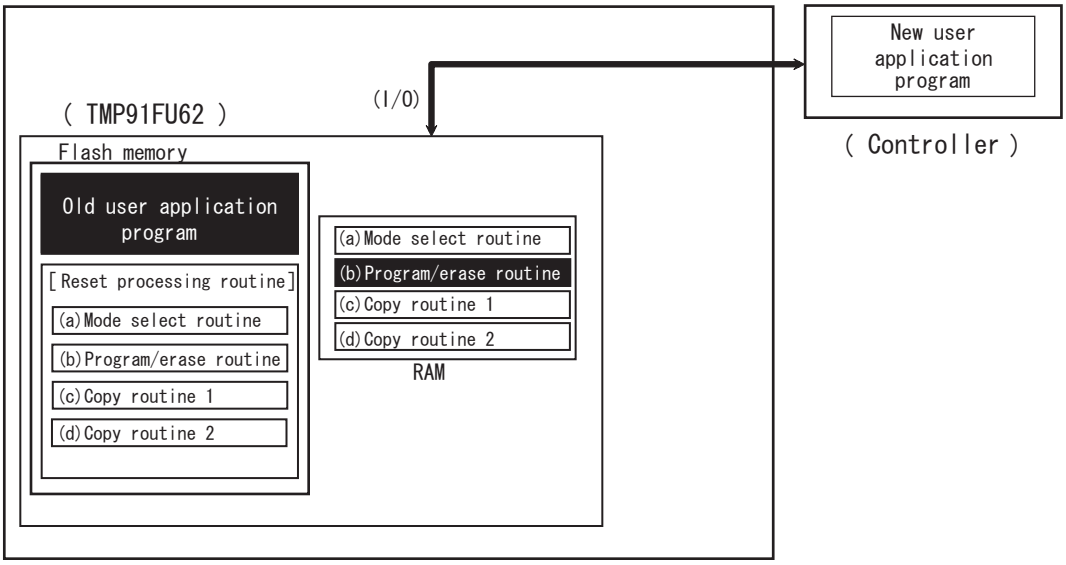
The copy routine 2 (d) in the RAM copies the routines (a) to (d) into the flash memory.

Note: If data is erased on a sector basis and the routines (a) to (d) are left in the flash memory, step 5 is not needed.



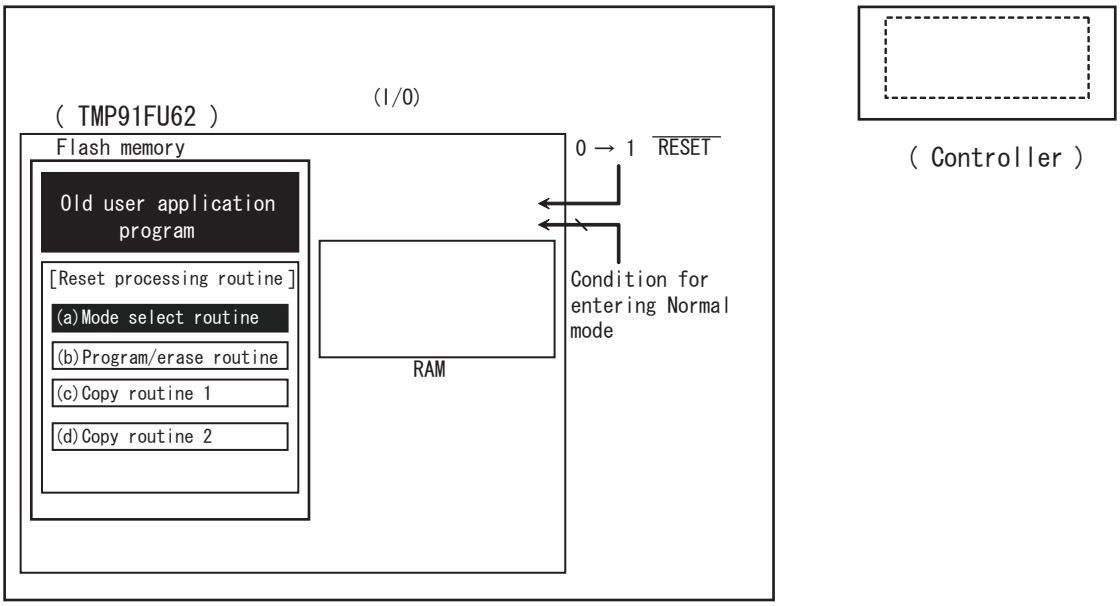
(Step-6) Writing the new user application program to the flash memory

The program/erase routine in the RAM is executed to load the new user application program from the controller into the erased area of the flash memory.



(Step-7) Executing the new user application program

The $\overline{\text{RESET}}$ input pin is driven Low (“0”) to reset the device. The mode setting condition is set for Normal mode. After reset release, the device will start executing the new user application program.



13.5.2 (1-B) Program/Erase Procedure Example 2

In this example, only the boot program (minimum requirement) is stored in the flash memory and other necessary routines are supplied from the controller.

(Step-1)Environment setup

First, the condition (e.g. pin status) for entering User Boot mode must be set and the I/O bus for transferring data must be determined. Then, the device's peripheral circuitry must be designed and a corresponding program must be written. Before mounting the device on the board, it is necessary to write the following two routines into one on the sectors in the flash memory.

(a)Mode select routine:

Selects Normal mode or User Boot mode.

(b)Transfer routine:

Loads the program/erase routine from an external source.

The following routines are prepared on the controller.

(c)Program/erase routine:

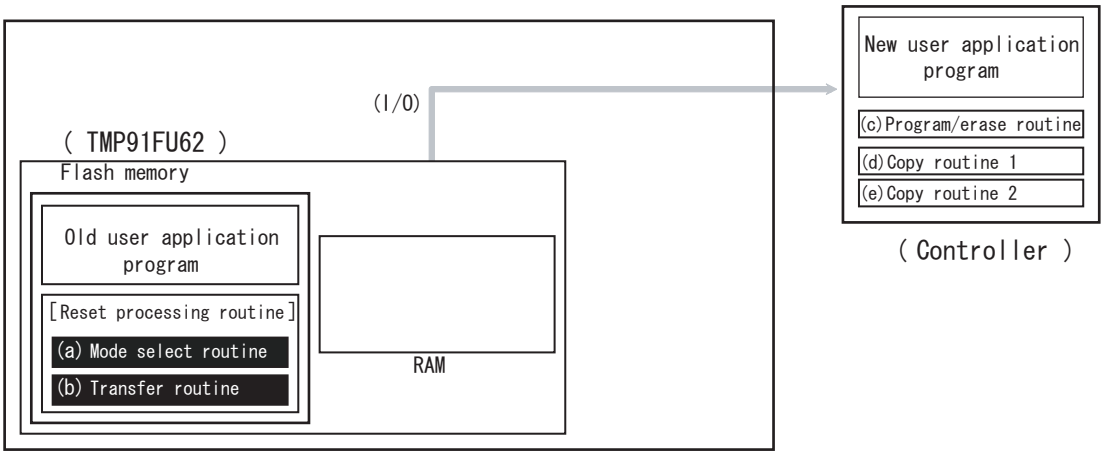
Programs/erases the flash memory.

(d)Copy routine 1:

Copies routines (a) and (b) into the internal RAM or external memory.

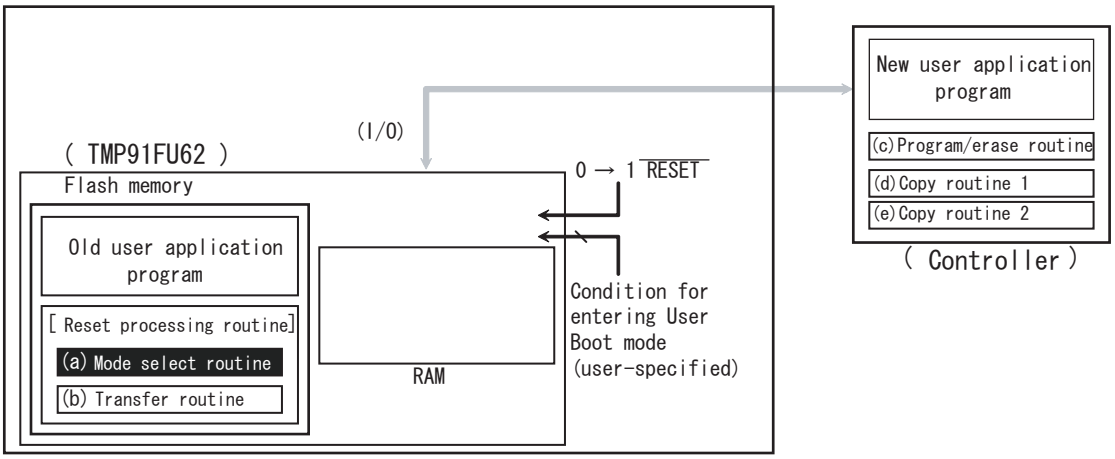
(e)Copy routine 2:

Copies routines (a) and (b) from the internal RAM or external memory into the flash memory.



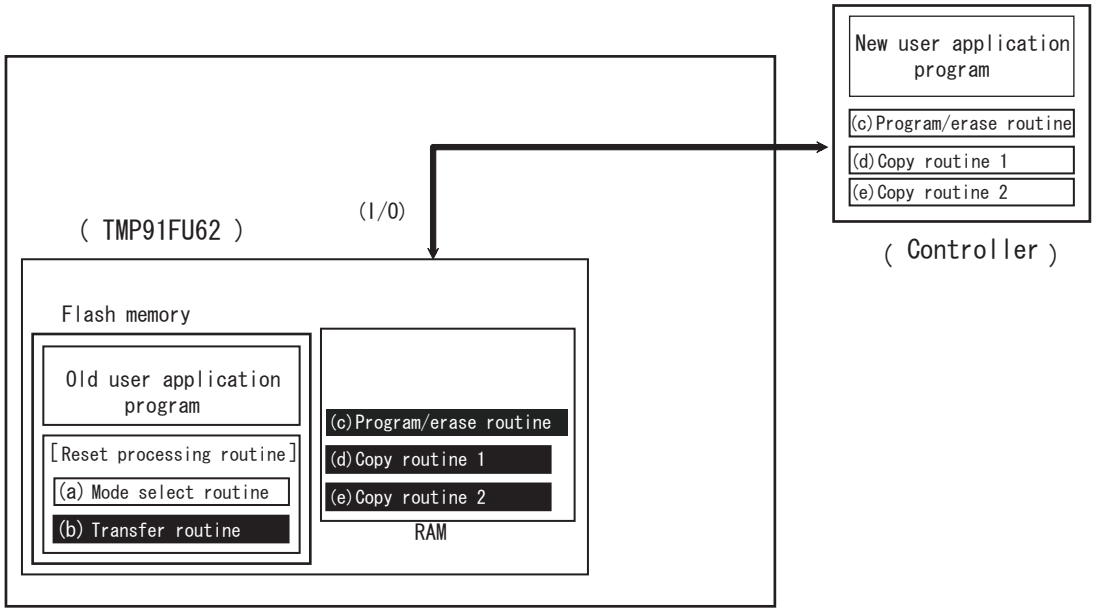
(Step-2) Entering User Boot mode (using the reset processing)

The following explanation assumes that these routines are incorporated in the reset processing program. After reset release, the reset processing program first determines whether or not the device should enter User Boot mode. If the condition for entering User Boot mode is true, User Boot mode is entered to program/erase the flash memory.



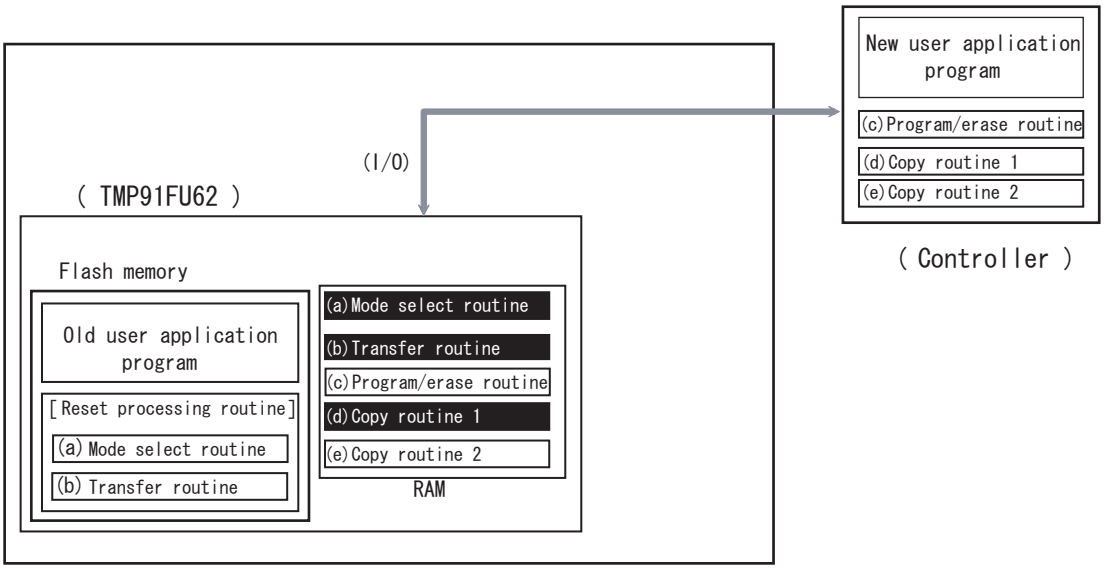
(Step-3) Copying the program/erase routine to the internal RAM

After the device has entered User Boot mode, the transfer routine (b) transfers the routines (c) to (e) from the controller to the internal RAM (or external memory). (The routines are copied into the internal RAM here.)



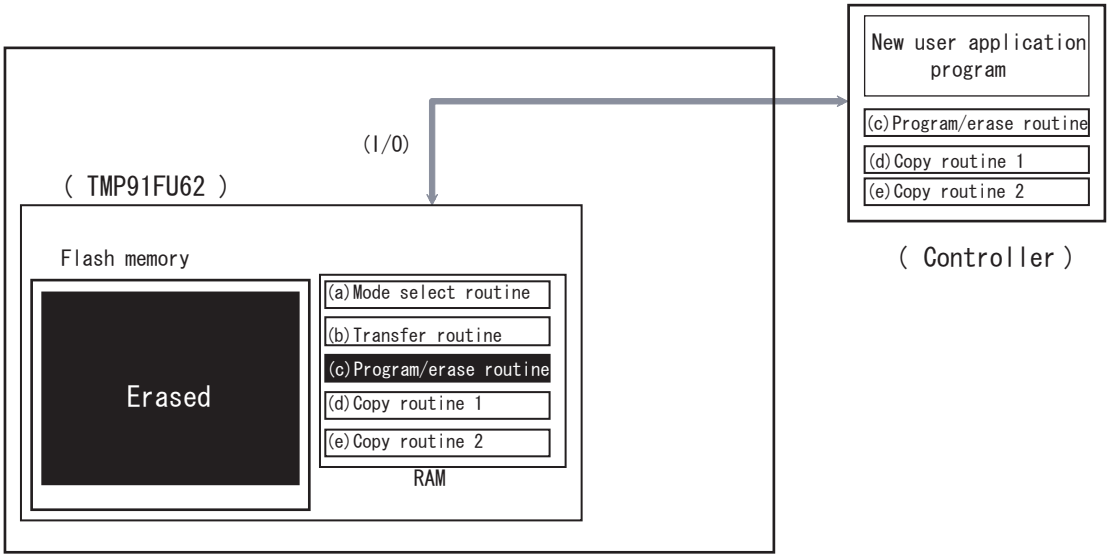
(Step-4) Executing the copy routine 1 in the internal RAM

Control jumps to the internal RAM and the copy routine 1 (d) copies the routines (a) and (b) into the internal RAM.



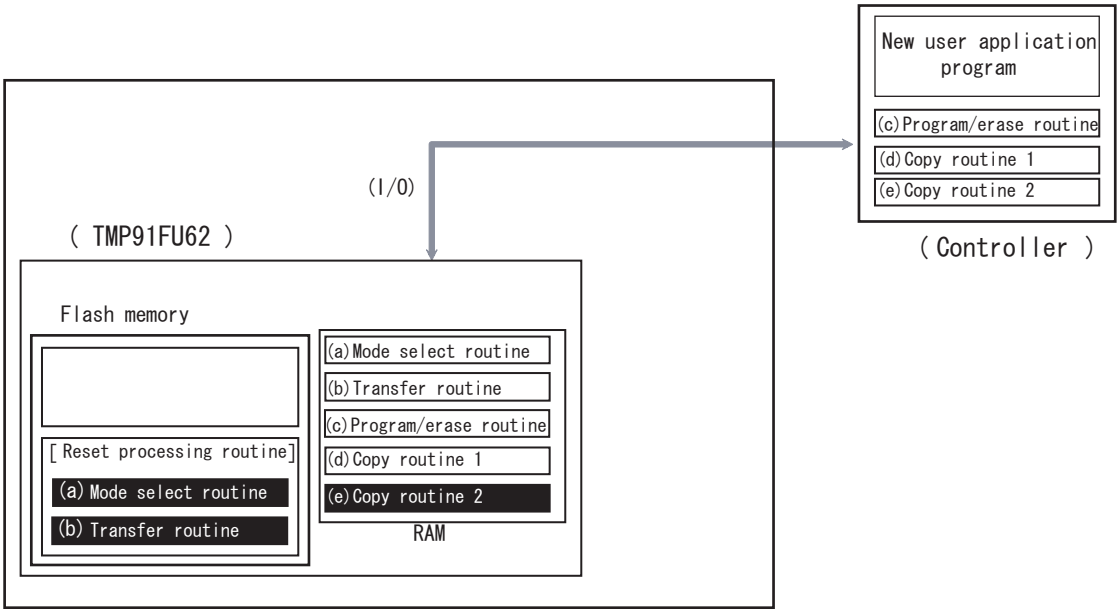
(Step-5) Erasing the flash memory by the program/erase routine

The program/erase routine (c) erases the old user program area.



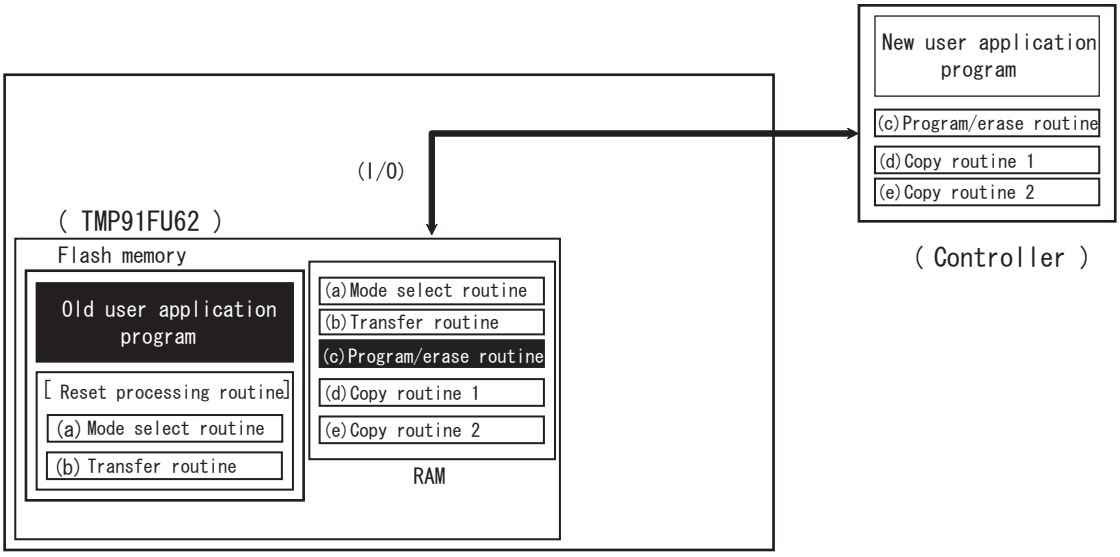
(Step-6) Restoring the user boot program in the flash memory

The copy routine (e) copies the routines (a) and (b) from the internal RAM into the flash memory.



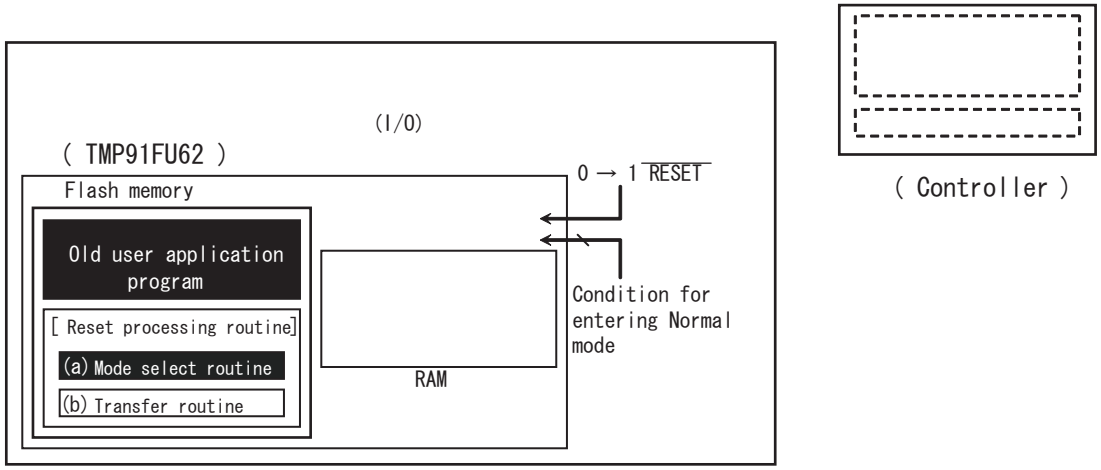
(Step-7) Writing the new user application program to the flash memory

The program/erase routine (c) in the RAM is executed to load the new user application program from the controller into the erased area of the flash memory.



(Step-8) Executing the new user application program

The $\overline{\text{RESET}}$ input pin is driven Low (“0”) to reset the device. The mode setting condition is set for Normal mode. After reset release, the device will start executing the new user application program.



13.6 Flash Memory Command Sequences

The operation of the flash memory is comprised of six commands, as shown in Table 13-20. Addresses specified in each command sequence must be in an area where the flash memory is mapped. For details, see Table 13-3.

Table 13-20 Command Sequences

	Command	1st Bus Write Cycle		2nd Bus Write Cycle		3rd Bus Write Cycle		4th Bus Write Cycle		5th Bus Write Cycle		6th Bus Write Cycle	
	Sequence	Addr.	Data	Addr.	Data	Addr.	Data	Addr.	Data	Addr.	Data	Addr.	Data
1	Single Word Program	AAAH	AAH	554H	55H	AAAH	A0H	PA (Note1)	PD (Note1)				
2	Sector Erase (8KB Erase)	AAAH	AAH	554H	55H	AAAH	80H	AAAH	AAH	554H	55H	SA (Note2)	30H
3	Chip Erase (All Erase)	AAAH	AAH	554H	55H	AAAH	80H	AAAH	AAH	554H	55H	AAAH	10H
4	Product ID Entry	AAAH	AAH	554H	55H	AAAH	90H						
5	Product ID Exit	xxH	F0H										
	Product ID Exit	AAAH	AAH	554H	55H	AAAH	F0H						
6	Read Protect Set	AAAH	AAH	554H	55H	AAAH	A5H	77EH	F0H (Note3)				
	Write Protect Set	AAAH	AAH	554H	55H	AAAH	A5H	77EH	0FH (Note3)				

Note 1: PA = Program Word address, PD = Program Word data
Set the address and data to be programmed. Even-numbered addresses should be specified here.

Note 2: SA = Sector Erase address, Each sector erase range is selected by address A23 to A13.

Note 3: When apply read protect and write protect, be sure to program the data of 00H.

Table 13-21 Hardware Sequence Flags

Status		D7	D6
During auto operation	Single Word Program	$\overline{D7}$	Toggle
	Sector Erase/Chip Erase	0	Toggle
	Read Protect Set/Write Protect Set	Cannot be used	Toggle

Note: D15 to D8 and D5 to D0 are “don't care”.

13.6.1 Single Word Program

The Single Word Program command sequence programs the flash memory on a word basis. The address and data to be programmed are specified in the 4th bus write cycle. It takes a maximum of 60 us to program a single word. Another command sequence cannot be executed until the write operation has completed. This can be checked by reading the same address in the flash memory repeatedly until the same data is read consecutively. While a write operation is in progress, bit 6 of data is toggled each time it is read.

Note: To rewrite data to Flash memory addresses at which data (including FFFFH) is already written, make sure to erase the existing data by “sector erase” or “chip erase” before rewriting data.

13.6.2 Sector Erase (8-Kbyte Erase)

The Sector Erase command sequence erases 8 Kbytes of data in the flash memory at a time. The flash memory address range to be erased is specified in the 6th bus write cycle. For the address range of each sector, see Table 13-3. This command sequence cannot be used in Programmer mode.

It takes a maximum of 75 ms to erase 8 Kbytes. Another command sequence cannot be executed until the erase operation has completed. This can be checked by reading the same address in the flash memory repeatedly until the same data is read consecutively. While a erase operation is in progress, bit 6 of data is toggled each time it is read.

13.6.3 Chip Erase (All Erase)

The Chip Erase command sequence erases the entire area of the flash memory.

It takes a maximum of 300 ms to erase the entire flash memory. Another command sequence cannot be executed until the erase operation has completed. This can be checked by reading the same address in the flash memory repeatedly until the same data is read consecutively. While a erase operation is in progress, bit 6 of data is toggled each time it is read.

Erase operations clear data to FFH.

13.6.4 Product ID Entry

When the Product ID Entry command is executed, Product ID mode is entered. In this mode, the vendor ID, flash macro ID, flash size ID, and read/write protect status can be read from the flash memory. In Product ID mode, the data in the flash memory cannot be read.

13.6.5 Product ID Exit

This command sequence is used to exit Product ID mode.

13.6.6 Read Protect Set

The Read Protect Set command sequence applies read protection on the flash memory. When read protection is applied, the flash memory cannot be read in Programmer mode and the RAM Transfer command cannot be executed in Single Boot mode.

To cancel read protection, it is necessary to execute the Chip Erase command sequence. To check whether or not read protection is applied, read xxx77EH in Product ID mode. It takes a maximum of 60 us to set read protection on the flash memory. Another command sequence cannot be executed until the read protection setting has completed. This can be checked by reading the same address in the flash memory repeatedly until the same data can be read consecutively. While a read protect operation is in progress, bit 6 of data is toggled each time it is read.

13.6.7 Write Protect Set

The Write Protect Set command sequence applies write protection on the flash memory. When write protection is applied, the flash memory cannot be written to in Programmer mode and the RAM Transfer command cannot be executed in Single Boot mode.

To cancel write protection, it is necessary to execute the Chip Erase command sequence. To check whether or not write protection is applied, read xxx77EH in Product ID mode. It takes a maximum of 60 us to set write protection. Another command sequence cannot be executed until the write protection setting has completed. This can be checked by reading the same address in the flash memory repeatedly until the same data can be read consecutively. While a write protect operation is in progress, bit 6 of data is toggled each time it is read.

13.6.8 Hardware Sequence Flags

The following hardware sequence flags are available to check the auto operation execution status of the flash memory.

1. Data polling (D7)

   When data is written to the flash memory, D7 outputs the complement of its programmed data until the write operation has completed. After the write operation has completed, D7 outputs the proper cell data. By reading D7, therefore, the operation status can be checked. While the Sector Erase or Chip Erase command sequence is being executed, D7 outputs “0”. After the command sequence is completed, D7 outputs “1” (cell data). Then, the data written to all the bits can be read after waiting for 1 us.

   When read/write protection is applied, the data polling function cannot be used. Instead, use the toggle bit (D6) to check the operation status.

2. Toggle bit (D6)

   When the Flash Memory Program, Sector Erase, Chip Erase, Write Protect Set, or Read Protect Set command sequence is executed, bit 6 (D6) of the data read by read operations outputs “0” and “1” alternately each time it is read until the processing of the executed command sequence has completed. The toggle bit (D6) thus provides a software means of checking whether or not the processing of each command sequence has completed. Normally, the same address in the flash memory is read repeatedly until the same data is read successively. The initial read of the toggle bit always returns “1”.

Note: The flash memory incorporated in the TMP91FU62 does not have an exceed-time-limit bit (D5). It is therefore necessary to set the data polling time limit and toggle bit polling time limit so that polling can be stopped if the time limit is exceeded.

13.6.9 Data Read

Data is read from the flash memory in byte units or word units. It is not necessary to execute a command sequence to read data from the flash memory.

13.6.10 Programming the Flash Memory by the Internal CPU

The internal CPU programs the flash memory by using the command sequences and hardware sequence flags described above. However, since the flash memory cannot be read during auto operation mode, the program/erase routine must be executed outside of the flash memory.

The CPU can program the flash memory either by using Single Boot mode or by using a user-created protocol in Single Chip mode (User Boot).

1. Single Boot:

   The microcontroller is started up in Single Boot mode to program the flash memory by the internal boot ROM program. In this mode, the internal boot ROM is mapped to an area including the interrupt vector table, in which the boot ROM program is executed. The flash memory is mapped to an address area different from the boot ROM area. The boot ROM program loads data into the flash memory by serial transfer. In Single Boot mode, interrupts must be disabled including non-maskable interrupts.

   For details, see " 13.4 Single Boot Mode "

2. User Boot:

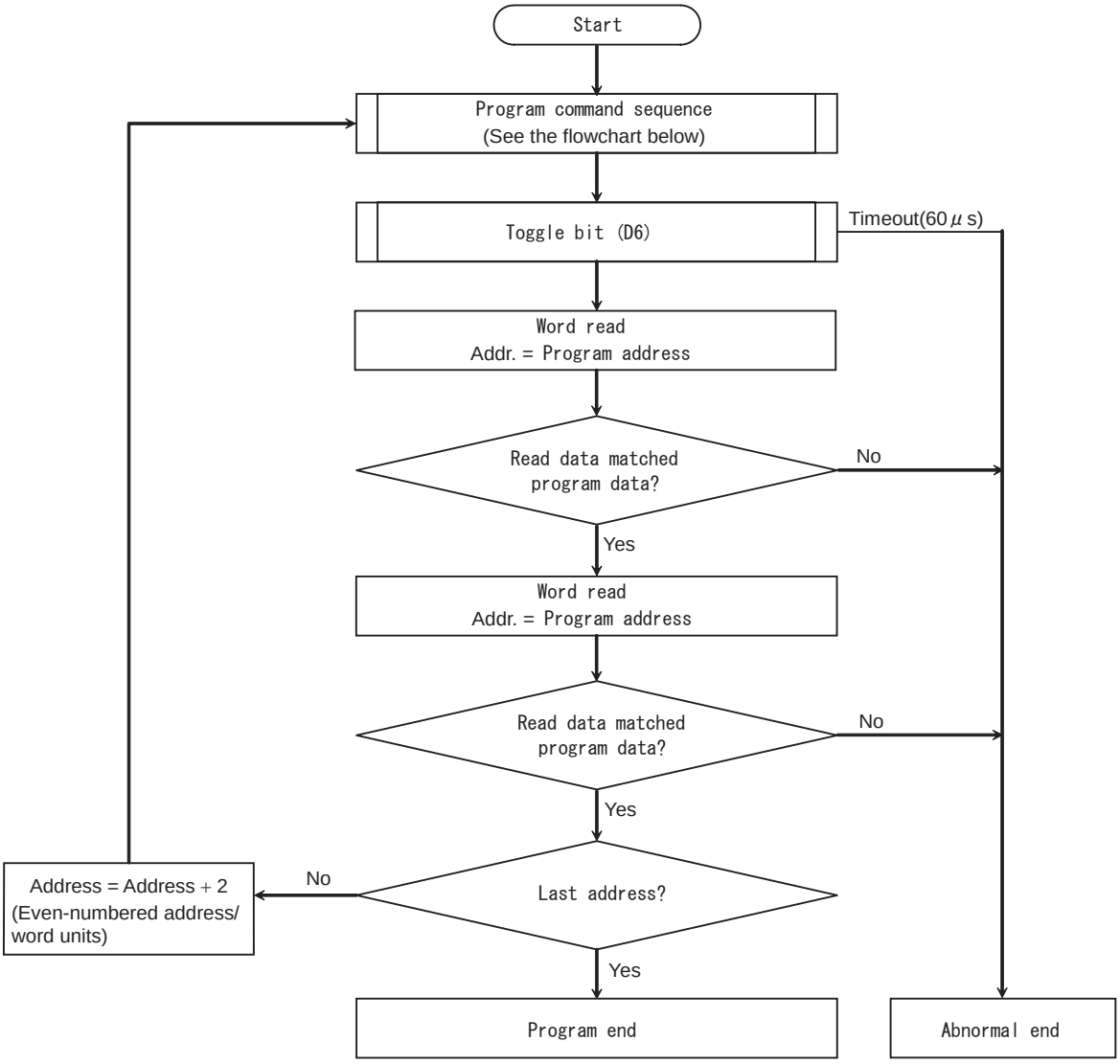
   In this method, the flash memory is programmed by executing a user-created routine in Single Chip mode (normal operation mode). In this mode, the user-created program/erase routine must also be executed outside of the flash memory. It is also necessary to disable interrupts including non-maskable interrupts.

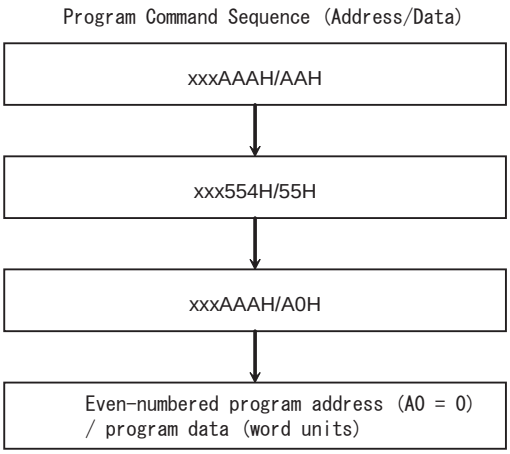
   The user should prepare a flash memory program/erase routine (including routines for loading write data and writing the loaded data into the flash memory). In the main program, normal operation is switched to flash memory programming operation to execute the flash memory program/erase routine outside of the flash memory area. For example, the flash memory program/erase routine may be transferred from the flash memory to the internal RAM and executed there or it may be prepared and executed in external memory.

   For details, see " 13.5 User Boot Mode (in Single Chip Mode) ".

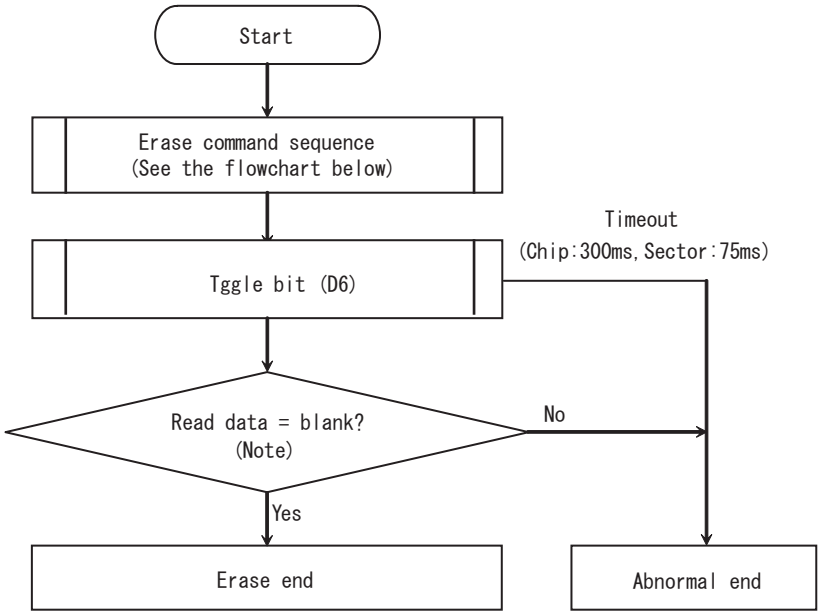
Flowcharts: Flash memory access by the internal CPU

Single Word Program

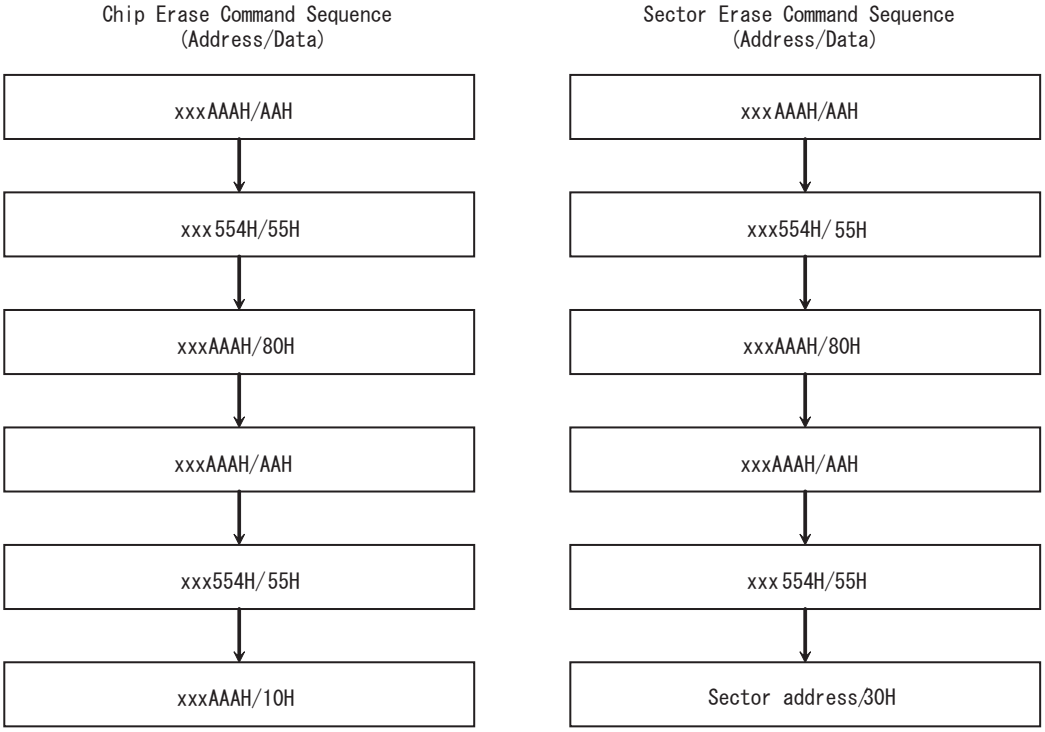




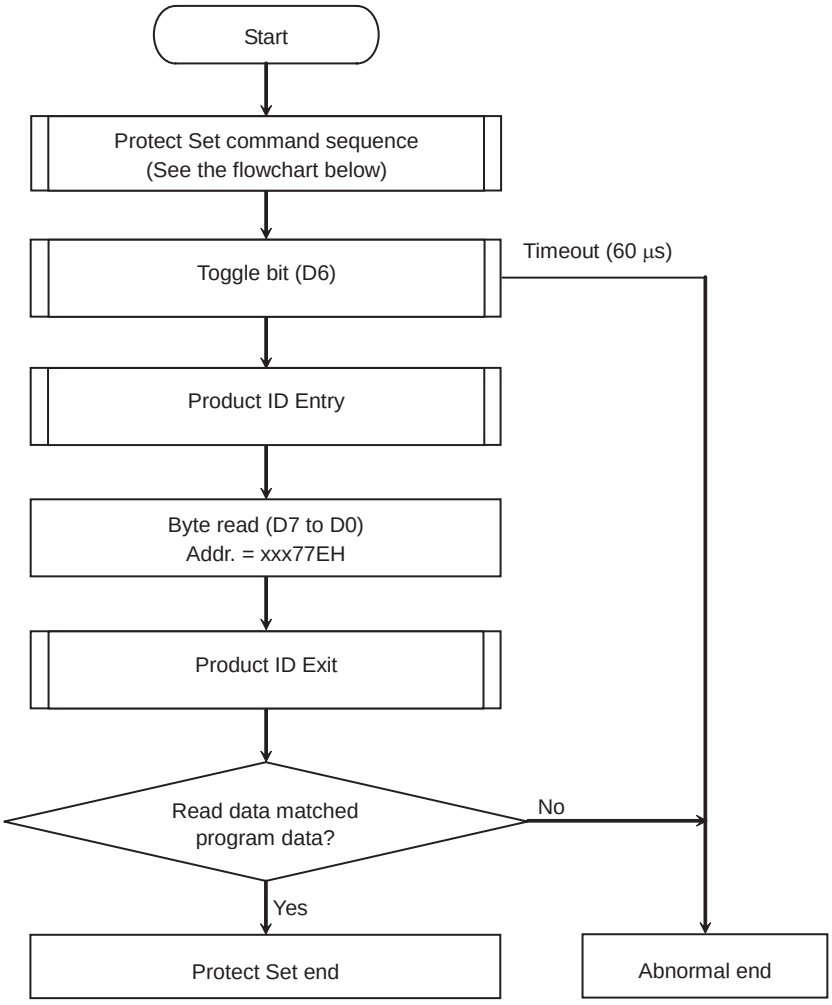
Chip Erase/Sector Erase



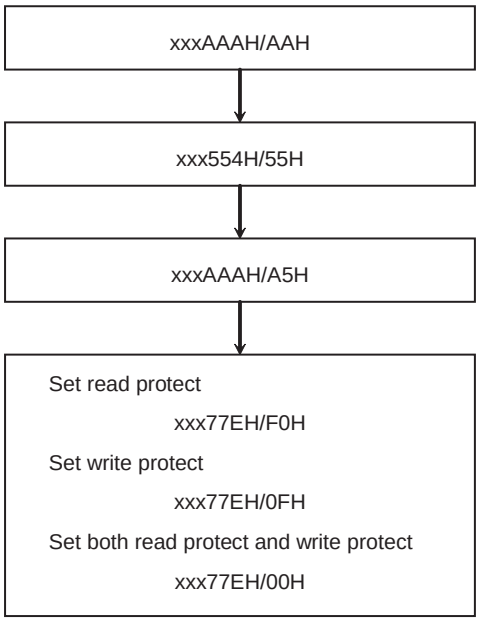
Note: In Chip Erase, whether or not the entire flash memory is blank is checked.
In Sector Erase, whether or not the selected sector is blank is checked.



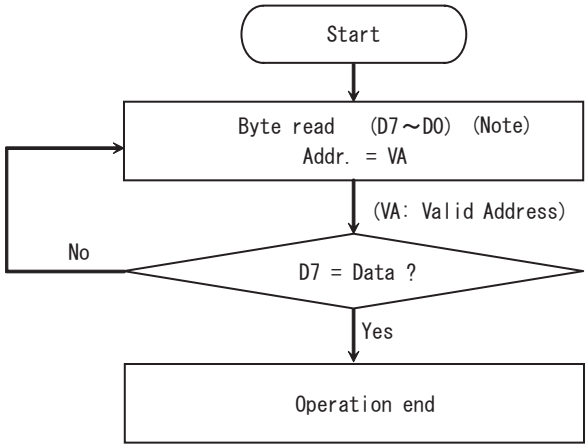
Read/Write Protect Set



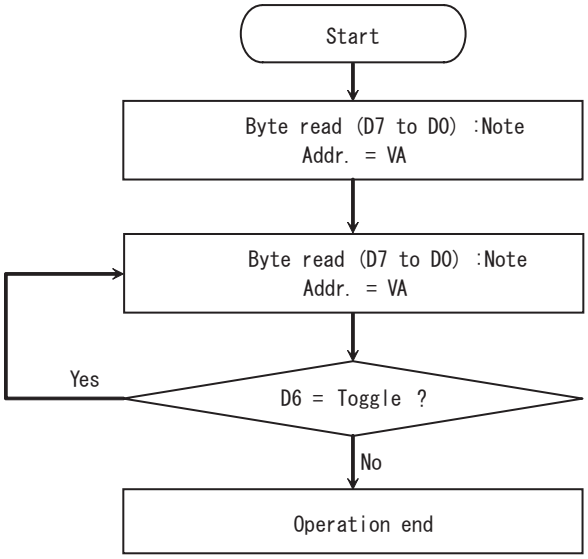
Protect Set Command Sequence
(Address/Data)



Data Polling (D7)



Toggle Bit (D6)



Note: Hardware sequence flags are read from the flash memory in byte units or word units.

VA: In Single Word Program, VA denotes the address to be programmed.

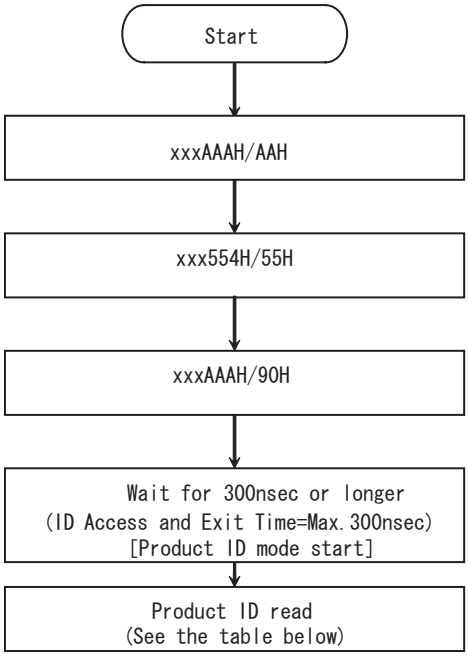
In Sector Erase, VA denotes any address in the selected sector.

In Chip Erase, VA denotes any address in the flash memory.

In Read Protect Set, VA denotes the protect set address (xx77EH).

In Write Protect Set, VA denotes the protect set address (xx77EH).

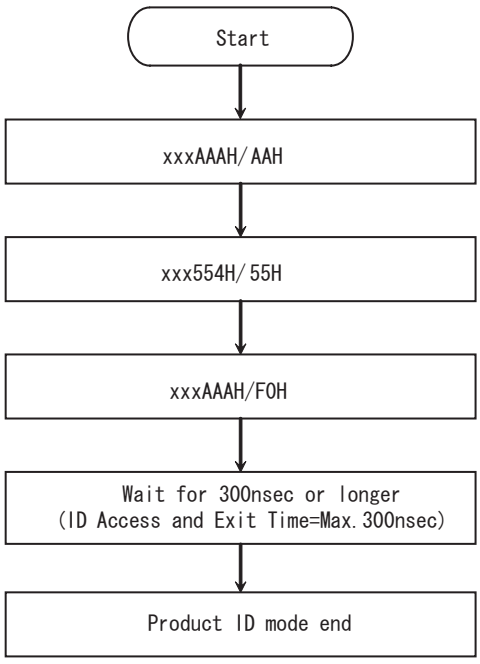
Product ID Entry

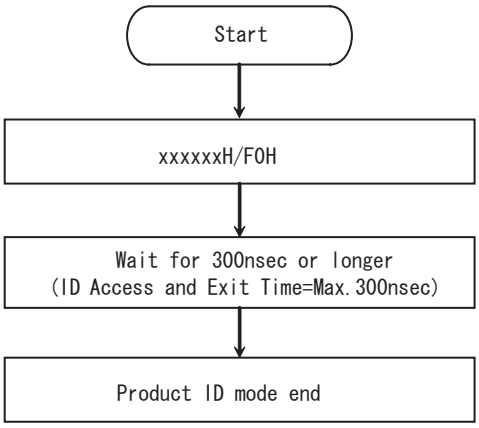


Read Values in Product ID Mode

	Address	Read Value
Vendor ID	xxxx00H	98H
Flash macro ID	xxxx02H	42H
Flash size ID	xxxx04H	17H
Read/Write Protect status	xxx77EH	Data programmed when protection is set. When protection is not set, FFH.

Product ID Exit





(Example: Program to be loaded and executed in RAM)

Erase the flash memory (chip erase) and then write 0706H to address FE8000H.

```
;#### Flash memory chip erase processing ####
    ld      XIX, 0xFE8000               ; set start address
CHIPERASE:
    ld      (0xFE8AAA), 0xAA            ;1st Bus Write Cycle
    ld      (0xFE8554), 0x55            ;2nd Bus Write Cycle
    ld      (0xFE8AAA), 0x80            ;3rd Bus Write Cycle
    ld      (0xFE8AAA), 0xAA            ;4th Bus Write Cycle
    ld      (0xFE8554), 0x55            ;5th Bus Write Cycle
    ld      (0xFE8AAA), 0x10            ;6th Bus Write Cycle

    cal     TOGGLECHK                   ; check toggle bit

CHIPERASE _ LOOP:
    ld      WA, (XIX+)                  ; read data from flash memory
    cp      WA, 0xFFFF                  ; blank data?
    j       ne,CHIPERASE _ ERR          ; if not blank data, jump to error processing
    cp      XIX, 0xFFFFFF               ; end address (0xFFFFFF)?
    j       ULT,CHIPERASE _ LOOP        ; check entire memory area and then end loop processing


;#### Flash memory program processing ####
    ld      XIX, 0xFE8000               ; set program address
    ld      WA, 0x0706                  ; set program data
PROGRAM:
    ld      (0xFE8AAA), 0xAA            ;1st Bus Write Cycle
    ld      (0xFE8554), 0x55            ;2nd Bus Write Cycle
    ld      (0xFE8AAA), 0xA0            ;3rd Bus Write Cycle
    ld      (XIX), WA                   ;4th Bus Write Cycle

    cal     TOGGLECHK                   ; check toggle bit

    ld      BC, (XIX)                   ; read data from flash memory
    cp      WA, BC
    j       ne, PROGRAM _ ERR           ; if programmed data cannot be read, error is determined
    ld      BC, (XIX)                   ; read data from flash memory
    cp      WA, BC
    j       ne, PROGRAM _ ERR           ; if programmed data cannot be read, error is determined

PROGRAM _ END:
    j       PROGRAM _ END               ; program operation end
```

```
;#### Toggle bit (D6) check processing ####
TOGGLECHK:
    ld      L, (XIX)
    and     L, 0y01000000           ; check toggle bit (D6)
    ld      H, L                    ; save first toggle bit data
TOGGLECHK1:
    ld      L, (XIX)
    and     L, 0y01000000           ; check toggle bit (D6)
    cp      L, H                    ; toggle bit = toggled?
    j       z, TOGGLECHK2           ; if not toggled, end processing
    ld      H, L                    ; save current toggle bit state
    j       TOGGLECHK1              ; recheck toggle bit
TOGGLECHK2:
    ret


;####  Error processing ####
CHIPERASE _ ERR:
    j       CHIPERASE _ ERR         ; chip erase error

PROGRAM _ ERR:
    j       PROGRAM _ ERR           ; program error
```

(Example: Program to be loaded and executed in RAM)

Erase data at addresses FF0000H to FF1FFFH (sector erase) and then write 0706H to address FF0000H.

```
;#### Flash memory sector erase processing ####
    ld      XIX, 0xFF0000               ; set start address
SECTORERASE:
    ld      (0xFE8AAA), 0xAA            ;1st Bus Write Cycle
    ld      (0xFE8554), 0x55            ;2nd Bus Write Cycle
    ld      (0xFE8AAA), 0x80            ;3rd Bus Write Cycle
    ld      (0xFE8AAA), 0xAA            ;4th Bus Write Cycle
    ld      (0xFE8554), 0x55            ;5th Bus Write Cycle
    ld      (XIX), 0x30                 ;6th Bus Write Cycle

    cal     TOGGLECHK                   ; check toggle bit

SECTORERASE _ LOOP:
    ld      WA, (XIX+)                  ; read data from flash memory
    cp      WA, 0xFFFF                  ; blank data?
    j       ne,SECTORERASE _ ERR        ; if not blank data, jump to error processing
    cp      XIX, 0xFF1FFF               ; end address (0xFF1FFF)?
    j       ULT,SECTORERASE _ LOOP      ; check erased sector area and then end loop processing


;#### Flash memory program processing ####
    ld      XIX, 0xFF0000               ; set program address
    ld      WA, 0x0706                  ; set program data
PROGRAM:
    ld      (0xFE8AAA), 0xAA            ;1st Bus Write Cycle
    ld      (0xFE8554), 0x55            ;2nd Bus Write Cycle
    ld      (0xFE8AAA), 0xA0            ;3rd Bus Write Cycle
    ld      (XIX), WA                   ;4th Bus Write Cycle

    cal     TOGGLECHK                   ; check toggle bit

    ld      BC, (XIX)                   ; read data from flash memory
    cp      WA, BC
    j       ne, PROGRAM _ ERR           ; if programmed data cannot be read, error is determined
    ld      BC, (XIX)                   ; read data from flash memory
    cp      WA, BC
    j       ne, PROGRAM _ ERR           ; if programmed data cannot be read, error is determined

PROGRAM _ END:
    j       PROGRAM _ END               ; program operation end
```

```
;#### Toggle bit (D6) check processing ####
TOGGLECHK:
    ld      L, (XIX)
    and     L, 0y01000000           ; check toggle bit (D6)
    ld      H, L                    ; save first toggle bit data
TOGGLECHK1:
    ld      L, (XIX)
    and     L, 0y01000000           ; check toggle bit (D6)
    cp      L, H                    ; toggle bit = toggled?
    j       z, TOGGLECHK2           ; If not toggled, end processing
    ld      H, L                    ; save current toggle bit state
    j       TOGGLECHK1              ; Recheck toggle bit
TOGGLECHK2:
    ret


;#### Error processing ####
SECTORERASE _ ERR:
    j       SECTORERASE _ ERR       ;  sector erase error

PROGRAM _ ERR:
    j       PROGRAM _ ERR           ; program error
```

(Example: Program to be loaded and executed in RAM)

Set read protection and write protection on the flash memory.

```
;#### Flash Memory Protect Set processing ####
    ld      XIX, 0xFE877E               ; set protect address
PROTECT:
    ld      (0xFE8AAA), 0xAA            ;1st Bus Write Cycle
    ld      (0xFE8554), 0x55            ;2nd Bus Write Cycle
    ld      (0xFE8AAA), 0xA5            ;3rd Bus Write Cycle
    ld      (XIX), 0x00                 ;4th Bus Write Cycle

    cal     TOGGLECHK                   ; check toggle bit
    cal     PID _ ENTRY                 ;
    ld      A, (XIX)                    ; read protected address
    cal     PID _ EXIT                  ;
    cp      A, 0x00                     ;(0xFE877E)=0x00?
    j       ne, PROTECT _ ERR           ; protected?

PROTECT _ END:
    j       PROTECT _ END               ; protect set operation completed

PROTECT _ ERR:
    j       PROTECT _ ERR               ; protect set error


;#### Product ID Entry processing ####
PID _ ENTRY:
    ld      (0xFE8AAA), 0xAA            ;1st Bus Write Cycle
    ld      (0xFE8554), 0x55            ;2nd Bus Write Cycle
    ld      (0xFE8AAA), 0x90            ;3rd Bus Write Cycle
; --- wait for 300 nsec or longer (execute NOP instruction [200nsec/@fFPH=20MHz] two times) ---
    nop
    nop                                 ; wait for 400 nsec
    ret


;#### Product ID Exit processing ####
PID _ EXIT:
    ld      (0xFE8000), 0xF0            ;1st Bus Write Cycle
; --- wait for 300 nsec or longer (execute NOP instruction [200nsec/@fFPH=20MHz] two times) ---
    nop
    nop                                 ; wait for 400 nsec
    ret
```

```
;####  Toggle bit (D6) check processing ####
TOGGLECHK:
    ld      L, (XIX)
    and     L, 0y01000000           ; check toggle bit (D6)
    ld      H, L                    ; save first toggle bit data
TOGGLECHK1:
    ld      L, (XIX)
    and     L, 0y01000000           ; check toggle bit (D6)
    cp      L, H                    ; toggle bit = toggled?
    j       z, TOGGLECHK2           ; if not toggled, end processing
    ld      H, L                    ; save current toggle bit state
    j       TOGGLECHK1              ; recheck toggle bit
TOGGLECHK2:
    ret
```

(Example: Program to be loaded and executed in RAM)

Read data from address FE8000H.

```
;#### Flash memory read processing ####
READ:
    ld      WA, (0xFE8000)          ; read data from flash memory
```

14. Electrical Characteristics

14.1 Absolute Maximum Ratings

Parameter	Symbol	Pin name	Rating	Unit
Supply voltage	V_{CC}		−0.5 to 6.0	V
Input voltage	V_{IN}		−0.5 to V_{CC} + 0.5	V
Output current (Per pin)	I_{OL1}	P5, P6, P96, P97	2	mA
Output current (Per pin)	I_{OL2}	P1, P3, P4, P7, P8, P90-P95, PA, PB	5	mA
Output current (Per pin)	I_{OL3}	P0	30	mA
Output current (Per pin)	I_{OH1}	P5, P6, P96, P97	−2	mA
Output current (Per pin)	I_{OH2}	P1, P3, P4, P7, P8, P90-P95, PA, PB	−5	mA
Output current (Per pin)	I_{OH3}	P0	−30	mA
Output current (Total)	ΣI_{OL}	P1, P3, P4, P5, P6, P7, P8, P9, PA, PB	80	mA
Output current (Total)	ΣI_{OH}	P1, P3, P4, P5, P6, P7, P8, P9, PA, PB	−80	mA
Output current (High current port Total)	ΣI_{OL3}	P0	120	mA
Output current (High current port Total)	ΣI_{OH3}	P0	−120	mA
Power dissipation (T_{OPR} = 85°C)	PD		600	mW
Soldering temperature (10 s)	T_{SOLDER}		260	°C
Storage temperature	T_{STG}		−65 to 150	°C
Operating temperature	T_{OPR}		−40 to 85	°C

Note: Absolute Maximum ratings are limiting values of operating and environmental conditions which should not be exceeded under the worst possible conditions. The equipment manufacturer should design so that no absolute maximum rating value is exceeded. Exposure to conditions beyond those listed above may cause permanent damage to the device or affect device reliability, which could increase potential risks of personal injury due to IC blowup and/or burning.

Solderability of lead free products

Test Parameter	Test Condition	Note
Solderability	Use of Sn-37Pb solder Bath Solder bath temperature 230 °C, Dipping time 5 [s] The number of times One, Use of R-type flux	Pass: solderability rate until forming ≥ 95%
	Use of Sn-3.0Ag-0.5 Cu solder Bath Solder bath temperature 245°C, Dipping time 5 [s] The number of times One, Use of R-type flux (use of lead free)	

14.2 DC Electrical Characteristics

Parameter		Symbol	Condition		Min	Typ.	Max	Unit
Power supply voltage	(AV_{CC} = DV_{CC}) (AV_{SS} = DV_{SS} = 0V)	V_{CC}	fc = 4 to 20 MHz fs = 30 to 34 kHz		4.5		5.5	V
	for erase/program operations of flash memory (AV_{CC} = DV_{CC}) (AV_{SS} = DV_{SS} = 0V)		fc = 4 to 20 MHz T_{OPR} = -10 to 40 °C		4.75		5.25	
Low-level input voltage	P00 to P17	V_{IL}	V_{CC} = 4.5 to 5.5 V		−0.3		0.8	V
	$\overline{RESET}$, P30 to PB2	V_{IL1}					0.25 V_{CC}	
	AM0, AM1	V_{IL2}					0.3	
	X1	V_{IL3}					0.2 V_{CC}	
High-level input voltage	P00 to P17	V_{IH}	V_{CC} = 4.5 to 5.5 V		2.2		V_{CC} + 0.3	V
	$\overline{RESET}$, P30 to PB2	V_{IH2}			0.75 V_{CC}			
	AM0, AM1	V_{IH3}			V_{CC} − 0.3			
	X1	V_{IH4}			0.8 V_{CC}			
Low-level output voltage		V_{OL}	I_{OL} = 1.6 mA (V_{CC} = 4.5 to 5.5 V)				0.45	V
High-level output voltage		V_{OH}	I_{OH} = −400 μA (V_{CC} = 4.5 to 5.5 V)		4.2			V
			I_{OH} = −1.6 mA (V_{CC} = 4.5 to 5.5 V)		2.4			
Low-level output current	High current port P0	I_{OL}	V_{OL} = 1.0V (V_{CC} = 4.5 to 5.5 V)			20		mA
Input leakage current		I_{LI}	0.0 ≤ V_{IN} ≤ V_{CC}			0.02	± 5	μA
Output leakage current		I_{LO}	0.2 ≤ V_{IN} ≤ V_{CC} − 0.2			0.05	± 10	
Power down voltage (while RAM is being backed up in STOP mode)		V_{STOP}	V_{IL2} = 0.2 V_{CC} V_{IH2} = 0.8 V_{CC}		2.0		5.5	V
$\overline{RESET}$ pull-up resistor		R_{RST}	V_{CC} = 4.5 to 5.5 V		50		230	kΩ
Pin capacitance		C_{IO}	fc = 1 MHz				10	pF
Schmitt width $\overline{RESET}$, INT0		V_{TH}	V_{CC} = 4.5 to 5.5 V		0.4	1.0		V
Programmable pull-up resistor		RKH	V_{CC} = 4.5 to 5.5 V		50		230	kΩ
NORMAL (Note 2)		I_{CC}	V_{CC} = 4.5 to 5.5 V fc = 20 MHz			25	35	mA
IDLE2						8	15	
IDLE1						3.5	8	
SLOW (Note 2)			V_{CC} = 4.5 to 5.5 V fs = 32.768 kHz			80	100	μA
STOP			T_{OPR} ≤ 50°C	V_{CC} = 4.5 to 5.5 V		0.5	10	μA
			T_{OPR} ≤ 70°C				25	
			T_{OPR} ≤ 85°C				50	
Peak current for Intermittent operation (Note 3,4)		I_{DDP-P}	V_{DD} = 5.5 V		–	20	–	mA

Note 1: Typical values show those at T_{OPR} = 25°C and V_{CC} = 5 V.

Note 2: I_{CC} measurement conditions (NORMAL, SLOW): All functions are operational; output pins are open and input pins are level fixed. Data and address bus CL = 30 pF loaded.

Note 3: When a program is executing in the flash memory or when data is being read from the flash memory, the flash memory operates in an intermittent manner, causing peak currents in the operation current, as shown in Figure 14-1.
In this case, the supply current I_{CC} (in NORMAL and SLOW modes) is defined as the sum of the average peak current and MCU current.

Note 4: When designing the power supply, make sure that peak currents can be supplied. In SLOW1 mode, the difference between the peak current and the average current becomes large.

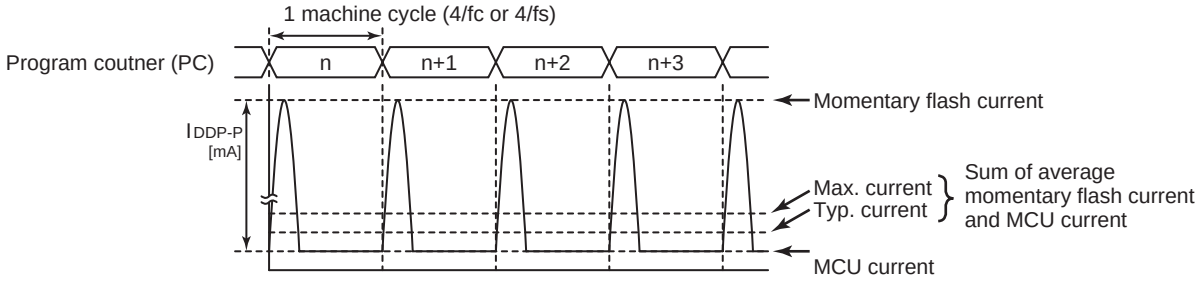


Figure 14-1 Intermittent Operation of Flash Memory

14.3 AD Conversion Characteristics

$AV_{CC} = DV_{CC}$, $AV_{SS} = DV_{SS}$

Parameter	Symbol	Variable	Min	Typ.	Max	Unit
Analog reference voltage (+)	AV_{CC}	V_{CC} = 4.5 to 5.5 V	DV_{CC} - 1.5 V	DV_{CC}	DV_{CC}	V
Analog reference voltage (-)	AV_{SS}		DV_{SS}	DV_{SS}	DV_{SS} + 0.2 V	V
Analog input voltage range	V_{AIN}		AV_{SS}		AV_{CC}	V
Error (Not including quantizing errors)	-			± 1.0	± 4.0	LSB

Note 1: 1LSB = $(AV_{CC} - AV_{SS})/1024$ [V]

Note 2: The operation above is guaranteed for fFPH ≥ 4 MHz.

Note 3: The value for I_{CC} includes the current which flows through the AV_{CC} pin.

14.4 Serial Channel Timing (I/O internal mode)

14.4.1 SCLK input mode

Parameter	Symbol	Variable		20 MHz		16 MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK period	t_{SCY}	16x		800		1000		ns
Output data → SCLK rising/falling edge*	t_{OSS}	$t_{SCY}/2 - 4x - 85$ (V_{CC} = 5V ± 10%)		115		165		ns
SCLK rising/falling edge* → Output data hold	t_{OHS}	$t_{SCY}/2 + 2x + 0$		500		625		ns
SCLK rising/falling edge* → Input data hold	t_{HSR}	3x + 10		160		198		ns
SCLK rising/falling edge* → Valid data input*	t_{SRD}		$t_{SCY} - 0$		800		1000	ns
Valid data input → SCLK rising/falling edge*	t_{RDS}	0		0		0		ns

Note: Symbol “x” in the above table means the period of clock “f_{FPH}”, it’s half period of the system clock “f_{SYS}” for CPU core. The period of f_{FPH} depends on the clock gear setting or the selection of high-/low-oscillator frequency.

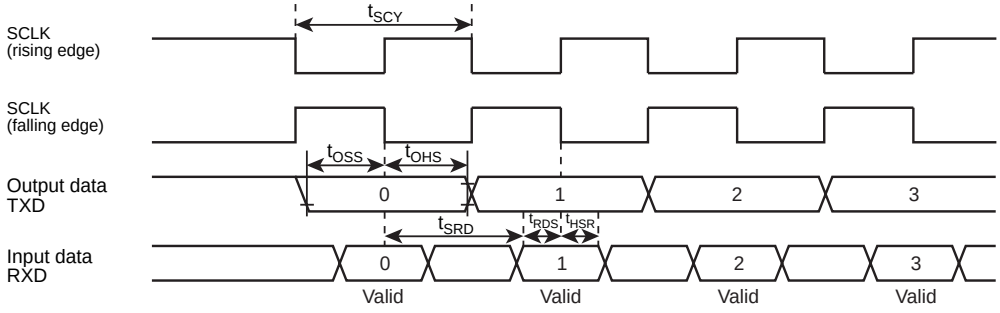
14.4.2 SCLK output mode

Parameter	Symbol	Variable		20 MHz		16 MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK period	t_{SCY}	16x	8192x	0.8	410	1.0	512	μs
Output data → SCLK rising/falling edge*	t_{OSS}	$t_{SCY}/2 - 40$		360		460		ns
SCLK rising/falling edge* → Output data hold	t_{OHS}	$t_{SCY}/2 - 40$		360		460		ns
SCLK rising/falling edge* → Input data hold	t_{HSR}	0		0		0		ns
SCLK rising/falling edge* → Valid data input	t_{SRD}		$t_{SCY} - 1x - 90$		660		847	ns
Valid data input → SCLK rising/falling edge*	t_{RDS}	1x + 90		140		153		ns

Note 1: *: SCLK rising/falling edge:The rising edge is used in SCLK rising mode.
The falling edge is used in SCLK falling mode.

Note 2: 20 MHz and 16 MHz values are calculated from t_{SCY} = 16x case.

Note 3: Symbol “x” in the above table means the period of clock “f_{FPH}”, it’s half period of the system clock “f_{SYS}” for CPU core. The period of f_{FPH} depends on the clock gear setting or the selection of high-/low-oscillator frequency.



14.5 Event Counter

TA0IN, TA4IN, TB0IN0, TB0IN1, TB1IN0,TB1IN1, TB2IN0,TB2IN1, TB3IN0,TB3IN1

Parameter	Symbol	Variable		20 MHz		16 MHz		Unit
		Min	Max	Min	Max	Min	Max	
Clock period	t_{VCK}	8x + 100		500		600		ns
Clock low-level width	t_{VCKL}	4x + 40		240		290		ns
Clock high-level width	t_{VCKH}	4x + 40		240		290		ns

Note: Symbol “x” in the above table means the period of clock “f_{FPH}”, it’s half period of the system clock “f_{SYS}” for CPU core. The period of f_{FPH} depends on the clock gear setting or the selection of high-/low-oscillator frequency.

14.6 Interrupt and Capture

14.6.1 INT0 to INT4 interrupts

Parameter	Symbol	Variable		20 MHz		16 MHz		Unit
		Min	Max	Min	Max	Min	Max	
INT0 to INT4 low-level width	t_{INTAL}	4x + 40		240		290		ns
INT0 to INT4 high-level width	t_{INTAH}	4x + 40		240		290		ns

Note: Symbol “x” in the above table means the period of clock “f_{FPH}”, it’s half period of the system clock “f_{SYS}” for CPU core. The period of f_{FPH} depends on the clock gear setting or the selection of high-/low-oscillator frequency.

14.6.2 INT1 to INT8 interrupts, capture

INT1 to INT8 input pulse width depend on the system clock selection and clock selection for prescaler. Below table show pulse width of each operation clock.

System Clock Selection SYSCR1 <SYSCK>	Clock Selection for Prescaler SYSCR0 <PRCK1>	t_{INTBL} (INT1 to INT8 low level pulse width)		t_{INTBH} (INT1 to INT8 high level pulse width)		Unit
		Variable	f_{FPH} = 20MHz	Variable	f_{FPH} = 20 MHz	
		Min	Min	Min	Min	
0 (fc)	0 (f_{FPH})	8x + 100	500	8x + 100	500	ns
	1 (fc/16)	128xc + 0.1	6.5	128xc + 0.1	6.5	us
1 (fc)	0 (f_{FPH})	8x + 0.1	244.3	8x + 0.1	244.3	

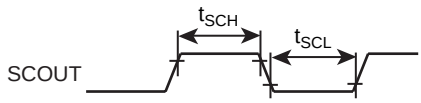
Note 1: “xc” shows period of clock fc in high frequency oscillator.

Note 2: Symbol “x” in the above table means the period of clock “f_{FPH}”, it’s half period of the system clock “f_{SYS}” for CPU core. The period of f_{FPH} depends on the clock gear setting or the selection of high-/low-oscillator frequency.

14.7 SCOUT Pin AC Characteristics

Parameter	Symbol	Variable		20 MHz		16 MHz		Condition	Unit
		Min	Max	Min	Max	Min	Max		
Low-level width	t_{SCH}	0.5T – 15		10		16		$V_{CC} \geq 4.5V$	ns
High-level width	t_{SCL}	0.5T – 15		10		16		$V_{CC} \geq 4.5V$	ns

Note: T = Period of SCOUT
Measuring conditions
Output level: High = 0.7 V_{CC}, Low = 0.3 V_{CC}, CL = 10 pF



14.8 Flash Characteristics

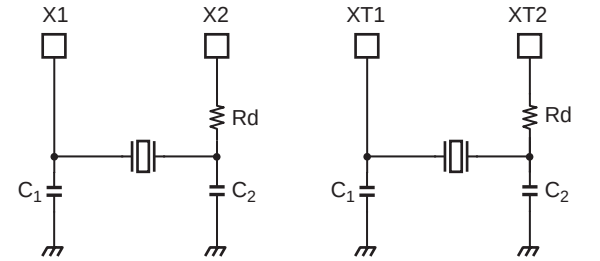
14.8.1 Write/Retention Characteristics

(V_{SS} = 0 V)

Parameter	Condition	Min	Typ.	Max.	Unit
Number of guaranteed writes to flash memory	V_{SS} = 0 V fc = 4 to 20 MHz T_{OPR} = −10 to 40°C	–	–	100	Times

14.9 Recommended Oscillating Conditions

The TMP91FU62 has been evaluated by the oscillator vender below. Use this information when selecting external parts.



(1) High-frequency Oscillation (2) Low-frequency Oscillation

Note 1: To ensure stable oscillation, the resonator position, load capacitance, etc. must be appropriate. Because these factors are greatly affected by board patterns, please be sure to evaluate operation on the board on which the device will actually be mounted.

Note 2: When using the device (oscillator) in places exposed to high electric fields such as cathode-ray tubes, we recommend electrically shielding the package in order to maintain normal operating condition.

Note 3: The product numbers and specifications of the resonators by Murata Manufacturing Co., Ltd. are subject to change. For up-to-date information, please refer to the following URL:
http://www.murata.co.jp/search/index.html

15. Table of SFR's

The special function registers (SFRs) include the I/O ports and peripheral control registers allocated to the 4-Kbyte address space from 000000H to 000FFFH.

1. I/O ports
2. I/O port control
3. Interrupt control
4. Clock gear
5. 8-bit timer
6. 16-bit timer
7. UART/serial channel
8. I^2C bus interface
9. AD converter
10. Watchdog timer
11. Special timer for CLOCK
12. Program patch logic

Table 15-1 SFR Address Map (PORT, INTC, CS/WAIT)

[1]PORT

Address	Name
0000H	P0
1H	P1
2H	P0CR
3H	
4H	P1CR
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	P3
DH	P3FC2
EH	P3CR
FH	P3FC

Address	Name
0010H	P4
1H	P4FC2
2H	P4CR
3H	
4H	P5
5H	SIOCHG1
6H	P5CR
7H	P5FC
8H	P6
9H	
AH	P6CR
BH	P6FC
CH	P7
DH	
EH	P7CR
FH	P7FC

Address	Name
0020H	P8
1H	
2H	P8CR
3H	P8FC
4H	P9
5H	SIOCHG0
6H	P9CR
7H	P9FC
8H	PA
9H	
AH	PACR
BH	PAFC
CH	PB
DH	
EH	PBCR
FH	

Address	Name
0030H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	ODE

[2]INTC

Address	Name
0080H	DMA0V
1H	DMA1V
2H	DMA2V
3H	DMA3V
4H	
5H	
6H	
7H	
8H	INTCLR
9H	DMAR
AH	DMAB
BH	
CH	IIMC
DH	
EH	
FH	

[2]INTC

Address	Name
0090H	INTE0AD
1H	INTE12
2H	INTE34
3H	INTE56
4H	INTE78
5H	
6H	INTETA01
7H	
8H	INTETA45
9H	INTETB0
AH	INTETB1
BH	INTETB2
CH	INTETB3
DH	
EH	INTETB01V
FH	INTETB23V

Address	Name
00A0H	INTERTC
1H	INTES0
2H	INTES1
3H	INTES2
4H	INTESBI0
5H	INTETC01
6H	INTETC23
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[4] CGEAR

Address	Name
00E0H	SYSCR0
1H	SYSCR1
2H	SYSCR2
3H	EMCCR0
4H	EMCCR1
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access to the unnamed addresses (e.g., addresses to which no register has been allocated).

Table 15-2 SFR Address Map (CGCR, TMRA, TMRB)

[5] TMRA

Address	Name
0100H	TA01RUN
1H	
2H	TA0REG
3H	TA1REG
4H	TA01MOD
5H	TA1FFCR
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Name
0110H	TA45RUN
1H	
2H	TA4REG
3H	TA5REG
4H	TA45MOD
5H	TA5FFCR
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[6] TMRB

Address	Name
0180H	TB0RUN
1H	
2H	TB0MOD
3H	TB0FFCR
4H	
5H	
6H	
7H	
8H	TB0RG0L
9H	TB0RG0H
AH	TB0RG1L
BH	TB0RG1H
CH	TB0CP0L
DH	TB0CP0H
EH	TB0CP1L
FH	TB0CP1H

[6] TMRB

Address	Name
0190H	TB1RUN
1H	
2H	TB1MOD
3H	TB1FFCR
4H	
5H	
6H	
7H	
8H	TB1RG0L
9H	TB1RG0H
AH	TB1RG1L
BH	TB1RG1H
CH	TB1CP0L
DH	TB1CP0H
EH	TB1CP1L
FH	TB1CP1H

Address	Name
01A0H	TB2RUN
1H	
2H	TB2MOD
3H	TB2FFCR
4H	
5H	
6H	
7H	
8H	TB2RG0L
9H	TB2RG0H
AH	TB2RG1L
BH	TB2RG1H
CH	TB2CP0L
DH	TB2CP0H
EH	TB2CP1L
FH	TB2CP1H

Address	Name
01B0H	TB3RUN
1H	
2H	TB3MOD
3H	TB3FFCR
4H	
5H	
6H	
7H	
8H	TB3RG0L
9H	TB3RG0H
AH	TB3RG1L
BH	TB3RG1H
CH	TB3CP0L
DH	TB3CP0H
EH	TB3CP1L
FH	TB3CP1H

[7] UART/SIO

Address	Name
0200H	SC0BUF
1H	SC0CR
2H	SC0MOD0
3H	BR0CR
4H	BR0ADD
5H	SC0MOD1
6H	
7H	
8H	SC1BUF
9H	SC1CR
AH	SC1MOD0
BH	BR1CR
CH	BR1ADD
DH	SC1MOD1
EH	
FH	

Address	Name
0210H	SC2BUF
1H	SC2CR
2H	SC2MOD0
3H	BR2CR
4H	BR2ADD
5H	SC2MOD1
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[8] I^2C

Address	Name
0240H	SBI0CR1
1H	SBI0DBR
2H	I2C0AR
3H	SBI0CR2/SBI0SR
4H	SBI0BR
5H	
6H	
7H	SBI0CR0
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Note: Do not access to the unnamed addresses (e.g., addresses to which no register has been allocated).

Table 15-3 SFR Address Map (UART/SIO, I^2C, ADC, WDT, RTC, ROMC)

[9]10bit ADC

Address	Name
02B0H	ADCCR1
1H	ADCCR2
2H	ADCDRL
3H	ADCDRH
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[10] WDT

Address	Name
0300H	WDMOD
1H	WDCR
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[11] RTC

Address	Name
0310H	RTCCR
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[12] ROMC

Address	Name
0400H	ROMCMP00
1H	ROMCMP01
2H	ROMCMP02
3H	
4H	ROMSUB0L
5H	ROMSUB0H
6H	
7H	
8H	ROMCMP10
9H	ROMCMP11
AH	ROMCMP12
BH	
CH	ROMSUB1L
DH	ROMSUB1H
EH	
FH	

Address	Name
0410H	ROMCMP20
1H	ROMCMP21
2H	ROMCMP22
3H	
4H	ROMSUB2L
5H	ROMSUB2H
6H	
7H	
8H	ROMCMP30
9H	ROMCMP31
AH	ROMCMP32
BH	
CH	ROMSUB3L
DH	ROMSUB3H
EH	
FH	

Address	Name
0420H	ROMCMP40
1H	ROMCMP41
2H	ROMCMP42
3H	
4H	ROMSUB4L
5H	ROMSUB4H
6H	
7H	
8H	ROMCMP50
9H	ROMCMP51
AH	ROMCMP52
BH	
CH	ROMSUB5L
DH	ROMSUB5H
EH	
FH	

Note: Do not access to the unnamed addresses (e.g., addresses to which no register has been allocated).

(1) I/O Ports

Symbol	Name	Address	7	6	5	4	3	2	1	0
P0	Port 0	00H	P07	P06	P05	P04	P03	P02	P01	P00
			R/W							
			Data from external port (Output latch register is undefined.)							
P1	Port 1	01H	P17	P16	P15	P14	P13	P12	P11	P10
			R/W							
			Data from external port (Output latch register is cleared to “0”.)							
P3	Port 3	0CH	–	–	–	–	P33	P32	P31	P30
			–	–	–	–	R/W			
			–	–	–	–	Data from external port (Output latch register is set to “1”.)			
P4	Port 4	10H	–	–	–	–	P43	P42	P41	P40
							R/W			
			–	–	–	–	Data from external port (Output latch register is set to “1”.)			
			–	–	–	–	0 (Output latch register): Pull-up resistor OFF 1 (Output latch register): Pull-up resistor ON			
P5	Port 5	14H	P57	P56	P55	P54	P53	P52	P51	P50
			R/W							
			Data from external port (Output latch register is set to “1”.)							
P6	Port 6	18H	P67	P66	P65	P64	P63	P62	P61	P60
			R/W							
			Data from external port (Output latch register is set to “1”.)							
P7	Port 7	1CH	–	–	P75	P74	P73	P72	P71	P70
			–	–	–					
			–	–	Data from external port (Output latch register is set to “1”.)					
P8	Port 8	20H	P87	P86	P85	P84	P83	P82	P81	P80
			R/W							
			Data from external port (Output latch register is set to “1”.)							
P9	Port 9	24H	P97	P96	P95	P94	P93	P92	P91	P90
			R/W							
			Data from external port (Output latch register is set to “1”.)							
PA	Port A	28H	–	–	–	–	PA3	PA2	PA1	PA0
			–	–	–	–	R/W			
			–	–	–	–	Data from external port (Output latch register is set to “1”.)			
PB	Port B	2CH	–	–	–	–	–	PB2	PB1	PB0
			–	–	–	–	–	R/W		
			–	–	–	–	–	Data from external port (Output latch register is set to “1”.)		

(2) I/O Port control

Symbol	Name	Address	7	6	5	4	3	2	1	0
P0CR	Port 0 control	02H (RMW instructions are prohibited.)	P07C	P06C	P05C	P04C	P03C	P02C	P01C	P00C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P1CR	Port 1 control	04H (RMW instructions are prohibited.)	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P3CR	Port 3 control	0EH (RMW instructions are prohibited.)	–	–	–	–	P33C	P32C	P31C	P30C
			–	–	–	–	W			
			–	–	–	–	0	0	0	0
			–	–	–	–	<<Refer to column of P3FC>>		<<Refer to column of P3FC2>>	
P3FC	Port 3 function	0FH (RMW instructions are prohibited.)	–	–	–	–	P33F	P32F	P31F	P30F
			–	–	–	–	W			
			–	–	–	–	0	0	0	0
			–	–	–	–	P33F/ P33C= 00:input port 01:output port 10: reserved 11:TB3OUT1	P32F/ P32C= 00:input port 01:output port 10: reserved 11:TB3OUT0	<<Refer to column of P3FC2>>	
P3FC2	Port 3 function 2	0DH (RMW instructions are prohibited.)	–	–	–	–	–	–	P31F2	P30F2
			–	–	–	–	–	–	W	
			–	–	–	–	–	–	0	0
			–	–	–	–	–	–	P31F2/ P31F/ P31C= 000:input port 001:output port 010:TB3IN1 /INT4 101: SCL0	P30F2/ P30F/ P30C= 000:input port 001:output port 010:TB3IN0 /INT3 101: SDA0

Symbol	Name	Address	7	6	5	4	3	2	1	0
P4CR	Port 4 control	12H (RMW instructions are prohibited.)	–	–	–	P44C	P43C	P42C	P41C	P40C
			–	–	–	W				
			–	–	–	0	0	0	0	0
			–	–	–	<<Refer to column of P4FC2>>				
P4FC2	Port 4 function 2	11H (RMW instructions are prohibited.)	–	–	–	–	P43F2	–	P41F2	P40F2
			–	–	–	–	W	–	W	
			–	–	–	–	0	–	0	0
			–	–	–	–	P43F2, P43C = 00 :input port 01:output port 10: reserved 11:SCLK2	P42C = 0: input port 1: output port	P41F2, P41C = 00: input port 01: output port 10: reserved 11:TXD2	P40F2, P40C = 00: input port 01:output port 10: reserved 11: SCOUT
SIOCHG1	SIO change register 1	15H (RMW instructions are prohibited.)	–	–	–	SIOCHG14	–	SIOCHG12	SIOCHG11	–
			–	–	–	W	–	W		–
			–	–	–	0	–	0	0	–
			–	–	–	P42 port 0: CMOS output 1: Open-drain output	–	0: Setting of P42C 1: TXD2	0: Setting of P41F2 and P41C 1: RXD2	–
P5CR	Port 5 control	16H (RMW instructions are prohibited.)	P57C	P56C	P55C	P54C	P53C	P52C	P51C	P50C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P5FC	Port 5 function	17H (RMW instructions are prohibited.)	P57F	P56F	P55F	P54F	P53F	P52F	P51F	P50F
			W							
			0	0	0	0	0	0	0	0
			P57 input 0: disable 1: enable	P56 input 0: disable 1: enable	P55 input 0: disable 1: enable	P54 input 0: disable 1: enable	P53 input 0: disable 1: enable	P52 input 0: disable 1: enable	P51 input 0: disable 1: enable	P50 input 0: disable 1: enable
P6CR	Port 6 control	1AH (RMW instructions are prohibited.)	P67C	P66C	P65C	P64C	P63C	P62C	P61C	P60C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P6FC	Port 6 function	1BH (RMW instructions are prohibited.)	P67F	P66F	P65F	P64F	P63F	P62F	P61F	P60F
			W							
			0	0	0	0	0	0	0	0
			P67 input 0: disable 1: enable	P66 input 0: disable 1: enable	P65 input 0: disable 1: enable	P64 input 0: disable 1: enable	P63 input 0: disable 1: enable	P62 input 0: disable 1: enable	P61 input 0: disable 1: enable	P60 input 0: disable 1: enable
P7CR	Port 7 control	1EH (RMW instructions are prohibited.)	–	–	P75C	P74C	P73C	P72C	P71C	P70C
			–	–	W					
			–	–	0	0	0	0	0	0
			–	–	0: Input 1: Output					
P7FC	Port 7 function	1FH (RMW instructions are prohibited.)	–	–	P75F	P74F	–	–	P71F	–
			–	–	W		–	–	W	–
			–	–	0	0	–	–	0	–
			–	–	0: Port 1: INT0	0: Port 1: TA5OUT	–	–	0: Port 1: TA1OUT	–

Symbol	Name	Address	7	6	5	4	3	2	1	0
P8CR	Port 8 control	22H (RMW instructions are prohibited.)	P87C	P86C	P85C	P84C	P83C	P82C	P81C	P80C
			W							
			0	0	0	0	0	0	0	0
			0: Input 1: Output							
P8FC	Port 8 function	23H (RMW instructions are prohibited.)	P87F	P86F	P85F	P84F	P83F	P82F	P81F	P80F
			W							
			0	0	0	0	0	0	0	0
			0: port 1: TB1OUT1	0: port 1: TB1OUT0	0: port 1: TB1IN1, INT8	0: port 1: TB1IN0, INT7	0: port 1: TB0OUT1	0: port 1: TB0OUT0	0: port 1: TB0IN1, INT6	0: port 1: TB0IN0, INT5
SIOCHG0	SIO change register 0	25H (RMW instructions are prohibited.)	–	–	SIOCHG05	SIOCHG04	SIOCHG03	SIOCHG02	SIOCHG01	SIOCHG00
			–	–	W					
			–	–	0	0	0	0	0	0
			–	–	P94 port 0: CMOS output 1: Open-drain output	0: Setting of P94C 1: TXD1	0: Setting of P93F and P93C 1: RXD1	P91 port 0: CMOS output 1: Open-drain output	0: Setting of P91C 1: TXD0	0: Setting of P90F and P90C 1: RXD0
P9CR	Port 9 control	26H (RMW instructions are prohibited.)	P97C	P96C	P95C	P94C	P93C	P92C	P91C	P90C
			W							
			1	1	0	0	0	0	0	0
			0: Input 1: Output							
P9FC	Port 9 function	27H (RMW instructions are prohibited.)	P97F	P96F	P95F	–	P93F	P92F	–	P90F
			W			–	W		–	W
			0	0	0	–	0	0	–	0
			Port 0: disable 1: enable	Port 0: disable 1: enable	0: port 1: SCLK1 output	–	0: port 1: TXD1 output	0: port 1: SCLK0 output	–	0: port 1: TXD0 output
PACR	Port A control	2AH (RMW instructions are prohibited.)	–	–	–	–	PA3C	PA2C	PA1C	PA0C
			–	–	–	–	W			
			–	–	–	–	0	0	0	0
			–	–	–	–	0: Input 1: Output			
PAFC	Port A function	2BH (RMW instructions are prohibited.)	–	–	–	–	PA3F	PA2F	PA1F	PA0F
			–	–	–	–	W			
			–	–	–	–	0	0	0	0
			–	–	–	–	0: port 1: TB2OUT1	0:port 1: TB2OUT0	0: port 1: TB2IN1, INT2	0: port 1: TB2IN0, INT1
PBCR	Port B control	2EH (RMW instructions are prohibited.)	–	–	–	–	–	PB2C	PB1C	PB0C
			–	–	–	–	–	W		
			–	–	–	–	–	0	0	0
			–	–	–	–	–	0: Input 1: Output		
ODE	Open-drain control register	3FH	–	–	–	ODE93	ODE90	ODE41	ODE31	ODE30
			–	–	–	R/W				
			–	–	–	0	0	0	0	0
			–	–	–	0: CMOS output 1:Open drain output				

(3) Interrupt control

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTE0AD	Interrupt enable 0 & AD	90H	INTAD				INT0			
			IADC	IADM2	IADM1	IADM0	I0C	I0M2	I0M1	I0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTAD	Interrput level			1: INT0	Interrput level		
INTE12	Interrupt enable 2 / 1	91H	INT2				INT1			
			I2C	I2M2	I2M1	I2M0	I1C	I1M2	I1M1	I1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INT2	Interrput level			1: INT1	Interrput level		
INTE34	Interrupt enable 4 / 3	92H	INT4				INT3			
			I4C	I4M2	I4M1	I4M0	I3C	I3M2	I3M1	I3M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INT4	Interrput level			1: INT3	Interrput level		
INTE56	Interrupt enable 6 / 5	93H	INT6				INT5			
			I6C	I6M2	I6M1	I6M0	I5C	I5M2	I5M1	I5M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INT6	Interrput level			1: INT5	Interrput level		
INTE78	Interrupt enable 8 / 7	94H	INT8				INT7			
			I8C	I8M2	I8M1	I8M0	I7C	I7M2	I7M1	I7M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INT8	Interrput level			1: INT7	Interrput level		
INTETA01	Interrupt enable timer A 1 / 0	96H	INTTA1(TMRA1)				INTTA0 (TMRA0)			
			ITA1C	ITA1M2	ITA1M1	ITA1M0	ITA0C	ITA0M2	ITA0M1	ITA0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTA1	Interrput level			1: INTTA0	Interrput level		
INTETA45	Interrupt enable timer A 5 / 4	98H	INTTA5 (TMRA5)				INTTA4 (TMRA4)			
			ITA5C	ITA5M2	ITA5M1	ITA5M0	ITA4C	ITA4M2	ITA4M1	ITA4M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTA5	Interrput level			1: INTTA4	Interrput level		

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTETB0	Interrupt enable TMRB 0	99H	INTTB01(TMRB0)				INTTB00(TMRB0)			
			ITB01C	ITB01M2	ITB01M1	ITB01M0	ITB00C	ITB00M2	ITB00M1	ITB00M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTB01	Interrput level			1: INTTB00	Interrput level		
INTETB1	Interrupt enable TMRB 1	9AH	INTTB11(TMRB1)				INTTB10(TMRB1)			
			ITB11C	ITB11M2	ITB11M1	ITB11M0	ITB10C	ITB10M2	ITB10M1	ITB10M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTB11	Interrput level			1: INTTB10	Interrput level		
INTETB2	Interrupt enable TMRB 2	9BH	INTTB21(TMRB2)				INTTB20(TMRB2)			
			ITB21C	ITB21M2	ITB21M1	ITB21M0	ITB20C	ITB20M2	ITB20M1	ITB20M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTB21	Interrput level			1: INTTB20	Interrput level		
INTETB3	Interrupt enable TMRB 3	9CH	INTTB31(TMRB3)				INTTB30(TMRB3)			
			ITB31C	ITB31M2	ITB31M1	ITB31M0	ITB30C	ITB30M2	ITB30M1	ITB30M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTB31	Interrput level			1: INTTB30	Interrput level		
INTETB01V	Interrupt enable TMRB 0/1 (Over flow)	9EH	INTTBOF1(TMRB1 over flow)				INTTBOF1(TMRB0 over flow)			
			ITF1C	ITF1M2	ITF1M1	ITF1M0	ITF0C	ITF0M2	ITF0M1	ITF0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTBOF1	Interrput level			1:INTTBOF0	Interrput level		
INTETB23V	Interrupt enable TMRB 2/3 (Over flow)	9FH	INTTBOF3(TMRB3 over flow)				INTTBOF2(TMRB2 over flow)			
			ITF3C	ITF3M2	ITF3M1	ITF3M0	ITF2C	ITF2M2	ITF2M1	ITF2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTBOF3	Interrput level			1:INTTBOF2	Interrput level		

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTERTC	Interrupt enable INTRTC	A0H	INTRTC				—			
			IRTCC	IRTCM2	IRTCM1	IRTCM0	—	—	—	—
			R	R/W			—	—		
			0	0	0	0	—	—	—	—
			1: INTRTC	Interrput level			—	—		
INTES0	Interrupt enable serial 0	A1H	INTTX0				INTRX0			
			ITX0C	ITX0M2	ITX0M1	ITX0M0	IRX0C	IRX0M2	IRX0M1	IRX0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTX0	Interrput level			1: INTRX0	Interrput level		
INTES1	Interrupt enable serial 1	A2H	INTTX1				INTRX1			
			ITX1C	ITX1M2	ITX1M1	ITX1M0	IRX1C	IRX1M2	IRX1M1	IRX1M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTX1	Interrput level			1: INTRX1	Interrput level		
INTES2	Interrupt enable serial 2	A3H	INTTX2				INTRX2			
			ITX2C	ITX2M2	ITX2M1	ITX2M0	IRX0C	IRX2M2	IRX2M1	IRX2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTX2	Interrput level			1: INTRX2	Interrput level		
INTESBI0	Interrupt enable SBI 0/1	A4H	—				INTSBI0			
			—	—	—	—	ISBI0C	ISBI0M2	ISBI0M1	ISBI0M0
			—	—			R	R/W		
			—	—	—	—	0	0	0	0
			—	—			1: INTSBI0	Interrput level		
INTETC01	Interrupt enable TC 0/1	A5H	INTTC1				INTTC0			
			ITC1C	ITC1M2	ITC1M1	ITC1M0	ITC0C	ITC0M2	ITC0M1	ITC0M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTC1	Interrput level			1: INTTC0	Interrput level		
INTETC23	Interrupt enable TC 2/3	A6H	INTTC3				INTTC2			
			ITC3C	ITC3M2	ITC3M1	ITC3M0	ITC2C	ITC2M2	ITC2M1	ITC2M0
			R	R/W			R	R/W		
			0	0	0	0	0	0	0	0
			1: INTTC3	Interrput level			1: INTTC2	Interrput level		

Symbol	Name	Address	7	6	5	4	3	2	1	0
DMA0V	DMA0 Start Vector	80H	—	—	DMA0V5	DMA0V4	DMA0V3	DMA0V2	DMA0V1	DMA0V0
			—	—	R/W					
			—	—	0	0	0	0	0	0
			—	—	DMA0 start vector					
DMA1V	DMA1 Start Vector	81H	—	—	DMA1V5	DMA1V4	DMA1V3	DMA1V2	DMA1V1	DMA1V0
			—	—	R/W					
			—	—	0	0	0	0	0	0
			—	—	DMA1 start vector					
DMA2V	DMA2 Start Vector	82H	—	—	DMA2V5	DMA2V4	DMA2V3	DMA2V2	DMA2V1	DMA2V0
			—	—	R/W					
			—	—	0	0	0	0	0	0
			—	—	DMA2 start vector					
DMA3V	DMA3 Start Vector	83H	—	—	DMA3V5	DMA3V4	DMA3V3	DMA3V2	DMA3V1	DMA3V0
			—	—	R/W					
			—	—	0	0	0	0	0	0
			—	—	DMA3 start vector					
INTCLR	Interrupt Clear Control	88H (RMW instructions are prohibited.)	—	—	CLRV5	CLRV4	CLRV3	CLRV2	CLRV1	CLRV0
			—	—	W					
			—	—	0	0	0	0	0	0
			—	—	Interrupt vector					
DMAR	DMA Software Request Register	89H (RMW instructions are prohibited.)	—	—	—	—	DMAR3	DMAR2	DMAR1	DMAR0
			—	—	—	—	R/W			
			—	—	—	—	0	0	0	0
			—	—	—	—	1: DMA software request			
DMAB	DMA Burst Register	8AH	—	—	—	—	DMAB3	DMAB2	DMAB1	DMAB0
			—	—	—	—	R/W			
			—	—	—	—	0	0	0	0
			—	—	—	—	1: DMA burst request			
IIMC	Interrupt input mode control	8CH (RMW instructions are prohibited.)	—	—	—	—	—	I0EDGE	I0LE	—
			W							
			0	0	0	0	0	0	0	0
			Always write “0”.	—	—	—	—	INT0 EDGE 0: Rising 1: Falling	INT0 mode 0: Edge 1: Level	—

(4) Clock control

Symbol	Name	Address	7	6	5	4	3	2	1	0
SYSCR0	System clock control register 0	E0H	XEN	XTEN	RXEN	RXTEN	RSYSCK	WUEF	PRCK1	–
			R/W							–
			1	0	1	0	0	0	0	–
			High-frequency oscillator 0:Stop 1:Oscillation	Low-frequency oscillator 0:Stop 1:Oscillation	High-frequency oscillator (fc) after release of STOP mode 0:Stop 1:Oscillation	Low-frequency oscillator (fs) after release of STOP mode 0:Stop 1:Oscillation	Selects clock after release of STOP mode 0:fc 1:fs	Warm-up timer control 0 Write: Don't care 1 Write: Start warm-up 0 Read: End warm-up 1 Read: Do not end warm-up	Select prescaler clock 0:f_{FPH} 1:fc/16	–
SYSCR1	System clock control register 1	E1H	–	–	–	–	SYSCK	GEAR2	GEAR2	GEAR2
			–	–	–	–	R/W			
			–	–	–	–	0	0	0	0
			–	–	–	–	Select system clock 0: fc 1: fs	Select gear value of high frequency (fc) 000:fc 001:fc/2 010:fc/4 011:fc/8 100:fc/16 101:reserved 110:reserved 111:reserved		
SYSCR2	System clock control register 1	E2H	–	SCOSEL	WUPTM1	WUPTM0	HALTM1	HALTM0	–	DRVE
			–	R/W					–	R/W
			–	0	1	0	1	1	–	0
			–	Select SCOUT 0:fs 1:f_{SYS}	Select warm-up time for oscillator 00:2^{18}/inputted frequency 01:2^{8}/inputted frequency 10:2^{14}/inputted frequency 11:2^{16}/inputted frequency		HALT mode 00:reserved 01:STOP mode 10:IDLE1 mode 11:IDLE2 mode		–	Pin state control in STOP mode 0: I/O off 1:Remains the state before HALT
EMCCR0	EMC control register 0	E3H	PROTECT	–	–	–	–	–	–	–
			R	R/W						
			0	0	1	0	0	0	1	1
			Protect flag 0:OFF 1:ON	Write "0".	Write "1".	Write "0".	Write "0".	Write "0".	Write "1".	Write "1".
EMCCR1	EMC control register 1	E4H	Protect OFF by writing "1FH". Protect ON by writing except "1FH".							

(5) 8-bit timer

Symbol	Name	Address	7	6	5	4	3	2	1	0
TA01RUN	8-bit timer RUN	100H	TA0RDE	–	–	–	I2TA01	TA01PRUN	TA1RUN	TA0RUN
			R/W	–	–	–	R/W			
			0	–	–	–	0	0	0	0
			Double buffer 0: Disable 1: Enable	–	–	–	IDLE2 0: Stop 1: Operate	TMRA01 prescaler 0: Stop and clear 1: Run (count up)	Up counter (UC1)	Up counter (UC0)
TA0REG	8-bit timer register 0	102H (RMW instructions are prohibited.)	–							
			W							
			0							
TA1REG	8-bit timer register 1	103H (RMW instructions are prohibited.)	–							
			W							
			0							
TA01MOD	8-bit timer source CLK & mode	104H	TA01M1	TA01M0	PWM01	PWM00	TA1CLK1	TA1CLK0	TA0CLK1	TA0CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		Input clock for TMRA1 00: TA0TRG 01: ϕT1 10: ϕT16 11: ϕT256		Input clock for TMRA0 00: TA0IN pin 01: ϕT1 10: ϕT4 11: ϕT16	
TA1FFCR	8-bit timer frip-flop control	105H	–	–	–	–	TA1FFC1	TA1FFC0	TA1FFIE	TA1FFIS
			–	–	–	–	R/W			
			–	–	–	–	1	1	0	0
			–	–	–	–	00: Invert TA1FF 01: Set TA1FF 10: Clear TA1FF 11: Don’t care		TA1FF control for inversion 0: Disable 1: Enable	TA1FF inversion select 0: TMRA0 1: TMRA1

Symbol	Name	Address	7	6	5	4	3	2	1	0
TA45RUN	8-bit timer RUN	110H	TA4RDE	–	–	–	I2TA45	TA45PRUN	TA5RUN	TA4RUN
			R/W	–	–	–	R/W			
			0	–	–	–	0	0	0	0
			Double buffer 0: Disable 1: Enable	–	–	–	IDLE2 0: Stop 1: Operate	TMRA45 prescaler	Up counter (UC5)	Up counter (UC4)
								0: Stop and clear 1: Run (count up)		
TA4REG	8-bit timer register 0	112H (RMW instructions are prohibited.)	–							
			W							
			0							
TA5REG	8-bit timer register 1	113H (RMW instructions are prohibited.)	–							
			W							
			0							
TA45MOD	8-bit timer source CLK & mode	114H	TA45M1	TA45M0	PWM41	PWM40	TA5CLK1	TA5CLK0	TA4CLK1	TA4CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Operation mode 00: 8-bit timer mode 01: 16-bit timer mode 10: 8-bit PPG mode 11: 8-bit PWM mode		PWM cycle 00: Reserved 01: 2^6 10: 2^7 11: 2^8		Input clock for TMRA5 00: TA4TRG 01: ϕT1 10: ϕT16 11: ϕT256		Input clock for TMRA4 00: TA4IN pin 01: ϕT1 10: ϕT4 11: ϕT16	
TA5FFCR	8-bit timer frip-flop control	115H	–	–	–	–	TA5FFC1	TA5FFC0	TA5FFIE	TA5FFIS
			–	–	–	–	R/W			
			–	–	–	–	1	1	0	0
			–	–	–	–	00: Invert TA5FF 01: Set TA5FF 10: Clear TA5FF 11: Don't care		TA5FF control for inversion 0: Disable 1: Enable	TA5FF inversion select 0: TMRA4 1: TMRA5

(6) 16-bit timer

Symbol	Name	Address	7	6	5	4	3	2	1	0
TB0RUN	16-bit timer control	180H	TB0RDE	–	–	–	I2TB0	TB0PRUN	–	TB0RUN
			R/W		–	–	R/W		–	R/W
			0	0	–	–	0	0	–	0
			Double Buffer 0: Disable 1: Enable	Always write 0.	–	–	IDLE2 0: Stop 1: Operate	TMRB0 prescaler	–	Up counter (UC0)
								0: Stop and Clear 1: Run (count up)		
TB0MOD	16-bit timer source CLK & mode	182H (RMW instructions are prohibited.)	TB0CT1	TB0ET1	TB0CP0I	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
			R/W		W*	R/W				
			0	0	1	0	0	0	0	0
			TB0FF1 inversion trigger 0: Trigger disable 1: Trigger enable		Software capture control 0: Software capture 1: Undefined	Capture timing 00: Disable INT5 occurs at rising edge 01: TB0IN0↑ TB0IN1↑ INT5 occurs at rising edge 10: TB0IN0↑ TB0IN0↓ INT5 occurs at falling edge 11: TA1OUT↑ TA1OUT↓ INT5 occurs at rising edge		Up counter control 0: Clear disable 1: Clear enable	TMRB0 input clock select 00: TB0IN0 pin input 01: ϕT1 10: ϕT4 11: ϕT16	
			Invert when UC0 is loaded into TB0CP1H/L	Invert when UC0 matches with TB0RG1H/L						
TB0FFCR	16-bit timer frip-flop control	183H (RMW instructions are prohibited.)	TB0FF1C1	TB0FF1C0	TB0C1T1	TB0C0T1	TB0E1T1	TB0E0T1	TB0FF0C1	TB0FF0C0
			W*		R/W				W*	
			1	1	0	0	0	0	1	1
			TB0FF1 control 00: Invert 01: Set 10: Clear 11: Don't care Note: Always read as 11.		TB0FF0 inversion trigger 0: Disable 1: Enable				TB0FF0 control 00: Invert 01: Set 10: Clear 11: Don't care Note: Always read as 11.	
					Invert when UC0 is loaded into TB0CP1H/L.	Invert when UC0 is loaded into TB0CP0H/L.	Invert when UC0 matches TB0RG1H/L.	Invert when UC0 matches TB0RG0H/L.		
TB0RG0L	16-bit timer register 0L	188H (RMW instructions are prohibited.)	–							
			W							
			Undefined							
TB0RG0H	16-bit timer register 0H	189H (RMW instructions are prohibited.)	–							
			W							
			Undefined							
TB0RG1L	16-bit timer register 1L	18AH (RMW instructions are prohibited.)	–							
			W							
			Undefined							
TB0RG1H	16-bit timer register 1H	18BH (RMW instructions are prohibited.)	–							
			W							
			Undefined							
TB0CP0L	Capture register 0L	18CH	–							
			R							
			Undefined							
TB0CP0H	Capture register 0H	18DH	–							
			R							
			Undefined							
TB0CP1L	Capture register 1L	18EH	–							
			R							
			Undefined							
TB0CP1H	Capture register 1H	18FH	–							
			R							
			Undefined							

Symbol	Name	Address	7	6	5	4	3	2	1	0
TB1RUN	16-bit timer control	190H	TB1RDE	–	–	–	I2TB1	TB1PRUN	–	TB1RUN
			R/W		–	–	R/W		–	R/W
			0	0	–	–	0	0	–	0
			Double Buffer 0: Disable 1: Enable	Always write 0.	–	–	IDLE2 0: Stop 1: Operate	TMRB1 prescaler	–	Up counter (UC1)
								0: Stop and Clear 1: Run (count up)		
TB1MOD	16-bit timer source CLK & mode	192H (RMW instructions are prohibited.)	TB1CT1	TB1ET1	TB1CP0I	TB1CPM1	TB1CPM0	TB1CLE	TB1CLK1	TB1CLK0
			R/W		W*	R/W				
			0	0	1	0	0	0	0	0
			TB1FF1 inversion trigger 0: Trigger disable 1: Trigger enable		Software capture control 0: Software capture 1: Undefined	Capture timing 00: Disable INT7 occurs at rising edge 01: TB1IN0↑ TB1IN1↑ INT7 occurs at rising edge 10: TB1IN0↑ TB1IN0↓ INT7 occurs at falling edge 11: TA1OUT↑ TA1OUT↓ INT7 occurs at rising edge		Up counter control 0: Clear disable 1: Clear enable	TMRB1 input clock select 00: TB1IN0 pin input 01: ϕT1 10: ϕT4 11: ϕT16	
			Invert when UC1 is loaded into TB1CP1H/L	Invert when UC1 matches with TB1RG1H/L						
TB1FFCR	16-bit timer frip-flop control	193H (RMW instructions are prohibited.)	TB1FF1C1	TB1FF1C0	TB1C1T1	TB1C0T1	TB1E1T1	TB1E0T1	TB1FF0C1	TB1FF0C0
			W*		R/W				W*	
			1	1	0	0	0	0	1	1
			TB1FF1 control 00: Invert 01: Set 10: Clear 11: Don't care Note: Always read as 11.		TB1FF0 inversion trigger 0: Disable 1: Enable				TB1FF0 control 00: Invert 01: Set 10: Clear 11: Don't care Note: Always read as 11.	
					Invert when UC1 is loaded into TB1CP1H/L.	Invert when UC1 is loaded into TB1CP0H/L.	Invert when UC1 matches TB1RG1H/L.	Invert when UC1 matches TB1RG0H/L.		
TB1RG0L	16-bit timer register 0L	198H (RMW instructions are prohibited.)	–							
			W							
			Undefined							
TB1RG0H	16-bit timer register 0H	199H (RMW instructions are prohibited.)	–							
			W							
			Undefined							
TB1RG1L	16-bit timer register 1L	19AH (RMW instructions are prohibited.)	–							
			W							
			Undefined							
TB1RG1H	16-bit timer register 1H	19BH (RMW instructions are prohibited.)	–							
			W							
			Undefined							
TB1CP0L	Capture register 0L	19CH	–							
			R							
			Undefined							
TB1CP0H	Capture register 0H	19DH	–							
			R							
			Undefined							
TB1CP1L	Capture register 1L	19EH	–							
			R							
			Undefined							
TB1CP1H	Capture register 1H	19FH	–							
			R							
			Undefined							

Symbol	Name	Address	7	6	5	4	3	2	1	0
TB2RUN	16-bit timer control	1A0H	TB2RDE	–	–	–	I2TB2	TB2PRUN	–	TB2RUN
			R/W		–	–	R/W		–	R/W
			0	0	–	–	0	0	–	0
			Double Buffer 0: Disable 1: Enable	Always write 0.	–	–	IDLE2 0: Stop 1: Operate	TMRB2 prescaler	–	Up counter (UC2)
								0: Stop and Clear 1: Run (count up)		
TB2MOD	16-bit timer source CLK & mode	1A2H (RMW instructions are prohibited.)	TB2CT1	TB2ET1	TB2CP0I	TB2CPM1	TB2CPM0	TB2CLE	TB2CLK1	TB2CLK0
			R/W		W*	R/W				
			0	0	1	0	0	0	0	0
			TB2FF1 inversion trigger 0: Trigger disable 1: Trigger enable		Software capture control 0: Software capture 1: Undefined	Capture timing 00: Disable INT1 occurs at rising edge 01: TB2IN0↑ TB2IN1↑ INT1 occurs at rising edge 10: TB2IN0↑ TB2IN0↓ INT1 occurs at falling edge 11: TA1OUT↑ TA1OUT↓ INT1 occurs at rising edge		Up counter control 0: Clear disable 1: Clear enable	TMRB2 input clock select 00: TB2IN0 pin input 01: ϕT1 10: ϕT4 11: ϕT16	
			Invert when UC2 is loaded into TB2CP1H/L	Invert when UC2 matches with TB2RG1H/L						
TB2FFCR	16-bit timer frip-flop control	1A3H (RMW instructions are prohibited.)	TB2FF1C1	TB2FF1C0	TB2C1T1	TB2C0T1	TB2E1T1	TB2E0T1	TB2FF0C1	TB2FF0C0
			W*		R/W				W*	
			1	1	0	0	0	0	1	1
			TB2FF1 control 00: Invert 01: Set 10: Clear 11: Don't care Note: Always read as 11.		TB2FF0 inversion trigger 0: Disable 1: Enable				TB2FF0 control 00: Invert 01: Set 10: Clear 11: Don't care Note: Always read as 11.	
					Invert when UC2 is loaded into TB2CP1H/L.	Invert when UC2 is loaded into TB2CP0H/L.	Invert when UC2 matches TB2RG1H/L.	Invert when UC2 matches TB2RG0H/L.		
TB2RG0L	16-bit timer register 0L	1A8H (RMW instructions are prohibited.)	–							
			W							
			Undefined							
TB2RG0H	16-bit timer register 0H	1A9H (RMW instructions are prohibited.)	–							
			W							
			Undefined							
TB2RG1L	16-bit timer register 1L	1AAH (RMW instructions are prohibited.)	–							
			W							
			Undefined							
TB2RG1H	16-bit timer register 1H	1ABH (RMW instructions are prohibited.)	–							
			W							
			Undefined							
TB2CP0L	Capture register 0L	1ACH	–							
			R							
			Undefined							
TB2CP0H	Capture register 0H	1ADH	–							
			R							
			Undefined							
TB2CP1L	Capture register 1L	1AEH	–							
			R							
			Undefined							
TB2CP1H	Capture register 1H	1AFH	–							
			R							
			Undefined							

Symbol	Name	Address	7	6	5	4	3	2	1	0
TB3RUN	16-bit timer control	1B0H	TB3RDE	–	–	–	I2TB3	TB3PRUN	–	TB3RUN
			R/W		–	–	R/W		–	R/W
			0	0	–	–	0	0	–	0
			Double Buffer 0: Disable 1: Enable	Always write 0.	–	–	IDLE2 0: Stop 1: Operate	TMRB3 prescaler	–	Up counter (UC3)
								0: Stop and Clear 1: Run (count up)		
TB3MOD	16-bit timer source CLK & mode	1B2H (RMW instructions are prohibited.)	TB3CT1	TB3ET1	TB3CP0I	TB3CPM1	TB3CPM0	TB3CLE	TB3CLK1	TB3CLK0
			R/W		W*	R/W				
			0	0	1	0	0	0	0	0
			TB3FF1 inversion trigger 0: Trigger disable 1: Trigger enable		Software capture control 0: Software capture 1: Undefined	Capture timing 00: Disable INT3 occurs at rising edge 01: TB3IN0↑ TB3IN1↑ INT3 occurs at rising edge 10: TB3IN0↑ TB3IN0↓ INT3 occurs at falling edge 11: Reserved		Up counter control 0: Clear disable 1: Clear enable	TMRB3 input clock select 00: TB3IN0 pin input 01: ϕT1 10: ϕT4 11: ϕT16	
			Invert when UC3 is loaded into TB3CP1H/L	Invert when UC3 matches with TB3RG1H/L						
TB3FFCR	16-bit timer frip-flop control	1B3H (RMW instructions are prohibited.)	TB3FF1C1	TB3FF1C0	TB3C1T1	TB3C0T1	TB3E1T1	TB3E0T1	TB3FF0C1	TB3FF0C0
			W*		R/W				W*	
			1	1	0	0	0	0	1	1
			TB3FF1 control 00: Invert 01: Set 10: Clear 11: Don't care Note: Always read as 11.		TB3FF0 inversion trigger 0: Disable 1: Enable				TB3FF0 control 00: Invert 01: Set 10: Clear 11: Don't care Note: Always read as 11.	
					Invert when UC3 is loaded into TB3CP1H/L.	Invert when UC3 is loaded into TB3CP0H/L.	Invert when UC3 matches TB3RG1H/L.	Invert when UC3 matches TB3RG0H/L.		
TB3RG0L	16-bit timer register 0L	1B8H (RMW instructions are prohibited.)	–							
			W							
			Undefined							
TB3RG0H	16-bit timer register 0H	1B9H (RMW instructions are prohibited.)	–							
			W							
			Undefined							
TB3RG1L	16-bit timer register 1L	1BAH (RMW instructions are prohibited.)	–							
			W							
			Undefined							
TB3RG1H	16-bit timer register 1H	1BBH (RMW instructions are prohibited.)	–							
			W							
			Undefined							
TB3CP0L	Capture register 0L	1BCH	–							
			R							
			Undefined							
TB3CP0H	Capture register 0H	1BDH	–							
			R							
			Undefined							
TB3CP1L	Capture register 1L	1BEH	–							
			R							
			Undefined							
TB3CP1H	Capture register 1H	1BFH	–							
			R							
			Undefined							

(7) UART/SIO

Symbol	Name	Address	7	6	5	4	3	2	1	0
SC0BUF	Serial channel 0 buffer	200H (RMW instructions are prohibited.)	RB7 / TB7	RB6 / TB6	RB5 / TB5	RB4 / TB4	RB3 / TB3	RB2 / TB2	RB1 / TB1	RB0 / TB0
			R (Receiving) / W (Transmission)							
			Undefined							
SC0CR	Serial channel 0 control	201H (RMW instructions are prohibited.)	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
			R	R/W		R (Cleared to "0" when read)			R/W	
			Undefined	0	0	0	0	0	0	0
			Received data bit8	Parity 0: Odd 1: Even	Parity addition 0: Disable 1: Enable	Overrun error flag 0: Undetect error 1: Detect error	Parity error flag 0: Undetect error 1: Detect error	Framing error flag 0: Undetect error 1: Detect error	Edge selection for SCLK pin (I/O mode) 0: SCLK↑ 1: SCLK↓	Edge selection for SCLK pin (I/O mode) 0: SCLK↑ 1: SCLK↓
SC0MOD0	Serial channel 0 mode 0	202H	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			0	0	0	0	0	0	0	0
			Transmission data bit8	Handshake function 0: Disable 1: Enable	Receive function 0: Disable 1: Enable	Wakeup function 0: Disable 1: Enable	Serial transmission mode 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial transmission clock (UART) 00: Timer TA0TRG 01: Baud rate generator 10: Internal clock f_{SYS} 11: External clock (SCLK input)	
BR0CR	Baud ratel control	203H	–	BR0ADDE	BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
			R/W							
			0	0	0	0	0	0	0	0
			Always write 0.	+ (16 - K)/16 division 0: Disable 1: Enable	Input clock selection for baud rate generator 00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Setting of the divided frequency “N”			
BR0ADD	Serial channel 0 K setting register	204H	–	–	–	–	BR0K3	BR0K2	BR0K1	BR0K0
			–	–	–	–	R/W			
			–	–	–	–	0	0	0	0
			–	–	–	–	Sets frequency divisor “K” (Divided by N + (16 - K)/16)			
SC0MOD1	Serial channel 0 mode 1	205H	I2S0	FDPX0	–	–	–	–	–	–
			R/W		–	–	–	–	–	–
			0	0	–	–	–	–	–	–
			IDLE2 0: Stop 1: Run	Duplex 0: Half 1: Full	–	–	–	–	–	–

Symbol	Name	Address	7	6	5	4	3	2	1	0
SC1BUF	Serial channel 1 buffer	208H (RMW instructions are prohibited.)	RB7 / TB7	RB6 / TB6	RB5 / TB5	RB4 / TB4	RB3 / TB3	RB2 / TB2	RB1 / TB1	RB0 / TB0
			R (Receiving) / W (Transmission)							
			Undefined							
SC1CR	Serial channel 1 control	209H (RMW instructions are prohibited.)	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
			R	R/W		R (Cleared to "0" when read)			R/W	
			Undefined	0	0	0	0	0	0	0
			Received data bit8	Parity 0: Odd 1: Even	Parity addition 0: Disable 1: Enable	Overrun error flag 0: Undetect error 1: Detect error	Parity error flag 0: Undetect error 1: Detect error	Framing error flag 0: Undetect error 1: Detect error	Edge selection for SCLK pin (I/O mode) 0: SCLK↑ 1: SCLK↓	Edge selection for SCLK pin (I/O mode) 0: SCLK↑ 1: SCLK↓
SC1MOD0	Serial channel 1 mode 0	20AH	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			0	0	0	0	0	0	0	0
			Transmission data bit8	Handshake function 0: Disable 1: Enable	Receive function 0: Disable 1: Enable	Wakeup function 0: Disable 1: Enable	Serial transmission mode 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial transmission clock (UART) 00: Timer TA0TRG 01: Baud rate generator 10: Internal clock f_{SYS} 11: External clock (SCLK input)	
BR1CR	Baud ratel control	20BH	–	BR1ADDE	BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
			R/W							
			0	0	0	0	0	0	0	0
			Always write “0”.	+ (16 - K)/16 division 0: Disable 1: Enable	Input clock selection for baud rate generator 00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Setting of the divided frequency “N”			
BR1ADD	Serial channel 1 K setting register	20CH	–	–	–	–	BR1K3	BR1K2	BR1K1	BR1K0
			–	–	–	–	R/W			
			–	–	–	–	0	0	0	0
			–	–	–	–	Sets frequency divisor “K” (Divided by N + (16 - K)/16)			
SC1MOD1	Serial channel 1 mode 1	20DH	I2S1	FDPX1	–	–	–	–	–	–
			R/W		–	–	–	–	–	–
			0	0	–	–	–	–	–	–
			IDLE2 0: Stop 1: Run	Duplex 0: Half 1: Full	–	–	–	–	–	–

Symbol	Name	Address	7	6	5	4	3	2	1	0
SC2BUF	Serial channel 2 buffer	210H (RMW instructions are prohibited.)	RB7 / TB7	RB6 / TB6	RB5 / TB5	RB4 / TB4	RB3 / TB3	RB2 / TB2	RB1 / TB1	RB0 / TB0
			R (Receiving) / W (Transmission)							
			Undefined							
SC2CR	Serial channel 2 control	211H (RMW instructions are prohibited.)	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
			R	R/W		R (Cleared to "0" when read)			R/W	
			Undefined	0	0	0	0	0	0	0
			Received data bit8	Parity 0: Odd 1: Even	Parity addition 0: Disable 1: Enable	Overrun error flag 0: Undetect error 1: Detect error	Parity error flag 0: Undetect error 1: Detect error	Framing error flag 0: Undetect error 1: Detect error	Edge selection for SCLK pin (I/O mode) 0: SCLK↑ 1: SCLK↓	Edge selection for SCLK pin (I/O mode) 0: SCLK↑ 1: SCLK↓
SC2MOD0	Serial channel 2 mode 0	212H	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			0	0	0	0	0	0	0	0
			Transmission data bit8	Handshake function 0: Disable 1: Enable	Receive function 0: Disable 1: Enable	Wakeup function 0: Disable 1: Enable	Serial transmission mode 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial transmission clock (UART) 00: Timer TA0TRG 01: Baud rate generator 10: Internal clock f_{SYS} 11: External clock (SCLK input)	
BR2CR	Baud ratel control	213H	–	BR2ADDE	BR2CK1	BR2CK0	BR2S3	BR2S2	BR2S1	BR2S0
			R/W							
			0	0	0	0	0	0	0	0
			Always write “0”.	+ (16 - K)/16 division 0: Disable 1: Enable	Input clock selection for baud rate generator 00: ϕT0 01: ϕT2 10: ϕT8 11: ϕT32		Setting of the divided frequency “N”			
BR2ADD	Serial channel 2 K setting register	214H	–	–	–	–	BR2K3	BR2K2	BR2K1	BR2K0
			–	–	–	–	R/W			
			–	–	–	–	0	0	0	0
			–	–	–	–	Sets frequency divisor “K” (Divided by N + (16 - K)/16)			
SC2MOD1	Serial channel 2 mode 1	215H	I2S2	FDPX2	–	–	–	–	–	–
			R/W		–	–	–	–	–	–
			0	0	–	–	–	–	–	–
			IDLE2 0: Stop 1: Run	Duplex 0: Half 1: Full	–	–	–	–	–	–

(8) I^2C bus interface

Symbol	Name	Address	7	6	5	4	3	2	1	0
SBI0CR1	Serial bus interface control register 1	240H (RMW instructions are prohibited.)	BC2	BC1	BC0	ACK	–	SCK2	SCK1	SCK0/ SWRMON
			W			R/W	–	W		R/W
			0	0	0	0	–	0	0	0/1
			Number of transferred bits 000: 8 001: 1 010: 2 011: 3 100: 4 101: 5 110: 6 111: 7			Acknowledge clock 0: Disable 1: Enable	–	<SCK2:0> at write Internal serial clock selection and software reset monitor 000: 4 001: 5 010: 6 011: 7 100: 8 101: 9 110: 10 111:Reserved <SWRMON> at read 0: During software reset		
SBI0DBR	SBI buffer register	241H (RMW instructions are prohibited.)	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
			R (Receiving) / W (Transmission)							
			Undefined							
I2C0AR	I^2C bus address register	242H (RMW instructions are prohibited.)	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ALS
			W							
			0	0	0	0	0	0	0	0
			Slave address selection for when device is operating as slave device							Address recognition 0: Enable 1: Disable
When read SBI0SR	Serial bus interface status register	243H (RMW instructions are prohibited.)	MST	TRX	BB	PIN	AL/ SBIM1	AAS/ SBIM0	AD0/ SWRST1	LRB/ SWRST0
			R/W							
			0	0	0	1	0	0	0	0
			0: Slave 1: Master	0:Receiver 1:Transmit	Bus status monitor 0: Free 1: Busy[	INTSBI request monitor 0: Request 1: Cancel	Arbitration lost detection monitor 1: Detect	Slave address match detection monitor 1:Detect	GENERAL CALL detection 1: Detect	Last receive bit monitor 0: “0” 1: “1”
When write SBI0CR2	Serial bus interface control register 2				Start/stop condition 0: Start condition 1: Stop condition	Cancel INTSBI interrupt request 0: – 1: Cancel	Serial bus interface operating mode selection 00: Port mode 01: Reserved 10: I^2C bus mode 11: Reserved		Software reset generate Write “10” and “01”, then an internal reset signal is generated.	
SBI0BR	Serial bus interface baud rate register	244H (RMW instructions are prohibited.)	–	I2SBI0	–	–	–	–	–	–
			W	R/W	–	–	–	–	–	R/W
			0	0	–	–	–	–	–	0
			Always write “0”	Operation in IDLE2 mode 0: Stop 1: Operate	–	–	–	–	–	Always write “0”
SBI0CR0	Serial bus interface control register 0	247H (RMW instructions are prohibited.)	SBI0EN	–	–	–	–	–	–	–
			R/W	R						
			0	0	0	0	0	0	0	0
			SBI operation 0: disable 1: enable	Always read "0".						

(9) AD converter

Symbol	Name	Address	7	6	5	4	3	2	1	0
ADCCR1	AD control register 1	2B0H	ADRS	AMD		AINEN	SAIN			
			R/W							
			0	0	0	0	0	0	0	0
			AD conversion start 0: - 1: AD conversion start	AD operating mode 00: AD operation disable 01: single mode 10: Reserved 11: Repeat mode		Analog input control 0: disable 1: enable	Analog input channel select 0000: AN0 0100: AN4 1000: AN8 1100: AN12 0001: AN1 0101: AN5 1001: AN9 1101: AN13 0010: AN2 0110: AN6 1010: AN10 1110: AN14 0011: AN3 0111: AN7 1011: AN11 1111: AN15			
ADCCR2	AD control register 2	2B1H (RMW instructions are prohibited.)	EOCF	ADBF	RSEL	I2AD	ACK			
			R		R/W					
			0	0	0	0	1	1	0	0
			AD conversion end flag 0:Before or during conversion 1: Conversion completed	AD conversion BUSY flag 0: During stop of AD conversion 1: During AD conversion	Storing of an AD conversion result 0: 10bit mode 1: 8bit mode	IDLE2 control 0:Stop 1:Operation	AD conversion time select 1010: 78 / fc [s] 1011: 156 / fc [s] 1100: 312 / fc [s] 1101: 624 / fc [s] 1110: 1248 / fc [s]			
ADCDRL	AD result register L	2B2H	AD07	AD06	AD05	AD04	AD03	AD02	AD01	AD00
			R							
			0	0	0	0	0	0	0	0
ADCDRH When 10-bit storing mode	AD result register H	2B3H	–	–	–	–	–	–	AD09	AD08
			R							
			0	0	0	0	0	0	0	0
ADCDRH When 8-bit storing mode			AD09	AD08	AD07	AD06	AD05	AD04	AD03	AD02
			R							
			0	0	0	0	0	0	0	0

(10) Watchdog timer

Symbol	Name	Address	7	6	5	4	3	2	1	0
WDMOD	WDT mode register	300H	WDTE	WDTP1	WDTP0	–	–	I2WDT	RESCR	–
			R/W			–	–	R/W		
			0	0	0	–	–	0	0	0
			WDT control 1: Enable	Select detecting time 00: $2^{15}/f_{SYS}$ 01: $2^{17}/f_{SYS}$ 10: $2^{19}/f_{SYS}$ 11: $2^{21}/f_{SYS}$		–	–	IDLE2 0: Stop 1: Operate	1: Internally connects WDT out to the reset pin	Always write “0”.
WDCR	WDT control	301H (RMW instructions are prohibited.)	–							
			W							
			–							
			B1H: WDT disable code 4EH: WDT clear code							

(11) Special timer for CLOCK

Symbol	Name	Address	7	6	5	4	3	2	1	0
RTCCR	RTC control register	310H	–	–	–	–	–	RTCSEL1	RTCSEL0	RTCRUN
			R/W	–	–	–	–	R/W		
			0	–	–	–	–	0	0	0
			Always write “0”.	–	–	–	–	00: 2^{14}/fs 01: 2^{13}/fs 10: 2^{12}/fs 11: 2^{11}/fs		0: Stop & clear 1: Count

(12) Program patch logic

Symbol	Name	Address	7	6	5	4	3	2	1	0
ROMCMP00	Address compare register 00	400H (RMW instructions are prohibited.)	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	–
			W							–
			0	0	0	0	0	0	0	–
			Target ROM address (Lower 7 bits)							–
ROMCMP01	Address compare register 01	401H (RMW instructions are prohibited.)	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Middle 8 bits)							
ROMCMP02	Address compare register 02	402H (RMW instructions are prohibited.)	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Upper 8 bits)							
ROMSUB0L	Address substitution register 0L	404H (RMW instructions are prohibited.)	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
			W							
			0	0	0	0	0	0	0	0
			Patch code (Lower 8 bits)							
ROMSUB0H	Address substitution register 0H	405H (RMW instructions are prohibited.)	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
			W							
			0	0	0	0	0	0	0	0
			Patch code (Upper 8 bits)							
ROMCMP10	Address compare register 10	408H (RMW instructions are prohibited.)	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	–
			W							–
			0	0	0	0	0	0	0	–
			Target ROM address (Lower 7 bits)							–
ROMCMP11	Address compare register 11	409H (RMW instructions are prohibited.)	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Middle 8 bits)							
ROMCMP12	Address compare register 12	40AH (RMW instructions are prohibited.)	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Upper 8 bits)							
ROMSUB1L	Address substitution register 1L	40CH (RMW instructions are prohibited.)	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
			W							
			0	0	0	0	0	0	0	0
			Patch code (Lower 8 bits)							
ROMSUB1H	Address substitution register 1H	40DH (RMW instructions are prohibited.)	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
			W							
			0	0	0	0	0	0	0	0
			Patch code (Upper 8 bits)							

Symbol	Name	Address	7	6	5	4	3	2	1	0
ROMCMP20	Address compare register 20	410H (RMW instructions are prohibited.)	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	–
			W							–
			0	0	0	0	0	0	0	–
			Target ROM address (Lower 7 bits)							–
ROMCMP21	Address compare register 21	411H (RMW instructions are prohibited.)	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Middle 8 bits)							
ROMCMP22	Address compare register 22	412H (RMW instructions are prohibited.)	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Upper 8 bits)							
ROMSUB2L	Address substitution register 2L	414H (RMW instructions are prohibited.)	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
			W							
			0	0	0	0	0	0	0	0
			Patch code (Lower 8 bits)							
ROMSUB2H	Address substitution register 2H	415H (RMW instructions are prohibited.)	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
			W							
			0	0	0	0	0	0	0	0
			Patch code (Upper 8 bits)							
ROMCMP30	Address compare register 30	418H (RMW instructions are prohibited.)	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	–
			W							–
			0	0	0	0	0	0	0	–
			Target ROM address (Lower 7 bits)							–
ROMCMP31	Address compare register 31	419H (RMW instructions are prohibited.)	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Middle 8 bits)							
ROMCMP32	Address compare register 32	41AH (RMW instructions are prohibited.)	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Upper 8 bits)							
ROMSUB3L	Address substitution register 3L	41CH (RMW instructions are prohibited.)	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
			W							
			0	0	0	0	0	0	0	0
			Patch code (Lower 8 bits)							
ROMSUB3H	Address substitution register 3H	41DH (RMW instructions are prohibited.)	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
			W							
			0	0	0	0	0	0	0	0
			Patch code (Upper 8 bits)							

Symbol	Name	Address	7	6	5	4	3	2	1	0
ROMCMP40	Address compare register 40	420H (RMW instructions are prohibited.)	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	–
			W							–
			0	0	0	0	0	0	0	–
			Target ROM address (Lower 7 bits)							–
ROMCMP41	Address compare register 41	421H (RMW instructions are prohibited.)	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Middle 8 bits)							
ROMCMP42	Address compare register 22	422H (RMW instructions are prohibited.)	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Upper 8 bits)							
ROMSUB4L	Address substitution register 4L	424H (RMW instructions are prohibited.)	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
			W							
			0	0	0	0	0	0	0	0
			Patch code (Lower 8 bits)							
ROMSUB4H	Address substitution register 4H	425H (RMW instructions are prohibited.)	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
			W							
			0	0	0	0	0	0	0	0
			Patch code (Upper 8 bits)							
ROMCMP50	Address compare register 50	428H (RMW instructions are prohibited.)	ROMC07	ROMC06	ROMC05	ROMC04	ROMC03	ROMC02	ROMC01	–
			W							–
			0	0	0	0	0	0	0	–
			Target ROM address (Lower 7 bits)							–
ROMCMP51	Address compare register 51	429H (RMW instructions are prohibited.)	ROMC15	ROMC14	ROMC13	ROMC12	ROMC11	ROMC10	ROMC09	ROMC08
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Middle 8 bits)							
ROMCMP52	Address compare register 52	42AH (RMW instructions are prohibited.)	ROMC23	ROMC22	ROMC21	ROMC20	ROMC19	ROMC18	ROMC17	ROMC16
			W							
			0	0	0	0	0	0	0	0
			Target ROM address (Upper 8 bits)							
ROMSUB5L	Address substitution register 5L	42CH (RMW instructions are prohibited.)	ROMS07	ROMS06	ROMS05	ROMS04	ROMS03	ROMS02	ROMS01	ROMS00
			W							
			0	0	0	0	0	0	0	0
			Patch code (Lower 8 bits)							
ROMSUB5H	Address substitution register 5H	42DH (RMW instructions are prohibited.)	ROMS15	ROMS14	ROMS13	ROMS12	ROMS11	ROMS10	ROMS09	ROMS08
			W							
			0	0	0	0	0	0	0	0
			Patch code (Upper 8 bits)							

16. I/O Port Equivalent-circuit Diagrams

- How to read circuit diagrams

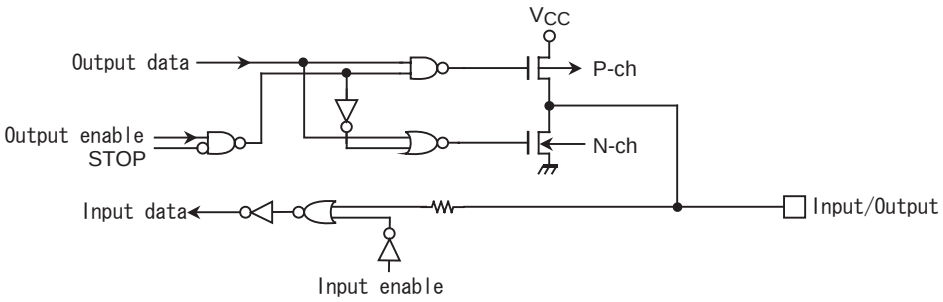
The circuit diagrams in this chapter are drawn using the same gate symbols as for the 74HCxx series standard CMOS logic ICs.

The signal named STOP has a unique function. This signal goes active-high if the CPU sets the HALT bit when the HALTM[1:0] field in the SYSCR2 register is programmed to 01 (e.g., STOP mode) and the drive enable (DRVE) bit in the same register is cleared. If the DRVE bit is set, the STOP signal remains inactive (at logic 0).

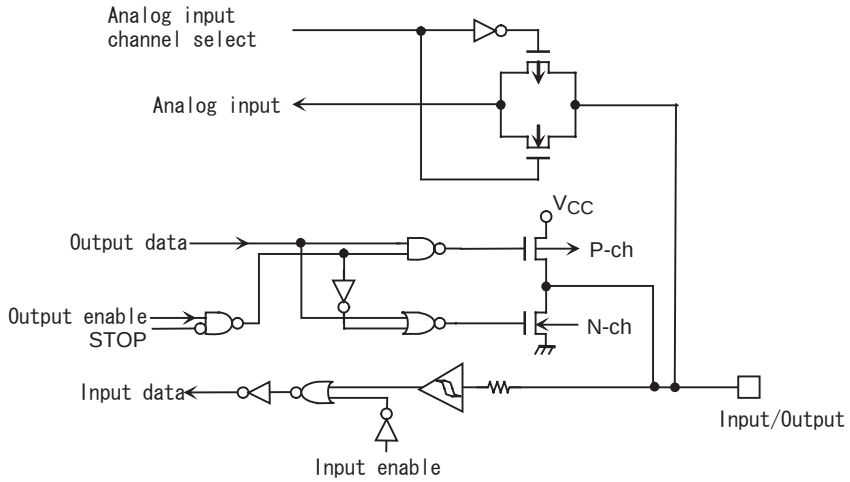
- The input protection circuit has a resistor in the range of several tens to several hundreds of ohms.

16.1 Equivalent circuit Diagrams

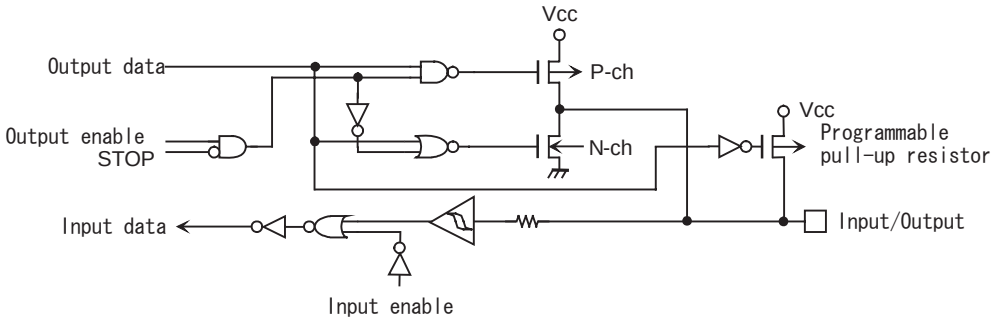
1. P0, P1



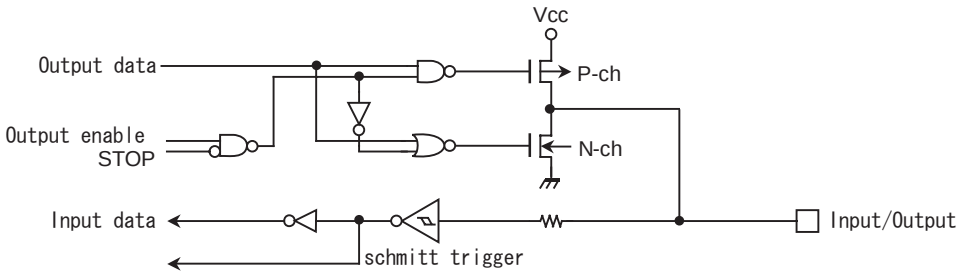
2. P5 (AN0 to AN7), P6 (AN8 to AN15)



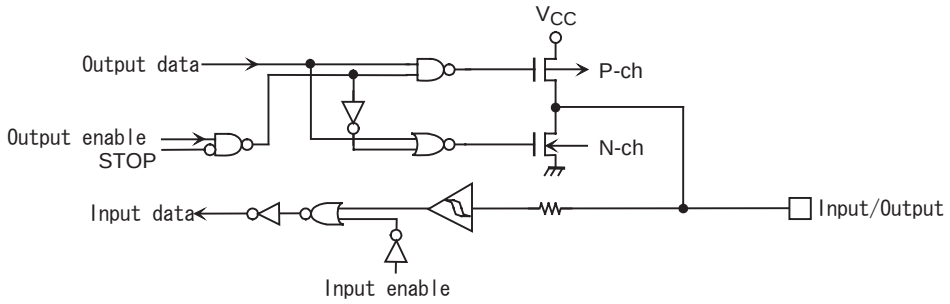
3. P40(SCOUT), P41(TXD2), P42(RXD2), P43(SCLK2/$\overline{\text{CTS2}}$)



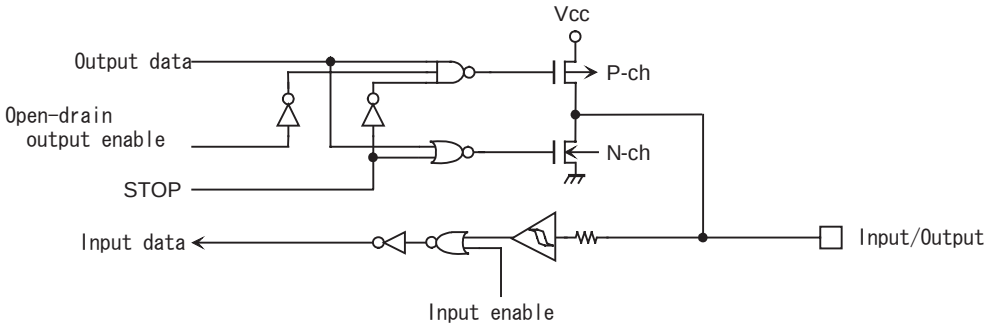
4. P75 (INT0)



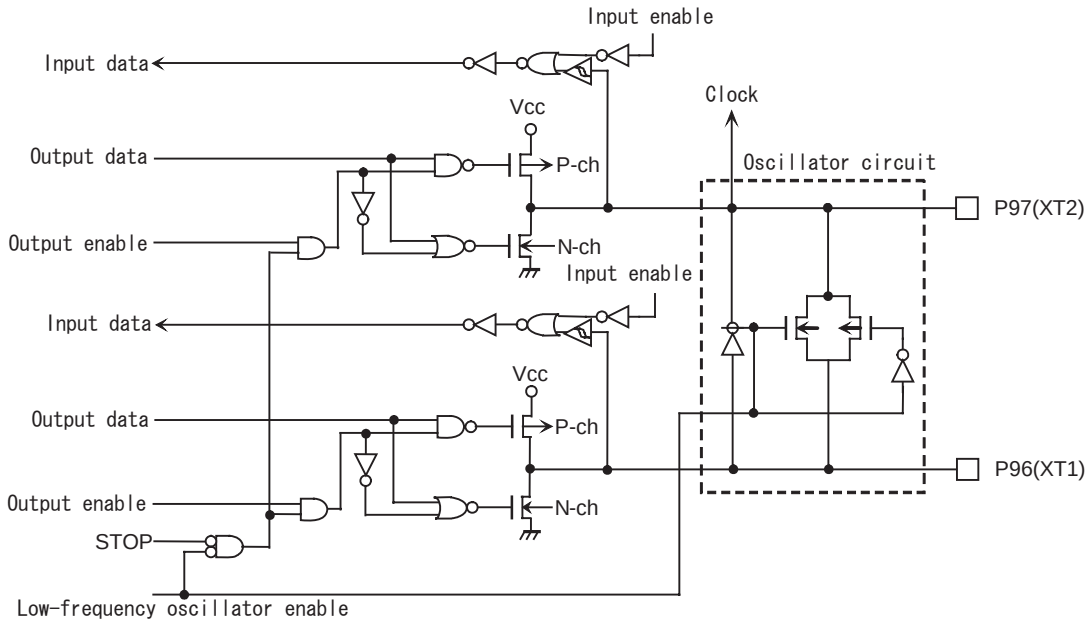
5. P32($\overline{\text{WAIT}}$/TB3OUT0), P33(TB3OUT1), P70(TA0IN), P71(TA1OUT), P72, P73(TA4IN), P74(TA5OUT), P80 to P87, P91(RXD0), P92(SCLK0/$\overline{\text{CTS0}}$), P94(RXD1), P95(SCLK1/$\overline{\text{CTS1}}$), PA0 to PA3, PB0 to PB2



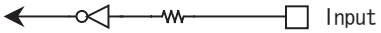
6. P30(TB3IN0/INT3/SDA0), P31(TB3IN1/INT4/SCL0), P90(TXD0), P93(TXD1)



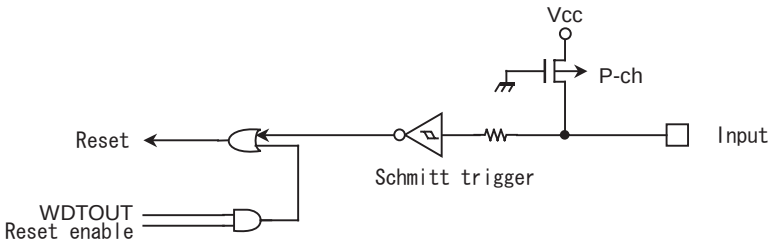
7. P96 (XT1), P97 (XT2)



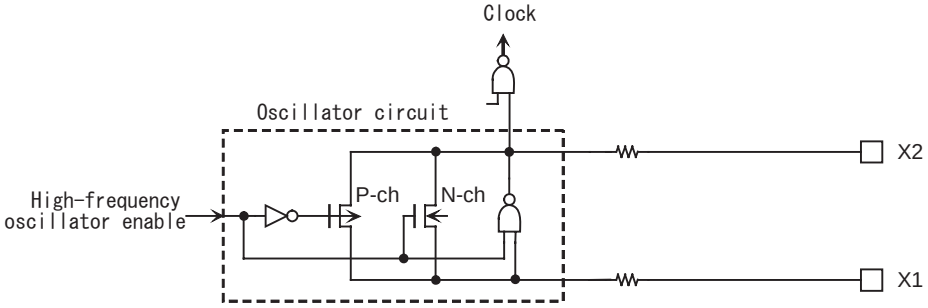
8. AM0 to AM1



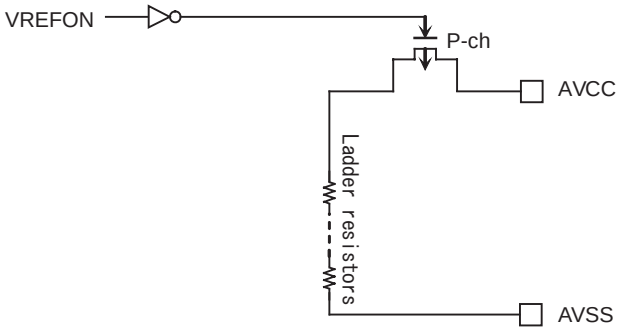
9. $\overline{\text{RESET}}$



10. X1, X2



11. AVCC, AVSS



17. Points to Note and Restrictions

17.1 Notation

a. The notation for built-in I/O registers is as follows register symbol <Bit symbol>

e.g.) TA01RUN<TA0RUN> denotes bit TA0RUN of register TA01RUN.

b. Read-modify-write instructions

An instruction in which the CPU reads data from memory and writes the data to the same memory location in one instruction.

Example 1: SET 3, (TA01RUN) ... Set bit3 of TA01RUN.

Example 2: INC 1, (100H) ... Increment the data at 100H.

- Examples of read-modify-write instructions on the TLCS-900

Exchange instruction

EX	(mem), R

Arithmetic operations

ADD	(mem), R/#	ADC	(mem), R/#
SUB	(mem), R/#	SBC	(mem), R/#
INC	#3, (mem)	DEC	#3, (mem)

Logic operations

AND	(mem), R/#	OR	(mem), R/#
XOR	(mem), R/#		

Bit manipulation operations

STCF	#3/A, (mem)	RES	#3, (mem)
SET	#3, (mem)	CHG	#3, (mem)
TSET	#3, (mem)		

Rotate and shift operations

RLC	(mem)	RRC	(mem)
RL	(mem)	RR	(mem)
SLA	(mem)	SRA	(mem)
SLL	(mem)	SRL	(mem)
RLD	(mem)	RRD	(mem)

c. f_{OSCH}, fc, fs, f_{FPH}, f_{SYS} and one state

The clock frequency input on pins X1 and 2 is called f_{OSCH} or fc.

The clock selected by SYSCR1<SYSCK> is called f_{FPH}. The clock frequency give by f_{FPH} divided by 2 is called f_{SYS}.

One cycle of f_{SYS} is referred to as one state.

17.2 Points of note

a. AM0 and AM1 pins

This pin is connected to the DVcc pin. Do not alter the level when the pin is active.

b. EMU0 pins

Open pins.

c. HALT mode (IDLE1)

When IDLE1 mode (in which oscillator operation only occurs) is used, set RTCCR<RTCRUN> to 0 stop the Special timer for CLOCK before the HALT instructions is executed.

d. Warm-up counter

The warm-up counter operates when STOP mode is released, even if the system is using an external oscillator. As a result a time equivalent to the warm-up time elapses between input of the release request and output of the system clock.

e. Programmable pull-up/pull-down resistances

The programmable pull-up/pull-down resistor can be turned ON/OFF by a program when the ports are set for use as input ports. When the ports are set for use as output prts, they cannot be turned ON/OFF by a program.

The data registers (e.g., P4) are used to turn the pull-up/pull-down resistors ON/OFF. Consequently read-modify-write instructions are prohibited.

f. Watchdog timer

The watchdog timer starts operation immediately after a reset is released. When the watchdog timer is not to be used, disable it.

When the bus is released, neither internal memory nor internal I/O can be accessed. However, the internal I/O continues to operate. Hence the watchdog timer continues to run. Therefore be careful about the bus releasing time and set the detection timer of watchdog timer.

g. CPU (Micro DMA)

Only the LCD cr, r and LDC r, cr instructions can be used to access the control registers in the CPU (e.g., the transfer source address register (DMASn)).

h. Undefined SFR

The value of an undefined bit in an SFR is undefined when read.

i. POP SR instruction

Please execute the POP SR instruction during DI condition.

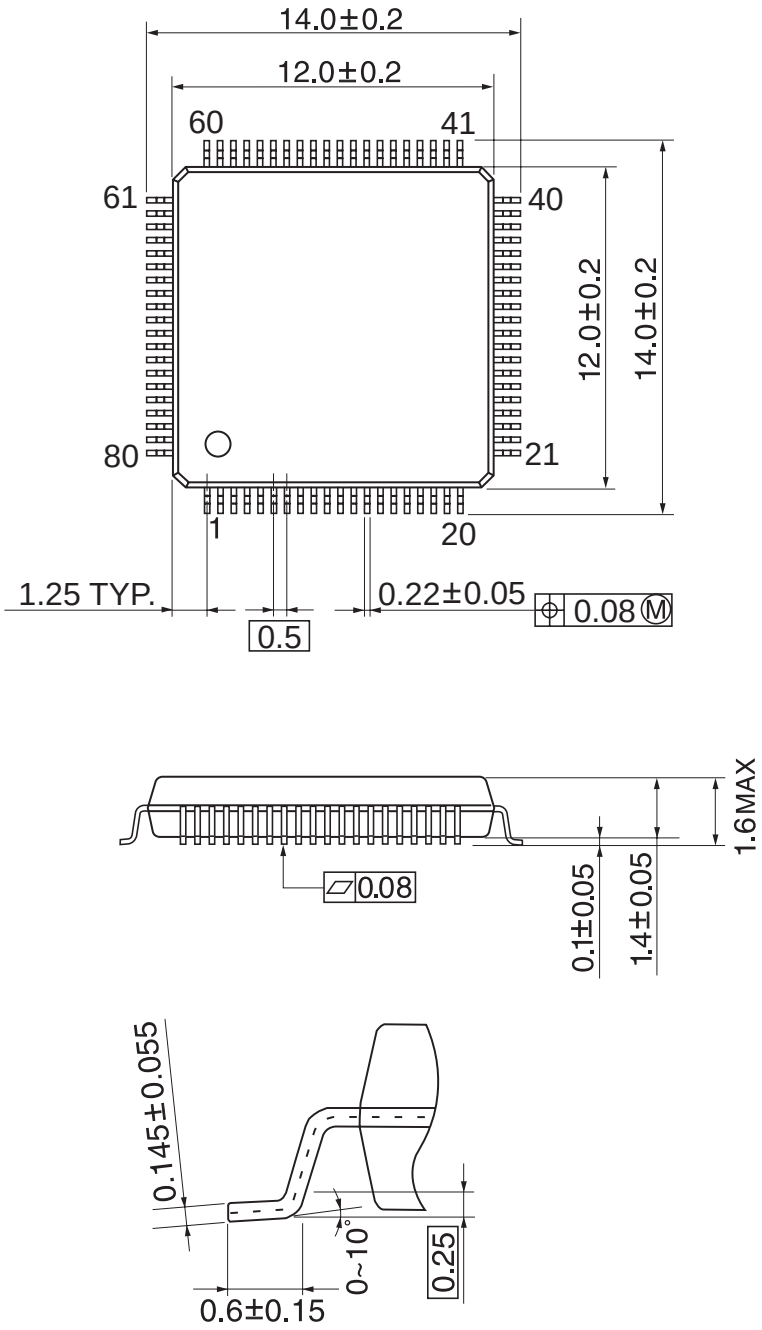
j. Clocks for serial channels (SIO)

As for the serial channels SIO0, SIO1 and SIO2, a baud rate generator is unavailable as an input clock of an I/O interface and a clock for a serial transfer if a prescaler clock is set to fc/16 when SYSCR0<PRCK1> is "1".

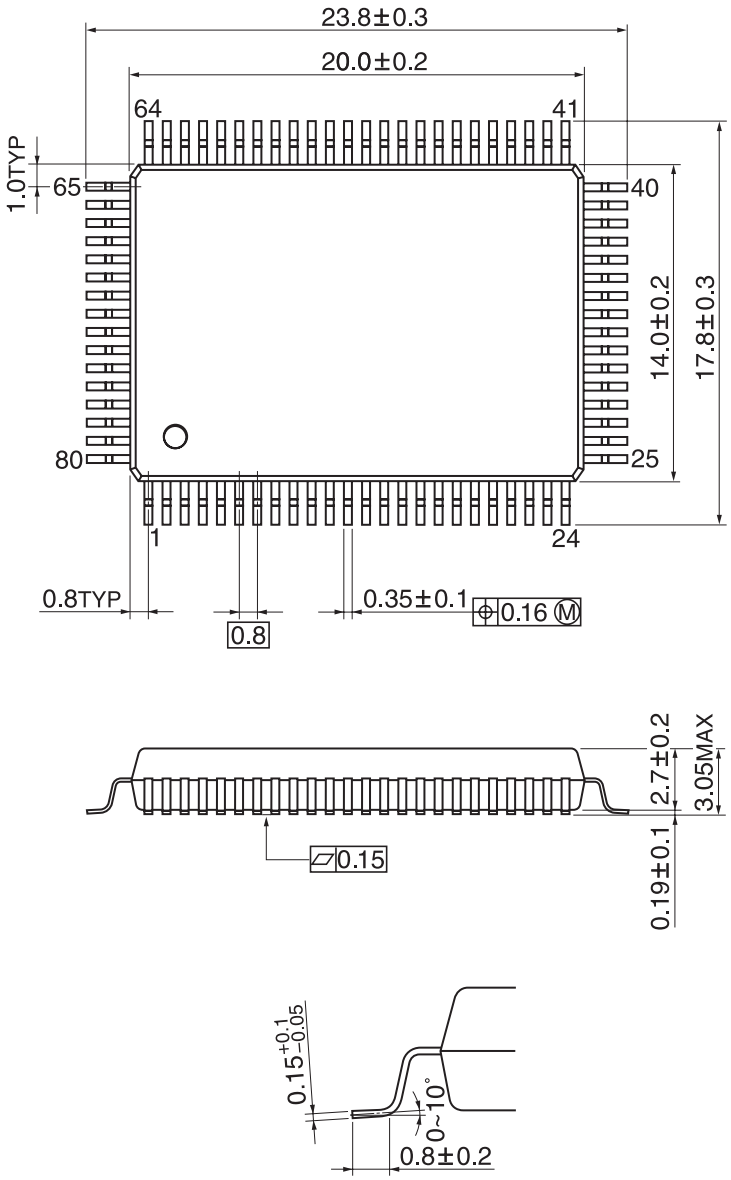
18. Package Dimension

LQFP80-P-1212-0.50E

Unit: mm



QFP80-P-1420-0.80B Unit: mm



Postscript

This is a technical document that describes the operating functions and electrical specifications of the 16-bit microcontroller series TLCS-900/L1 (LSI).

Toshiba provides a variety of development tools and basic software to enable efficient software development.

These development tools have specifications that support advances in microcomputer hardware (LSI) and can be used extensively. Both the hardware and software are supported continuously with version updates.

The recent advances in CMOS LSI production technology have been phenomenal and microcomputer systems for LSI design are constantly being improved. The products described in this document may also be revised in the future. Be sure to check the latest specifications before using.

Toshiba is developing highly integrated, high-performance microcomputers using advanced MOS production technology and especially well proven CMOS technology.

We are prepared to meet the requests for custom packaging for a variety of application areas.
We are confident that our products can satisfy your application needs now and in the future.