TOSHIBA

32-Bit TX System RISC
TX19 Family
TMP19A23FYXBG/FG

Rev1.2 March 10, 2008

32-bit RISC RISC Microprocessor - TX19 Family

TMP19A23FYFG/XBG

1. Overview and Features

The TX19 family is a high-performance 32-bit RISC processor series that TOSHIBA originally developed by integrating the MIPS16^{TM}ASE (Application Specific Extension), which is an extended instruction set of high code efficiency.

TMP19A23 is a 32-bit RISC microprocessor with a TX19A processor core and various peripheral functions integrated into one package. It can operate at low voltage with low power consumption.

Features of TMP19A23 are as follows:

RESTRICTIONS ON PRODUCT USE 070122EBP

- The information contained herein is subject to change without notice. 021023_D
- 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. 021023_A
- 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. 021023_B
- 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. 060106_Q
- 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 patents or other rights of TOSHIBA or the third parties. 070122_C
- The products described in this document are subject to foreign exchange and foreign trade control laws. 060925_E
- For a discussion of how the reliability of microcontrollers can be predicted, please refer to Section 1.3 of the chapter entitled Quality and Reliability Assurance/Handling Precautions. 030619_S

(1) TX19A processor core

1) Improved code efficiency and operating performance have been realized through the use of two ISA (Instruction Set Architecture) modes - 16- and 32-bit ISA modes.

- The 16-bit ISA mode instructions are compatible with the MIPS16e-TX instructions of superior code efficiency at the object level.
- The 32-bit ISA mode instructions are compatible with the TX39 instructions of superior operating performance at the object level.

2) Both high performance and low power consumption have been achieved.

● High performance

- Almost all instructions can be executed with one clock.
- High performance is possible via a three-operand operation instruction.
- 5-stage pipeline
- Built-in high-speed memory
- DSP function: A 32-bit multiplication and accumulation operation can be executed with one clock.

● Low power consumption

- Optimized design using a low power consumption library
- Standby function that stops the operation of the processor core

3) High-speed interrupt response suitable for real-time control

- Independency of the entry address
- Automatic generation of factor-specific vector addresses
- Automatic update of interrupt mask levels

(2) On Chip program memory and data memory

Product name	On chip Flash ROM	On chip RAM
TMP19A23FYFG	256Kbyte	24Kbyte
TMP19A23FYXBG	256Kbyte	24Kbyte

- ROM correction function : 8word x 12blocks

(3) External memory expansion

- 16M-byte off-chip address for code and data
- External data bus:
  Separate bus/multiplexed bus: Dynamic bus sizing for 8- and 16-bit widths ports.
- Chip select/wait controller : 4 channels

(4) DMA controller : 8 channels

- Data to be transferred to internal memory, internal I/O, external memory, and external I/O

(5) 16-bit timer : 12 channels

- 16-bit interval timer mode
- 16-bit event counter mode
- 16-bit PPG output
- High accuracy 16-bit PPG output : 2channels
- Event capture function
- 2-phase pulse input counter function (1 channel assigned to perform this function): Multiplication-by-4 mode

(7) General-purpose serial interface : 3 channels

- Either UART mode or synchronous mode can be selected.

(8) High speed serial interface : 1 channels

- Either UART mode or synchronous mode can be selected.

(9) Serial bus interface : 2 channel

- Either I²C bus mode or synchronous mode can be selected.

(10) 10-bit A/D converter with (S/H) : 13 channels

- Conversion speed: 1.22 μs@54 MHz / 1.15us@40MHz
- Start by an internal timer trigger
- Fixed channel/scan mode
- Single/repeat mode
- High-priority conversion mode
- Timer monitor function

(11) Watchdog timer : 1 channel

(12) Interrupt source

- CPU: 2 factors .....software interrupt instruction
- Internal: 38 factors.....The order of precedence can be set over 7 levels (except the watchdog timer interrupt).
- External: 16 factors…The order of precedence can be set over 7 levels (BGA: 15 factors).

(13) Input/output ports

- QFP : 111pins
- BGA : 103pins

(13) Standby mode

- standby modes (IDLE、STOP)

(14) Clock generator

- On-chip PLL(multiplication by 16)
- Clock gear function: The high-speed clock can be divided into 8/8, 4/8, 2/8 or 1/8.

(15) Endian: Bi-endian (big-endian/little-endian)

(16) Maximum operating frequency
- 54MHz (QFP:PLL multiplication)
- 40MHz (BGA:PLL multiplication)

(17) Operating voltage range
- QFP

  I/O, ADC : 3.0V-3.6V
- BGA

  Core : 1.35V-1.65V

  I/O, ADC : 2.7V-3.6V

(18) Package
- LQFP144-P-2020-0.50 (20mm × 20mm, 0.50mm pitch)
- TFBGA141-P-0909-0.65A5 (9mm × 9mm, 0.65mm pitch)

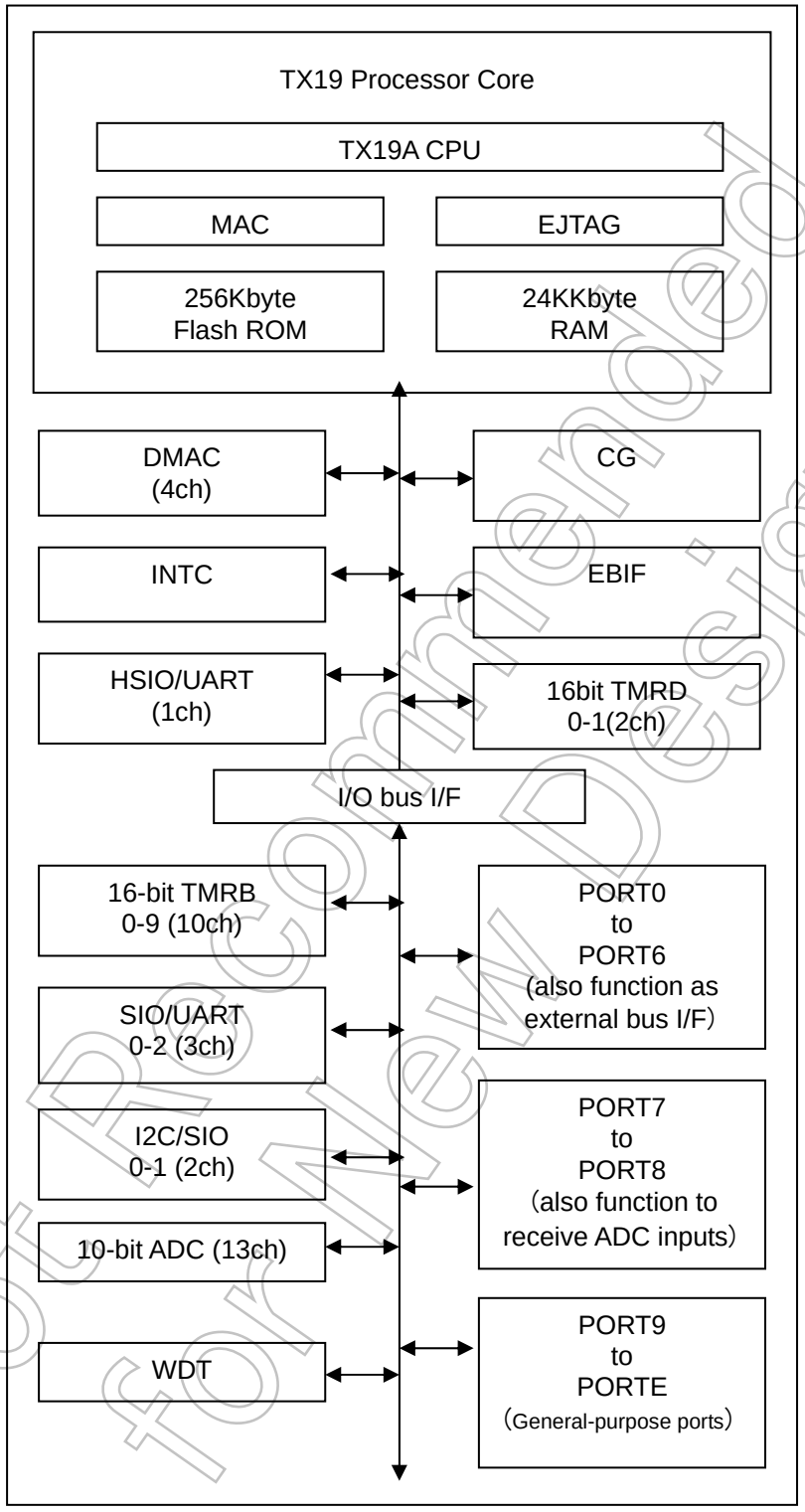


Fig 1-1 TMP19A23 Block Diagram

2. Pin Layout and Pin Functions

2.1 TMP19A23FYFG (LQFP) Pin Layout (Top view)

Fig. 2-1 shows the pin layout of TMP19A23FYFG.

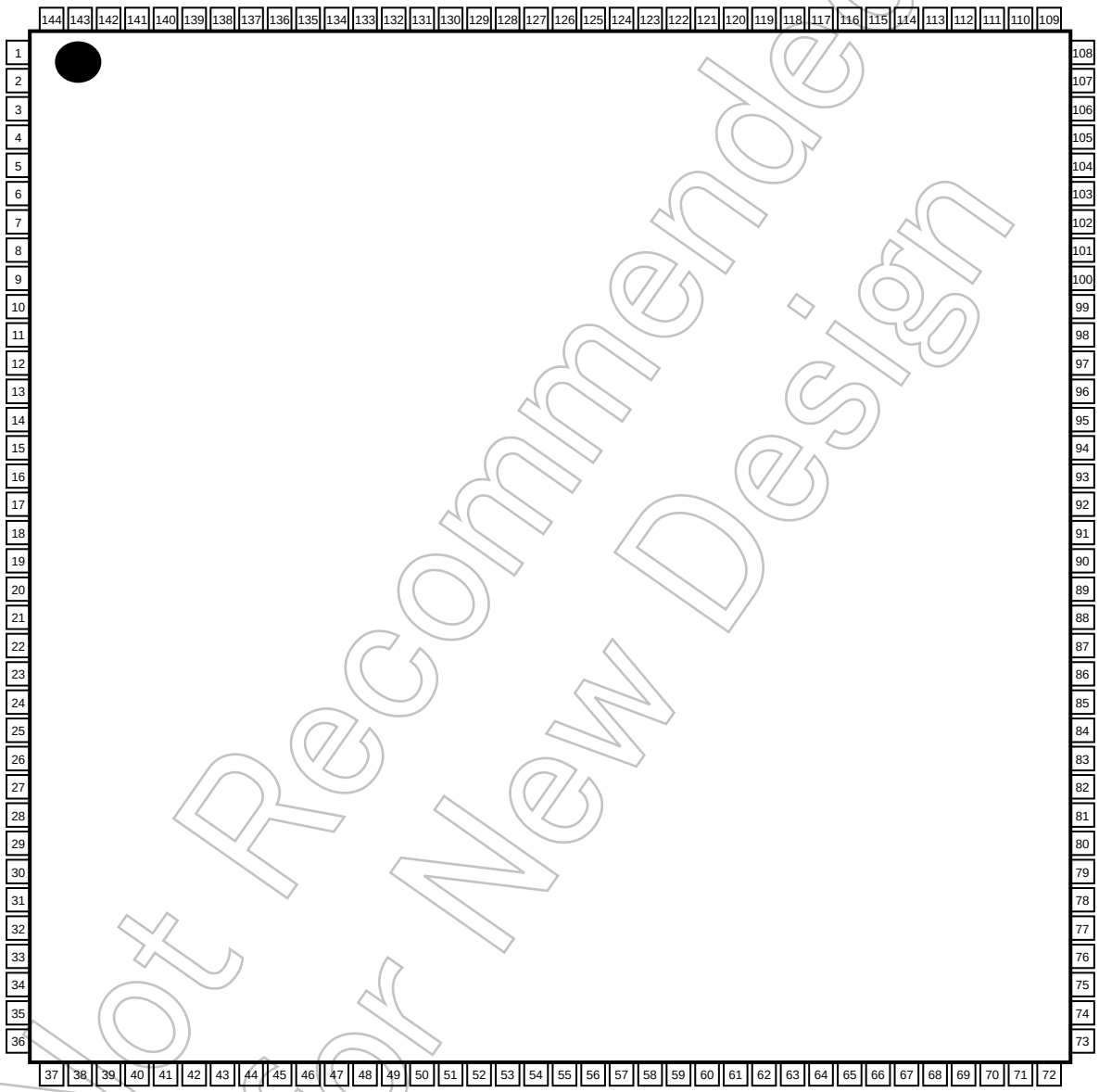


Fig 2-1 Pin Layout Diagram (LQFP144)

Table 2-1 Pin Numbers and Names (LQFP144)

Pin No.	Pin name	Pin No.	Pin name	Pin No.	Pin name	Pin No.	Pin name
1	DVCC3	37	TEST3	73	REGVOUT	109	DVCC3
2	P27/A23/A7/A23	38	P30/RD	74	DVCC15	110	TEST1
3	P26/A22/A6/A22	39	P31/WR	75	CVCC15	111	PA0/SO0/SDA0/TB6OUT
4	P25/A21/A5/A21	40	P32/HWR	76	X1	112	PA1/SI0/SCL0/TB7OUT
5	P24/A20/A4/A20	41	P33/WAIT/RDY	77	CVSS	113	PA2/SCK0/INTA
6	P23/A19/A3/A19	42	P34/BUSRQ	78	X2	114	PA3/TB1OUT/DREQ2
7	P22/A18/A2/A18	43	P35/BUSAK	79	RESET	115	PA4/SO1/SDA1
8	P21/A17/A1/A17	44	P36/R/W	80	BW0	116	PA5/SI1/SCL1
9	P20/A16/A0/A16	45	P37/ALE	81	PC0/TB3OUT / $\overline{\text{ADTRG}}$	117	PA6/SCK1/INTB
10	TEST2	46	AVCC	82	PC1/TB4OUT	118	PA7/TB2OUT/DACK2
11	P17/D15/AD15/A15	47	VREFH	83	PC2/INTE/TPC	119	PB7/TB5OUT
12	P16/D14/AD14/A14	48	P70/AIN0	84	PC3/INTF/TOVR	120	PB6/SCLK2/CTS2/INTD
13	P15/D13/AD13/A13	49	P71/AIN1	85	DVCC3	121	PB5/RXD2
14	P14/D12/AD12/A12	50	P72/AIN2	86	BW1	122	PB4/TXD2/TB5IN1
15	P13/D11/AD11/A11	51	P73/AIN3	87	PD7/TPC7/TPD7 /TB9IN1	123	PB3/INT2/TB5IN0
16	P12/D10/AD10/A10	52	P74/AIN4	88	PD6/TPC6/TPD6 /TB9IN0	124	PB2/HSCLK0/HCTS0
17	P11/D9/AD9/A9	53	P75/AIN5	89	PD5/TPC5/TPD5 /TB8IN1	125	PB1/HRXD0
18	P10/D8/AD8/A8	54	P76/AIN6	90	PD4/TPC4/TPD4 /TB8IN0	126	PB0/HTXD0
19	P07/D7/AD7	55	P77/AIN7	91	PD3/TPC3/TPD3	127	DVSS
20	P06/D6/AD6	56	P80/AIN8	92	PD2/TPC2/TPD2	128	P67/A15/TB7IN1 /TD1OUT1/TD0OUT1
21	P05/D5/AD5	57	P81/AIN9	93	PD1/TPC1/TPD1	129	P66/A14/TB7IN0 /TD1OUT0/TD0OUT0
22	P04/D4/AD4	58	P82/AIN10	94	PD0/TPC0/TPD0	130	P65/A13/TB6IN1 /TD0OUT1/TD1OUT1
23	P03/D3/AD3	59	P83/AIN11	95	DVSS	131	P64/A12/TB6IN0 /TD0OUT0/TD1OUT0
24	P02/D2/AD2	60	P84/AIN12	96	PE5/DCLK	132	P63/A11/TB0IN1/INT6
25	P01/D1/AD1	61	AVSS(VREFL)	97	PE4/PCST4	133	P62/A10/TB0IN0/INT5
26	P00/D0/AD0	62	P90/TXD0	98	PE3/PCST3	134	P61/A9/TB3IN1 /TB0OUT1
27	DVSS	63	P91/RXD0/INTC	99	PE2/PCST2	135	P60/A8/TB3IN0 /TB0OUT0
28	DVCC3	64	P92/SCLK0/CTS0 /INT7	100	PE1/PCST1	136	P57/A7/TB2IN1/DACK3
29	P40/CS0	65	P93/INT1	101	PE0/PCST0	137	P56/A6/TB2IN0/DREQ3
30	P41/CS1	66	P94/TXD1	102	EJE	138	P55/A5/TB1IN1/INT4
31	P42/CS2	67	P95/RXD1	103	DINT	139	P54/A4/TB1IN0/INT3
32	P43/CS3	68	P96/SCLK1/CTS1 /INT8	104	TCK	140	P53/A3/TB9OUT
33	P44/BUSMD	69	P97/INT9	105	TMS	141	P52/A2/TB8OUT
34	P45/ENDIAN	70	REGNOISE	106	TDO	142	P51/A1/TB7OUT
35	P46/SCOUT	71	REGGND	107	TDI	143	P50/A0/TB6OUT
36	P47/INT0	72	REGVIN	108	TRST	144	BOOT

2.2 TMP19A23FYXBG (TFBGA) Pin Layout (Top view)

Fig. 2-2 shows the pin layout of TMP19A23FYXBG.

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12
B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12
C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12
D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12
E1	E2	E3	E4	E5	E6	E7	E8	E9	E10	E11	E12
F1	F2	F3	F4	F5	F6		F8	F9	F10	F11	F12
G1	G2	G3	G4	G5			G8	G9	G10	G11	G12
H1	H2	H3	H4	H5	H6	H7	H8	H9	H10	H11	H12
J1	J2	J3	J4	J5	J6	J7	J8	J9	J10	J11	J12
K1	K2	K3	K4	K5	K6	K7	K8	K9	K10	K11	K12
L1	L2	L3	L4	L5	L6	L7	L8	L9	L10	L11	L12
M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12

Fig 2-2 Pin Layout Diagram (TFBGA141)

Table 2-2 Pin Numbers and Names (TFBGA141)

Pin No.	Pin name	Pin No.	Pin name	Pin No.	Pin name	Pin No.	Pin name
A1	NC	D1	RESET	G2	PE4/PCST4	K4	PA5/SI1/SCL1
A2	P96/SCLK1/CTS1 /INT8	D2	X2	G3	PE0/PCST0	K5	PB5/RXD2
A3	P93/INT1	D3	CVSS	G4	DVSS	K6	PB2/HSCLK0 /HCTS0/INTC
A4	P90/TXD0	D4	P94/TXD1	G5	DVCC3	K7	P65/A13/TB6IN1 /TD0OUT1
A5	P83/AIN11	D5	AVSS	G8	TEST2	K8	P60/A8/TB3IN0 /TB0OUT0
A6	P77/AIN7	D6	P81/AIN9	G9	P13/D11/AD11/A11	K9	P54/A4/TB1IN0 /INT3
A7	P74/AIN4	D7	P72/AIN2	G10	P12/D10/AD10/A10	K10	P26/A22/A6/A22
A8	AVCC	D8	P37/ALE	G11	P11/D9/AD9/A9	K11	P24/A20/A4/A20
A9	P36/R/W	D9	P45/ENDIAN	G12	P10/D8/AD8/A8	K12	P23/A19/A3/A19
A10	P33/WAIT/RDY	D10	P42/CS2	H1	PE2/PCST2	L1	DVCC3
A11	P31/WR	D11	P41/CS1	H2	PE1/PCST1	L2	TEST1
A12	TEST3	D12	P40/CS0	H3	EJE	L3	PA2/SCK0/INTA
B1	DVCC15	E1	PC2/INTE	H4	TMS	L4	PA6/SCK1/INTB
B2	P97/INT9	E2	PC1/TB4OUT	H5	PB7/TB5OUT	L5	PB3/INT2/TB5IN0
B3	P95/RXD1	E3	C0/TB3OUT/ $\overline{\text{ADTRG}}$	H6	DVSS	L6	PB0/HTXD0
B4	P91/RXD0/INTC	E4	BW0	H7	BOOT	L7	P67/A15/TB7IN1 /TD1OUT1
B5	P82/AIN10	E5	DVSS	H8	DVCC3	L8	P63/A11/TB0IN1 /INT6
B6	P76/AIN6	E6	VREFL	H9	P17/D15/AD15/A15	L9	P55/A5/TB1IN1 /INT4
B7	P75/AIN5	E7	VREFH	H10	P16/D14/AD14/A14	L10	P52/A2/TB8OUT
B8	P71/AIN1	E8	DVCC3	H11	P14/D12/AD12/A12	L11	P27/A23/A7/A23
B9	P35/BUSAK	E9	P00/D0/AD0	H12	P15/D13/AD13/A13	L12	P25/A21/A5/A21
B10	P32/HWR	E10	P02/D2/AD2	J1	DINT	M1	NC
B11	P47/INT0	E11	P03/D3/AD3	J2	TCK	M2	PA1/SI0/SCL0 /TB7OUT
B12	P46/SCOUT	E12	P01/D1/AD1	J3	TDI	M3	PA4/SO1/SDA1
C1	X1	F1	PD0/TPC0	J4	PA3/TB1OUT/DREQ2	M4	PA7/TB2OUT /DACK2
C2	CVCC15	F2	BW1	J5	PB6/SCLK2/CTS2 /INTD	M5	PB4/TXD2/TB5IN1
C3	DVCC15	F3	PE3/PCST3	J6	P64/A12/TB6IN0 /TD0OUT0	M6	PB1/HRXD0
C4	P92/SCLK0/CTS0 /INT7	F4	TEST4	J7	P61/A9/TB3IN1 /TB0OUT1	M7	P66/A14/TB7IN0 /TD1OUT0
C5	P84/AIN12	F5	DVCC3	J8	P57/A7/TB2IN1 /DACK3	M8	P62/A10/TB0IN0 /INT5
C6	P80/AIN8	F6	NC	J9	P50/A0/TB6OUT	M9	P56/A6/TB2IN0 /DREQ3
C7	P73/AIN3	F8	DVSS	J10	P22/A18/A2/A18	M10	P53/A3/TB9OUT
C8	P70/AIN0	F9	P04/D4/AD4	J11	P21/A17/A1/A17	M11	P51/A1/TB7OUT
C9	P34/BUSRQ	F10	P05/D5/AD5	J12	P20/A16/A0/A16	M12	NC
C10	P30/RD	F11	P06/D6/AD6	K1	TDO		
C11	P44/BUSMD	F12	P07/D7/AD7	K2	TRST		
C12	P43/CS3	G1	PE5/DCLK	K3	PA0/SO0/SDA0 /TB6OUT		

2. 3 Pin names and Functions

2. 3. 1 Pin Usage Information: TMP19A23FYFG

Table2-3 lists the input and output pins of the TMP19A23FYFG, including alternate pin names and functions for multi-function pins.

Table 2-3 Pin Names and Functions (1/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
PS	1	DVCC3	−	Power supply pin			
Function	2	P27	I/O	I/O port	Pull up		
		A23	O	Address bus 23			
		A7	O	Address bus 7			
	3	P26	I/O	I/O port	Pull up		
		A22	O	Address bus 22			
		A6	O	Address bus 6			
	4	P25	I/O	I/O port	Pull up		
		A21	O	Address bus 21			
		A5	O	Address bus 5			
	5	P24	I/O	I/O port	Pull up		
		A20	O	Address bus 20			
		A4	O	Address bus 4			
	6	P23	I/O	I/O port	Pull up		
		A19	O	Address bus 19			
		A3	O	Address bus 3			
	7	P22	I/O	I/O port	Pull up		
		A18	O	Address bus 18			
		A2	O	Address bus 2			
	8	P21	I/O	I/O port	Pull up		
		A17	O	Address bus 17			
		A1	O	Address bus 1			
	9	P20	I/O	I/O port	Pull up		
		A16	O	Address bus 16			
		A0	O	Address bus 0			
Test	10	TEST2	I	TEST pin: Set to OPEN.			
Function	11	P17	I/O	I/O port	Pull up		
		D15	I/O	Data bus 15			
		AD15	I/O	Address/ data bus 15			
		A15	O	Address bus 15			
	12	P16	I/O	I/O port	Pull up		
		D14	I/O	Data bus 14			
		AD14	I/O	Address/ data bus 14			
		A14	O	Address bus 14			
	13	P15	I/O	I/O port	Pull up		
		D13	I/O	Data bus 13			
		AD13	I/O	Address/ data bus 13			
		A13	O	Address bus 13			
	14	P14	I/O	I/O port	Pull up		
		D12	I/O	Data bus 12			
		AD12	I/O	Address/ data bus 12			
		A12	O	Address bus 12			
	15	P13	I/O	I/O port	Pull up		
		D11	I/O	Data bus 11			
		AD11	I/O	Address/ data bus 11			
		A11	O	Address bus 11			

Table 2-3 Pin Names and Functions (2/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	16	P12 D10 AD10 A10	I/O I/O I/O O	I/O port Data bus 10 Address/ data bus 10 Address bus 10	Pull up		
	17	P11 D9 AD9 A9	I/O I/O I/O O	I/O port Data bus 9 Address/ data bus 9 Address bus 9	Pull up		
	18	P10 D8 AD8 A8	I/O I/O I/O O	I/O port Data bus 8 Address/ data bus 8 Address bus 8	Pull up		
	19	P07 D7 D7	I/O I/O I/O	I/O port Data bus 7 Address/ data bus 7	Pull up		
	20	P06 D6 D6	I/O I/O I/O	I/O port Data bus 6 Address/ data bus 6	Pull up		
	21	P05 D5 D5	I/O I/O I/O	I/O port Data bus 5 Address/ data bus 5	Pull up		
	22	P04 D4 D4	I/O I/O I/O	I/O port Data bus 4 Address/ data bus 4	Pull up		
	23	P03 D3 D3	I/O I/O I/O	I/O port Data bus 3 Address/ data bus 3	Pull up		
	24	P02 D2 D2	I/O I/O I/O	I/O port Data bus 2 Address/ data bus 2	Pull up		
	25	P01 D1 D1	I/O I/O I/O	I/O port Data bus 1 Address/ data bus 1	Pull up		
	26	P00 D0 AD0	I/O I/O I/O	I/O port Data bus 0 Address/ data bus 0	Pull up		
PS	27	DVSS	−	GND pin			
	28	DVCC3	−	Power supply pin			
Function	29	P40 $\overline{CS0}$	I/O O	I/O port Chip select 0	Pull up		○
	30	P41 $\overline{CS1}$	I/O I	I/O port Chip select 1	Pull up		○
	31	P42 $\overline{CS2}$	I/O I	I/O port Chip select 2	Pull up		○
	32	P43 $\overline{CS3}$	I/O I	I/O port Chip select 3	Pull up		○
	33	P44 BUSMD	I/O I	I/O port Pin for setting an external bus mode: When performing a reset operation, pull it up or down according to a bus mode to be used. Multiplex bus mode: H Separate bus mode: L	Pull up	○	○

Table 2-3 Pin Names and Functions (3/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	34	P45 ENDIAN	I/O I	I/O port Pin for selecting a little/ big endian. When performing a reset operation, pull it up or down according to the type of endian to be used. Big endian: H Little endian: L	Pull up	○	○
	35	P46 SCOUT	I/O I	I/O port System clock output: Selectable between high- and low-speed clock outputs, as in the case of CPU.	Pull up		○
	36	P47 INT0	I/O I	I/O port Interrupt request pin	Pull up	○ w/ noise filter	○
Test	37	TEST3	I	TEST pin: Set to OPEN.			
Function	38	P30 $\overline{RD}$	I/O O	I/O port Read Strobe	Pull up		
	39	P31 $\overline{WR}$	I/O O	I/O port Write Strobe: Strobe signal for writing data of D0 to D7 pins	Pull up		
	40	P32 $\overline{HWR}$	I/O O	I/O port Higher Write Strobe: Strobe signal for writing data of D8 to D15 pins	Pull up		
	41	P33 $\overline{WAIT}$ $\overline{RDY}$	I/O I I	I/O port Wait: Pin for requesting CPU to put a bus in a wait state Ready: Pin for notifying CPU that a bus is ready	Pull up		
	42	P34 $\overline{BUSRQ}$	I/O I	I/O port Bus request: Signal requesting CPU to allow an external master to take the bus control authority	Pull up		
	43	P35 $\overline{BUSAK}$	I/O O	I/O port Bus acknowledge: Signal notifying that CPU has released the bus control authority in response to $\overline{BUSRQ}$	Pull up		
	44	P36 R/$\overline{W}$	I/O O	I/O port Read/write: "1" shows a read cycle or a dummy cycle. "0" shows a write cycle.	Pull up		
	45	P37 ALE	I/O O	I/O port Address latch enable (address latch is enabled only when using multiplex bus mode)	Pull up		
PS	46	AVCC	−	Pin for supplying the A/D converter with a power supply. Connect it to a power supply even if the A/D converter is not used.			
	47	VREFH	−	Pin (L) for supplying the A/D converter with a reference power supply. Connect this pin to AVCC3 if the A/D converter is not used.			

Table 2-3 Pin Names and Functions (4/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	48	P70 AIN0	I I	Port used exclusively for input Analog input	Pull up		
	49	P71 AIN1	I/O I	Port used exclusively for input Analog input	Pull up		
	50	P72 AIN2	I/O I	Port used exclusively for input Analog input	Pull up		
	51	P73 AIN3	I/O I	Port used exclusively for input Analog input	Pull up		
	52	P74 AIN4	I/O I	Port used exclusively for input Analog input	Pull up		
	53	P75 AIN5	I/O I	Port used exclusively for input Analog input	Pull up		
	54	P76 AIN6	I/O I	Port used exclusively for input Analog input	Pull up		
	55	P77 AIN7	I/O I	Port used exclusively for input Analog input	Pull up		
	56	P80 AIN8	I I	Port used exclusively for input Analog input	Pull up		
	57	P81 AIN9	I I	Port used exclusively for input Analog input	Pull up		
	58	P82 AIN10	I I	Port used exclusively for input Analog input	Pull up		
	59	P83 AIN11	I I	Port used exclusively for input Analog input	Pull up		
	60	P84 AIN12	I I	Port used exclusively for input Analog input	Pull up		
PS	61	AVSS	−	A/D converter: GND pin (0V) Connect it to a power supply even if the A/D converter is not used.			
Function	62	P90 TXD0	I/O O	I/O port Sending serial data **(5V tolerant input available)**		○	For open drain Exclusive use * Attach a pull up resistor to output a high pulse
	63	P91 RXD0 INTC	I/O I I	I/O port I/O port Receiving serial data Interrupt request pin **(5V tolerant input available)**		○ w/ noise filter	
	64	P92 SCLK0 CTS0 INT7	I/O I/O I I	I/O port Serial clock input/ output Handshake input pin Interrupt request pin **(5V tolerant input available)**		○ w/ noise filter	
	65	P93 INT 1	I/O I	I/O port Interrupt request pin **(5V tolerant input available)**		○ w/ noise filter	
	66	P94 TXD1	I/O O	I/O port Sending serial data **(5V tolerant input available)**		○	
	67	P95 RXD1	I/O I	I/O port Receiving serial data **(5V tolerant input available)**		○	

Table 2-3 Pin Names and Functions (5/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	68	P96 SCLK1 $\overline{\text{CTS1}}$ INT8	I/O I/O I I	I/O port Serial clock input/ output Handshake input pin Interrupt request pin **(5V tolerant input available)**		○ w/ noise filter	For open drain Exclusive use * Attach a pull up resistor to output a high pulse
	69	P97 INT9	I/O I	port97 I/O port Interrupt request pin **(5V tolerant input available)**			
PS	70	REGNOISE	—	Output reduce pin: Please connect condenser in between REGNOISE pin and GND to reduce the output noise voltage (recommended condenser volume is under estimation of 0.01μF).			
	71	DVSS	—	GND pin			
	72	REGVIN	—	Regulator input pin: 3V power supply (Recommended condenser volume: 1μF).			
	73	REGVOUT	—	Regulator output pin: Connect with DVCC15/CVCC15 (Recommended condenser volume: 10μF).			
	74	DVCC15	—	Connect with REGVOUT pin			
	75	CVCC15	—	Pin for supplying a high-frequency oscillator with power: Connect with REGVOUT pin			
Clock	76	X1	I/O	Pin for connecting a high-speed oscillator.		○	
PS	77	CVSS	—	GND pin			
Clock	78	X2	I/O	Pin for connecting a high-speed oscillator.			
Reset	79	$\overline{\text{RESET}}$	I	Reset input pin	Set to Pull up	○ w/ noise filter	
Test	80	BW0	I	TEST pin: To be fixed to DVCC3		○	
Function	81	PC0 TB3OUT $\overline{\text{ADTRG}}$	I/O O I	I/O port Timer B output Inputting ADC trigger.	Pull up	○ w/ noise filter	○
	82	PC1 TB4OUT	I/O O	I/O port Timer B output	Pull up		○
Function/ Debug	83	PC2 INTE TPC	I/O I O	I/O port Interrupt request pin Outputting trace data from the program counter	Pull up	○ w/ noise filter	○
	84	PC3 INTF TOVR	I/O O O	I/O port Interrupt request pin Outputting trace data from the program counter	Pull up	○ w/ noise filter	○
PS	85	DVCC3	—	Power supply pin			
	86	BW1	—	TEST pin: To be fixed to DVCC3		○	
Function/ Debug	87	PD7 TPC7 TPD7 TB9IN1	I/O O O	I/O port Outputting trace data from the program counter Outputting trace data from the data access address For inputting the capture trigger of timer B	Pull up	○	
	88	PD6 TPC6 TPD6 TB9IN0	I/O O O	I/O port Outputting trace data from the program counter Outputting trace data from the data access address For inputting the capture trigger of timer B	Pull up	○	

Table 2-3 Pin Names and Functions (6/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull-up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function/ Debug	89	PD5 TPC5 TPD5 TB8IN1	I/O O O 	I/O port Outputting trace data from the program counter Outputting trace data from the data access address For inputting the capture trigger of timer B	Pull up	○	
	90	PD4 TPC4 TPD4 TB8IN0	I/O O O I	I/O port Outputting trace data from the program counter Outputting trace data from the data access address For inputting the capture trigger of timer B	Pull up		
	91	PD3 TPC3 TPD3	I/O O O	I/O port Outputting trace data from the program counter Outputting trace data from the data access address	Pull up		
	92	PD2 TPC2 TPD2	I/O O O	I/O port Outputting trace data from the program counter Outputting trace data from the data access address	Pull up		
	93	PD1 TPC1 TPD1	I/O O O	I/O port Outputting trace data from the program counter Outputting trace data from the data access address	Pull up		
	94	PD0 TPC0 TPD0	I/O O O	I/O port Outputting trace data from the program counter Outputting trace data from the data access address	Pull up		
PS	95	DVSS	−	GND pin			
Function/ Debug	96	PE5 DCLK	I/O O	I/O port Debug clock: Signal for DSU-ICE	Pull up		
	97	PE4 PCST4	I/O O	I/O port PC trace status: signal for DSU-ICE	Pull up		
	98	PE3 PCST3	I/O O	I/O port PC trace status: signal for DSU-ICE	Pull up		
	99	PE2 PCST2	I/O O	I/O port PC trace status: signal for DSU-ICE	Pull up		
	100	PE1 PCST1	I/O O	I/O port PC trace status: signal for DSU-ICE	Pull up		
	101	PE0 PCST0	I/O O	I/O port PC trace status: signal for DSU-ICE	Pull up		
Debug	102	$\overline{EJE}$	I	EJTAG enable: Signal for DSU	Set to Pull up	○ w/ noise filter	
	103	$\overline{DINT}$	I	Debug interrupt: Signal for DSU-ICE	Set to Pull up	○ w/ noise filter	
	104	TCK	I	Test clock input: Signal for DSU	Set to Pull up	○ w/ noise filter	
	105	TMS	I	Test mode select input: Signal for DSU	Set to Pull up	○ w/ noise filter	
	106	TDO	O	Test data output: Signal for DSU			

Table 2-3 Pin Names and Functions (7/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull-up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Debug	107	TDI	I	Test data input: Signal for DSU	Set to Pull up	○	
	108	$\overline{\text{TRST}}$	I	Test reset input: Signal for DSU	Set to P-down	○ w/ noise filter	
PS	109	DVCC3	−	Power supply pin			
Test	110	TEST1	−	TEST pin: Set to OPEN.			
Function	111	PA0 SO0 SDA0 TB6OUT	I/O O I/O O	I/O port Pin for sending data if the serial bus interface operates in the SIO mode. Pin for sending and receiving data if the serial bus interface operates in the I2C mode. Timer B output	Pull up	○	○
	112	PA1 SI0 SCL0 TB7OUT	I/O I I/O O	I/O port Pin for receiving data if the serial bus interface operates in the SIO mode. Pin for inputting a clock if the serial bus interface operates in the I2C mode Timer B output	Pull up	○	○
	113	PA2 SCK0 INTA	I/O I/O I	I/O port Pin for inputting and outputting a clock if the serial bus interface operates in the SIO mode. Interrupt request pin	Pull up	○ w/ noise filter	○
	114	PA3 TB1OUT $\overline{\text{DREQ2}}$	I/O O I	I/O port For inputting the capture trigger of timer B DMA request signal: For inputting the request to transfer data by DMA from an external I/O device to DMAC.	Pull up		
	115	PA4 SO1 SDA1	I/O O I/O	Port A4: I/O port Pin for sending data if the serial bus interface operates in the SIO mode. Pin for sending and receiving data if the serial bus interface operates in the I2C mode. **(5V tolerant input available)**		○	For open drain Exclusive use * Attach a pull up resistor to output a high pulse
	116	PA5 SI1 SCL1	I/O I I/O	I/O port Pin for receiving data if the serial bus interface operates in the SIO mode. Pin for inputting a clock if the serial bus interface operates in the I2C mode **(5V tolerant input available)**		○	
	117	PA6 SCK1 INTB	I/O I/O I	I/O port Pin for inputting and outputting a clock if the serial bus interface operates in the SIO mode. Interrupt request pin **(5V tolerant input available)**		○	
	118	PA7 TB2OUT $\overline{\text{DACK2}}$	I/O O O	I/O port Timer B output DMA acknowledge signals: Signal showing that DREQ2 has acknowledged a DMA transfer request	Pull up		
	119	PB7 TB5OUT	I/O I	I/O port Timer B output	Pull up		

Table 2-3 Pin Names and Functions (8/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	120	PB6 SCLK2 CTS2 INTD	I/O I/O I I	I/O port Serial clock input / output Handshake input pin Interrupt request pin	Pull up	○ w/ noise filter	○
	121	PB5 RXD2	I/O I	I/O port Receiving serial data	Pull up		
	122	PB4 TB5IN1 TXD2	I/O I O	I/O port For inputting the capture trigger of timer B Sending serial data	Pull up	○	○
	123	PB3 TB5IN0 INT2	I/O I I	I/O port For inputting the capture trigger of timer B Interrupt request pin	Pull up	○ w/ noise filter	
	124	PB2 HSCLK0 $\overline{\text{HCTS0}}$	I/O I/O I	I/O port High-speed serial clock input/output 0 Handshake input pin	Pull up		○
	125	PB1 HRXD0	I/O I	I/O port Receiving serial data at high speed	Pull up		
	126	PB0 HTXD0	I/O O	I/O port Sending serial data 0 at high speeds	Pull up		○
PS	127	DVSS	−	GND pin			
Function	128	P67 A15 TB7IN1 TD1OUT1 TD0OUT1	I/O O I O O	I/O port Address bus 15 For inputting the capture trigger of timer B Timer D PPG output Timer D PPG output	Pull up	○	
	129	P66 A14 TB7IN0 TD1OUT0 TD0OUT0	I/O O I O O	I/O port Address bus 14 For inputting the capture trigger of timer B Timer D PPG output Timer D PPG output	Pull up	○	
	130	P65 A13 TB6IN1 TD0OUT1 TD1OUT1	I/O O I O O	I/O port Address bus 13 For inputting the capture trigger of timer B Timer D PPG output Timer D PPG output	Pull up	○	
	131	P64 A12 TB6IN0 TD0OUT0 TD1OUT0	I/O O I O O	I/O port Address bus 12 For inputting the capture trigger of timer B Timer D PPG output Timer D PPG output	Pull up	○	
	132	P63 A11 TB0IN1 INT6	I/O I I I	I/O port Address bus 11 For inputting the capture trigger of timer B Interrupt request pin	Pull up	○ w/ noise filter	
	133	P62 A10 TB0IN0 INT5	I/O O I I	I/O port Address bus 10 For inputting the capture trigger of timer B Interrupt request pin	Pull up	○ w/ noise filter	
	134	P61 A9 TB3IN1 TB0OUT1	I/O O I O	I/O port Address bus 9 For inputting the capture trigger of timer B 16-bit timer 0 output 1: 2 phase counter output pin. Open drain output pin depending on the program used.	Pull up	○	

Table 2-3 Pin Names and Functions (9/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	135	P60 A8 TB3IN0 TB0OUT0	I/O O I O	I/O port Address bus 8 For inputting the capture trigger of timer B 16-bit timer 0 output 0: 2 phase counter output pin. Open drain output pin depending on the program used.	Pull up	○	
	136	P57 A7 TB2IN1 $\overline{\text{DACK3}}$	I/O O I O	I/O port Address bus 7 For inputting the capture trigger of timer B DMA acknowledge signals	Pull up	○	
	137	P56 A6 TB2IN0 $\overline{\text{DREQ3}}$	I/O O I I	I/O port Address bus 6 For inputting the capture trigger of timer B DMA request signal	Pull up	○	
	138	P55 A5 TB1IN1 INT4	I/O O I I	I/O port Address bus 5 For inputting the capture trigger of timer B Interrupt request pin	Pull up	○ w/ noise filter	
	139	P54 A4 TB1IN0 INT3	I/O O I I	I/O port Address bus 4 For inputting the capture trigger of timer B Interrupt request pin	Pull up	○ w/ noise filter	
	140	P53 A3 TB9OUT	I/O O O	I/O port Address bus 3 Timer B output	Pull up		○
	141	P52 A2 TB8OUT	I/O O O	I/O port Address bus 2 Timer B output	Pull up		○
	142	P51 A1 TB7OUT	I/O O O	I/O port Address bus 1 Timer B output	Pull up		○
	143	P50 A0 TB6OUT	I/O O O	I/O port Address bus 0 Timer B output	Pull up		○
Boot	144	$\overline{\text{BOOT}}$	I	Pin for setting a single boot mode: This pin goes into single boot mode by sampling "L" at the rise of a reset signal. It is used to overwrite internal flash memory. By sampling "H (DVCC3) level" at the rise of a reset signal, it performs a normal operation. This pin should be pulled up under normal operating conditions. Pull it up when resetting. (with internal pull up resistor)	Pull up	○	

2.3.2 Pin Usage Information:TMP19A23FYXBG

Table2-4 lists the input and output pins of the TMP19A23FYXBG, including alternate pin names and functions for multi-function pins.

Table 2-4 Pin Names and Functions (1/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
	A1	NC	—	Non Connection *No connections allowed			
Function	A2	P96 SCLK1 $\overline{CTS1}$ INT8	I/O I/O I I	I/O port Serial clock input/ output Handshake input pin Interrupt request pin **(5V tolerant input available)**		○ w/ noise filter	For open drain Exclusive use * Attach a pull up resistor to output a high pulse
	A3	P93 INT1	I/O I	I/O port Interrupt request pin **(5V tolerant input available)**		○ w/ noise filter	
	A4	P90 TXD0	I/O O	I/O port Sending serial data **(5V tolerant input available)**		○	
	A5	P83 AIN11	I I	Port used exclusively for input Analog input	Pull up		
	A6	P77 AIN7	I/O I	Port used exclusively for input Analog input	Pull up		
	A7	P74 AIN4	I/O I	Port used exclusively for input Analog input	Pull up		
PS	A8	AVCC	—	Pin for supplying the A/D converter with a power supply. Connect it to a power supply even if the A/D converter is not used.			
Function	A9	P36 R/$\overline{W}$	I/O O	I/O port Read/write: "1" shows a read cycle or a dummy cycle. "0" shows a write cycle.	Pull up		
	A10	P33 $\overline{WAIT}$ $\overline{RDY}$	I/O I I	I/O port Wait: Pin for requesting CPU to put a bus in a wait state Ready: Pin for notifying CPU that a bus is ready	Pull up		
	A11	P31 $\overline{WR}$	I/O O	I/O port Write Strobe: Strobe signal for writing data of D0 to D7 pins	Pull up		
Test	A12	TEST3	I	TEST pin: Set to OPEN.			
PS	B1	DVCC15	—	1.5V Power supply pin			
Function	B2	P97 INT9	I/O I	Port 97 I/O port Interrupt request pin **(5V tolerant input available)**		○ w/ noise filter	For open drain Exclusive use * Attach a pull up resistor to output a high pulse
	B3	P95 RXD1	I/O I	I/O port Receiving serial data **(5V tolerant input available)**		○	
	B4	P91 RXD0 INTC	I/O I I	I/O port Receiving serial data Interrupt request pin **(5V tolerant input available)**		○ w/ noise filter	
	B5	P82 AIN10	I I	Port used exclusively for input Analog input	Pull up		

Table 2-4 Pin Names and Functions (2/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	B6	P76 AIN6	I/O I	Port used exclusively for input Analog input	Pull up		
	B7	P75 AIN5	I/O I	Port used exclusively for input Analog input	Pull up		
	B8	P71 AIN1	I/O I	Port used exclusively for input Analog input	Pull up		
	B9	P35 $\overline{\text{BUSAK}}$	I/O O	I/O port Bus acknowledge: Signal notifying that CPU has released the bus control authority in response to $\overline{\text{BUSRQ}}$	Pull up		
	B10	P32 $\overline{\text{HWR}}$	I/O O	I/O port Higher Write Strobe: Strobe signal for writing data of D8 to D15 pins	Pull up		
	B11	P47 INT0	I/O I	I/O port Interrupt request pin	Pull up	○ w/ noise filter	○
	B12	P46 SCOUT	I/O I	I/O port System clock output: Selectable between high- and low-speed clock outputs, as in the case of CPU.	Pull up		○
Clock	C1	X1	I/O	Pin for connecting a high-speed oscillator.		○	
PS	C2	CVCC15	−	Pin for supplying a high-frequency oscillator with power:1.5V Power supply pin			
	C3	DVCC15	−	1.5V Power supply pin			
Function	C4	P92 SCLK0 $\overline{\text{CTS0}}$ INT7	I/O I/O I I	I/O port Serial clock input/ output Handshake input pin Interrupt request pin **(5V tolerant input available)**		○ w/ noise filter	For open drain Exclusive use * Attach a pull up resistor to output a high pulse
	C5	P84 AIN12	I I	Port used exclusively for input Analog input	Pull up		
	C6	P80 AIN8	I I	Port used exclusively for input Analog input	Pull up		
	C7	P73 AIN3	I/O I	Port used exclusively for input Analog input	Pull up		
	C8	P70 AIN0	I I	Port used exclusively for input Analog input	Pull up		
	C9	P34 $\overline{\text{BUSRQ}}$	I/O I	I/O port Bus request: Signal requesting CPU to allow an external master to take the bus control authority	Pull up		
	C10	P30 $\overline{\text{RD}}$	I/O O	I/O port Read Strobe	Pull up		
	C11	P44 BUSMD	I/O I	I/O port Pin for setting an external bus mode: When performing a reset operation, pull it up or down according to a bus mode to be used. Multiplex bus mode: H Separate bus mode: L	Pull up	○	○

Table 2-4 Pin Names and Functions (3/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	C12	P43 CS3	I/O I	I/O port Chip select 3	Pull up		○
Reset	D1	$\overline{\text{RESET}}$	I	Reset input pin	Set to Pull up	○ w/ noise filter	
Clock	D2	X2	I/O	Pin for connecting a high-speed oscillator.			
PS	D3	CVSS	−	GND pin for supplying a high-frequency oscillator			
Function	D4	P94 TXD1	I/O O	I/O port Sending serial data **(5V tolerant input available)**		○	For open drain Exclusive use * Attach a pull up resistor to output a high pulse
PS	D5	AVSS	−	A/D converter: GND pin (0V) Connect it to a GND pin even if the A/D converter is not used.			
Function	D6	P81 AIN9	I I	Port used exclusively for input Analog input	Pull up		
Function	D7	P72 AIN2	I/O I	Port used exclusively for input Analog input	Pull up		
	D8	P37 ALE	I/O O	I/O port Address latch enable (address latch is enabled only when using multiplex bus mode)	Pull up		
	D9	P45 ENDIAN	I/O I	I/O port Pin for selecting a little/ big endian. When performing a reset operation, pull it up or down according to the type of endian to be used. Big endian: H Little endian: L	Pull up	○	○
	D10	P42 CS2	I/O I	I/O port Chip select 2	Pull up		○
	D11	P41 CS1	I/O I	I/O port Chip select 1	Pull up		○
	D12	P40 CS0	I/O O	I/O port Chip select 0	Pull up		○
Function/ Debug	E1	PC2 INTE	I/O I O	I/O port Interrupt request pin Outputting trace data from the program counter	Pull up	○ w/ noise filter	○
Function	E2	PC1 TB4OUT	I/O O	I/O port Timer B output	Pull up		○
	E3	PC0 TB3OUT $\overline{\text{ADTRG}}$	I/O O I	I/O port Timer B output Inputting ADC trigger.	Pull up	○ w/ noise filter	○
Test	E4	BW0	I	TEST pin: To be fixed to DVCC3		○	

Table 2-4 Pin Names and Functions (4/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
PS	E5	DVSS	−	GND pin			
	E6	VREFL	−	Pin (L) for supplying the A/D converter with a reference power supply. Connect this pin to AVSS if the A/D converter is not used.			
	E7	VREFH	−	Pin (H) for supplying the A/D converter with a reference power supply. Connect this pin to AVCC3 if the A/D converter is not used.			
	E8	DVCC3	−	Power supply pin			
Function	E9	P00 D0 AD0	I/O I/O I/O	I/O port Data bus 0 Address/ data bus 0	Pull up		
	E10	P02 D2 AD2	I/O I/O I/O	I/O port Data bus 2 Address/ data bus 2	Pull up		
	E11	P03 D3 AD3	I/O I/O I/O	I/O port Data bus 3 Address/ data bus 3	Pull up		
	E12	P01 D1 AD1	I/O I/O I/O	I/O port Data bus 1 Address/ data bus 1	Pull up		
Function/ Debug	F1	PD0 TPC0	I/O O	I/O port Outputting trace data from the program counter	Pull up		
PS	F2	BW1	−	TEST pin: To be fixed to DVCC3		○	
Function/ Debug	F3	PE3 PCST3	I/O O	I/O port PC trace status: signal for DSU-ICE	Pull up		
Test	F4	TEST4	I	TEST pin: To be fixed to DVSS.			
PS	F5	DVCC3	−	3V Power supply pin			
	F6	NC	−	Non Connection *No connections allowed			
PS	F8	DVSS	−	GND pin			
Function	F9	P04 D4 AD4	I/O I/O I/O	I/O port Data bus 4 Address/ data bus 4	Pull up		
	F10	P05 D5 AD5	I/O I/O I/O	I/O port Data bus 5 Address/ data bus 5	Pull up		
	F11	P06 D6 AD6	I/O I/O I/O	I/O port Data bus 6 Address/ data bus 6	Pull up		
	F12	P07 D7 AD7	I/O I/O I/O	I/O port Data bus 7 Address/ data bus 7	Pull up		
Function/ Debug	G1	PE5 DCLK	I/O O	I/O port Debug clock: Signal for DSU-ICE	Pull up		
	G2	PE4 PCST4	I/O O	I/O port PC trace status: signal for DSU-ICE	Pull up		
	G3	PE0 PCST0	I/O O	I/O port PC trace status: signal for DSU-ICE	Pull up		
PS	G4	DVSS	−	GND pin			
	G5	DVCC3	−	3V Power supply pin			
Test	G8	TEST2	I	TEST pin: Set to OPEN.			

Table 2-4 Pin Names and Functions (5/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	G9	P13 D11 AD11 A11	I/O I/O I/O O	I/O port Data bus 11 Address/ data bus 11 Address bus 11	Pull up		
	G10	P12 D10 AD10 A10	I/O I/O I/O O	I/O port Data bus 10 Address/ data bus 10 Address bus 10	Pull up		
	G11	P11 D9 AD9 A9	I/O I/O I/O O	I/O port Data bus 9 Address/ data bus 9 Address bus 9	Pull up		
	G12	P10 D8 AD8 A8	I/O I/O I/O O	I/O port Data bus 8 Address/ data bus 8 Address bus 8	Pull up		
Function/ Debug	H1	PE2 PCST2	I/O O	I/O port PC trace status: signal for DSU-ICE	Pull up		
	H2	PE1 PCST1	I/O O	I/O port PC trace status: signal for DSU-ICE	Pull up		
Debug	H3	$\overline{\text{EJE}}$	I	EJTAG enable: Signal for DSU	Set to Pull up	○ w/ noise filter	
	H4	TMS	I	Test mode select input: Signal for DSU	Set to Pull up	○ w/ noise filter	
Function	H5	PB7 TB5OUT	I/O I	I/O port Timer B output	Pull up		
PS	H6	DVSS	−	GND pin			
Boot	H7	$\overline{\text{BOOT}}$	I	Pin for setting a single boot mode: This pin goes into single boot mode by sampling "L" at the rise of a reset signal. It is used to overwrite internal flash memory. By sampling "H (DVCC3) level" at the rise of a reset signal, it performs a normal operation. This pin should be pulled up under normal operating conditions. Pull it up when resetting. (with internal pull up resistor)	Pull up	○	
PS	H8	DVCC3	−	3V Power supply pin			
Function	H9	P17 D15 AD15 A15	I/O I/O I/O O	I/O port Data bus 15 Address/ data bus 15 Address bus 15	Pull up		
	H10	P16 D14 AD14 A14	I/O I/O I/O O	I/O port Data bus 14 Address/ data bus 14 Address bus 14	Pull up		
	H11	P14 D12 AD12 A12	P14 D12 AD12 A12	I/O port Data bus 12 Address/ data bus 12 Address bus 12	Pull up		

Table 2-4 Pin Names and Functions (6/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	H12	P15 D13 AD13 A13	I/O I/O I/O O	I/O port Data bus 13 Address/ data bus 13 Address bus 13	Pull up		
Debug	J1	$\overline{\text{DINT}}$	I	Debug interrupt: Signal for DSU-ICE	Set to Pull up	○ w/ noise filter	
	J2	TCK	I	Test clock input: Signal for DSU	Set to Pull up	○ w/ noise filter	
	J3	TDI	I	Test data input: Signal for DSU	Set to Pull up	○	
Function	J4	PA3 TB1OUT $\overline{\text{DREQ2}}$	I/O O I	I/O port For inputting the capture trigger of timer B DMA request signal: For inputting the request to transfer data by DMA from an external I/O device to DMAC.	Pull up		
	J5	PB6 SCLK2 CTS2 INTD	I/O I/O I I	I/O port Serial clock input/ output Handshake input pin Interrupt request pin	Pull up	○ w/ noise filter	○
	J6	P64 A12 TB6IN0 TD0OUT0	I/O O I O O	I/O port Address bus 12 For inputting the capture trigger of timer B Timer D PPG output Timer D PPG output	Pull up	○	
	J7	P61 A9 TB3IN1 TB0OUT1	I/O O I O	I/O port Address bus 9 For inputting the capture trigger of timer B 16-bit timer 0 output 1: 2 phase counter output pin. Open drain output pin depending on the program used.	Pull up	○	
	J8	P57 A7 TB2IN1 $\overline{\text{DACK3}}$	I/O O I O	I/O port Address bus 7 For inputting the capture trigger of timer B DMA acknowledge signals	Pull up	○	
	J9	P50 A0 TB6OUT	I/O O O	I/O port Address bus 0 Timer B output	Pull up		○
	J10	P22 A18 A2	I/O O O	I/O port Address bus 18 Address bus 2	Pull up		
	J11	P21 A17 A1	I/O O O	I/O port Address bus 17 Address bus 1	Pull up		
	J12	P20 A16 A0	I/O O O	I/O port Address bus 16 Address bus 0	Pull up		
Debug	K1	TDO	O	Test data output: Signal for DSU			
	K2	$\overline{\text{TRST}}$	I	Test reset input: Signal for DSU	Set to P-down	○ w/ noise filter	

Table 2-4 Pin Names and Functions (7/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	K3	PA0 SO0 SDA0 TB6OUT	I/O O I/O O	I/O port Pin for sending data if the serial bus interface operates in the SIO mode. Pin for sending and receiving data if the serial bus interface operates in the I2C mode. Timer B output	Pull up	○	○
	K4	PA5 SI1 SCL1	I/O I I/O	I/O port Pin for receiving data if the serial bus interface operates in the SIO mode. Pin for inputting a clock if the serial bus interface operates in the I2C mode. **(5V tolerant input available)**		○	For open drain Exclusive use * Attach a pull up resistor to output a high pulse
	K5	PB5 RXD2	I/O I	I/O port Receiving serial data	Pull up		
	K6	PB2 HSCLK0 $\overline{\text{HCTS0}}$	I/O I/O I	I/O port High-speed serial clock input/output 0 Handshake input pin	Pull up		○
	K7	P65 A13 TB6IN1 TD0OUT1	I/O O I O	I/O port Address bus 13 For inputting the capture trigger of timer B Timer D PPG output	Pull up	○	
	K8	P60 A8 TB3IN0 TB0OUT0	I/O O I O	I/O port Address bus 8 For inputting the capture trigger of timer B 16-bit timer 0 output 0: 2 phase counter output pin. Open drain output pin depending on the program used.	Pull up	○	
	K9	P54 A4 TB1IN0 INT3	I/O O I I	I/O port Address bus 4 For inputting the capture trigger of timer B Interrupt request pin	Pull up	○ w/ noise filter	
	K10	P26 A22 A6	I/O O O	I/O port Address bus 22 Address bus 6	Pull up		
	K11	P24 A20 A4	I/O O O	I/O port Address bus 20 Address bus 4	Pull up		
	K12	P23 A19 A3	I/O O O	I/O port Address bus 19 Address bus 3	Pull up		
PS	L1	DVCC3	—	3V Power supply pin			
Test	L2	TEST1	—	TEST pin: Set to OPEN.			
Function	L3	PA2 SCK0 INTA	I/O I/O I	I/O port Pin for inputting and outputting a clock if the serial bus interface operates in the SIO mode. Interrupt request pin	Pull up	○ w/ noise filter	○

Table 2-4 Pin Names and Functions (8/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	L4	PA6 SCK1 INTB	I/O I/O I	I/O port Pin for inputting and outputting a clock if the serial bus interface operates in the SIO mode. Interrupt request pin **(5V tolerant input available)**		○	For open drain Exclusive use * Attach a pull up resistor to output a high pulse
	L5	PB3 TB5IN0 INT2	I/O I I	I/O port For inputting the capture trigger of timer B Interrupt request pin	Pull up	○ w/ noise filter	
	L6	PB0 HTXD0	I/O O	I/O port Sending serial data 0 at high speeds	Pull up		○
	L7	P67 A15 TB7IN1 TD1OUT1	I/O O I O	I/O port Address bus 15 For inputting the capture trigger of timer B Timer D PPG output	Pull up	○	
	L8	P63 A11 TB0IN1 INT6	I/O I I I	I/O port Address bus 11 For inputting the capture trigger of timer B Interrupt request pin	Pull up	○ w/ noise filter	
	L9	P55 A5 TB1IN1 INT4	I/O O I I	I/O port Address bus 5 For inputting the capture trigger of timer B Interrupt request pin	Pull up	○ w/ noise filter	
	L10	P52 A2 TB8OUT	I/O O O	I/O port Address bus 2 Timer B output	Pull up		○
	L11	P27 A23 A7	I/O O O	I/O port Address bus 23 Address bus 7	Pull up		
	L12	P25 A21 A5	I/O O O	I/O port Address bus 21 Address bus 5	Pull up		
	M1	NC	−	Non Connection *No connections allowed			
Function	M2	PA1 SI0 SCL0 TB7OUT	I/O I I/O O	I/O port Pin for receiving data if the serial bus interface operates in the SIO mode. Pin for inputting a clock if the serial bus interface operates in the I2C mode Timer B output	Pull up	○	○

Table 2-4 Pin Names and Functions (9/9)

Type	# of Pins	Pin Name	Input/ Output	Function	Programmable Pull up/ Pull down	Schmitt trigger	Programmable Open Drain Output
Function	M3	PA4 SO1 SDA1	I/O O I/O	Port A4:I/O port Pin for sending data if the serial bus interface operates in the SIO mode. Pin for sending and receiving data if the serial bus interface operates in the I2C mode. **(5V tolerant input available)**		○	For open drain Exclusive use * Attach a pull up resistor to output a high pulse
	M4	PA7 TB2OUT $\overline{\text{DACK2}}$	I/O O O	I/O port Timer B output DMA acknowledge signals: Signal showing that DREQ2 has acknowledged a DMA transfer request	Pull up		
	M5	PB4 TXD2 TB5IN1	I/O I O	I/O port For inputting the capture trigger of timer B Sending serial data	Pull up	○	○
	M6	PB1 HRXD0	I/O I	I/O port Receiving serial data at high speed	Pull up		
	M7	P66 A14 TB7IN0 TD1OUT0	I/O O I O	I/O port Address bus 14 For inputting the capture trigger of timer B Timer D PPG output	Pull up	○	
	M8	P62 A10 TB0IN0 INT5	I/O O I I	I/O port Address bus 10 For inputting the capture trigger of timer B Interrupt request pin	Pull up	○ w/ noise filter	
	M9	P56 A6 TB2IN0 $\overline{\text{DREQ3}}$	I/O O I I	I/O port Address bus 6 For inputting the capture trigger of timer B DMA request signal	Pull up	○	
	M10	P53 A3 TB9OUT	I/O O O	I/O port Address bus 3 Timer B output	Pull up		○
	M11	P51 A1 TB7OUT	I/O O O	I/O port Address bus 1 Timer B output	Pull up		○
	M12	NC	−	Non Connection *No connections allowed			

2.4 Pin Names and Power Supply Pins

Table 2-5 Pin Names and Power Supplies

Pin name	Power supply	Pin name	Power supply
P0	DVCC3	PCST4-0	DVCC3
P1	DVCC3	DCLK	DVCC3
P2	DVCC3	$\overline{EJE}$	DVCC3
P3	DVCC3	$\overline{TRST}$	DVCC3
P4	DVCC3	TDI	DVCC3
P5	DVCC3	TDO	DVCC3
P6	DVCC3	TMS	DVCC3
P7	AVCC3	TCK	DVCC3
P8	AVCC3	$\overline{DINT}$	DVCC3
P9	DVCC3	$\overline{RESET}$	DVCC3
PA	DVCC3	$\overline{BOOT}$	DVCC3
PB	DVCC3	X1、X2	CVCC15
PC	DVCC3		
PD	DVCC3		
PE	DVCC3		

2.5 Pin Numbers and Power Supply Pins

Table 2-6 Pin Numbers and Power Supplies for QFP

Power supply	Pin number	Voltage range
DVCC3	13, 37, 73, 100	3.0V-3.6V
AVCC	118	3.0V-3.6V
REGVIN	144	3.0V-3.6V

Note: Please connect REGVOUT, DVCC15 and CVCC15 outside of mocrocomputer.

Table 2-7 Pin Numbers and Power Supplies for BGA

Power supply	Pin number	Voltage range
DVCC15	B1, C3,	1.35V-1.65V
DVCC3	E8, F5, G5, H8	2.7V-3.6V
AVCC	A8	2.7V-3.6V
CVCCH	C2	1.35V-1.65V

3. Processor Core

The TMP19A23 has a high-performance 32-bit processor core (TX19A processor core). For information on the operations of this processor core, please refer to the "TX19A Family Architecture."

This chapter describes the functions unique to the TMP19A23 that are not explained in that document.

3.1 Reset Operation

To reset the device, ensure that the power supply voltage is in the operating voltage range, the oscillation of the internal high-frequency oscillator has stabilized at the specified frequency and that the $\overline{\text{RESET}}$ input has been "0" for at least 12 system clocks (1.78 μs during external 13.5MHz operation/ 2.4 μs during external 10 MHz operation).

Note that the PLL multiplication clock is quadrupled and the clock gear is initialized to the 1/8 mode during the reset period. When the reset request is authorized, the system control coprocessor (CP0) register of the TX19A processor core is initialized. For further details, please refer to the chapter about architecture.

After the reset exception handling is executed, the program branches off to the exception handler. The address to which the program branches off to (address where exception handling starts) is called an exception vector address. This exception vector address of a reset exception (for example, non-maskable interrupt) is 0xBFC0_0000 (virtual address).

The register of the internal I/O is initialized.

The port pin (including the pin that can also be used by the internal I/O) is set to a general-purpose input or output port mode.

(Note 1) Set the **$\overline{\text{RESET}}$** pin to "0" before turning the power on. Cancel the reset after the power supply voltage has stabilized sufficiently within the operating range.

(Note 2) After turning the power on, make sure that the power supply voltage and oscillation have stabilized, wait for 500 μs or longer, and cancel the reset.

(Note 3) In the FLASH program, the reset period of 0.5μs or longer is required independently of the system clock.

(Note 4) The reset operation may alter the internal RAM state, but does not alter data in the backup RAM.

4. Memory Map

Fig. 4-1 shows the memory map of the TMP19A23.

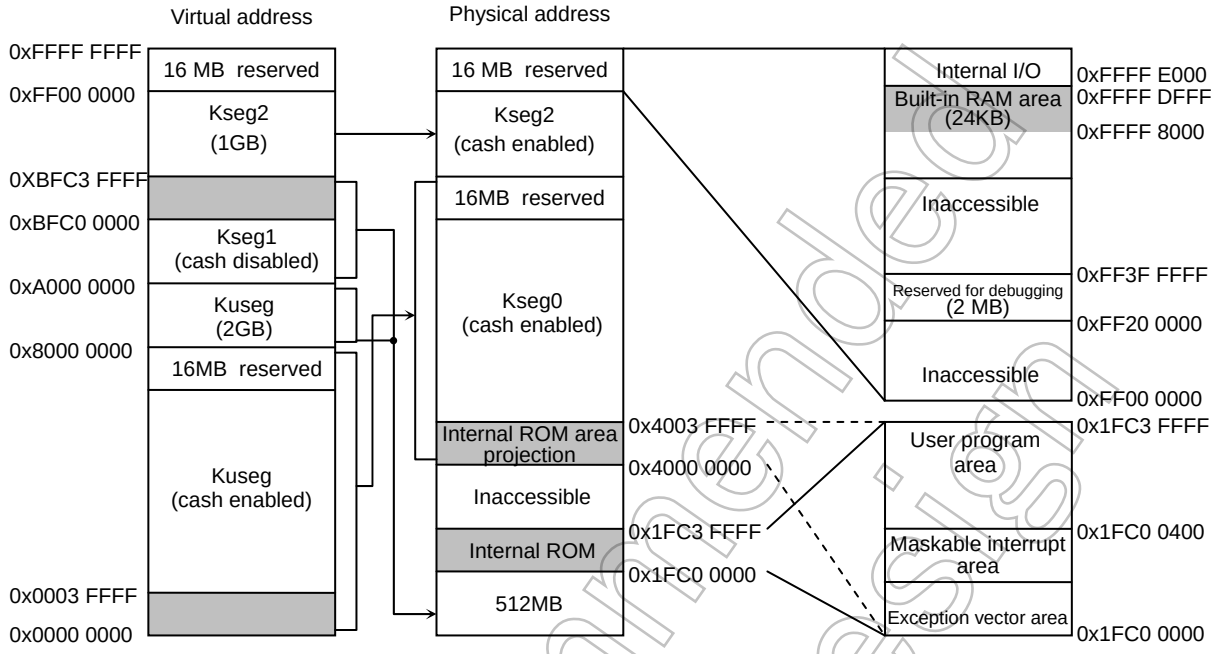


Fig. 4-1 Memory Map

(Note 1) The internal ROM is mapped to: 0x1FC0_0000~0x1FC3_FFFF (256KB)
The internal RAM is mapped to: 0xFFFF_8000~0xFFFF_DFFF (24KB)

(Note 2) For the TMP19A23, a physical space of only 16 MB is available as external address space to be accessed. It is possible to place this 16-MB physical address space in a chip select area of your choice inside the 3.5-GB physical address space of the CPU.
Access to internal memory, internal I/O space and reserved areas is given priority over access to the external address space. Therefore, access to the external address space is denied if any of the internal memory, internal I/O space or reserved areas are being accessed.

(Note 3) Do not place an instruction in the last four words of a physical area, specifically the last four words of an area where memory is mounted for external ROM extension (this varies depending on the system of the user).
Internal ROM: 0x1FC3_FFF0~0x1FC3_FFFF(256KB)

5. Clock/ Standby Control

The system operation modes contain the standby modes in which the processor core operations are stopped to reduce power dissipation. Fig. 5-1 State Transition Diagram of Single Clock Mode is shown below.

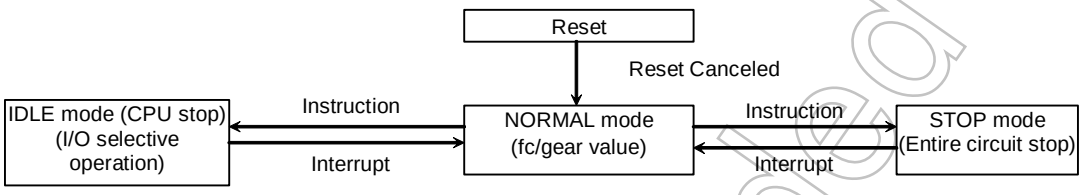


Fig. 5-1 State Transition Diagram of Single Clock Mode

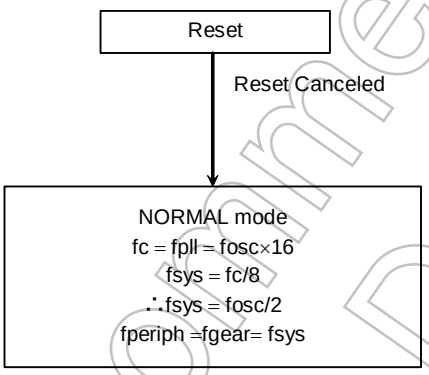


Fig. 5-2 Default State of the System Clock

fosc : Clock frequency to be input via the X1 and X2 pins

fpll : Clock frequency multiplied (16 times) by the PLL

fc : High-frequency clock frequency

fgear : Clock frequency selected by the system control register SYSCR1<GEAR2:0> in the clock generator

fsys : System clock frequency

The CPU, ROM, RAM, DMAC, INTC, HSIO and TMRD all operate according to this clock. The internal peripheral I/O operates according to the fsys/2 clock.

fperiph : Clock frequency selected by SYSCR1<FPSEL>

（Clock to be input to the peripheral I/O prescaler)

5.1 Clock System Block Diagram

5.1.1 Main System Clock

- Allows for oscillator connection or external clock input.
- Clock gear (1/2, 1/4, 1/8)
  (Default is 1/8.)
- Input frequency (high frequency)

	Input frequency range	Maximum operating frequency	Lowest operating frequency
QFP	8-13.5 MHz	54 MHz（Note 1）	4 MHz (Note 2)
BGA	8-10.0 MHz	40 MHz	4 MHz (Note 2)

(Note 1） Temporary
(Note 2) Clock gear 1/8 (default) is used when 8 MHz (MIN) is input.

(Note) (precautions for switching the high-speed clock gear)

Switching of clock gear is executed when a value is written to the SYSCR1<GEAR2:0> register. There are cases where switching does not occur immediately after the change in the register setting but the original clock gear is used for execution of instructions. If it is necessary to use the new clock for execution of the instructions following to the clock gear switching instruction, insert a dummy instruction (to execute a write cycle) and a SYNC instruction.
To use the clock gear, ensure that you make the time setting such that ϕTn of the prescaler output from each block in the peripheral I/O is calibrated to ϕTn<fsys/2 (ϕTn becomes slower than fsys/2). Do not switch the clock gear during operation of the timer counter or other peripheral I/O.

5.1.2 Clock Gear

- The high-speed clock is divided into 1/2, 1/4 or 1/8.
- The internal I/O prescaler clock ϕT0: fperiph/2, fperiph/4, fperiph/8 and fperiph/16

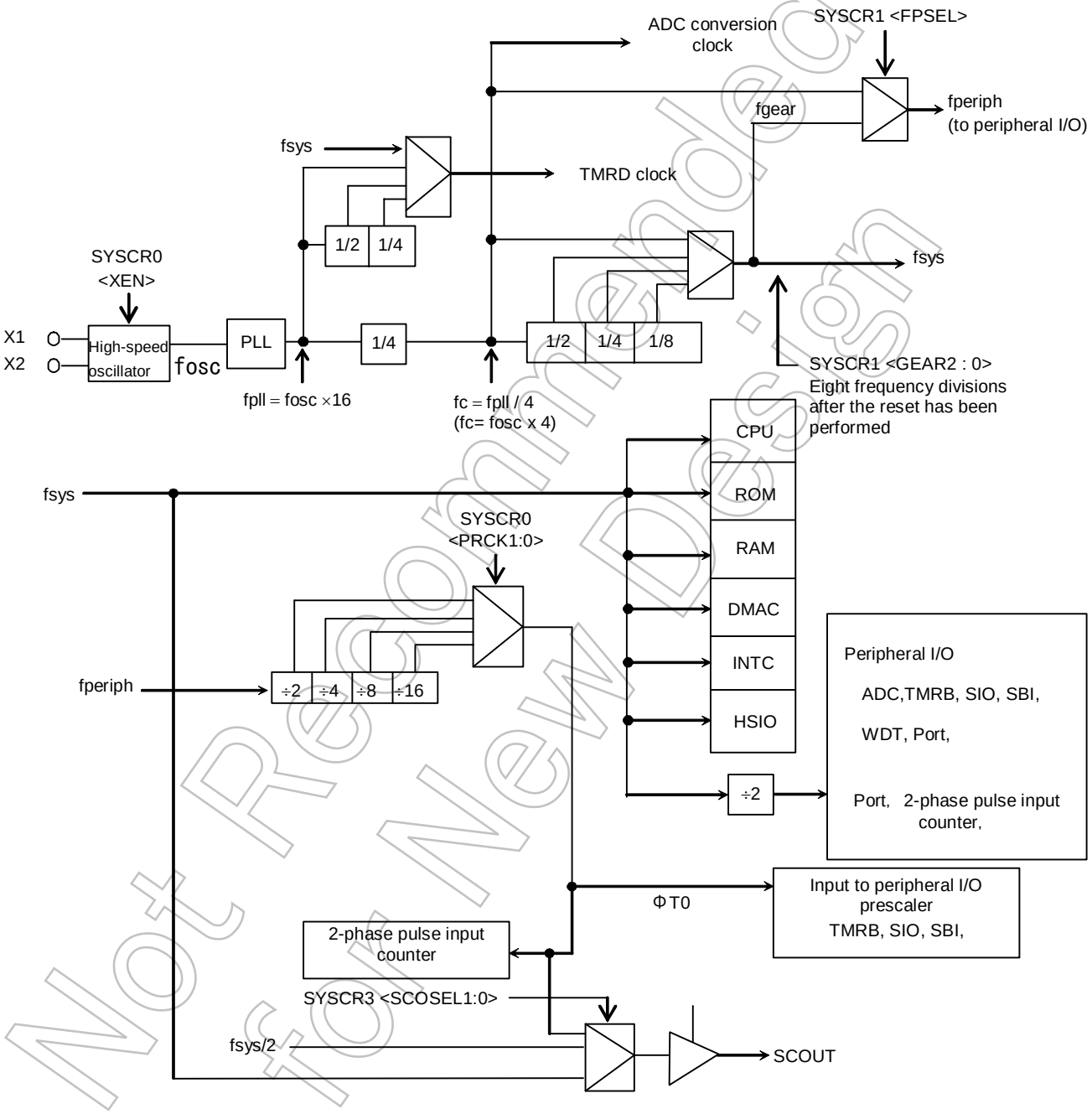


Fig. 5-3 Clock Related Block Diagram

5.2 CG Registers

5.2.1 System Control Registers

SYSCR0

LITTLE (0xFFFF_EE00)
BIG (0xFFFF_EE03)

	7	6	5	4	3	2	1	0
Bit symbol	XEN		RXEN			WUEF	PRCK1	PRCK0
Read/Write	R/W	R	R/W	R	R	R/W	R/W	R/W
After reset	1	0	1	0	0	0	0	0
Function	High-speed oscillator 0: Stop 1:Oscillation	This can be read as "0."	High-speed oscillator after the STOP mode is released 0: Stop 1:Oscillation	This can be read as "0."	This can be read as "0."	Control of warm-up timer (WUP) for oscillator 0 write: don't care 1 write: WUP start 0 read: WUP finished 1 read: WUP operating	Select prescaler clock 00:fperiph/16 01:fperiph/8 10:fperiph/4 11: fperiph/2	

SYSCR1

LITTLE (0xFFFF_EE01)
BIG (0xFFFF_EE02)

	7	6	5	4	3	2	1	0
Bit symbol				FPSEL		GEAR2	GEAR1	GEAR0
Read/Write	R	R	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	1	1	1
Function	This can be read as "0."	This can be read as "0."	Write "0".	Select fperiph 0:fgear 1:fc	Write "0".	Select gear of high-speed clock (fc) 000:fc 100:fc/2 001:Reserved 101:Reserved 010:Reserved 110:fc/4 011:Reserved 111:fc/8		

SYSCR2

LITTLE (0xFFFF_EE02)
BIG (0xFFFF_EE01)

	7	6	5	4	3	2	1	0
Bit symbol	DRVOSCH		WUPT1	WUPT0	STBY1	STBY0		DRVE
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
After reset	0	0	1	0	1	1	0	0
Function	High-speed oscillator current control 0: High capability 1: Low capability	This can be read as "0."	Select oscillator warm-up time 00:No WUP 01: 2^8/ Input frequency 10: 2^{14}/ Input frequency 11: 2^{16}/ Input frequency		Select standby mode 00: Reserved 01: STOP 10: Reserved 11:IDLE		This can be read as "0."	1: Drive the pin even in the STOP mode.

SYSCR3

LITTLE (0xFFFF_EE03)
BIG (0xFFFF_EE00)

	7	6	5	4	3	2	1	0
Bit symbol		SCOSEL1	SCOSEL0	ALESEL				
Read/Write	R	R/W	R/W	R/W	R			
After reset	0	0	1	1	0			
Function	This can be read as "0."	Select SCOUT output 00:Reserved 01:fsys/2 10:fsys 11:φT0		Set ALE output width 0:fsys×1 1:fsys×2	This can be read as "0."			

- **Don't switch the SYSCK and the GEAR<2:0> simultaneously.**
- **If the system enters the STOP mode with SYSCR2<DRVOSCH> set at 1 (low capability), the setting will change to 0 (high capability) after the STOP mode is released.**

PWMCG mode register

PWMCG (0xFFFF_EE28)

	7	6	5	4	3	2	1	0
bit Symbol			PWMGEAR1	PWMGEAR0				PWMEN
Read/Write	R				R			
After reset	0	0	1	1	0	0	0	0
Function	This can be read as "0."		Select source clock 00: PLL multiplied (16 times) 01: PLL octupled 10: PLL quadrupled 11: GCLK		This can be read as "0."			0 : PWMCLK OFF 1 : PWMCLK ON

Clock to be provided to high accuracy PPG is selected.

5.3 System Clock Controller

By resetting the system clock controller, the controller status is initialized to <XEN>="1 and <GEAR2:0>="111," and the system clock fsys changes to fc/8. For example, when a 13.5-MHz oscillator is connected to the X1 or X2 pin, fsys becomes 6.75 MHz (=13.5×16x1/4×1/8=6.75) after the reset. when a 10-MHz oscillator is connected to the X1 or X2 pin, fsys becomes 5 MHz (=10×16×1/4×1/8=5) after the reset.

Similarly, when the oscillator is not connected and an external oscillator is used to input a clock instead, fsys becomes the frequency obtained from the calculation "input frequency×16×1/4×1/8."

5.3.1 System Clock Pin Output Function

The system clock, fsys, fsys/2 or the peripheral I/O prescaler input clock ϕT0, can be output from the P46/SCOUT pin. By setting the port 4 related registers, P4CR<P46C> to "1" and P4FC<P46F> to "1," the P46/SCOUT pin becomes the SCOUT output pin. The output clock is selected by setting the SYSCR3<SCOSEL1:0>.

Table 5-4 SCOUT Output State in Each Standby Mode shows the pin states in each standby mode when the P46/SCOUT pin is set to the SCOUT output.

Table 5-4 SCOUT Output State in Each Standby Mode

MODE / SCOUT selection	NORMAL	STANDBY	
		IDLE	STOP
<SCOSEL1:0> = "00"	No Reserved Setup required.		
<SCOSEL1:0> = "01"	Output the fsys/2 clock.	Fixed to "0" or "1."	
<SCOSEL1:0> = "10"	Output the fsys clock.		
<SCOSEL1:0> = "11"	Output the ϕT0 clock.		

(Note) The phase difference (AC timing) between the system clock output by the SCOUT and the internal clock is not guaranteed.

5.3.2 Reducing the Oscillator Driving Capability

This function is intended for restricting oscillation noise generated from the oscillator and reducing the power dissipation of the oscillator when it is connected to the oscillator connection pin.

Setting the SYSCR2<DRVOSCH> to "1" reduces the driving capability of the high-speed oscillator (low capability).

This is reset to the default setting "0." When the power is turned on, oscillation starts with the normal driving capability (high capability). This is automatically set to the high driving capability state (<DRVOSCH> ="0") whenever the oscillator starts oscillation due to mode transition.

● Reducing the driving capability of the high-speed oscillator

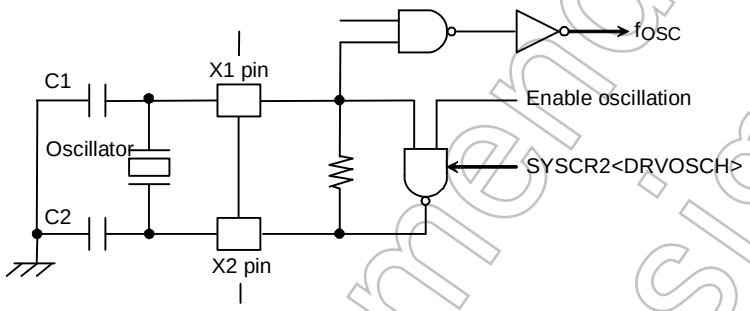


Fig. 5-5 Oscillator Driving Capability

5.4 Prescaler Clock Controller

Each internal I/O (TMRB0-9, SIO0-3 and SBI0-1) has a prescaler for dividing a clock. The clock ϕT0 to be input to each prescaler is obtained by selecting the "fperiph" clock at the SYSCR1<FPSEL> and the SYSCR0<PRCK1:0> and then dividing the clock according to the setting of SYSCR0<PRCK1:0>. After the controller is reset, fperiph/16 is selected as ϕT0.

5.5 Clock Multiplication Circuit (PLL)

This circuit outputs the fpll clock that is 16 times the high-speed oscillator output clock, fosc and adopts fpll×1/4 as a system clock, fsys. lowers the oscillator input frequency while increasing the internal clock speed.

5.6 Standby Controller

The TX19A core has several low-dissipation modes. To shift to the STOP or IDLE (Halt or Doze) mode, set the RP bit in the CPO status register, and then execute the WAIT instruction.

Before shifting to the mode, you need to select the standby mode at the system control register (SYSCR2).

The IDLE and STOP modes have the following features:

IDLE: Only the CPU is stopped in this mode.

The internal I/O has one bit of the ON/OFF setting register for operation in the IDLE mode in the register of each module. This enables operation settings for the IDLE mode. When the internal I/O has been set not to operate in the IDLE mode, it stops operation and holds the state when the system enters the IDLE mode.

Table 5-6 shows a list of IDLE setting registers.

Table 5-6 Internal I/O setting registers for the IDLE mode

Internal I/O	IDLE mode setting register
TMRB0-9	TBxRUN<I2TBx>
TMRD0-1	TDxRUN<I2TDx>
SIO0-2	SCxMOD1<I2Sx>
HSIO0	HSCxMOD1<I2Sx>
I2C/SIO(SBI0-1)	SBIBRx<I2SBIx>
A/D converter	ADMOD1<I2AD>
WDT	WDMOD<I2WDT>

(Note 1) The Halt mode is activated by setting the RP bit in the status register to "0," executing the WAIT command and shifting to the standby mode. In this mode, the TX19A processor core stops the processer operation while holding the status of the pipeline. The TX19A gives no response to the bus control authority request from the internal DMA, so the bus control authority is maintained in this mode.

(Note 2) The Doze mode is activated by setting the RP bit in the status register to "1" and shifting to the standby mode. In this mode, the TX19A processor core stops the processer operation while holding the status of the pipeline. The TX19A can respond to the bus control authority request given from the outside of the processor core.

STOP: All the internal circuits are brought to a stop.

The standby mode selection Status<RP > of CP0 is selected by the combination.

Please do not execute the WAIT instruction in the setting of "X" in the following table.

	STBY 1:0	HALT RP=0	DOZE RP=1
RESERVED	00	X	X
STOP	01	STOP	X
IDLE	11	HALT	DOZE

5.6.1 CG Operations in Each Mode

Table 5-7 Status of CG in Each Operation Mode

Clock source	Mode	Oscillation circuit	PLL	Clock supply to peripheral I/O	Clock supply to CPU
Oscillator	Normal	○	○	○	○
	Idle (Halt)	○	○	Selectable	×
	Idle (Doze)	○	○	Selectable	×
	Stop	×	×	×	×

○: ON or clock supply ×: OFF or no clock supply

5.6.2 Block Operations in Each Mode

Table 5-8 Block Operating Status in Each Operation Mode

Block	NORMAL	IDLE (Doze)	IDLE (Halt)	STOP
TX19A processor core	○	×	×	×
DMAC	○	○	×	×
INTC	○	○	○	×
External bus I/F	○	○	×	×
IO port	○	○	×	×
ADC	○	ON/OFF selectable for each module		×
SIO	○			×
HSIO	○			×
I2C	○			×
TMRB	○			×
TMRD	○			×
2-phase pulse input counter	○			×
CG	○	○	○	×
High-speed oscillator (fc)	○	○	○	×

○:ON ×:OFF

5.6.3 Releasing the Standby State

The standby state can be released by an interrupt request when the interrupt level is higher than the interrupt mask level, or by a reset. The standby release source that can be used is determined by a combination of the standby mode and the state of the interrupt mask register <IM15:8> assigned to the status register in the system control coprocessor (CPO) of the TX19A processor core. Details are shown in Table 5-9.

- Release by an interrupt request

  Operations of releasing the standby state using an interrupt request vary depending on the interrupt enabled state. If the interrupt level specified before the system enters the standby mode is higher than the value of the interrupt mask register, an interrupt handling operation is executed by the trigger after the standby is released, and the processing is started at the instruction next to the standby shift instruction (WAIT instruction). If the interrupt request level is equal to or lower than the value of the interrupt mask register, the processing is not executed.

  For a non-maskable interrupt, an interrupt handling is executed after the standby state is released irrespectively of the mask register value.

- Release by the reset

  Any standby state can be released by the reset.
  Note that releasing of the STOP mode requires sufficient reset time to allow the oscillator operation to become stable (the time required to be stable + 500μs and more).
  Please refer to "6. Exceptions/Interrupts" for details of STOP release and ordinary interrupts.

Standby Release Sources and Standby Release Operations
(Interrupt level)>(Interrupt mask)

Table 5-9

Interrupt accepting state			Interrupt enabled EI= "1"		Interrupt disabled EI= "0"	
Standby mode			IDLE (programmable)	STOP	IDLE (programmable)	STOP
Standby release source	Interrupt	INTWDT	◎	×	◎	-
		INT0-F	◎	◎ (Note 1)	○	○ (Note 1)
		INTTB0-9	◎	×	○	×
		INTRX0-2,INTTX0-2	◎	×	○	×
		HINTRX0,HINTTX0	◎	×	○	×
		INTS0-1	◎	×	○	×
		INTAD/INTADHP/INTADM	◎	×	○	×

◎: Starts the interrupt handling after the standby mode is released. (The LSI is initialized by the reset.)
○: Starts the processing at the address next to the standby instruction (without executing the interrupt handling) after the standby mode is released.
×: Cannot be used for releasing the standby mode.
-: Cannot execute masking with an interruption mask when a nonmaskable interrupt is selected.

(Note 1) The standby mode is released after the warm-up time has elapsed.

(Note 2) To release the standby mode by using the level mode interrupt in the interruptible state, keep the level until the interrupt handling is started. Changing the level before then doesn't allow of starting the interrupt processing properly.

(Note 3) To recover from the standby mode when the CPU has disabled the acceptance of interrupts, set the interrupt level higher than the interrupt mask (Interrupt level > Interrupt mask). If the interrupt level is equal to or lower than the interrupt mask (Interrupt level ≤ Interrupt mask), the system cannot recover from the standby mode.

5.6.4 STOP Mode

In the STOP mode, all the internal circuits, including the internal oscillators, are brought to a stop. The pin states in the STOP mode vary depending on the setting of the SYSCR2<DRVE>. Table 5.12 shows the pin states in the STOP mode. When the STOP mode is released, the system clock output is started after the elapse of warm-up time at the warm-up counter to allow the internal oscillators to stabilize. After the STOP mode is released, the system returns to the operation mode that was active immediately before the STOP mode (NORMAL), and starts the operation.

It is necessary to make these settings before the instruction to enter the STOP mode is executed. Specify the warm-up time at the SYSCR2<WUPT1:0>.

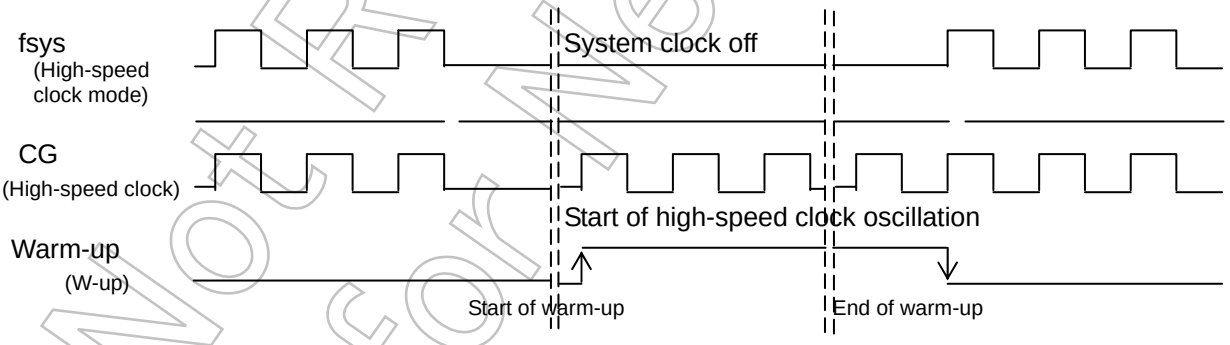
(Note) To shift from the NORMAL mode to the STOP mode on the TMP19A23, do not set the SYSCR2<WUPT1:0> to "00" or "01" for the warm-up time setting. The internal system recovery time cannot be satisfied when the system recovers from the STOP mode.

Table 5-10 Warm-up Settings for Transitions of Operation Modes

Transition of operation mode	Warm-up setting
NORMAL→IDLE	Not required
NORMAL→STOP	Not required
IDLE→NORMAL	Not required
STOP→NORMAL	Required

5.6.5 Recovery from the STOP or SLEEP Mode

Transition of operation modes: NORMAL → STOP → NORMAL



Selection of warm-up time SYSCR2<WUPT1:0>	Warm-up time (fosc=13.5MHz)	Warm-up time (fosc=10MHz)
01(2^8/fosc)	Setting disabled	Setting disabled
10(2^{14}/fosc)	1.214ms	1.638ms
11(2^{16}/fosc)	4.855ms	6.554ms

(Note) The internal system recovery time cannot be satisfied. Do not set <WUPT1:0> to "01."

Table 5-11

Pin States in the STOP Mode in Each State of SYSCR2<DRVE> (1/2)

Pin name	Input/output	<DRVE>=0	<DRVE>=1
P00-P07	Input mode	- *	- *
	Output mode	- *	- *
	Output mode AD0-AD7, D0-D7	- *	- *
P10-P17	Input mode	- *	- *
	Output mode	- *	- *
	Output mode AD8-AD15, D8-D15, A8-A15	- *	- *
P20-P27	Input mode	-	Input
	Output mode	-	Output
	Output mode A0-A7, A16-A23	Output	Output
P30-P32,P36	Input mode	-	Input
	Output mode	-	Output
	Output mode /RD, /WR, /HWR, R/W	Output	Output
P33,34	Input mode	-	Input
	Output mode	-	Output
	Input mode /BUSRQ, /WAIT	-	Input
P35,P37	Input mode	-	Input
	Output mode	-	Output
	Output mode /BUSAK, /ALE,	Output	Output
P40-P43	Input mode	-	Input
	Output mode	-	Output
	Output mode /CS0-/CS3	Output	Output
P44-P46	Input mode	-	Input
	Output mode	-	Output
	Input mode BUSMD, ENDIAN	-	Input
	Output mode SCOUT	-	Output
P47	Input mode	-	Input
	Output mode	-	Output
	Input mode INT0	Input	Input
P50-P57	Input mode	-	Input
	Output mode	-	Output
	Input mode INT3, INT4	Input	Input
	Output mode A0-A7	Output	Output
P60-P67	Input mode	-	Input
	Output mode	-	Output
	Input mode INT5, INT6	Input	Input
	Output modeA8-A15	Output	Output
P70-77	Input mode	-	-
P80-P84	Input mode	-	-
P90,P94,P95	Input mode	-	Input
	Output mode	-	Output
P91-P93, P96,P97	Input mode	-	Input
	Output mode	-	Output
	Input mode INT1, INT7, INT8, INT9, INTC	Input	Input
PA0,PA1, PA3-PA5,PA7	Input mode	-	Input
	Output mode	-	Output
PA2,PA6	Input mode	-	Input
	Output mode	-	Output
	Input mode INTA, INTB	Input	Input
PB0-PB2, PB4,PB5,PB7	Input mode	-	Input
	Output mode	-	Output
PB3,PB6	Input mode	-	Input
	Output mode	-	Output
	Input mode INT2, INTD	Input	Input

Table 5-12
Pin States in the STOP Mode in Each State of SYSCR2<DRVE> (1/2)

Pin name	Input/Output	<DRVE>=0	<DRVE>=1
PC0,PC1	Input mode	-	Input
	Output mode	-	Output
PC2,PC3	Input mode	-	Input
	Output mode	-	Output
	Input mode INTE, INTF	Input	Input
PD0-PD7	Input mode	-	Input
	Output mode	-	Output
PE0-PE5	Input mode	-	Input
	Output mode	-	Output
$\overline{\text{RESET}}$	Input pin	Input	Input
TEST	Input pin	Input	Input
X1	Input pin	-	-
X2	Output pin	"H" level output	"H" level output

- :Indicates that the input is disabled for the input mode and the input pin and the impedance becomes high for the output mode and the output pin.

Input :The input gate is active. To prevent the input pin from floating, fix the input voltage to the "L" or "H" level.

Output : The pin is in the output state.

* :Input or output is accepted only while accessing the pin. The state in Stop mode that does not accept accessing the pin is the same as "-" shown above.

6. Exceptions/ Interrupts

6.1 Overview

The TMP19A23 device is configured with the following 18 maskable interrupt factors and 15 exceptions including NMI. General exceptions, debug exceptions and interrupts are shown below.

- General exceptions
  - Reset exception
  - Non-maskable interrupt (NMI)
  - Address error exception (instruction fetch)
  - Address error exception (load/store)
  - Bus error exception (instruction fetch)
  - Bus error exception (data access)
  - Co-processor unusable exception
  - Reserved instruction exception
  - Integer overflow exception
  - Trap exception
  - System call exception
  - Breakpoint exception
- Debug exception
  - Single step exception
  - Debug breakpoint exception
- Interrupts
  - Maskable software interrupts (2 factors)
  - Maskable hardware interrupts:
  - QFP 38 internal factors and 16 external factors (INT0 - F)
  - BGA 37 internal factors and 15 external factors (INT0 - E)

The TMP19A23 device not only processes interrupt requests from internal hardware peripherals and external inputs but also forces transition to exception handling processes as a means of notifying any error status generated in normal instruction sequences.

By using the register bank called “shadow register set” newly implemented in the TX19A processor core, it is now unnecessary to save the general purpose register (GPR) contents elsewhere upon interrupt response thus leading to very fast interrupt response.

The device is capable of handling multiple interrupts according to seven programmable interrupt levels (priority orders). Also, it can mask interrupt requests with a priority level the same or lower than a specified mask level.

6.2 Exception Vector

The starting address of an exception handler is defined to be “exception vector address.” The exception vector address for a reset exception and non-maskable interrupts is 0xBFC0_0000. The exception vector address for a debug exception can be either 0xBFC0_0480 (EJTAG ProbEn = 0) or 0xFF20_0200 (EJTAG ProbEn = 1) depending on the internal signal <ProbeEn>. For other exceptions, the corresponding exception vector addresses are determined depending on the values of Status <BEV> and Cause <IV> of the system control coprocessor register (CP0).

Table 6-1 Exception Vector Table (Virtual Address)

Exception	BEV=0	BEV=1
Reset, NMI	0xBFC0_0000	0xBFC0_0000
Debug exceptions (En=0)	0xBFC0_0480	0xBFC0_0480
Debug exceptions (En=1)	0xFF20_0200	0xFF20_0200
Interrupts (IV=0)	0x8000_0180	0xBFC0_0380
Interrupts (IV=1)	0x8000_0200	0xBFC0_0400
Others general exceptions	0x8000_0180	0xBFC0_0380

(Note) If exception vector addresses are to be placed in internal ROM, set the status bit <BEV> of the system control coprocessor register (CP0) to “1.”

6.3 Reset Exception

A reset exception is generated by either setting the external reset pin to “L” or counting the WDT beyond a “reset” count. When a reset exception is generated, peripheral hardware registers and the CP0 register are initialized and it jumps to the exception vector address 0xBFC0_0000. The PC value of reset exception generation will be stored in ErrorEPC of the CP0 register.

Since a reset exception causes to set the status bit <ERL> of the CP0 register to “1” disabling interrupt requests, the Status <ERL> bit must be cleared to “0” in a startup routine (reset exception handler) or by any other means if interrupts are to be used.

Refer to the section “Exception Handling, Reset Exception” of the separate volume “TX19A Core Architecture” for detailed operation upon generation of reset exception.

6.4 Non-maskable Interrupt (NMI)

An NMI interrupt is generated when WDT is counted to an NMI set count or when a bus error area is accessed by store access including DMA transfer. When an NMI interrupt is generated, the status bits <ERL> and <NMI> of the CP0 register are set to “1” and it jumps to the exception vector address 0xBFC0_0000.

The PC value of NMI generation will be stored in ErrorEPC of the CP0 register. Note that any NMI due to a bus error upon a store instruction causes an exception that is not synchronized with instruction sequence. Therefore, the PC value of an instruction being executed at the time of error generation will be stored instead of the PC value for the instruction that actually caused the error. Upon NMI generation, when the shadow register set is enabled, SSCR <PSS> will be overwritten by the value of SSCR <CSS> but the register bank will not be switched because the value of SSCR <CSS> is not updated. The reason why only the SSCR <PSS> value is updated is because it is necessary to prevent the register bank from being changed when SSCR <PSS> is overwritten by the value of SSCR <CSS> due to an ERET instruction executed upon returning from NMI.

The cause of NMI generation can be determined by NMIFLG <WDT> and <WBER> of CG (refer to the Section 6.10 NMI Flag Register). Refer to the section “Exception Handling, Non-Maskable Interruptions” of the separate volume “TX19A Core Architecture” for detailed operation upon generation of NMI.

6.5 General Exceptions (Other than Reset Exception and NMI)

A general exception will be generated when a specific instruction such as SYSCALL is executed or when any abnormalities such as an illegal instruction fetch is detected. When a general exception is generated and if Status <BEV> of the CP0 register is “1,” it jumps to the exception vector address 0xBFC0_380. The cause of a general exception can be determined by Cause <ExCode> of the CP0 register.

The PC value at a general exception will be stored in EPC of the CP0 register. Note that any bus error exception (data access) is not synchronized with instruction sequence so the PC value of an instruction being executed at the time of error generation will be stored instead of the PC value for the instruction that actually caused the error. Upon a general exception, when the shadow register set is enabled, SSCR <PSS> will be overwritten by the value of SSCR <CSS> but the register bank will not be switched because the value of SSCR <CSS> is not updated. The reason why only the SSCR <PSS> value is updated is because it is necessary to prevent the register bank from being changed when SSCR <PSS> is overwritten by the value of SSCR <CSS> due to an ERET instruction executed upon returning from the exception.

Any illegal address that caused an address error exception (instruction fetch or load/store) will be stored in BadVAddr of the CP0 register.

Refer to the corresponding sections of “Exception Handling” of the separate volume “TX19A Core Architecture” for detailed operation upon generation of general exceptions.

(Note 1) Address error exceptions (load/store) will not be generated in DMS transfer operations. In DMA transfer, address errors can be detected as configuration errors (CSRx <Conf> of DMAC).

(Note 2) Bus errors (data access) may be generated either by load instructions or by load accesses of DMA transfer operations.

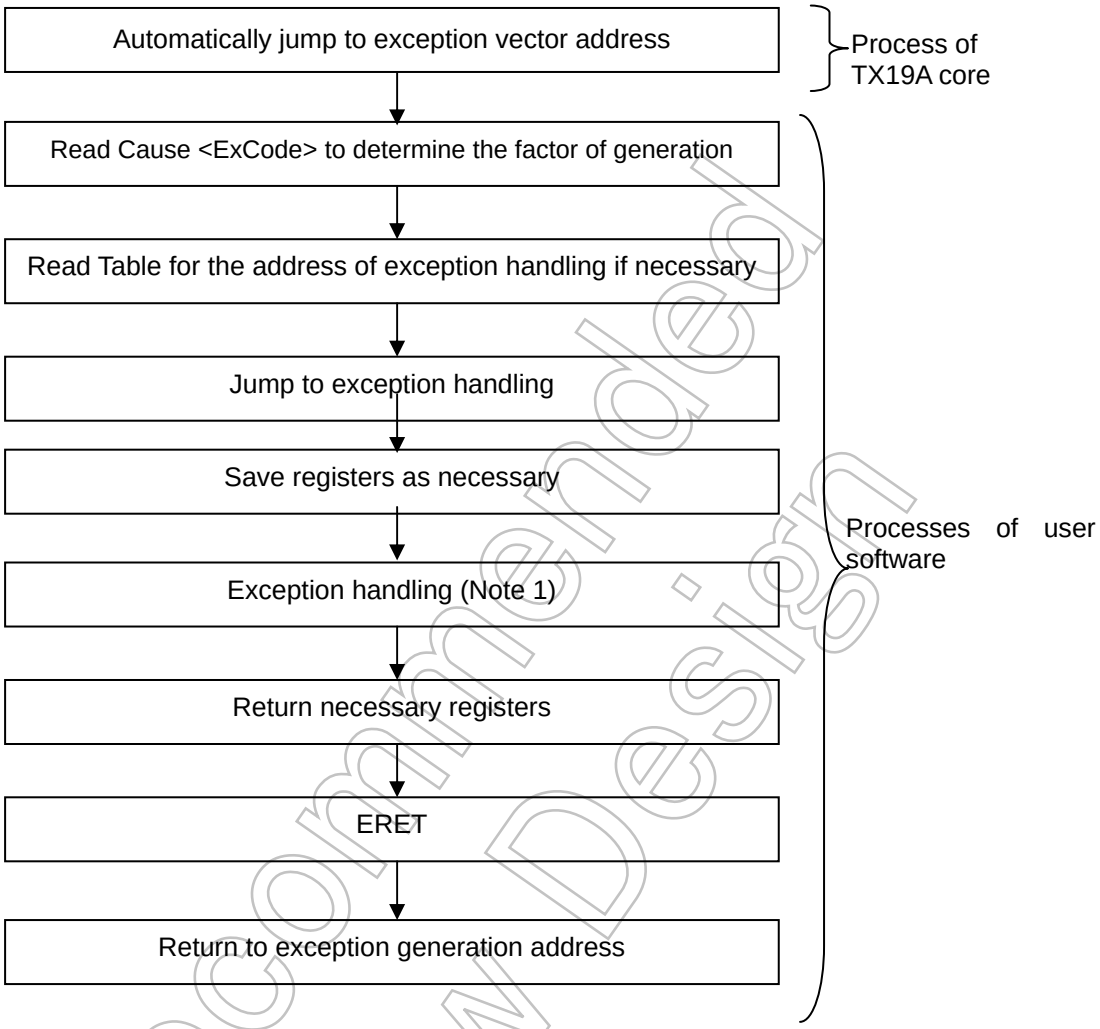


Fig. 6-1 Example Sequence of General Exceptions (Other than Reset Exception and NMI)

(Note 1) Since general exceptions (other than reset exception/NMI and excluding trap exceptions, system call exceptions, and breakpoint exceptions) indicate some sort of abnormal conditions, the system tends to be reset.

(Note 2) Upon generation of a general exception other than reset exception/NMI, excluding bus error exceptions (instruction fetch/data access), the PC that caused the exception will be stored in EPC. Therefore, returning the system by simply using ERET may cause the same exception again.

6.6 Debug Exceptions

Single step exceptions and debug breakpoint exceptions are the types of debug exceptions. These types of exceptions are seldom used in user programs.

Also note that enabling the shadow register set will not be effective in debug exceptions.

Refer to the section “Exception Handling, Debug Exception” of the separate volume “TX19A Core Architecture” for detailed operation upon generation of debug exceptions.

6.7 Maskable Software Interrupts

Two-factor maskable software interrupts (hereinafter referred to simply as “software interrupts”) can be generated by individually setting “1” to the Cause <IP [1:0]> bits of the CP0 register.

Software interrupts can be accepted in no less than three clocks after setting values to the Cause <IP [1:0]> bits of the CP0 register.

In order for a software interrupt request to be accepted, it is necessary regarding the CP0 register that its Status <IE> is set to “1” and Status <ERL/EXL> is cleared to “0” while Status <IM [1:0]> is “1.” Also, software interrupts can be individually masked by setting Status <IM [1:0]> of the CP0 register to “0.” If software and hardware interrupts coincide, the hardware interrupt overrides the software interrupt.

Upon software interrupts, when the shadow register set is enabled, SSCR <PSS> will be overwritten by the value of SSCR <CSS> but the register bank will not be switched because the value of SSCR <CSS> is not updated. The reason why only the SSCR <PSS> value is updated is because it is necessary to prevent the register bank from being changed when SSCR <PSS> is overwritten by the value of SSCR <CSS> due to an ERET instruction executed upon returning from the software interrupt. Software interrupts are processed in a process flow such as shown in

Fig. 6-2.

(Note 1) “Software interrupt” is different from the idea of “software set” to be used as one of hardware interrupt factors, as described later.

(Note 2) IVR readout must take place after the software interrupt. Hardware interrupt is notified to the core upon executing IVR readout.

(Note 3) The idea of “Software set” is to generate a hardware interrupt by setting the value other than “000” to IMC0 <IL2:0>.

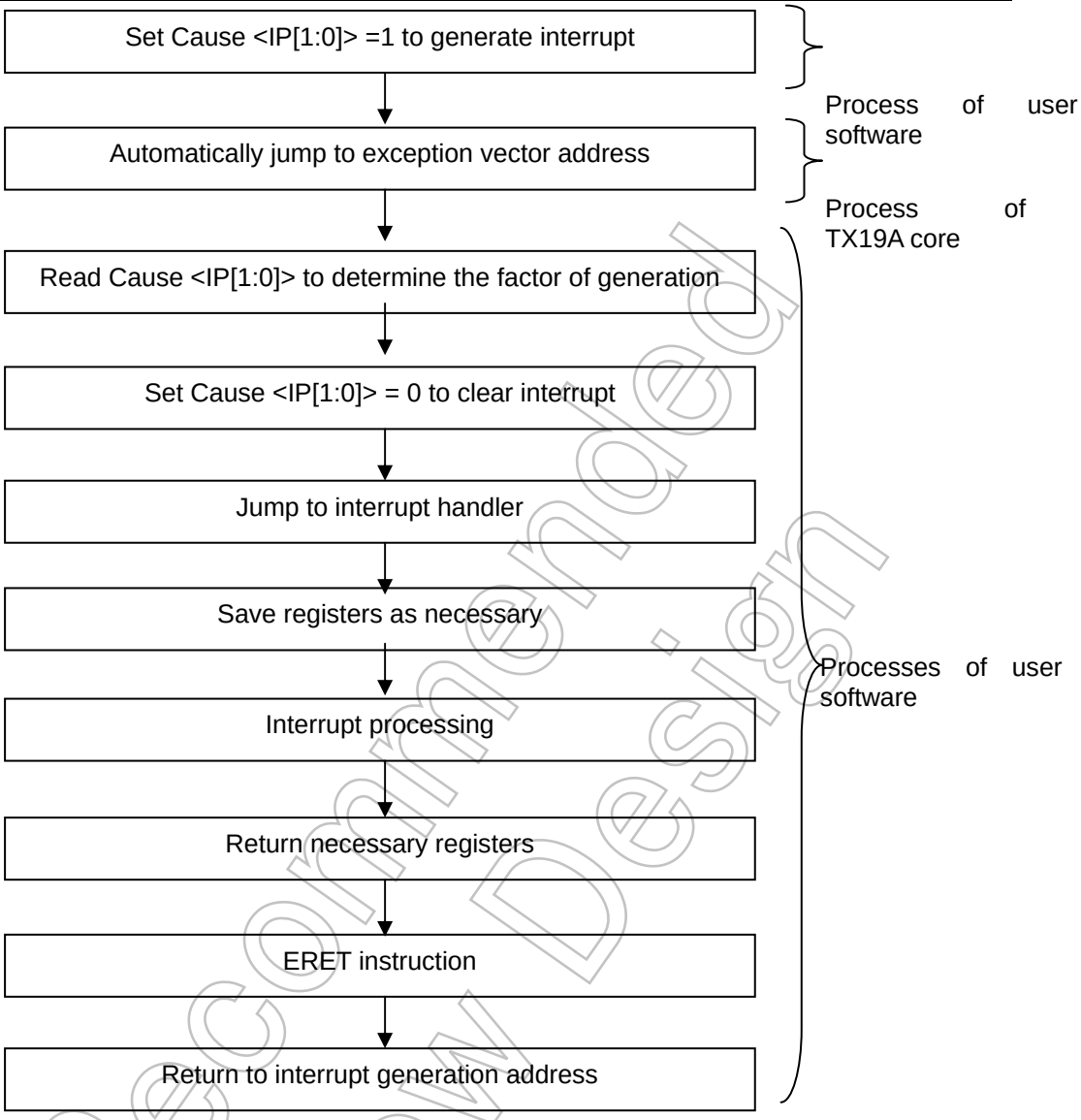


Fig. 6-2 Example of Software Interrupt Operation

(Note) A software interrupt is accepted in no less than three clocks after the instruction that enabled the interrupt and the PC at the time of acceptance is stored in EPC.

6.8 Maskable Hardware Interrupts

6.8.1 Features

The maskable hardware interrupts (hereinafter referred to as “hardware interrupts”) are 55 factor interrupt requests for which the interrupt controller (INTC) can individually assign an interrupt level out of seven interrupt (priority) levels.

In order for a hardware interrupt request to be accepted, it is necessary regarding the CP0 register that its Status <IE> is set to “1” and Status <ERL/EXL> is cleared to “0” while Status <IM [4:2]> is set to “1.”

If more than one interrupts are generated at the same time, the hardware interrupts are accepted in accordance with the priority order of the interrupt levels. If more than one interrupts of a same interrupt level are generated at the same time, these interrupts are accepted in the order of the interrupt number as listed in Table 6-2.

When an interrupt request is accepted, the Status <EXL> bit of the CP0 register is set to “1,” further interrupts are disabled, and ILEV<CMASK> of INTC is automatically updated to the interrupt level set for the interrupt request. Note that Status <IE> of the CP0 register remains set to “1” in interrupt response operations.

In processing hardware interrupts, each interrupt level is associated with a register bank called a “shadow register set.” When an interrupt request is accepted, the register bank is switched to the register bank of which number is the same as with the corresponding interrupt level. Through this mechanism, it is unnecessary for the user program to save the general purpose register (GPR) contents elsewhere upon interrupt response thus ensuring fast interrupt response. (CP0 register SSCR<SSD>=“0”)

For accepting multiple interrupts, Status <EXL> of the CP0 register is cleared to “0” to permit further interrupts. In this, because ILEV <CMASK> of INTC has been updated to the interrupt level set for the interrupt request already accepted, only further interrupts of which level is higher than the present interrupt level can be accepted. Refer to Section 0 “Example of Multiple Interrupt Setting“ for more details of multiple interrupts.

Also, by appropriately setting the ILEV <CMASK> register of INTC, you can mask interrupt request of which interrupt level is lower than a programmed mask level.

Any interrupt request can be used as a trigger to start a DMA transfer sequence.

While detailed operation of hardware interrupts is provided below, please also refer to the section “Exception Handling, Maskable Interrupts (Interrupts)” of the separate volume “TX19A Core Architecture” for more details.

Fig. 6-3 Interrupt Notification Diagram

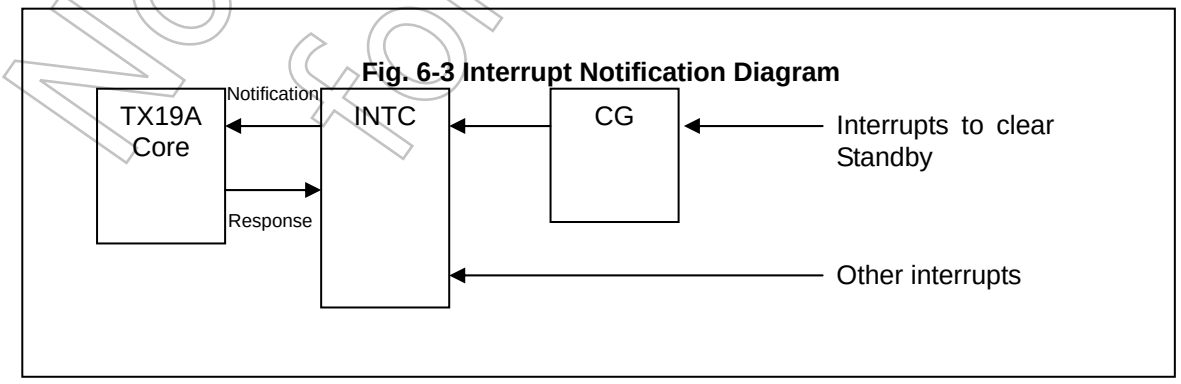


Table 6-2 List of Hardware Interrupt Factors

Interrupt Number	IVR[7:0]	Interrupt Factor	Interrupt Control Register	Address
0	0x000	Software set	IMC0	0xFFFF_E000
1	0x004	INT0 pin		
2	0x008	INT1 pin		
3	0x00C	INT2 pin		
4	0x010	INT3 pin	IMC1	0xFFFF_E004
5	0x014	INT4 pin		
6	0x018	INT5 pin		
7	0x01C	INT6 pin		
8	0x020	INT7 pin	IMC2	0xFFFF_E008
9	0x024	INT8 pin		
10	0x028	INT9 pin		
11	0x02C	INTA pin		
12	0x030	INTB pin	IMC3	0xFFFF_E00C
13	0x034	INTC pin		
14	0x038	INTD pin		
15	0x03C	INTE pin		
16	0x040	INTF pin **(Not embedded in BGA)**	IMC4	0xFFFF_E010
17	0x044	INTRX0 : Serial receive (channel.0)		
18	0x048	INTTX0 : Serial transmit (channel.0)		
19	0x04C	INTRX1 : Serial receive (channel.1)		
20	0x050	INTTX1 : Serial transmit (channel.1)	IMC5	0xFFFF_E014
21	0x054	INTRX2 : Serial receive (channel.2)		
22	0x058	INTTX2 : Serial transmit (channel.2)		
23	0x05C	HINTRX0 : High speed serial receive (Hchannel.0)		
24	0x060	HINTTX0 : High speed serial transmit (Hchannel.0)	IMC6	0xFFFF_E018
25	0x064	INTSBI0 : Serial bus interface 0		
26	0x068	INTADHP : Highest priority ADC complete interrupt		
27	0x06C	INTADM0 : ADC monitor function interrupt 0		
28	0x070	INTADM1 : ADC monitor function interrupt 1	IMC7	0xFFFF_E01C
29	0x074	INTTB0 : 16bitTMRB 0		
30	0x078	INTTB1 : 16bitTMRB 1		
31	0x07C	INTTB2 : 16bitTMRB 2		
32	0x080	INTTB3 : 16bitTMRB 3	IMC8	0xFFFF_E020
33	0x084	INTTB4 : 16bitTMRB 4		
34	0x088	INTTB5 : 16bitTMRB 5		
35	0x08C	INTTB6 : 16bitTMRB 6		
36	0x090	INTTB7 : 16bitTMRB 7	IMC9	0xFFFF_E024
37	0x094	INTTB8 : 16bitTMRB 8		
38	0x098	INTTB9 : 16bitTMRB 9		
39	0x09C	INTSBI1 : Serial bus interface 1		
40	0x0A0	INTD0CMP0 : 16bitTMRD0 CMP 0	IMCA	0xFFFF_E028
41	0x0A4	INTD0CMP1 : 16bitTMRD0 CMP 1		
42	0x0A8	INTD0CMP2 : 16bitTMRD0 CMP 2		
43	0x0AC	INTD0CMP3 : 16bitTMRD0 CMP 3		
44	0x0B0	INTD0CMP4 : 16bitTMRD0 CMP 4	IMCB	0xFFFF_E02C
45	0x0B4	INTD1CMP0 : 16bitTMRD1 CMP 0		
46	0x0B8	INTD1CMP1 : 16bitTMRD1 CMP 1		
47	0x0BC	INTD1CMP2 : 16bitTMRD0 CMP 2		
48	0x0C0	INTD1CMP3 : 16bitTMRD1 CMP 3	IMCC	0xFFFF_E030
49	0x0C4	INTD1CMP4 : 16bitTMRD1 CMP 4		
50	0x0C8	INTAD : ADC completed		
51	0x0CC	INTDMA0 : Completion of DMA transfer (channel 0)		
52	0x0D0	INTDMA1 : Completion of DMA transfer (channel 1)	IMCD	0xFFFF_E034
53	0x0D4	INTDMA2 : Completion of DMA transfer (channel 2)		
54	0x0D8	INTDMA3 : Completion of DMA transfer (channel 3)		
55	0x0DC	Reserve		
56	0x0E0	Reserve	IMCE	0xFFFF_E038
57	0x0E4	Reserve		
58	0x0E8	Reserve		
59	0x0EC	Reserve		
60	0x0F0	Reserve	IMCF	0xFFFF_E03C
61	0x0F4	Reserve		
62	0x0F8	Reserve		
63	0x0FC	Reserve		

(Note 1) While IMCxx is a 32 bit register, 8 bit/16 bit access is also accepted.

(Note 2) Each factor can clear the IDLE mode.

Table 6-3 Interrupt Factors to Cancel Stop Modes

Number	Interrupt Factor	Note
0	INT0	External interrupt 0
1	INT1	External interrupt 1
2	INT2	External interrupt 2
3	INT3	External interrupt 3
4	INT4	External interrupt 4
5	INT5	External interrupt 5
6	INT6	External interrupt 6
7	INT7	External interrupt 7
8	INT8	External interrupt 8
9	INT9	External interrupt 9
10	INTA	External interrupt A
11	INTB	External interrupt B
12	INTC	External interrupt C
13	INTD	External interrupt D
14	INTE	External interrupt E
15	INTF **(Not embedded in BGA)**	External interrupt F

* Number 0 to 15 interrupt factors can cancel Stop and Idle modes.
* BGA does not include INTF.

6.8.2 Detecting Interrupt Requests

Each of interrupt factors has its own interrupt detection sequence as described in Table 6-4. Upon detection, an interrupt request is notified to INTC for priority arbitration and then notified to the TX19A processor core. Refer to Table 6-5 for the detection level available for each interrupt factor.

Table 6-4 Location of Interrupt Request Detection

Interrupt	Detected by	Interrupt Notification Route
(1) Interrupts from external pins INT0 - INTF	CG	PORT → CG (detection) → INTC (arbitration) → TX19A core
	INTC	PORT → INTC (detection/arbitration) → TX19A core
(2) Other interrupts	INTC	Peripheral circuit → INTC (detection/arbitration) → TX19A core

6.8.3 Interrupt Priority Arbitration

1. Seven levels of interrupt priority

   Each of interrupt factors can be individually set to one of the seven interrupt priority levels by INTC.

   The interrupt level to be applied is set by IMCxx <ILxxx> of INTC. The higher the interrupt level is set, the higher the priority becomes. If the value is set to "000" meaning interrupt level of 0, no interrupts will be generated by the factor. Also note that any factors of interrupt level 0 are not suspended.

2. Interrupt Level Notification

   When an interrupt request is generated, INTC compares the interrupt level with the mask level. If the interrupt level is higher than the mask level set in ILEV <CMASK>, it notifies the TX19A processor of the interrupt request.

   If more than one interrupts are generated at the same time, the interrupts are notified in accordance with the priority order of these interrupt levels. If more than one interrupts of a same interrupt level are generated at the same time, these interrupts are notified in the order of the interrupt number as listed in Table 6-2.

   When an interrupt request of the same interrupt factor is received again before the previous interrupt has been cleared, only the first interrupt can be accepted.

3. INTC Register Update

   When an interrupt request is accepted by the TX19A core, the highest interrupt level at that point in time will be set to ILEV <CMASK> and the corresponding vector value is set to IVR. Once CMASK and IVR are set, any interrupt with a higher interrupt level cannot update them or cause notification to the core until the IVR value is read.

(Note) Please read the IVR value before attempting to change the ILEV value. If the ILEV value is changed before reading IVR, an unexpected interrupt request may be generated.

6.8.4 Hardware Interrupt Operation

When a hardware interrupt is generated, the TX19A core will go through the following steps to jump to the corresponding exception vector address as given in Table 6-1 according to the Status <BEV> and Cause <IV> bits of the CP0 register.

(1) Sets Status <EXL> of CP0 register to “1.”
(2) Sets the PC value at the interrupt generation to EPC of the CP0 register.
(3) If the shadow register set is enabled (CP0 register SSCR <SSD> = 0), SSCR <CSS/PSS> of the CP0 register will be updated and it switches to the register bank of the same interrupt level number.
(4) The values of ILEV <CMASK/PMASKx> of INTC will be updated and the mask level is set to the interrupt level of the interrupt request accepted.
(5) Sets IVR [7:0] to the corresponding value listed in Table 6-2.

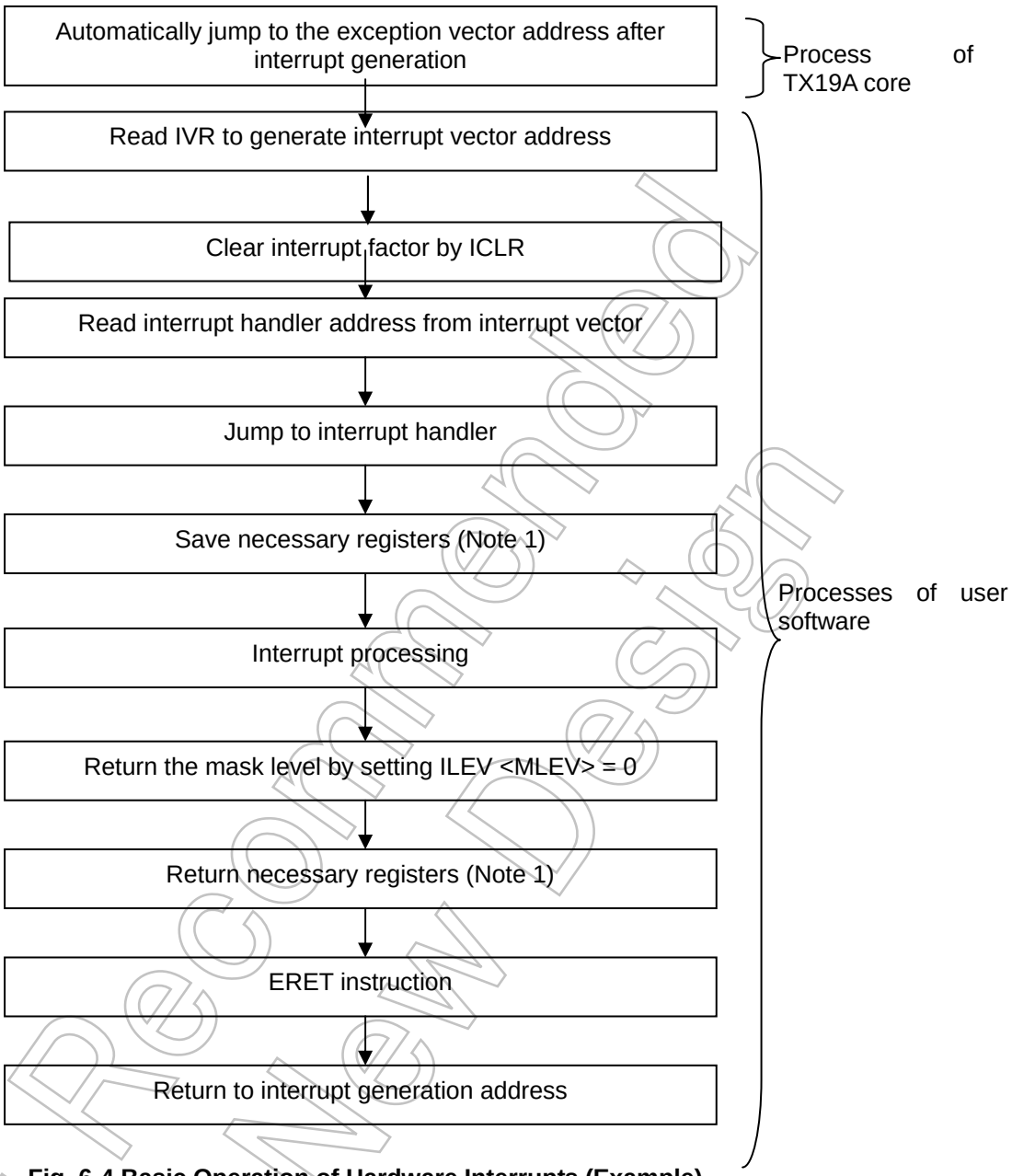


Fig. 6-4 Basic Operation of Hardware Interrupts (Example)

(Note) By using the shadow register set (setting CP0 register SSCR <SSD> = 0), most of general purpose register contents can be automatically saved in TX19A core.

6.8.5 Initialization for Interrupts

Before using interrupts, it is necessary to appropriately configure them. Necessary settings that have to be made regardless of the interrupt factors are described in Section 6.8.5.1 “Common Initialization“ and settings specifically required for certain factors and applications are described in Section 6.8.5.2 “Initialization for Individual Interrupt Factors.”

6.8.5.1 Common Initialization

In order to use interrupts, the following settings are necessary:

(1) Set Status <IM [4:2]> of CP0 register to “111.”

(2) Set the base address of the interrupt vector table to IVR [31:8] of INTC.

(3) Set the interrupt handler addresses for the respective interrupt factors to the addresses obtained as the sum of the base address of “the interrupt vector table and the IVR [7:0] values corresponding to the respective interrupt factors.

Example of the above step (1): When the interrupt exception vector address 0xBFC00400 is used

```
lui     r2,0x1040           ; CU0=1 ,BEV =1 (r2 =0x1040_xxxx)
addiu   r2,r2,0x1C00        ; IM4,IM3,IM2 =1 (r2 =0x1040_1C00)
mtc0    r2, r12
```

Example of the above step (2): If Vector Table is used as the label of the interrupt vector table

```
lui     r3,hi(VectorTable)
addiu   r3,r3,lo(VectorTable)   ; r3 =VectorTable address
lui     r2,hi(IVR)              ; r2 =0xFFFF_xxxx(Upper 16 bits of IVR address)
sw      r3,lo(IVR)(r2)          ; Set address of VectorTable to IVR[31:8]
```

Example of the above step (3): If the base address of interrupt vector is set to 0xBFC20000

```
_VectorTable section code isa32 abs=0xBFC20000
VectorTable:
dw      _SWINT          ; 0 --- software interrupt
dw      _INT0           ; 1 --- INT0
dw      _INT1           ; 2 --- INT1
dw      _INT2           ; 3 --- INT2
dw      _INT3           ; 4 --- INT3
dw      _INT4           ; 5 --- INT4
dw      _INT5           ; 6 --- INT5
dw      _INT6           ; 7 --- INT6
dw      _INT7           ; 8 --- INT7
```

(Note) The above examples assume the use of Assembler made by Toshiba. If any third party Assembler is used, it may generate syntax errors; you are advised to modify the above statements according to the Assembler to be used.

6.8.5.2 Initialization for Individual Interrupt Factors

The registers to be set in using different interrupt factors are as listed below:

Table 6-5 Registers to be Set for Detecting Interrupts

Interrupt	Detected by	Registers to be Set	Interrupt detection levels available (setting in active condition)
(1) Interrupts from external pins INT0 - INTF	INTC	PxFC(PORT) PxCR(PORT) IMCxx(INTC)	With INTC, “L” and “H” levels and falling/rising edges can be set.
	CG	PxFC(PORT) PxCR(PORT) IMCGx(CG) IMCxx(INTC)	If it is to be used for recovery from Standby mode, set “L” and “H” levels and falling and rising edges for CG while INTC must be set to “H.”
(2) Interrupts from dual phase counter	INTC	PxFC(PORT) PxCR(PORT) IMCxx(INTC)	Set for rising edge with INTC.
(3) Other interrupts	INTC	IMCxx(INTC)	With INTC, “L” and “H” levels and falling/rising edges can be set.

(Note 1) In level detection, the value is checked at internal clock timing each time. Edge detection is made by comparing the previous value with the current value at internal clock timing. Edge detection for CG/KWUP (static setting) is made by detecting input signal instead of using internal clock.

(Note 2) In interrupt initialization, follow the order of the interrupt detection route as indicated in Table 6-4 before enabling the interrupts with the CP0 register. If any different setting order is used, an unexpected interrupt may be generated. So, be sure to clear interrupt factors before setting interrupt permission. Similarly, if interrupts are to be disabled, first disable the interrupt by the CP0 register and then set the registers accordingly in the reverse order of interrupt detection.

(1) Interrupts from external pins (INT0 to INTF)

- Use PORT PxCR and PxIE to enable an input port. (Refer to 7. Port Function)
- Use PORT PxFC to set pin functions to INT0 - INTF. (Refer to 7. Port Function)
- Use PORT PxPUP to set pull up connections as appropriate. (Refer to 7. Port Function)
- Use INTC IMCx <EIMxx> to set active state. (Refer to 5.3.3 Interrupt-related Registers)
- Use IMCGx <EMCGxx> of CG for setting to enable/disable clearing of standby modes. (Refer to INTCG Registers, Interrupts to Clear STOP and IDLE)
- Use INTC IMCx <EIMxx> to set active state of internal interrupt signals to be notified from CG. If rising or falling edge is set with INTC IMCx <EIMxx>, set it to falling edge (set IMCx <EIMxx> to “10”). For H/L level setting, set it to “L” level (set IMCx <EIMxx> to “00”). (Refer to 6.8.8 Registers.)

・An example setting when an external interrupt “INT3” is used to clear Stop by the falling edge:

```
Status<IE> =“0”                    ; Interrupt is disabled
P5CR<P54C> =“0”                    ; The port is set to an input port
P5FC<P54F> =“0”                    ; The port is assigned to INT3
IMCGA<EMCG32:30> =“010”            ; INT3 is set to falling edge
IMCGA<INT3EN> =“1”                 ; INT3 is set to clear Standby mode
EICRCG<ICRCG3:0> =“0011”           ; Clears the INT3 standby clear request
IMC1<EIM41:40> =“01”               ; INT3 is set to level detection
INTCLR<EICLR7:0> =“010”            ; Clears the INT3 interrupt request
IMC1<IL42:40> =“101”               ; Interrupt level of INT3 is set to “5.”
ILEV<MLEV>/<CMASK> =“1”/”xxx”      ; Mask level is set to “xxx.”
                                   (To be set simultaneously with ILEV <MLEV>)
SYNC instruction                   ; (To be set simultaneously with ILEV <MLEV>)
Status<IE> =“1”                    ; Interrupt is enabled
```

・An example setting when an external interrupt “INT3” is to be disabled:

```
Status<IE> =“0”                    ; Interrupt is disabled
IMC1<IL42:40> =“000”               ; INT3 interrupt is disabled.
INTCLR<EICLR7:0> =“010”            ; Clears the INT3 interrupt request
```

(2) Other hardware interrupts

- Settings are made to use peripheral hardware devices.
- Set INTC IMCxx <EIMxx> to “10.” (Refer to 6.8.8 Registers.)

(Note) In interrupt initialization, set INTC registers before enabling interrupts with the CP0 register. Similarly, if interrupt is to be disabled, first disable interrupt by the CP0 register and then set INTC.

6.8.5.3 Interrupt Enable

In order for an interrupt request to be accepted, all the following parameters must be set to enable the interrupt in addition to the initial settings described in Section 6.8.5 “Initialization for Interrupts.

- Set Status <ERL> of the CP0 register to “0.”
- Set Status <EXL> of the CP0 register to “0.”
- Set Status <IE> of the CP0 register to “1.”

By these settings, interrupt is enabled two clocks after execution of the instruction and the registers are set. Note that one of the following methods may be used in setting Status <IE> of the CP0 register to “1.”

1. Set Status <IE> of the CPO register to “1”using the MTCO instruction (32 bit ISA instruction).
2. Set IER of the CP0 register to any value other than “0” using the MTC0 instruction (32 bit ISA instruction). (Note 1)
3. Set Status <IE> of the CPO register to “1”using the MTCO instruction (16 bit ISA instruction).
4. Execute the EI instruction of 16 bit ISA. (Note 2)

(Note 1) This method is suitable for 32 bit ISA to avoid too many codes. If Toshiba C compiler is used, this is executed by the 32 bit ISA instruction “__EI ( ) embedded function”.

(Note 2) This method is suitable for 16 bit ISA to avoid too many codes. If Toshiba C compiler is used, this is executed by the 16 bit ISA instruction “__EI ( ) embedded function”.

(Note 3) The methods 2 and 4 that can avoid too many codes and provide faster process are recommended.

6.8.5.4 Interrupt Disable

To disable interrupts, either one of the following setting procedures must be performed." When interrupts are disabled, any interrupt request described in Section 6.8.5 "Interrupts will be suspended. Also note that TMP19A23 doesn't suspend any interrupt factor that is set to interrupt level 0.

- Set Status <ERL> of the CP0 register to "1."
- Set Status <EXL> of the CP0 register to "1."
- Set Status <IE> of the CP0 register to "0."

By these settings, interrupts are disabled immediately after execution of the instruction and the registers are set two clocks later. Status <ERL> and <EXL> of the CP0 register is set by an interrupt or an exception and cleared by ERET instruction. Therefore, setting status <IE> of the CP0 register to "0" is suitable for disabling interrupts. Refer 6.8.7 Example of Multiple Interrupt Setting for disabling interrupts by using multiple interrupts. Note that either of the following methods may be used in setting Status <IE> of the CP0 register to "0."

1. Set Status <IE> of the CP0 register to "0" using the MTC0 instruction of 32 bit ISA.
2. Set IER of the CP0 register to "0" using the MTC0 instruction of 32 bit ISA (Note 1).
3. Set Status <IE> of the CP0 register to "0" using the MTC0 instruction of 16 bit ISA.
4. Execute the DI instruction of 16-bit mode ISA (Note 2).

(Note 1) This method is suitable for 32 bit ISA to avoid too many codes. If Toshiba C compiler is used, this instruction is executed by the 32 bit ISA instruction "_ _DI ( ) embedded function."

(Note 2) This method is suitable for 16 bit ISA to avoid too many codes. If Toshiba C compiler is used, this is executed by the 16 bit ISA instruction "_ _DI ( ) embedded function".

(Note 3) The methods 2 and 4 that can avoid too many codes and provide faster process are recommended.

If the factors once enabled are to be individually disabled again (IMCx<ILxxx> ="000") after setting interrupt levels by IMCx <ILxxx> of INTC, first set the Status <ERL/EXL/EI> bits of the CP0 register to disable interrupts and then disable relevant factors individually.

Example statements to individually disable interrupt factors:

```
mtc0    r0, IER          ; Interrupt is disabled (Status<IE> = "0")
sb      r0, IMCxx        ; Disable interrupt factors
sync                     ; Stall until it is write-enabled.
mtc0    r29, IER         ; Interrupt is enabled (Status<IE> = "1")
```

(Note) The above examples assume uses of Assembler made by Toshiba. If any third party Assembler is used, it may generate syntax errors; you are advised to modify the above statements according to the Assembler to be used.

6.8.6 Interrupt Processing

This section describes detailed operation of interrupt processing using the basic flow chart of Fig. 6-4.

6.8.6.1 Interrupt Response and Return

1) Hardware processes to accept interrupts

After interrupt request arbitration, INTC sets the interrupt vector and interrupt level of the interrupt request accepted to IVR and ILEV<CMASK>, respectively, to notify the TX19A processor core of the interrupt level. When the interrupt level is notified, the TX19A processor core sets Status <EXL> of the CP0 register to “1” to disable interrupts and saves the PC value at the interrupt generation to EPC. If the shadow register set is enabled (CP0 register SSCR <SSD> = 0), the processor core sets the interrupt level to SSCR <CSS> of the CP0 register and switches the register bank.

When an interrupt is accepted, any ongoing execution is suspended and it automatically jumps to the exception vector address (for interrupts). Fig. 6-5 shows the sequence of accepting interrupts.

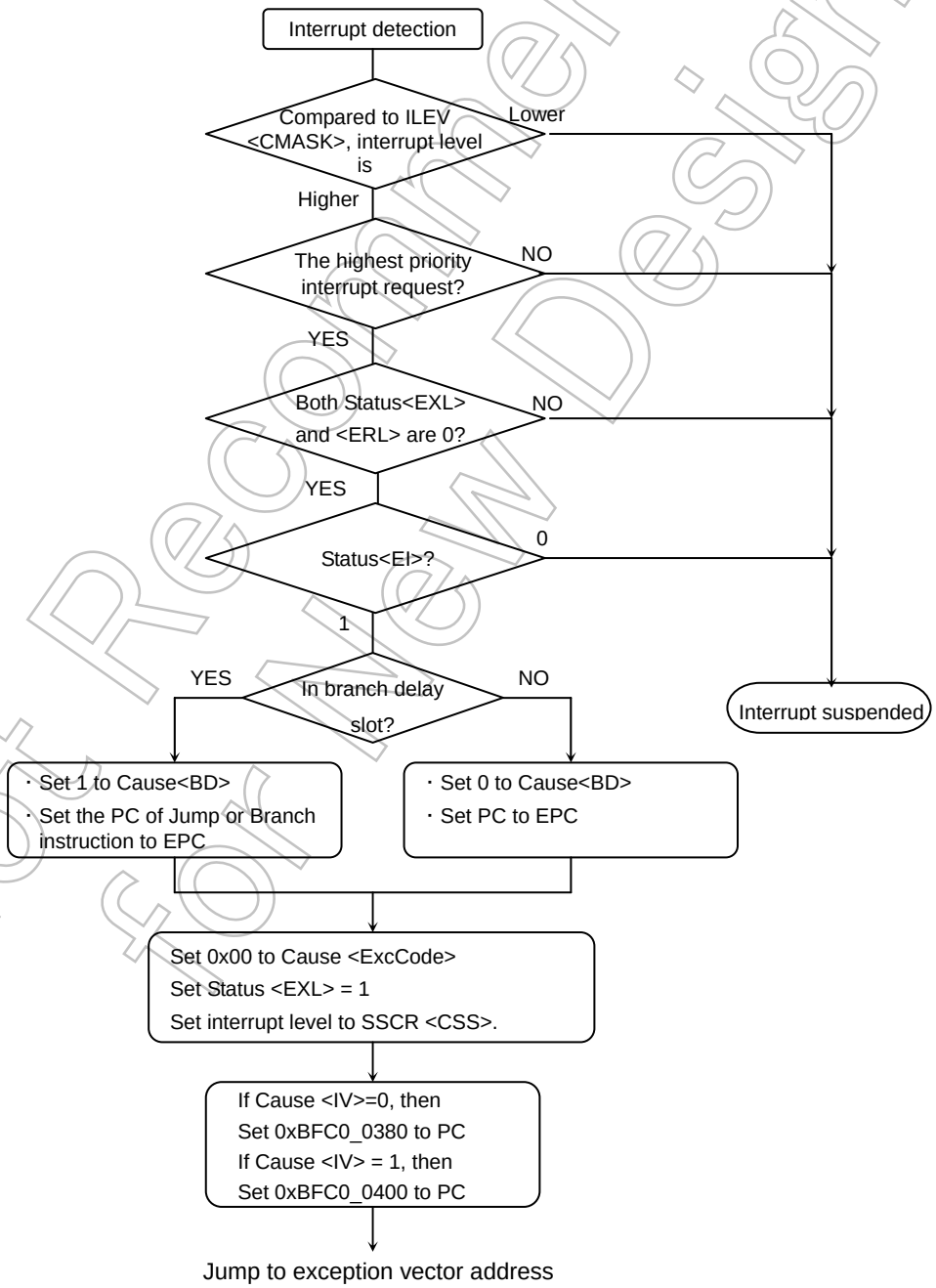


Fig. 6-5 Hardware Process Flow to Accept Interrupts

2) Processes to be performed by the exception handler

After an interrupt request is accepted, it automatically jumps to the exception handler where the interrupt vector address is read from INTC IVR and the user program generates the address of the interrupt handler. As in the example statements presented in Section 6.8.5, “Initialization for Interrupts,” the interrupt vector base address is set to IVR[31:8] so that the IVR value becomes the interrupt vector address.

After reading the INTC IVR value, the interrupt factor is cleared. If the interrupt factor is cleared before IVR is read, correct value cannot be read because the IVR value is also cleared.

Example exception handler statement: Exception vector address (interrupt) is 0xBFC0_0400.

```
VECTOR_INT section code isa32 abs=0xBFC00400
__InterruptVector:
lui     r26,hi(IVR)
lw      r26,lo(IVR)(r26)        ; Read IVR for interrupt vector address
lui     r27,hi(INTCLR)
sh      r26,lo(INTCLR)(r27)     ; Interrupt request is cleared
lw      r26,0(r26)              ; Read interrupt handler address from interrupt vector
jr      r26                     ; Jump to interrupt handler
nop
```

(Note) The above example assumes use of Assembler made by Toshiba. If any third party Assembler is used, it may generate syntax errors; you are advised to modify the above statement according to the Assembler to be used.

3) Processes to be performed by the interrupt handler

Typical tasks of the interrupt handler are to save appropriate registers and to process interrupts. If the shadow register set is enabled (CP0 register SSCR <SSD> = 0), the general purpose register values other than r26, r27, r28, and r29 (Shadow Register Set number 1 to 7) are automatically saved so the user program doesn't have to save these. Refer to the separate volume “TX19A Core Architecture” for details of general purpose registers that are to be automatically saved.

In general, registers other than GPR are dependent on user programs. The Status, EPC, SSCR, HI, LO, Cause, and Config values of the CP0 register shall be saved as appropriate.

For using multiple interrupts, interrupts are enabled by clearing Status <EXL> of the CP0 register to “0” after appropriate saving processes.

(Note) Note that general exceptions can be accepted even when interrupts are disabled. So, even when you don't use multiple interrupts, it is desirable to save any general purpose register and the CP0 register that could be overwritten by general exceptions.

Example interrupt handler settings to be necessary:

```
Save from SSCR to stack      ; Save SSCR values (as appropriate)
NOP instruction      ; Stall until SSCR is switched
NOP instruction      ; Stall until SSCR is switched
Save from EPC to stack       ; Save EPC values (as appropriate)
Save from Status to stack    ; Save Status values (as appropriate)
NOP instruction      ; Stall before executing ERET instruction
NOP instruction      ; Stall before executing ERET instruction
Status<EXL> = “0”  ; Interrupt enable (only for multiple interrupts)
```

(Note) After overwriting SSCR of the CP0 register, wait for two cycles to allow for register bank switching before attempting a register access.

4) Returning from the interrupt handler

For returning from the interrupt handler to the main process, return the register values saved at the top of the interrupt handler process and set “0” to INTC ILEV <MLEV> to clear the interrupt mask level. By executing the ERET instruction after all the return tasks are completed, Status <EXL> of the CP0 register is cleared to “0” and the EPC address returns to PC for the main process to be resumed. If the shadow register set has been enabled (CP0 register SSCR <SSD> = 0), SSCR <CSS> is updated by the ERET instruction and the Shadow Register Set number is automatically decremented for automatically returning the general purpose registers saved in the register bank.

If multiple interrupts are used, it is necessary to set Status <EXL> of the CP0 register to “1” to disable interrupts prior to executing the return process.

Example settings to return from the interrupt hander:

```
Status<EXL> =“1”          ; Interrupt disable (only for multiple interrupts)
ILEV<MLEV> =“0”           ; Decrement the mask level
SYNC instrucition          ; Stall until mask level is decremented
Return to SSCR             ; Return SSCR values saved (as appropriate)
NOP instruction            ; Stall until SSCR is switched
NOPinstruction             ; Stall until SSCR is switched
Return to EPC              ; Return EPC values saved (as appropriate)
Return to Status           ; Return Status values saved (as appropriate)
NOP instruction            ; Stall before executing ERET instruction
NOP instruction            ; Stall before executing ERET instruction
ERET instruction           ; Status<EXL> = “0,” EPC to PC, SSCR<PSS> to
                             SSCR<CSS>
```

(Note 1) After overwriting SSCR of the CP0 register, wait for two cycles to allow for register bank switching before attempting a register access.

(Note 2) Don't access the CP0 register at executing two instructions prior to the ERET instruction.

6.8.7 Example of Multiple Interrupt Setting

In “multiple interrupt” processing, a higher interrupt level interrupt is processed while an interrupt is being processed. With TMP19A23, multiple interrupts are processed through the interrupt priority arbitration function of INTC. When an interrupt request is accepted, ILEV <CMASK> of INTC is automatically updated to the interrupt level of the interrupt accepted to enable arbitration to use the priority preset by the user program.

1) Additional processes required for multiple interrupts

When an interrupt is accepted, Status <EXL> of the CP0 register is set to “1” disabling further interrupts. In order to allow multiple interrupts, it is necessary to save the registers that could be overwritten by the second and the following interrupts before enabling the multiple interrupt process. For this purpose, in addition to the typical exception handler and interrupt handler processes, save the following registers before setting Status <EXL> of the CP0 register to “0” to enable interrupts.

CP0 registers that must be saved:

- EPC
- SSCR
- Status

Save the HI, LO, Cause, and Config registers as appropriate.

(Note) Some of the registers may be automatically saved and returned by using some interrupt function of Toshiba C compiler. Refer to “TX19A C Compiler Reference” provided with the Toshiba C compiler for more details.

2) Additional return processes required for multiple interrupts

Before returning registers in the interrupt return process, it is necessary to disable interrupts using the method described in Section 6.8.5.4 “Interrupt Disable.” This is to prevent the returned register values from being corrupted by multiple interrupts. Note that the ERET instruction automatically clears Status <EXL> of the CP0 register to “0.” So, by setting Status <EXL> of the CP0 register to “1” to disable interrupts in the returning process, you can return from the interrupt with interrupts enabled automatically.

3) Proper use of Status <EXL> and Status <IE>

While there is no significant distinction between the Status <EXL> and Status <IE> parameters, Status <EXL> is automatically set to “1” upon interrupt generation and cleared to “0” by the ERET instruction automatically. In saving and returning register values at the initial and final phases of an interrupt process, where interrupts have to be disabled, hardware controlled Status <EXL> is normally used. Status <IE> is used for other general interrupt enable/disable control functions.

Applicable interrupt enable/disable control sequences are described in Section 6.8.7.1, “Interrupt Control for Multiple Interrupts.”

6.8.7.1 Interrupt Control for Multiple Interrupts

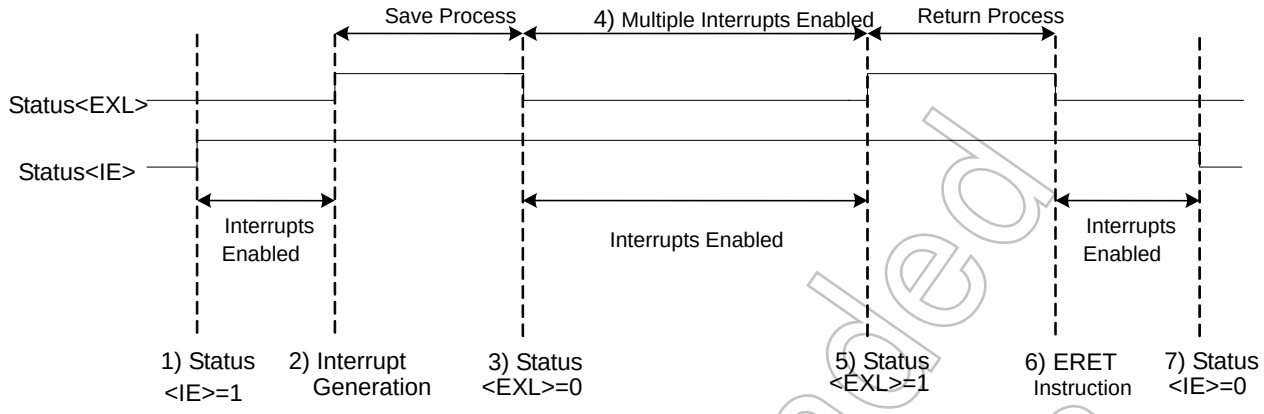


Fig. 6-6 Interrupt Enable/Disable Control Sequence for Multiple Interrupts

1) Status<IE>=1

Interrupts can be enabled by setting Status <IE> of the CP0 register to "1" while Status <EXL> is set to "0." This optional setting is made by the software program when it is necessary.

2) Interrupt Generation

When an interrupt is generated, Status <EXL> of the CP0 register is set to "1" disabling further interrupts. This process is automatically performed by hardware.

3) Status<EXL>=0

If multiple interrupts are to be enabled, it is necessary to set Status <EXL> of the CP0 register to "0" to enable interrupts after relevant registers are saved. If interrupts are enabled before saving registers, a higher priority level interrupt could corrupt the register data. This optional setting is made by the software program when it is necessary.

4) Multiple Interrupts Enabled

This is the period multiple interrupts are enabled. Interrupts with a level higher than the present interrupt level (ILEV <CMASK>) are to be accepted. If it is desired to disable interrupts during this period, set Status <IE> of the CP0 register to "0."

5) Status<EXL>=1

If multiple interrupts are enabled, it is necessary to set Status <EXL> of the CP0 register to "1" to disable interrupts before returning relevant register values. If registers are saved before disabling interrupts, a higher priority level interrupt could corrupt the register data. This optional setting is made by the software program when it is necessary.

6) ERET instruction

This instruction returns the system to the state before the interrupt generation. If this instruction is executed while Status <EXL> of the CP0 register is set to "1," the Status <EXL> will be automatically set to "0" and interrupt is enabled (provided that Status <IE> of the CP0 register is set to "1").

7) Status<IE>=0

Interrupts can be disabled by setting Status <IE> of the CP0 register to "0." This optional setting is made by the software program when it is necessary.

6.8.8 Registers

6.8.8.1 Register Map

Table 6-6 INTC Register Map

Address	Register symbol	Register	Corresponding interrupt number
0xFFFF_E000	IMC0	Interrupt mode control register 00	0 - 3
0xFFFF_E004	IMC1	Interrupt mode control register 04	4 - 7
0xFFFF_E008	IMC2	Interrupt mode control register 08	8 - 11
0xFFFF_E00C	IMC3	Interrupt mode control register 12	12 - 15
0xFFFF_E010	IMC4	Interrupt mode control register 16	16 - 19
0xFFFF_E014	IMC5	Interrupt mode control register 20	20 - 23
0xFFFF_E018	IMC6	Interrupt mode control register 24	24 - 27
0xFFFF_E01C	IMC7	Interrupt mode control register 28	28 - 31
0xFFFF_E020	IMC8	Interrupt mode control register 32	32 - 35
0xFFFF_E024	IMC9	Interrupt mode control register 36	36 - 39
0xFFFF_E028	IMCA	Interrupt mode control register 40	40 - 43
0xFFFF_E02C	IMCB	Interrupt mode control register 44	44 - 47
0xFFFF_E030	IMCC	Interrupt mode control register 48	48 - 51
0xFFFF_E034	IMCD	Interrupt mode control register 52	52 - 55
0xFFFF_E038	IMCE	Interrupt mode control register 56	56 - 59
0xFFFF_E03C	IMCF	Interrupt mode control register 60	60 - 63
0xFFFF_E040	IVR	Interrupt vector register	
0xFFFF_E060	INTCLR	Interrupt request clear register	
0xFFFF_E10C	ILEV	Interrupt mask level register	

(Note) While the interrupt mode control register (IMCxx) is a 32 bit register, 8 bit/16 bit access is also accepted.

6.8.8.2 Interrupt Vector Registers (IVR)

For an interrupt generated, the IVR register indicates the interrupt vector address of the corresponding interrupt factor. When an interrupt request is accepted, the corresponding value as listed in Table 6-2 is set to IVR [7:0]. By setting the base address of interrupt vectors to IVR [31:8], a read/write register, simply reading the IVR value can provide the corresponding interrupt vector address.

Interrupt Vector Register

IVR (0xFFFF_E040)

	7	6	5	4	3	2	1	0
bit Symbol	IVR7	IVR6	IVR5	IVR4	IVR3	IVR2	IVR1	IVR0
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	The vector of the interrupt factor generated is set.						Always reads “0.”	
	15	14	13	12	11	10	9	8
bit Symbol	IVR15	IVR14	IVR13	IVR12	IVR11	IVR10	IVR9	IVR8
Read/Write	R/W							R
After reset	0	0	0	0	0	0	0	0
Function								Always reads “0.”
	23	22	21	20	19	18	17	16
bit Symbol	IVR23	IVR22	IVR21	IVR20	IVR19	IVR18	IVR17	IVR16
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function								
	31	30	29	28	27	26	25	24
bit Symbol	IVR31	IVR30	IVR29	IVR28	IVR27	IVR26	IVR25	IVR24
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function								

6.8.8.3 Interrupt Level Register (ILEV)

ILEV is the register to control the interrupt level to be used by INTC in notifying interrupt requests to the TX19A processor core.

Interrupts with interrupt levels not higher than ILEV <CMASK> are suspended. The interrupt priority level "7" is the highest priority and "1" the lowest. Note that any interrupt with interrupt level 0 is not suspended.

When a new interrupt is generated, the corresponding interrupt level is stored in <CMASK> and any previously stored values are incremented in mask levels such that the previous CMASK is saved in PMASK0 and PMASK0 is saved in PMASK1 and so on. For writing a new value to <CMASK>, set "1" to <MLEV> and write <CMASK> simultaneously. Writing a new value to <PMASKx> cannot be made.

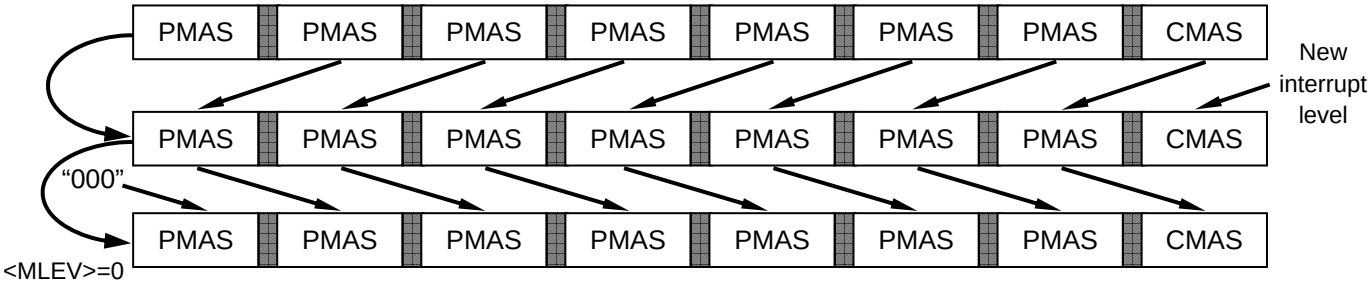
When <MLEV> is set to "0," the interrupt mask levels in the register shift back to the previous state such that PMASK0 is moved to CMASK and PMASK1 is moved to PMASK0, and so on. The last <PMASK6> is set to "000." If it is used in returning from an interrupt process, be sure to set <MLEV> to "0" before executing the ERET instruction. <MLEV> always reads "0."

Interrupt Level Register

ILEV (0xFFFF_E10C)

	7	6	5	4	3	2	1	0
Bit symbol	—	PMASK0			—	CMASK		
Read/Write	R					R/W		
After reset	0	000			0	000		
Function		Interrupt mask level (previous) 0				Interrupt mask level (current)		
	15	14	13	12	11	10	9	8
Bit symbol	—	PMASK2			—	PMASK1		
Read/Write	R							
After reset	0	000			0	000		
Function		Interrupt mask level (previous) 2				Interrupt mask level (previous) 1		
	23	22	21	20	19	18	17	16
Bit symbol	—	PMASK4			—	PMASK3		
Read/Write	R							
After reset	0	000			0	000		
Function		Interrupt mask level (previous) 4				Interrupt mask level (previous) 3		
	31	30	29	28	27	26	25	24
Bit symbol	MLEV	PMASK6			—	PMASK5		
Read/Write	W	R						
After reset	0	000			0	000		
Function	0: Return mask level 1: Change CMASK	Interrupt mask level (previous) 6				Interrupt mask level (previous) 5		

(Note 1) This register must be 32-bit accessed.
(Note 2) Be sure to read the IVR value before changing the ILEV value. If the ILEV value is changed before reading IVR, an unexpected interrupt request may be generated. Please set the mask level and <MLEV> individually.
(Note 3) Bit manipulation instructions cannot be used to access this register.



6.8.8.4 Interrupt mode control register (IMCxx)

IMCxx is comprised of <Ilxxx>, which determines the interrupt levels of individual interrupt factors, <DMxx>, which is used to set activation factors of DMA transfer, and <EIMXX>, which determines active state of interrupt requests.

IMC0 (0xFFFF_E000)

	7	6	5	4	3	2	1	0
Bit symbol		EIM01	EIM00	DM0		IL02	IL01	IL00
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: Disable 10: Disable 11: Disable **Be sure to set "00."**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 0 is set as the activation factor.	Always reads "0."	If DM0 = 0, select the interrupt level for interrupt number 0 (software set). 000: Disable Interrupt 001-111: 1-7 If DM0 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	15	14	13	12	11	10	9	8
Bit symbol		EIM11	EIM10	DM1		IL12	IL11	IL10
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: "H" level 10: Falling edge 11: Rising edge **CG setting "01".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 1 is set as the activation factor.	Always reads "0."	If DM1 = 0, select the interrupt level for interrupt number 1 (INT0). 000: Disable Interrupt 001-111: 1-7 If DM1 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	23	22	21	20	19	18	17	16
Bit symbol		EIM21	EIM20	DM2		IL22	IL21	IL20
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: "H" level 10: Falling edge 11: Rising edge **CG setting "01".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 2 is set as the activation factor.	Always reads "0."	If DM2 = 0, select the interrupt level for interrupt number 2 (INT1). 000: Disable Interrupt 001-111: 1-7 If DM2 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	31	30	29	28	27	26	25	24
Bit symbol		EIM31	EIM30	DM3		IL32	IL31	IL30
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: "H" level 10: Falling edge 11: Rising edge **CG setting "01".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 3 is set as the activation factor.	Always reads "0."	If DM3 = 0, select the interrupt level for interrupt number 3 (INT2). 000: Disable Interrupt 001-111: 1-7 If DM3 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

IMC1
(0xFFFF_E004)

	7	6	5	4	3	2	1	0
Bit symbol		EIM41	EIM40	DM4		IL42	IL41	IL40
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: "H" level 10: Falling edge 11: Rising edge **CG setting "01".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 4 is set as the activation factor.	Always reads "0."	If DM4 = 0, select the interrupt level for interrupt number 4 (INT3) 000: Disable Interrupt 001-111: 1-7 If DM4 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	15	14	13	12	11	10	9	8
Bit symbol		EIM51	EIM50	DM5		IL52	IL51	IL50
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: "H" level 10: Falling edge 11: Rising edge **CG setting "01".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 5 is set as the activation factor.	Always reads "0."	If DM5 = 0, select the interrupt level for interrupt number 5 (INT4). 000: Disable Interrupt 001-111: 1-7 If DM5 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	23	22	21	20	19	18	17	16
Bit symbol		EIM61	EIM60	DM6		IL62	IL61	IL60
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: "H" level 10: Falling edge 11: Rising edge **CG setting "01".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 6 is set as the activation factor.	Always reads "0."	If DM6 = 0, select the interrupt level for interrupt number 6 (INT5). 000: Disable Interrupt 001-111: 1-7 If DM6 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	31	30	29	28	27	26	25	24
Bit symbol		EIM71	EIM70	DM7		IL72	IL71	IL70
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: "H" level 10: Falling edge 11: Rising edge **CG setting "01".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 7 is set as the activation factor.	Always reads "0."	If DM7 = 0, select the interrupt level for interrupt number 7 (INT6). 000: Disable Interrupt 001-111: 1-7 If DM7 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

IMC2
(0xFFFF_E008)

	7	6	5	4	3	2	1	0
Bit symbol		EIM81	EIM80	DM8		IL82	IL81	IL80
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: "H" level 10: Falling edge 11: Rising edge **CG setting "01".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 8 is set as the activation factor.	Always reads "0."	If DM8 = 0, select the interrupt level for interrupt number 8 (INT7). 000: Disable Interrupt 001-111: 1-7 If DM8 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	15	14	13	12	11	10	9	8
Bit symbol		EIM91	EIM90	DM9		IL92	IL91	IL90
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: "H" level 10: Falling edge 11: Rising edge **CG setting "01".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 9 is set as the activation factor.	Always reads "0."	If DM9 = 0, select the interrupt level for interrupt number 9 (INT8). 000: Disable Interrupt 001-111: 1-7 If DM9 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	23	22	21	20	19	18	17	16
Bit symbol		EIMA1	EIMA0	DMA		ILA2	ILA1	ILA0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: "H" level 10: Falling edge 11: Rising edge **CG setting "01".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 10 is set as the activation factor.	Always reads "0."	If DMA = 0, select the interrupt level for interrupt number 10 (INT9). 000: Disable Interrupt 001-111: 1-7 If DMA = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	31	30	29	28	27	26	25	24
Bit symbol		EIMB1	EIMB0	DMB		ILB2	ILB1	ILB0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: "H" level 10: Falling edge 11: Rising edge **CG setting "01".**		Set as DMAC activation factor. 0: Non-activation factor 1:Interrupt Number 11 is set as the activation factor.	Always reads "0."	If DMB = 0, select the interrupt level for interrupt number 11 (INTA) 000: Disable Interrupt 001-111: 1-7 If DMB = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

IMC3
(0xFFFF_E00C)

	7	6	5	4	3	2	1	0
Bit symbol		EIMC1	EIMC0	DMC		ILC2	ILC1	ILC0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: "H" level 10: Falling edge 11: Rising edge **CG setting "01".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 12 is set as the activation factor.	Always reads "0."	If DMC = 0, select the interrupt level for interrupt number 12 (INTB) 000: Disable Interrupt 001-111: 1-7 If DMC = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	15	14	13	12	11	10	9	8
Bit symbol		EIMD1	EIMD0	DMD		ILD2	ILD1	ILD0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: "H" level 10: Falling edge 11: Rising edge **CG setting "01".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 13 is set as the activation factor.	Always reads "0."	If DMD = 0, select the interrupt level for interrupt number 13 (INTC) 000: Disable Interrupt 001-111: 1-7 If DMD = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	23	22	21	20	19	18	17	16
Bit symbol		EIME1	EIME0	DME		ILE2	ILE1	ILE0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: "H" level 10: Falling edge 11: Rising edge **CG setting "01".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 14 is set as the activation factor.	Always reads "0."	If DME = 0, select the interrupt level for interrupt number 14 (INTD) 000: Disable Interrupt 001-111: 1-7 If DME = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	31	30	29	28	27	26	25	24
Bit symbol		EIMF1	EIMF0	DMF		ILF2	ILF1	ILF0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: "H" level 10: Falling edge 11: Rising edge **CG setting "01".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 15 is set as the activation factor.	Always reads "0."	If DMF = 0, select the interrupt level for interrupt number 15 (INTE) 000: Disable Interrupt 001-111: 1-7 If DMF = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

IMC4
(0xFFFF_E010)

	7	6	5	4	3	2	1	0
Bit symbol		EIMC1	EIMC0	DMC		ILC2	ILC1	ILC0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level 01: "H" level 10: Falling edge 11: Rising edge **CG setting "01".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 16 is set as the activation factor.	Always reads "0."	If DMC = 0, select the interrupt level for interrupt number 12 (INTF) 000: Disable Interrupt 001-111: 1-7 If DMC = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		
Bit symbol								
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function								

Not embedded in BGA.

	15	14	13	12	11	10	9	8
Bit symbol		EIM111	EIM110	DM11		IL112	IL111	IL110
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 17 is set as the activation factor.	Always reads "0."	If DM11 = 0, select the interrupt level for interrupt number 17 (INTRX0) 000: Disable Interrupt 001-111: 1-7 If DM11 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	23	22	21	20	19	18	17	16
Bit symbol		EIM121	EIM120	DM12		IL122	IL121	IL120
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 18 is set as the activation factor.	Always reads "0."	If DM12 = 0, select the interrupt level for interrupt number 18 (INTRX0) 000: Disable Interrupt 001-111: 1-7 If DM12 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	31	30	29	28	27	26	25	24
Bit symbol		EIM131	EIM130	DM13		IL132	IL131	IL130
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 19 is set as the activation factor.	Always reads "0."	If DM13 = 0, select the interrupt level for interrupt number 19 (INTRX1) 000: Disable Interrupt 001-111: 1-7 If DM13 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

Note: Default values of EIMxx0 and EIMxx1 are different from the values to be used. Properly set them to the specified values before use.

IMC5
(0xFFFF_E014)

	7	6	5	4	3	2	1	0
Bit symbol		EIM141	EIM140	DM14		IL142	IL141	IL140
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 20 is set as the activation factor.	Always reads "0."	If DM14 = 0, select the interrupt level for interrupt number 20 (INTTX1) 000: Disable Interrupt 001-111: 1-7 If DM14 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	15	14	13	12	11	10	9	8
Bit symbol		EIM151	EIM150	DM15		IL152	IL151	IL150
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 21 is set as the activation factor.	Always reads "0."	If DM15 = 0, select the interrupt level for interrupt number 21 (INTRX2) 000: Disable Interrupt 001-111: 1-7 If DM15 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	23	22	21	20	19	18	17	16
Bit symbol		EIM161	EIM160	DM16		IL162	IL161	IL160
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 22 is set as the activation factor.	Always reads "0."	If DM16 = 0, select the interrupt level for interrupt number 22 (INTTX2) 000: Disable Interrupt 001-111: 1-7 If DM16 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	31	30	29	28	27	26	25	24
Bit symbol		EIM171	EIM170	DM17		IL172	IL171	IL170
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 23 is set as the activation factor.	Always reads "0."	If DM17 = 0, select the interrupt level for interrupt number 23 (HINTRX0) 000: Disable Interrupt 001-111: 1-7 If DM17 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

Note: Default values of EIMxx0 and EIMxx1 are different from the values to be used. Properly set them to the specified values before use.

IMC6
(0xFFFF_E018)

	7	6	5	4	3	2	1	0
Bit symbol		EIM181	EIM180	DM18		IL182	IL181	IL180
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 24 is set as the activation factor.	Always reads "0."	If DM18 = 0, select the interrupt level for interrupt number 24 (HINTTX0) 000: Disable Interrupt 001-111: 1-7 If DM18 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	15	14	13	12	11	10	9	8
Bit symbol		EIM191	EIM190	DM19		IL192	IL191	IL190
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 25 is set as the activation factor.	Always reads "0."	If DM19 = 0, select the interrupt level for interrupt number 25 (INTSBI0) 000: Disable Interrupt 001-111: 1-7 If DM19 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	23	22	21	20	19	18	17	16
Bit symbol		EIM1A1	EIM1A0	DM1A		IL1A2	IL1A1	IL1A0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 26 is set as the activation factor.	Always reads "0."	If DM1A = 0, select the interrupt level for interrupt number 26 (INTADHP) 000: Disable Interrupt 001-111: 1-7 If DM1A = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	31	30	29	28	27	26	25	24
Bit symbol		EIM1B1	EIM1B0	DM1B		IL1B2	IL1B1	IL1B0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 27 is set as the activation factor.	Always reads "0."	If DM1B = 0, select the interrupt level for interrupt number 27 (INTADM0) 000: Disable Interrupt 001-111: 1-7 If DM1B = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

Note: Default values of EIMxx0 and EIMxx1 are different from the values to be used. Properly set them to the specified values before use.

IMC7
(0xFFFF_E01C)

	7	6	5	4	3	2	1	0
Bit symbol		EIM1C1	EIM1C0	DM1C		IL1C2	IL1C1	IL1C0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 28 is set as the activation factor.	Always reads "0."	If DM1C = 0, select the interrupt level for interrupt number 28 (INTADM1) 000: Disable Interrupt 001-111: 1-7 If DM1C = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	15	14	13	12	11	10	9	8
Bit symbol		EIM1D1	EIM1D0	DM1D		IL1D2	IL1D1	IL1D0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 29 is set as the activation factor.	Always reads "0."	If DM1D = 0, select the interrupt level for interrupt number 29 (INTTB0) 000: Disable Interrupt 001-111: 1-7 If DM1D = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	23	22	21	20	19	18	17	16
Bit symbol		EIM1E1	EIM1E0	DM1E		IL1E2	IL1E1	IL1E0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 30 is set as the activation factor.	Always reads "0."	If DM1E = 0, select the interrupt level for interrupt number 30 (INTTB1) 000: Disable Interrupt 001-111: 1-7 If DM1E = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	31	30	29	28	27	26	25	24
Bit symbol		EIM1F1	EIM1F0	DM1F		IL1F2	IL1F1	IL1F0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 31 is set as the activation factor.	Always reads "0."	If DM1F = 0, select the interrupt level for interrupt number 31 (INTTB2) 000: Disable Interrupt 001-111: 1-7 If DM1F = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

Note: Default values of EIMxx0 and EIMxx1 are different from the values to be used. Properly set them to the specified values before use.

IMC8
(0xFFFF_E020)

	7	6	5	4	3	2	1	0
Bit symbol		EIM201	EIM200	DM20		IL202	IL201	IL200
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 32 is set as the activation factor.	Always reads "0."	If DM20 = 0, select the interrupt level for interrupt number 32 (INTTB3) 000: Disable Interrupt 001-111: 1-7 If DM20 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	15	14	13	12	11	10	9	8
Bit symbol		EIM211	EIM210	DM21		IL212	IL211	IL210
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 33 is set as the activation factor.	Always reads "0."	If DM21 = 0, select the interrupt level for interrupt number 33 (INTTB4) 000: Disable Interrupt 001-111: 1-7 If DM21 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	23	22	21	20	19	18	17	16
Bit symbol		EIM221	EIM220	DM26		IL222	IL221	IL220
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 34 is set as the activation factor.	Always reads "0."	If DM22 = 0, select the interrupt level for interrupt number 34 (INTTB5) 000: Disable Interrupt 001-111: 1-7 If DM22 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	31	30	29	28	27	26	25	24
Bit symbol		EIM231	EIM230	DM23		IL232	IL231	IL230
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 35 is set as the activation factor.	Always reads "0."	If DM23 = 0, select the interrupt level for interrupt number 35 (INTTB6) 000: Disable Interrupt 001-111: 1-7 If DM23 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

Note: Default values of EIMxx0 and EIMxx1 are different from the values to be used. Properly set them to the specified values before use.

IMC9
(0xFFFF_E024)

	7	6	5	4	3	2	1	0
Bit symbol		EIM241	EIM240	DM24		IL242	IL241	IL240
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 36 is set as the activation factor.	Always reads "0."	If DM24 = 0, select the interrupt level for interrupt number 36 (INTTB7) 000: Disable Interrupt 001-111: 1-7 If DM24 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	15	14	13	12	11	10	9	8
Bit symbol		EIM251	EIM250	DM25		IL252	IL251	IL250
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 37 is set as the activation factor.	Always reads "0."	If DM25 = 0, select the interrupt level for interrupt number 37 (INTTB8) 000: Disable Interrupt 001-111: 1-7 If DM25 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	23	22	21	20	19	18	17	16
Bit symbol		EIM261	EIM260	DM26		IL262	IL261	IL260
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 38 is set as the activation factor.	Always reads "0."	If DM26 = 0, select the interrupt level for interrupt number 38 (INTTB9) 000: Disable Interrupt 001-111: 1-7 If DM26 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	31	30	29	28	27	26	25	24
Bit symbol		EIM271	EIM270	DM27		IL272	IL271	IL270
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 39 is set as the activation factor.	Always reads "0."	If DM27 = 0, select the interrupt level for interrupt number 39 (INTSBI1) 000: Disable Interrupt 001-111: 1-7 If DM27 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

Note: Default values of EIMxx0 and EIMxx1 are different from the values to be used. Properly set them to the specified values before use.

IMCA
(0xFFFF_E028)

	7	6	5	4	3	2	1	0
Bit symbol		EIM281	EIM280	DM28		IL282	IL281	IL280
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 40 is set as the activation factor.	Always reads "0."	If DM28 = 0, select the interrupt level for interrupt number 40 (INTD0CMP0) 000: Disable Interrupt 001-111: 1-7 If DM28 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	15	14	13	12	11	10	9	8
Bit symbol		EIM291	EIM290	DM29		IL292	IL291	IL290
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 41 is set as the activation factor.	Always reads "0."	If DM29 = 0, select the interrupt level for interrupt number 41 (INTD0CMP1) 000: Disable Interrupt 001-111: 1-7 If DM29 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	23	22	21	20	19	18	17	16
Bit symbol		EIM2A1	EIM2A0	DM2A		IL2A2	IL2A1	IL2A0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 42 is set as the activation factor.	Always reads "0."	If DM2A = 0, select the interrupt level for interrupt number 42 (INTD0CMP2) 000: Disable Interrupt 001-111: 1-7 If DM2A = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	31	30	29	28	27	26	25	24
Bit symbol		EIM2B1	EIM2B0	DM2B		IL2B2	IL2B1	IL2B0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 43 is set as the activation factor.	Always reads "0."	If DM2B = 0, select the interrupt level for interrupt number 43 (INTD0CMP3) 000: Disable Interrupt 001-111: 1-7 If DM2B = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

Note: Default values of EIMxx0 and EIMxx1 are different from the values to be used. Properly set them to the specified values before use.

IMCB
(0xFFFF_E02C)

	7	6	5	4	3	2	1	0
Bit symbol		EIM2C1	EIM2C0	DM2C		IL2C2	IL2C1	IL2C0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 44 is set as the activation factor.	Always reads "0."	If DM2C = 0, select the interrupt level for interrupt number 44 (INTD0CMP4) 000: Disable Interrupt 001-111: 1-7 If DM2C = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	15	14	13	12	11	10	9	8
Bit symbol		EIM2D1	EIM2D0	DM2D		IL2D2	IL2D1	IL2D0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 45 is set as the activation factor.	Always reads "0."	If DM2D = 0, select the interrupt level for interrupt number 45 (INTD1CMP0) 000: Disable Interrupt 001-111: 1-7 If DM2D = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	23	22	21	20	19	18	17	16
Bit symbol		EIM2E1	EIM2E0	DM2E		IL2E2	IL2E1	IL2E0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 46 is set as the activation factor.	Always reads "0."	If DM2E = 0, select the interrupt level for interrupt number 46 (INTD1CMP1) 000: Disable Interrupt 001-111: 1-7 If DM2E = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	31	30	29	28	27	26	25	24
Bit symbol		EIM2F1	EIM2F0	DM2F		IL2F2	IL2F1	IL2F0
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 47 is set as the activation factor.	Always reads "0."	If DM2F = 0, select the interrupt level for interrupt number 47 (INTD1CMP2) 000: Disable Interrupt 001-111: 1-7 If DM2F = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

Note: Default values of EIMxx0 and EIMxx1 are different from the values to be used. Properly set them to the specified values before use.

IMCC (0xFFFF_E030)

	7	6	5	4	3	2	1	0
Bit symbol		EIM301	EIM300	DM30		IL302	IL301	IL300
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 48 is set as the activation factor.	Always reads "0."	If DM30 = 0, select the interrupt level for interrupt number 48 (INTD1CMP3) 000: Disable Interrupt 001-111: 1-7 If DM30 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	15	14	13	12	11	10	9	8
Bit symbol		EIM311	EIM310	DM31		IL312	IL311	IL310
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 1: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 49 is set as the activation factor.	Always reads "0."	If DM31 = 0, select the interrupt level for interrupt number 49 (INTD1CMP4) 000: Disable Interrupt 001-111: 1-7 If DM31 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 11: Rising edge **Be sure to set "11".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 50 is set as the activation factor.	Always reads "0."	If DM32 = 0, select the interrupt level for interrupt number 50 (INTAD) 000: Disable Interrupt 001-111: 1-7 If DM32= 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	31	30	29	28	27	26	25	24
Bit symbol		EIM331	EIM330	DM33		IL332	IL331	IL330
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level **Be sure to set "00".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 51 is set as the activation factor.	Always reads "0."	If DM33 = 0, select the interrupt level for interrupt number 51 (INTDMA0) 000: Disable Interrupt 001-111: 1-7 If DM33 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

(Note 1): Default values of EIMxx0 and EIMxx1 are different from the values to be used. Properly set them to the specified values before use.

(Note 2): The access to the DMAC register by DMAC is prohibited.

IMCD
(0xFFFF_E034)

	7	6	5	4	3	2	1	0
Bit symbol		EIM341	EIM340	DM34		IL342	IL341	IL340
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0		
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level **Be sure to set "00".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 52 is set as the activation factor.	Always reads "0."	If DM34 = 0, select the interrupt level for interrupt number 52 (INTDMA1) 000: Disable Interrupt 001-111: 1-7 If DM34 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	15	14	13	12	11	10	9	8
Bit symbol		EIM351	EIM350	DM35		IL352	IL351	IL350
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level **Be sure to set "00".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 53 is set as the activation factor.	Always reads "0."	If DM35 = 0, select the interrupt level for interrupt number 53 (INTDMA2) 000: Disable Interrupt 001-111: 1-7 If DM35 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	23	22	21	20	19	18	17	16
Bit symbol		EIM361	EIM360	DM36		IL362	IL361	IL360
Read/Write	R	R/W			R	R/W		
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."	Selects active state of interrupt request. 00: "L" level **Be sure to set "00".**		Set as DMAC activation factor. 0: Non-activation factor 1: Interrupt number 54 is set as the activation factor.	Always reads "0."	If DM36 = 0, select the interrupt level for interrupt number 54 (INTDMA3) 000: Disable Interrupt 001-111: 1-7 If DM36 = 1, select the DMAC channel. 000 to 011: 0 to 3 1xx: ---		

	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."							

(Note 1): Default values of EIMxx0 and EIMxx1 are different from the values to be used. Properly set them to the specified values before use.

(Note 2): The access to the DMAC register by DMAC is prohibited.

Note 1: Please ensure that the type of active state is selected before enabling an interrupt request.

Note 2: When making interrupt requests as DMAC activation factors, please ensure that you put the DMAC into standby mode after setting the INTC.

Note 3: An active condition must be changed after an interrupt output of the corresponding device becomes negate especially when you change it to the level detection.

(1) IL Set to "0" if it is set to "Excluding 0"

(2) Change Detection condition (EIM)

(3) INTCLR Clear pertinent interrupt.

(4) IL Set to"Excluding 0".

6.8.8.5 Interrupt Request Clear Registers (INTCLR)

Setting the IVR [7:0] for the corresponding interrupt factor into the INTCLR register enables to clear any interrupt request being suspended. Do not clear an interrupt request before reading the IVR value. When an interrupt request is cleared, the IVR value is also cleared and the interrupt factor cannot be determined anymore.

IVR <IVR7:0> value setting to clear the interrupt request

INTCLR (0xFFFF_E060)

	7	6	5	4	3	2	1	0
Bit symbol	EICLR7	EICLR6	EICLR5	EICLR4	EICLR3	EICLR2	EICLR1	EICLR0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Set the IVR <IVR7:0> value that corresponds to the interrupt request that you would like to clear.							

	15	14	13	12	11	10	9	8
Bit symbol								
Read/Write	R							
After reset	0							
Function	Always reads "0."							

	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write	R							
After reset	0							
Function	Always reads "0."							

	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write	R							
After reset	0							
Function	Always reads "0."							

(Note 1) This register must be 16-bit accessed.

(Note 2) In order to maintain interrupt factors regardless of the active state setting of INTC IMCx <EIMxx>, clear the interrupt request in any case whether in "H" level, "L" level, rising edge, or falling edge.

(Note 3) Bit manipulation instructions cannot be used to access this register.

(Note 4) External transfer requests due to DMAC interrupt factors are not cleared. Once an external transfer request is accepted, it will not be canceled until the DMA transfer is executed. Therefore, any unnecessary external transfer request should be cleared by executing DMA transfer, by disabling interrupts using IMCx <ILxxx> or by canceling the corresponding DMAC activation factors using IMCx<DMxx> before accepting such external transfer requests.

(Note 5) Be sure to clear the corresponding interrupt number with INTCLR after IMCx register setting.

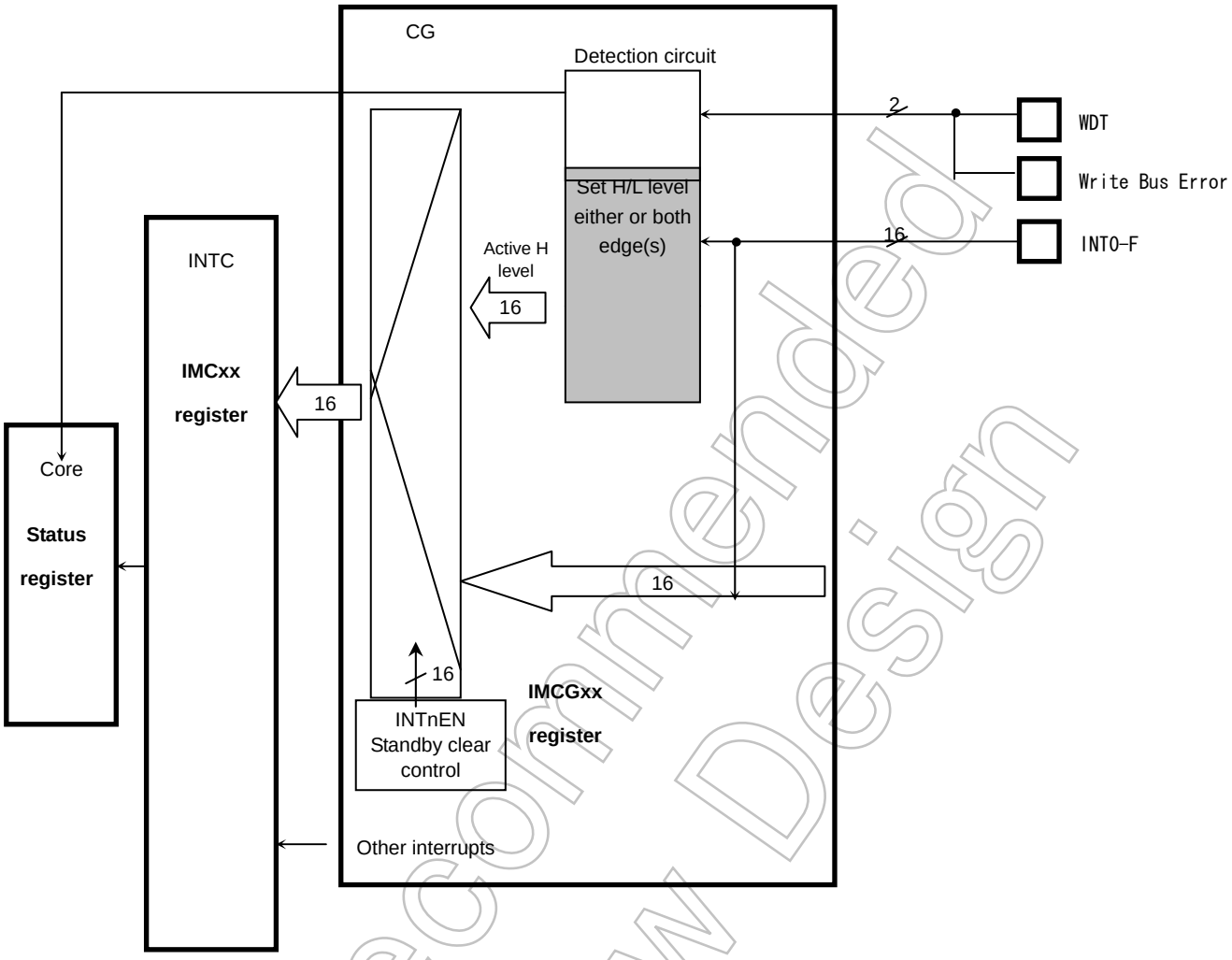


Fig. 6-7 Interrupt Connection Diagram

6.9 INTCG Registers (Interrupts to Clear STOP and IDLE)

INT0-INTF: STOP/ IDLE clear interrupt

IMCGA

(0xFFFF_EE10)

	7	6	5	4	3	2	1	0
Bit symbol		EIMCG02	EIMCG01	EIMCG00	EIMST01	EIMST00		INT0EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	Always reads "0."	Set active state of INT0 standby clear request. 000: "L" level 001: "H" level 010: Falling edge 011: Rising edge 100: Both edges			Active status of INT0 standby clear request 00: ー 01: Rising edge 10: Falling edge 11: Both edges		Always reads "0."	INT0 Clear input 0: Disable 1: Enable

	15	14	13	12	11	10	9	8
Bit symbol		EIMCG12	EIMCG11	EIMCG10	EIMST11	EIMST10		INT1EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	Always reads "0."	Set active state of INT1 standby clear request. 000: "L" level 001: "H" level 010: Falling edge 011: Rising edge 100: Both edges			Active status of INT1 standby clear request 00: ー 01: Rising edge 10: Falling edge 11: Both edges		Always reads "0."	INT1 Clear input 0: Disable 1: Enable

	23	22	21	20	19	18	17	16
Bit symbol		EMCG22	EMCG21	EMCG20	EMST21	EMST20		INT2EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	Always reads "0."	Set active state of INT2 standby clear request. 000: "L" level 001: "H" level 010: Falling edge 011: Rising edge 100: Both edges			Active status of INT2 standby clear request 00: ー 01: Rising edge 10: Falling edge 11: Both edges		Always reads "0."	INT2 Clear input 0: Disable 1: Enable

	31	30	29	28	27	26	25	24
Bit symbol		EMCG32	EMCG31	EMCG30	EMST31	EMST30		INT3EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	Always reads "0."	Set active state of INT3 standby clear request. 000: "L" level 001: "H" level 010: Falling edge 011: Rising edge 100: Both edges			Active status of INT3 standby clear request 00: ー 01: Rising edge 10: Falling edge 11: Both edges		Always reads "0."	INT3 Clear input 0: Disable 1: Enable

(Note 1) Refer to EMSTxx bit to know the active condition which is used for clearing standby.
(Note 2) Please specify the bit for the edge first and then specify the bit for the <INTxEN>. Setting them simultaneously is prohibited.

IMCGB
(0xFFFF_EE14)

	7	6	5	4	3	2	1	0
Bit symbol		EMCG42	EMCG41	EMCG40	EMST41	EMST40		INT4EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	Always reads “0.”	Set active state of INT4 standby clear request. 000: “L” level 001: “H” level 010: Falling edge 011: Rising edge 100: Both edges			Active status of INT4 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges		Always reads “0.”	INT4 Clear input 0: Disable 1: Enable

	15	14	13	12	11	10	9	8
Bit symbol		EMCG52	EMCG51	EMCG50	EMST51	EMST50		INT5EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	Always reads “0.”	Set active state of INT5 standby clear request. 000: “L” level 001: “H” level 010: Falling edge 011: Rising edge 100: Both edges			Active status of INT5 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges		Always reads “0.”	INT5 Clear input 0: Disable 1: Enable

	23	22	21	20	19	18	17	16
Bit symbol		EMCG62	EMCG61	EMCG60	EMST61	EMST60		INT6EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	Always reads “0.”	Set active state of INT6 standby clear request. 000: “L” level 001: “H” level 010: Falling edge 011: Rising edge 100: Both edges			Active status of INT6 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges		Always reads “0.”	INT6 Clear input 0: Disable 1: Enable

	31	30	29	28	27	26	25	24
Bit symbol		EMCG72	EMCG71	EMCG70	EMST71	EMST70		INT7EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	Always reads “0.”	Set active state of INT7 standby clear request. 000: “L” level 001: “H” level 010: Falling edge 011: Rising edge 100: Both edges			Active status of INT7 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges		Always reads “0.”	INT7 Clear input 0: Disable 1: Enable

(Note) Refer to EMSTxx bit to know the active condition which is used for clearing standby.
(Note 2) Please specify the bit for the edge first and then specify the bit for the <INTxEN>. Setting them simultaneously is prohibited.

IMCGC
(0xFFFF_EE18)

	7	6	5	4	3	2	1	0
Bit symbol		EMCG82	EMCG81	EMCG80	EMST81	EMST80		INT8EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	Always reads “0.”	Set active state of INT8 standby clear request. 000: “L” level 001: “H” level 010: Falling edge 011: Rising edge 100: Both edges 101~111:---			Active status of INT8 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges		Always reads “0.”	INT8 Clear input 0: Disable 1: Enable

	15	14	13	12	11	10	9	8
Bit symbol		EMCG92	EMCG91	EMCG90	EMST91	EMST90		INT9EN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	Always reads “0.”	Set active state of INT9 standby clear request. 000: “L” level 001: “H” level 010: Falling edge 011: Rising edge 100: Both edges 101~111:---			Active status of INT9 standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges		Always reads “0.”	INT9 Clear input 0: Disable 1: Enable

	23	22	21	20	19	18	17	16
Bit symbol		EMCGA2	EMCGA1	EMCGA0	EMSTA1	EMSTA0		INTAEN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	Always reads “0.”	Set active state of INTA standby clear request. 000: “L” level 001: “H” level 010: Falling edge 011: Rising edge 100: Both edges 101~111:---			Active status of INTA standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges		Always reads “0.”	INTA Clear input 0: Disable 1: Enable

	31	30	29	28	27	26	25	24
Bit symbol		EMCGB2	EMCGB1	EMCGB0	EMSTB1	EMSTB0		INTBEN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	Always reads “0.”	Set active state of INTB standby clear request. 000: “L” level 001: “H” level 010: Falling edge 011: Rising edge 100: Both edges 101~111:---			Active status of INTB standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges		Always reads “0.”	INTB Clear input 0: Disable 1: Enable

(Note) Refer to EMSTxx bit to know the active condition which is used for clearing standby.
(Note 2) Please specify the bit for the edge first and then specify the bit for the <INTxEN>. Setting them simultaneously is prohibited.

IMCGD
(0xFFFF_EE18)

	7	6	5	4	3	2	1	0
Bit symbol		EMCGC2	EMCGC1	EMCGC0	EMSTC1	EMSTC0		INTCEN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	Always reads “0.”	Set active state of INTC standby clear request. 00: “L” level 01: “H” level 10: Falling edge 11: Rising edge 100: Both edges			Active status of INTC standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges		Always reads “0.”	INTC Clear input 0: Disable 1: Enable
	15	14	13	12	11	10	9	8
Bit symbol		EMCGD2	EMCGD1	EMCGD0	EMSTD1	EMSTD0		INTDEN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	Always reads “0.”	Set active state of INTD standby clear request. 00: “L” level 01: “H” level 10: Falling edge 11: Rising edge 100: Both edges			Active status of INTD standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges		Always reads “0.”	INTD Clear input 0: Disable 1: Enable
	23	22	21	20	19	18	17	16
Bit symbol		EMCGE2	EMCGE1	EMCGE0	EMSTE1	EMSTE0		INTEEN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	Always reads “0.”	Set active state of INTE standby clear request. 00: “L” level 01: “H” level 10: Falling edge 11: Rising edge 100: Both edges			Active status of INTE standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges		Always reads “0.”	INTE Clear input 0: Disable 1: Enable
	31	30	29	28	27	26	25	24
Bit symbol		EMCGF2	EMCGF1	EMCGF0	EMSTF1	EMSTF0		INTFEN
Read/Write	R	R/W			R		R	R/W
After reset	0	0	1	0	0	0	0	0
Function	Always reads “0.”	Set active state of INTF standby clear request. 00: “L” level 01: “H” level 10: Falling edge 11: Rising edge 100: Both edges			Active status of INTF standby clear request 00: − 01: Rising edge 10: Falling edge 11: Both edges		Always reads “0.”	INTF Clear input 0: Disable 1: Enable

Not embedded in BGA.

(Note) Refer to EMSTxx bit to know the active condition which is used for clearing standby.
(Note 2) Please specify the bit for the edge first and then specify the bit for the <INTxEN>. Setting them simultaneously is prohibited.

Be sure to set active state of the clear request if interrupt is enabled for clearing the Stop or Idle standby mode.

(Note1) When using interrupts, be sure to follow the sequence of actions shown below:

1) If shared with other general ports, enable the target interrupt input.
2) Set conditions such as active state upon initialization.
3) Clear interrupt requests.
4) Enable interrupts.

(Note 2) Settings must be performed while interrupts are disabled.

(Note 3) For clearing the Stop mode with TMP19A23, 16 factors, INT0 to INTF, are available as clearing interrupts. You can use CG for selecting edge/level of active state and judging whether INT0 to INTF are to be used as STOP/IDLE interrupts clearing factor.

(Note 4) Among the above 16 factors to be assigned as Stop/Idle clear request interrupts, INT0 to INTB don't have to be set with CG if they are to be used as normal interrupts. Use INTC to specify either H/L level, rising/falling edge, or both edges.

Interrupt factors other than those assigned as Stop//Idle clear requests are set in the INTC block.

EICRCG
(0xFFFF_EE20)

	7	6	5	4	3	2	1	0
Bit symbol					ICRCG3	ICRCG2	ICRCG1	ICRCG0
Read/Write	R				W			
After reset	0				0	0	0	0
Function	Always reads “0.”				Always reads “0.” Clear interrupt requests. 0000: INT0 0101: INT5 1010: INTA 0001: INT1 0110: INT6 1011: INTB 0010: INT2 0111: INT7 1100: INTC 0011: INT3 1000: INT8 1101: INTD 0100: INT4 1001: INT9 1110: INTE 1111: INTF			

	15	14	13	12	11	10	9	8
Bit symbol								
Read/Write	R							
After reset	0							
Function	Always reads “0.”							

	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write								
After reset								
Function	Always reads “0.”							

	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write	R							
After reset	0							
Function	Always reads “0.”							

(Note) In case of clearing interrupt request of the above 16 factors that are assigned to clear Stop/ Idle modes,

-for INT0 to INTF, use the EIRCG register in the above CG block if they are used as clear factors.

-use the INTCLR to clear the interrupt factor if they are not used as clear factors.

NMI Flag Register

NMIFLG (0xFFFF_EE24)

	7	6	5	4	3	2	1	0
Bit symbol							WDT	WBER
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	Always reads “0.”						NMI factor 1: NMI generated by WDT interrupt	NMI factor 1: NMI generated by write bus error
	15	14	13	12	11	10	9	8
Bit symbol								
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	Always reads “0.”							
	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	Always reads “0.”							
	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	Always reads “0.”							

(Note) WDT and WBER are cleared to “0” when they are read.

Although TMP19A23 doesn't have NMI interrupts as external pin inputs, NMI interrupts are available as internal interrupt factors.

6.10 Cautions in Using Interrupts

The following paragraphs describe some points to be kept in mind in using interrupts. User programs must be written in a manner to satisfy the following details.

6.10.1 Cautions Related to TX19A Processor Core

- Exceptions cannot be disabled. Note that there are some cases where two different instructions can be distinguished only by exception generation. So, properly use them according to the specific usage.
- Software interrupts are different from the “software set” to be used as one of hardware interrupt factors.
- Immediately after overwriting SSCR of the CP0 register, add two NOP instructions to allow for register bank switching as it takes two clock cycles.
- In case multiple interrupts of the same interrupt level are accepted by changing ILEV <CMASK>, it is necessary for the user program to save because the register bank will not be switched.
- Only 32-bit ISA access can be used to access IER of the CP0 register.
- Different stack pointers (r29) are used for Shadow Register Set number 0 and Shadow Register Set numbers 1 to 7; it is necessary to set them separately (twice). If it is desired to use a common stack pointer, set SSCR<CSS> to “1” in the main process to use Shadow Register Set number 1. In this case, when a level 1 interrupt is accepted, it is necessary for the user program to save because the register bank will not be switched.
- If an ERET instruction is executed while interrupts are disabled by setting Status <ERL> of the CP0 register to “1,” it returns to the main process by using ErrorEPC of the CP0 register as the return address. As the TX19A processor core saves the interrupt return address to EPC, you should be careful if Status <ERL> is to be used for disabling interrupts.
- Don't execute an ERET instruction within two clock cycles after accessing Status, ErrorEPC, EPC, or SSCR of the CP0 register.
- If Status <ERL/EXL/IE> of the CP0 register is set to disable interrupts, interrupts are disabled at the time of instruction execution (E stage) but any value set to the register is reflected only two clocks later.
- If Status <ERL/EXL/IE> of the CP0 register is set to enable interrupts, interrupts are enabled two clocks after the instruction execution (E stage); any value set to the register is also reflected two clocks after the instruction execution (E stage).

6.10.2 Cautions Related to INTC

- If more than one interrupts of a same interrupt level are generated at the same time, interrupts are accepted from the factor of the smallest interrupt number.
- Any factor of interrupt level 0 is not suspended.
- Disabling interrupt factors individually (by setting interrupt level 0) is acceptable only while interrupts are disabled.
- Default settings of IMCx <EIMxx> of INTC may be different from the settings to be used.
- The INTC ILEV register must be 32-bit accessed.
- The INTC INTCLR register must be 32-bit accessed.
- When enabling interrupts, be sure to do so in the order of the detection route (from external to internal). When disabling, use the reverse order of the detection route (from internal to external).
- When a new value is written to INTC ILEV <CMASK>, set <MLEV> to “1” at the same time.
- During a level detection, the signal should not be changed until IVR is completely read.

7 Input/Output Ports

7.1 Port registers

Px :Port register

To read/ write port data.

PxCR :Control register

To control input/output

* Need to enable the input with PxIE register even when input is set.

PxFCn :Function register

To set functions. An assigned function can be activated by setting “1”.

PxOD :Open drain control register

To switch the input of a register that can be set as programmable open drain.

PxPUP :Pull up control register

To control program pull ups.

PxSEL :Serial setting register

To set for using serial function.

PxIE :Input control enable register

To control inputs. “0”cannot be set as a default to avoid through current. All the ports other than P0 and P1 require the setting.

7.2 Port 0 (P00 through P07)

The port 0 is a general-purpose, 8-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Outputs can be set by using the control register P0CR. A reset allows all bits of P0CR to be cleared to "0" and the port 0 to be put in output disable mode.

Besides the general-purpose input/output function, the port 0 performs other functions: D0 through D7 function as a data bus and AD0 through AD7 function as an address data bus. When external memory is accessed, the port 0 automatically functions as either a data bus or an address data bus, and all bits of P0CR are cleared to "0."

If the BUSMD pin (port P45) is set to "L" level during a reset, the port 0 is put in separate bus mode (D0 to D7). If it is set to "H" level during a reset, the port 0 is put in multiplexed mode (AD0 to AD7).

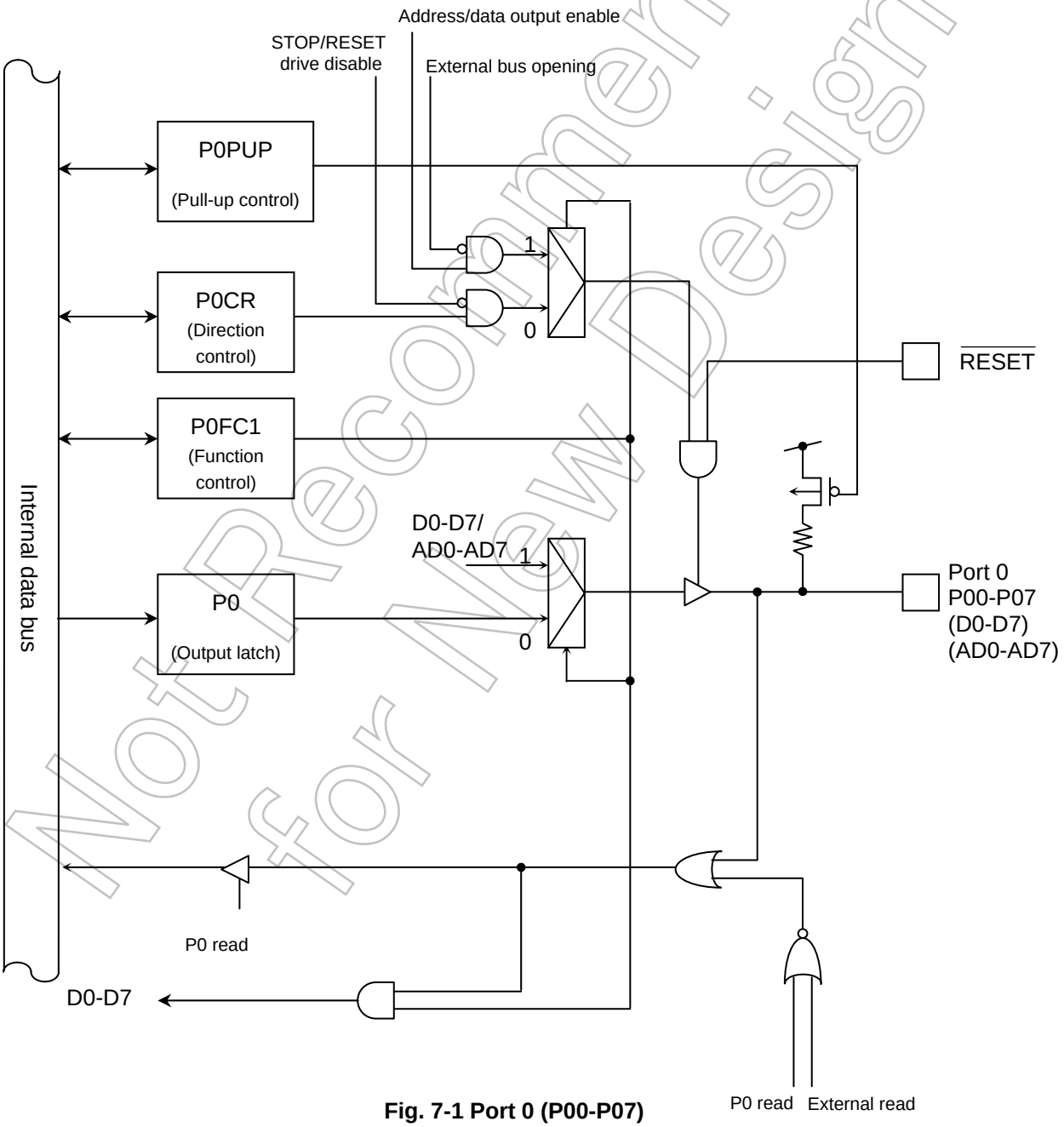


Fig. 7-1 Port 0 (P00-P07)

Port 0 register

		7	6	5	4	3	2	1	0
P0	Bit Symbol	P07	P06	P05	P04	P03	P02	P01	P00
(0xFFFF_F000)	Read/Write	R/W							
	After reset	Input mode (output latch register is cleared to “0.”)							

Port 0 control register

		7	6	5	4	3	2	1	0
P0CR	Bit Symbol	P07C	P06C	P05C	P04C	P03C	P02C	P01C	P00C
(0xFFFF_F004)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	0: Output disable 1: Output enable							

Port 0 function register 1

		7	6	5	4	3	2	1	0
P0FC1	Bit Symbol	P07F1	P06F1	P05F1	P04F1	P03F1	P02F1	P01F1	P00F1
(0xFFFF_F005)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	0: PORT 1:External bus setting							

Port 0 pull up control register

		7	6	5	4	3	2	1	0
P0PUP	Bit Symbol	PE07	PE06	PE05	PE04	PE03	PE02	PE01	PE00
(0xFFFF_F007)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up

7.3 Port 1(P10 through P17)

The port 1 is a general-purpose, 8-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Outputs can be set by using the function register P1FC 1, P1FC2 and the control register P1CR. A reset allows all bits of the output latch P1, P1CR, P1FC1 and P1FC2 to be cleared to "0" and the port 1 to be put in output disable mode.

Besides the general-purpose input/output function, the port 1 performs other functions: D8 through D15 function as a data bus, AD8 through AD15 function as an address data bus, and A8 through A15 function as an address bus. To access external memory, registers P1CR, P1FC1 and P1FC2 must be provisioned to allow the port 1 to function as either an address bus or an address data bus.

If the BUSMD pin (port 44) is set to "L" level during a reset, the port 1 is put in separate bus mode (D8 to D15). If it is set to "H" level during a reset, the port 1 is put in multiplexed mode (AD8 to AD15 or A8 to A15).

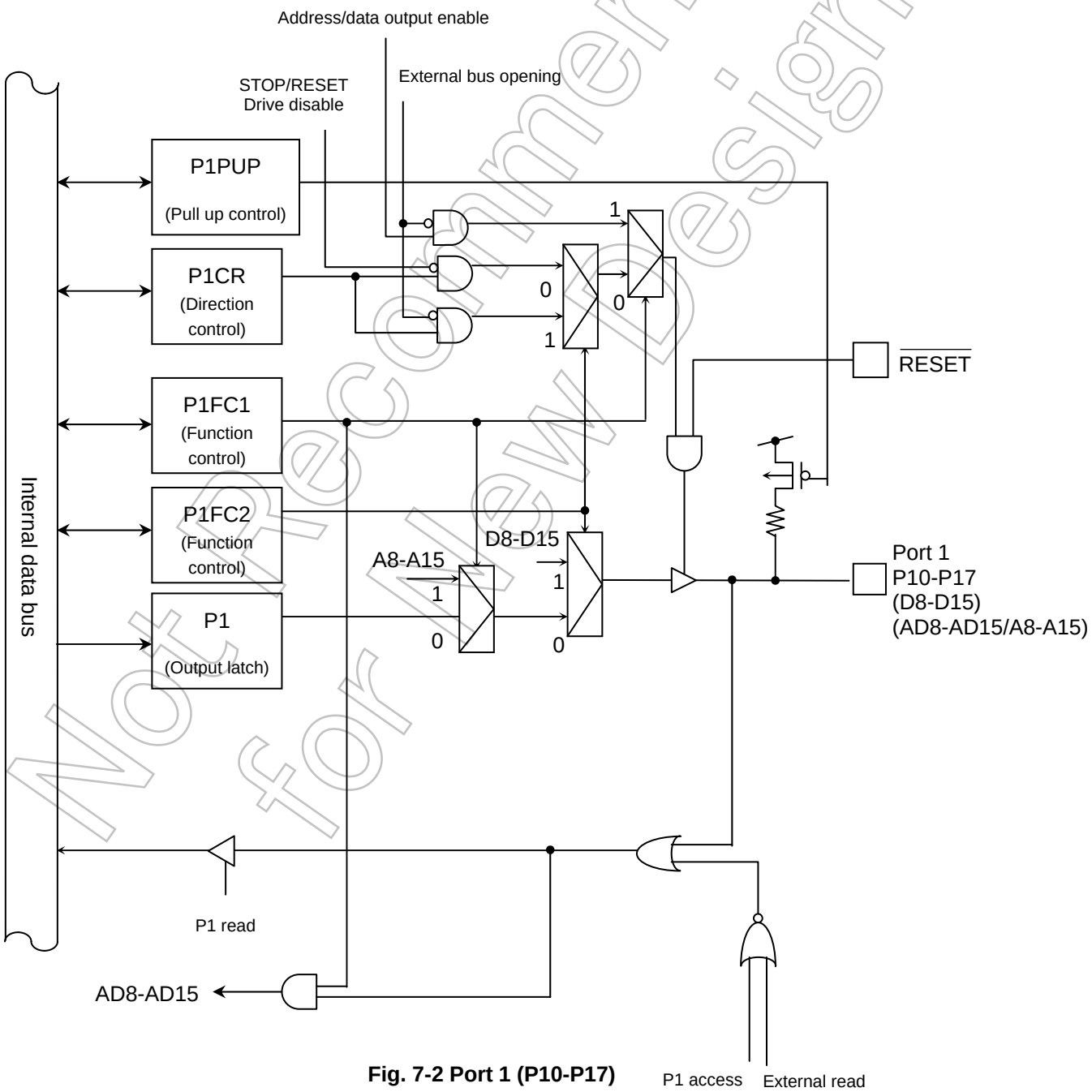


Fig. 7-2 Port 1 (P10-P17)

Port 1 register

P1 (0xFFFF_F001)

	7	6	5	4	3	2	1	0
Bit Symbol	P17	P16	P15	P14	P13	P12	P11	P10
Read/Write	R/W							
After reset	Input mode (output latch register is cleared to “0.”)							

Port 1 control register

P1CR (0xFFFF_F014)

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

Port 1 function register 1

P1FC1 (0xFFFF_F015)

	7	6	5	4	3	2	1	0
Bit Symbol	P17F1	P16F1	P15F1	P14F1	P13F1	P12F1	P11F1	P10F1
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0: PORT 1:External bus setting							

Port 1 function register 2

P1FC2 (0xFFFF_F016)

	7	6	5	4	3	2	1	0
Bit Symbol	P17F2	P16F2	P15F2	P14F2	P13F2	P12F2	P11F2	P10F2
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0: PORT 1:External bus setting							

Port 1 pull up control register

P1PUP (0xFFFF_F017)

	7	6	5	4	3	2	1	0
Bit Symbol	PE17	PE16	PE15	PE14	PE13	PE12	PE11	PE10
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up

7.4 Port 2 (P20 through P27)

The port 2 is a general-purpose, 8-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Outputs can be set by using the function register P2FC1, P2FC2 and the control register P2CR. A reset allows all bits of the output latch P2 to be set to "1," all bits of P2CR, P2FC1 and P2FC2 to be cleared to "0," and the port 2 to be put in output disable mode.

A reset allows P2CR, P2FC1 and P2FC2 to be cleared to "0" and the port 2 to function as an input port. Besides the general-purpose input/output port function, the port 2 performs another function: A0 through A7 function as one address bus and A16 through A23 function as the other address bus. To access external memory, registers P2CR, P2FC1 and P2FC2 must be provisioned to allow the port 2 to function as an address bus.

If the BUSMD pin (port P44) is set to "L" level during a reset, the port 2 is put in separate bus mode (A16 to A23). If it is set to "H" level during a reset, the port 2 is put in multiplexed mode (A0 through A7 or A16 through A23).

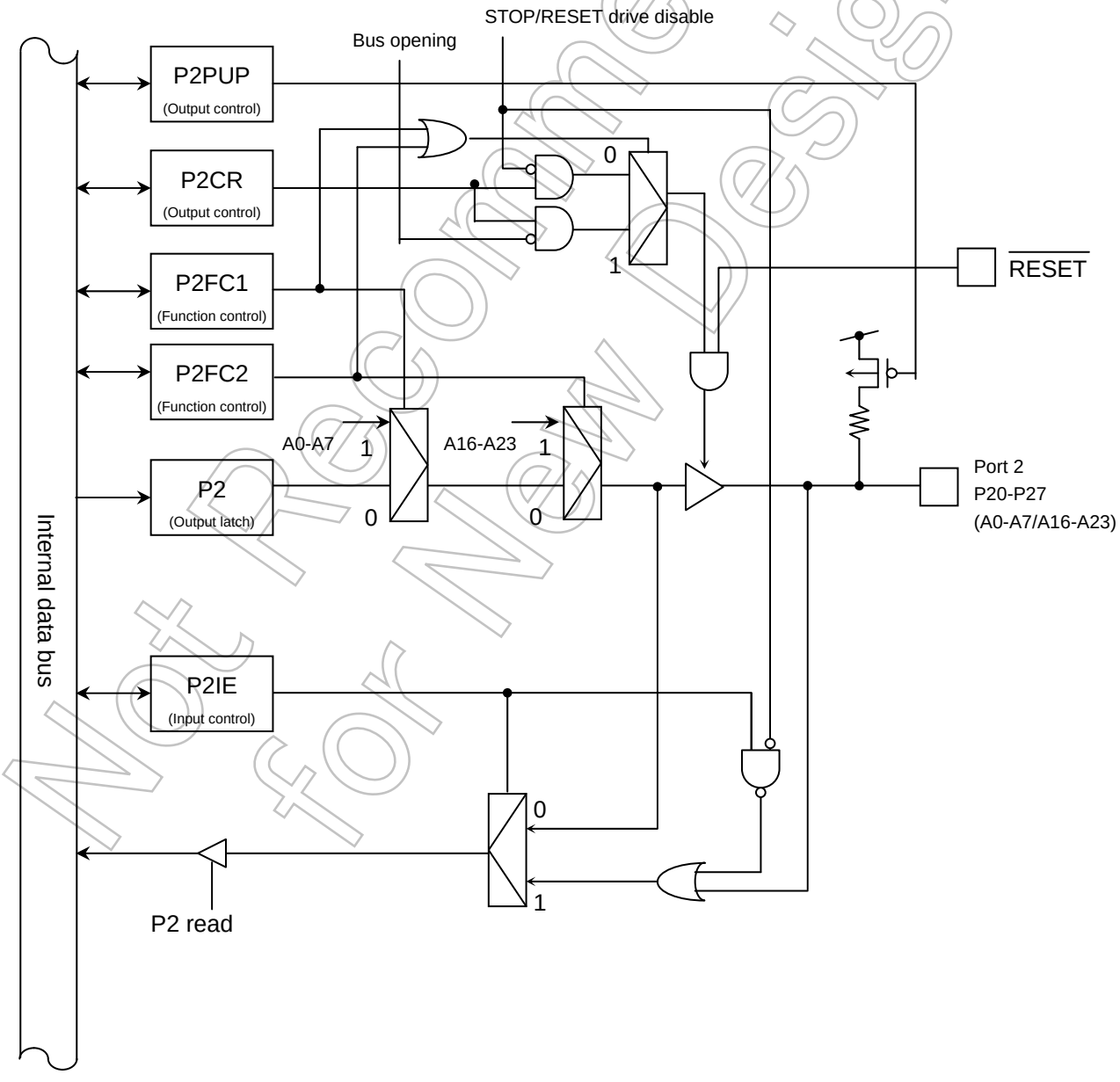


Fig. 7-3 Port 2(P20-P27)

Port 2 register

P2 (0xFFFF_F020)

	7	6	5	4	3	2	1	0
Bit Symbol	P27	P26	P25	P24	P23	P22	P21	P20
Read/Write	R/W							
After reset	Input mode (output latch register is set to "1.")							

Port 2 control register

P2CR (0xFFFF_F021)

	7	6	5	4	3	2	1	0
Bit Symbol	P27C	P26C	P25C	P24C	P23C	P22C	P21C	P20C
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0: Output disable 1: Output enable							

Port 2 function register 1

P2FC1 (0xFFFF_F022)

	7	6	5	4	3	2	1	0
Bit Symbol	P27F	P26F	P25F	P24F	P23F	P22F	P21F	P20F
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0: PORT 1:External bus setting							

Port 2 function register 2

P2FC2 (0xFFFF_F023)

	7	6	5	4	3	2	1	0
Bit Symbol	P27F2	P26F2	P25F2	P24F2	P23F2	P22F2	P21F2	P20F2
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0: PORT 1:External bus setting							

Port 2 pull up control register

P2PUP (0xFFFF_F02B)

	7	6	5	4	3	2	1	0
Bit Symbol	PE27	PE26	PE25	PE24	PE23	PE22	PE21	PE20
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up

Port 2 Input enable control register

P2IE (0xFFFF_F02E)

	7	6	5	4	3	2	1	0
Bit Symbol	PIE27	PIE26	PIE25	PIE24	PIE23	PIE22	PIE21	PIE20
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable

7.5 Port 3 (P30 through P37)

The port 3 is a general-purpose, 8-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Outputs can be set by using the control register P3CR and the function register P3FC1. A reset allows the output latches P30 and 31 to be set to "1."

In addition to above functions, a function of inputting and outputting the control and status signals of CPU is provided. If the P30 pin is set to $\overline{RD}$ signal output mode (<P30F>="1"), the $\overline{RD}$ strobe is output only when an external address area is accessed. Likewise, if the P31 pin is set to $\overline{WR}$ signal output mode (<P31F>="1"), the $\overline{WR}$ strobe is output only when an external address area is accessed.

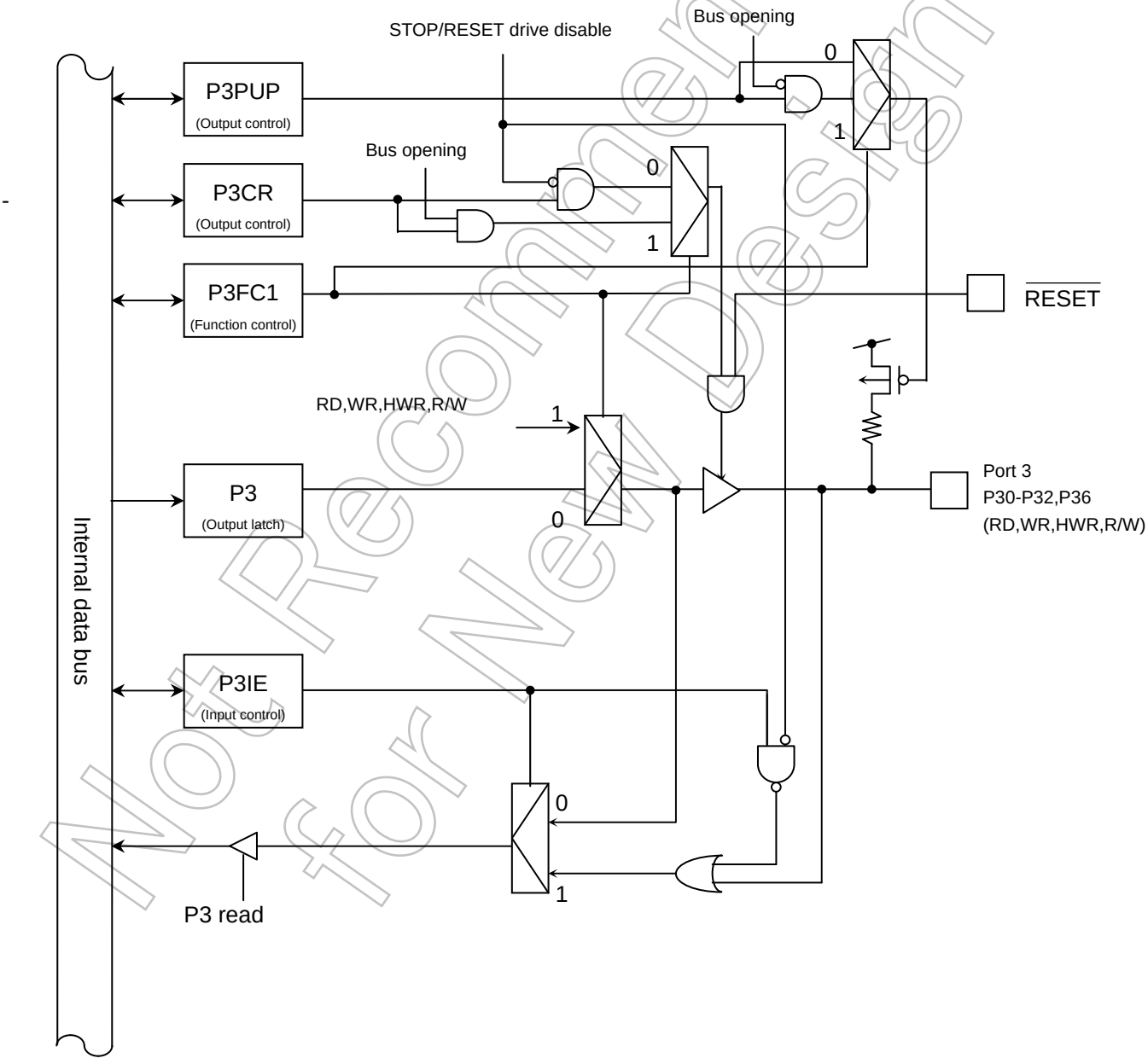


Fig. 7-4 Port 3 (P30-P32, P36)

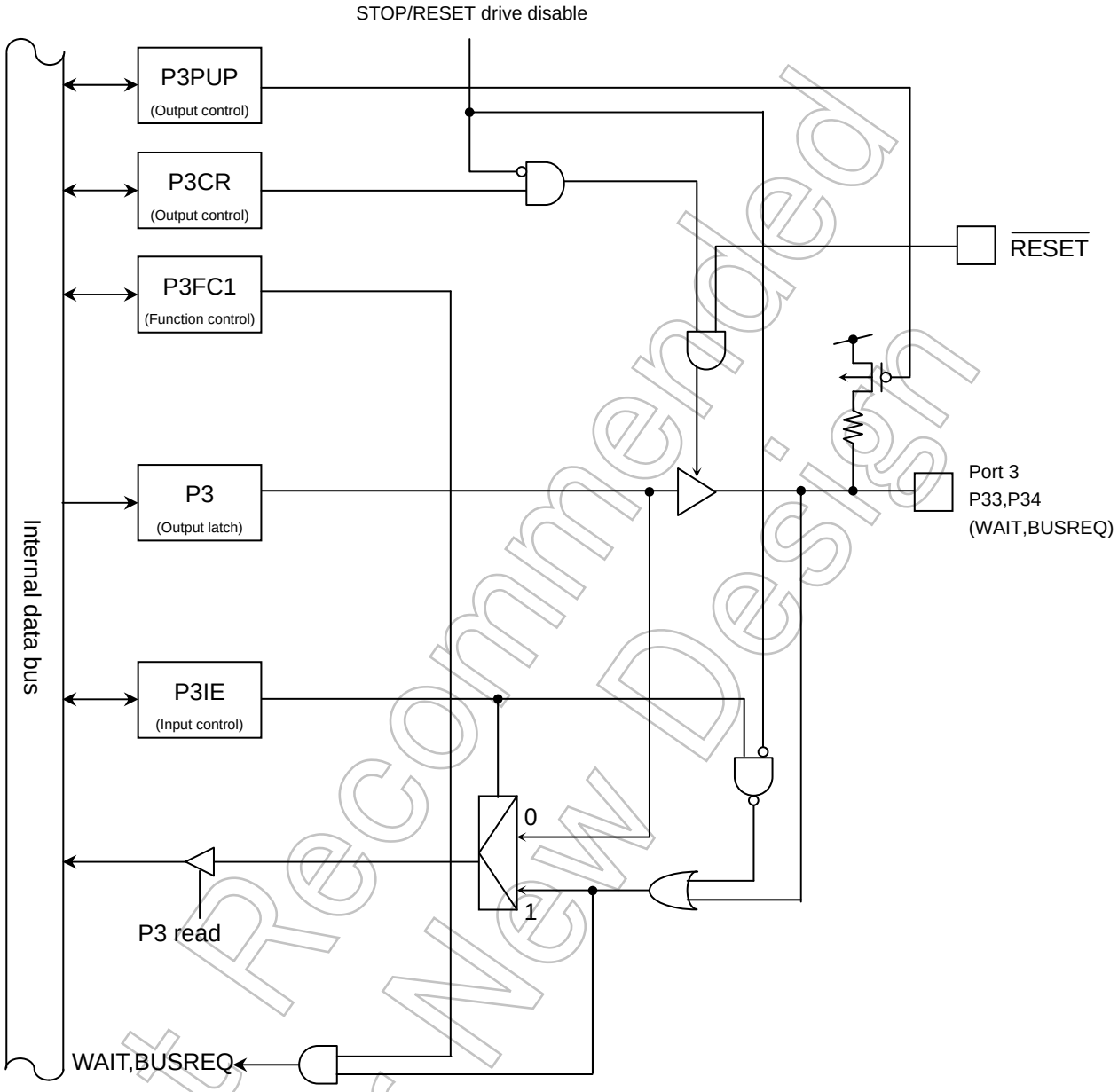


Fig. 7-5 Port 3 (P33,P34)

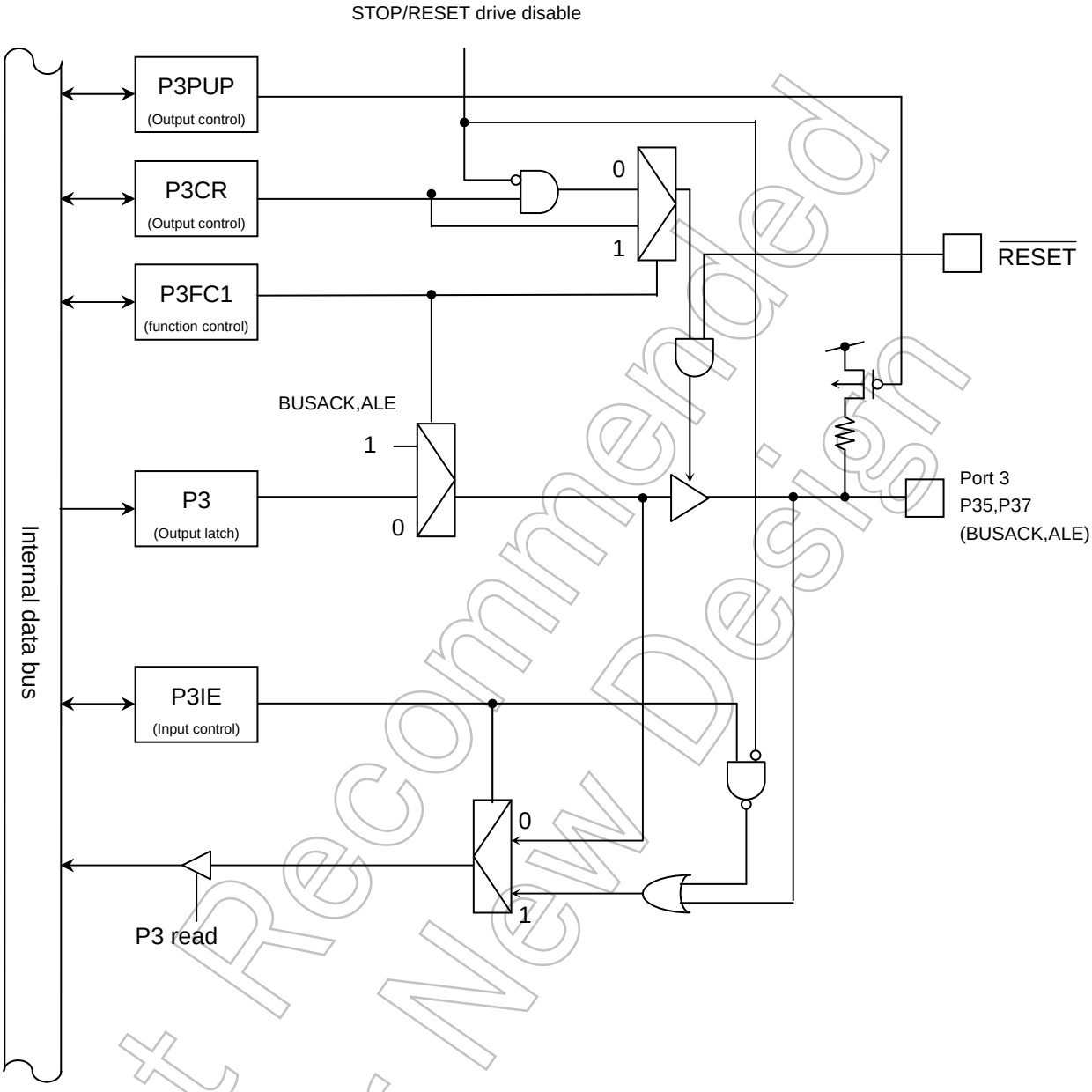


Fig. 7-6 Port 3 (P35,P37)

Port 3 register

P3 (0xFFFF_F030)

	7	6	5	4	3	2	1	0
Bit Symbol	P37	P36	P35	P34	P33	P32	P31	P30
Read/Write	R/W							
After reset	0	Input mode (output latch register is set to “1.”)						

Port 3 control register

P3CR (0xFFFF_F031)

	7	6	5	4	3	2	1	0
Bit Symbol	P37C	P36C	P35C	P34C	P33C	P32C	P31C	P30C
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0: Output disable 1: Output enable							

Port 3 function register 1

P3FC1 (0xFFFF_F032)

	7	6	5	4	3	2	1	0
Bit Symbol	P37F1	P36F1	P35F1	P34F1	P33F1	P32F1	P31F1	P30F1
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0:PORT 1:---	0:PORT 1R/W	0:PORT 1:BUSACK	0:PORT 1:BUSREQ	0:PORT /WAIT 1:RDY	0:PORT 1:HWR	0:PORT 1:WR	0:PORT 1:RD

Port 3 pull up control register

P3PUP (0xFFFF_F03B)

	7	6	5	4	3	2	1	0
Bit Symbol	PE37	PE36	PE35	PE34	PE33	PE32	PE31	PE30
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up

Port 3 Input enable control register

P3IE (0xFFFF_F03E)

	7	6	5	4	3	2	1	0
Bit Symbol	PIE37	PIE36	PIE35	PIE34	PIE33	PIE32	PIE31	PIE30
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable

7.6 Port 4 (P40 through P47)

The port 4 is a general-purpose, 8-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Outputs can be set by using the control register P4CR and the function register P4FC1.

Besides the general-purpose input/output port function, the port 4 performs other functions: P40 through P43 output the chip select signal ($\overline{CS0}$ to $\overline{CS3}$), P46 functions as the SCOUT output pin for outputting internal clocks, and P47 for external interrupt. By making necessary settings during a reset, P44 and P45 function as a BUSMD pin and ENDIAN setting pin respectively for setting external bus modes.

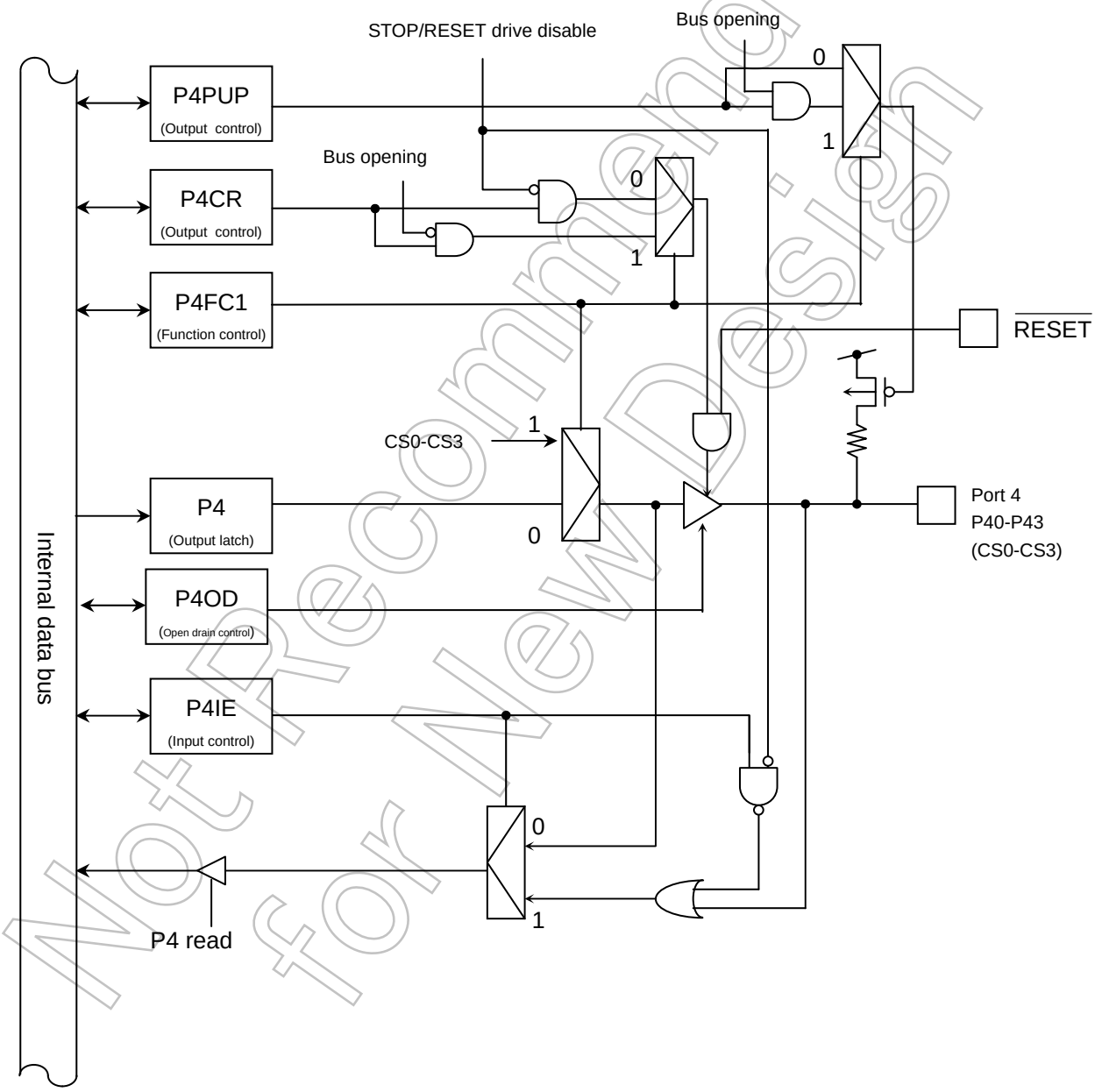


Fig. 7-7 Port 4 (P40-P43)

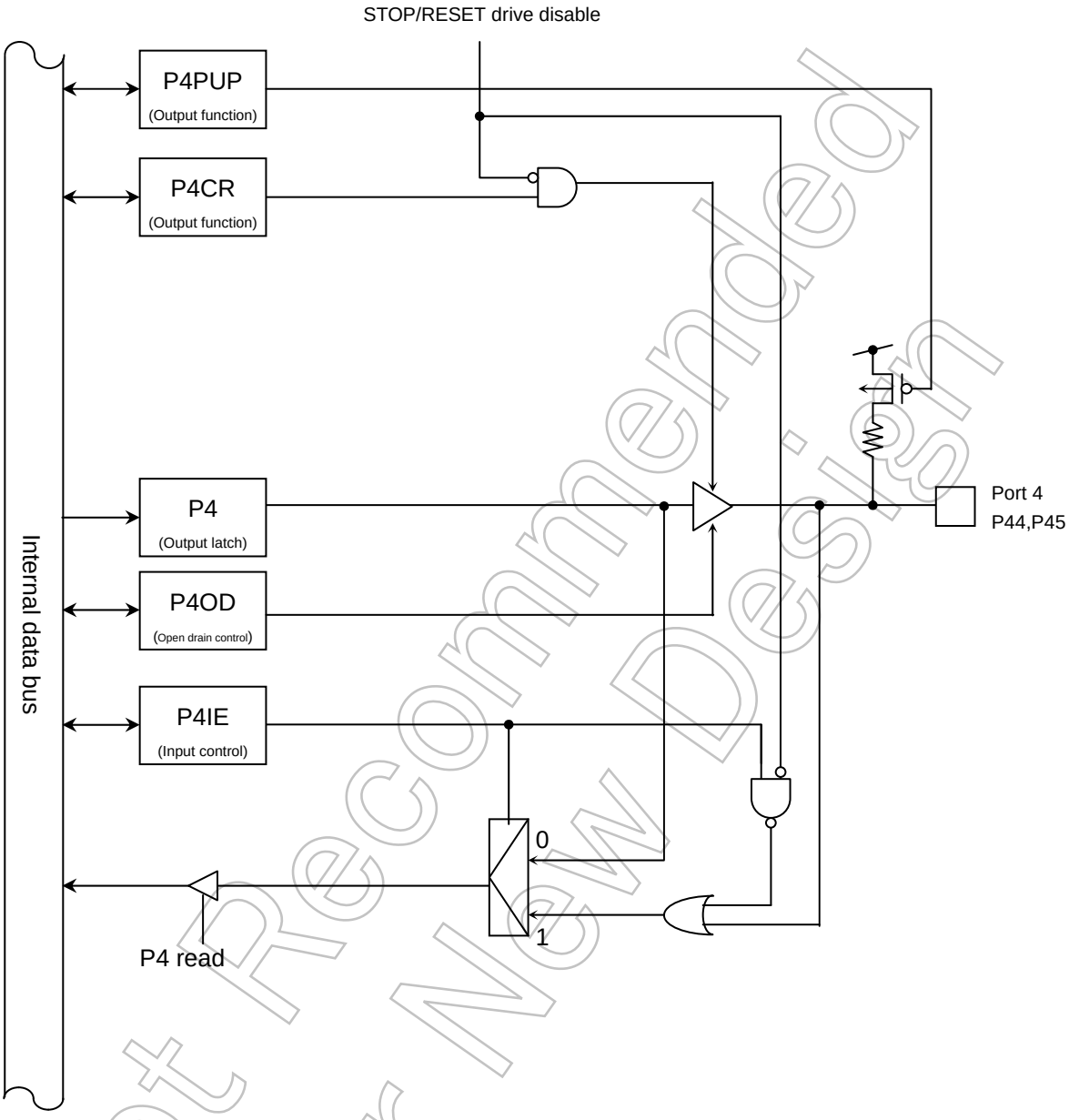


Fig. 7-8 Port 4 (P44, P45)

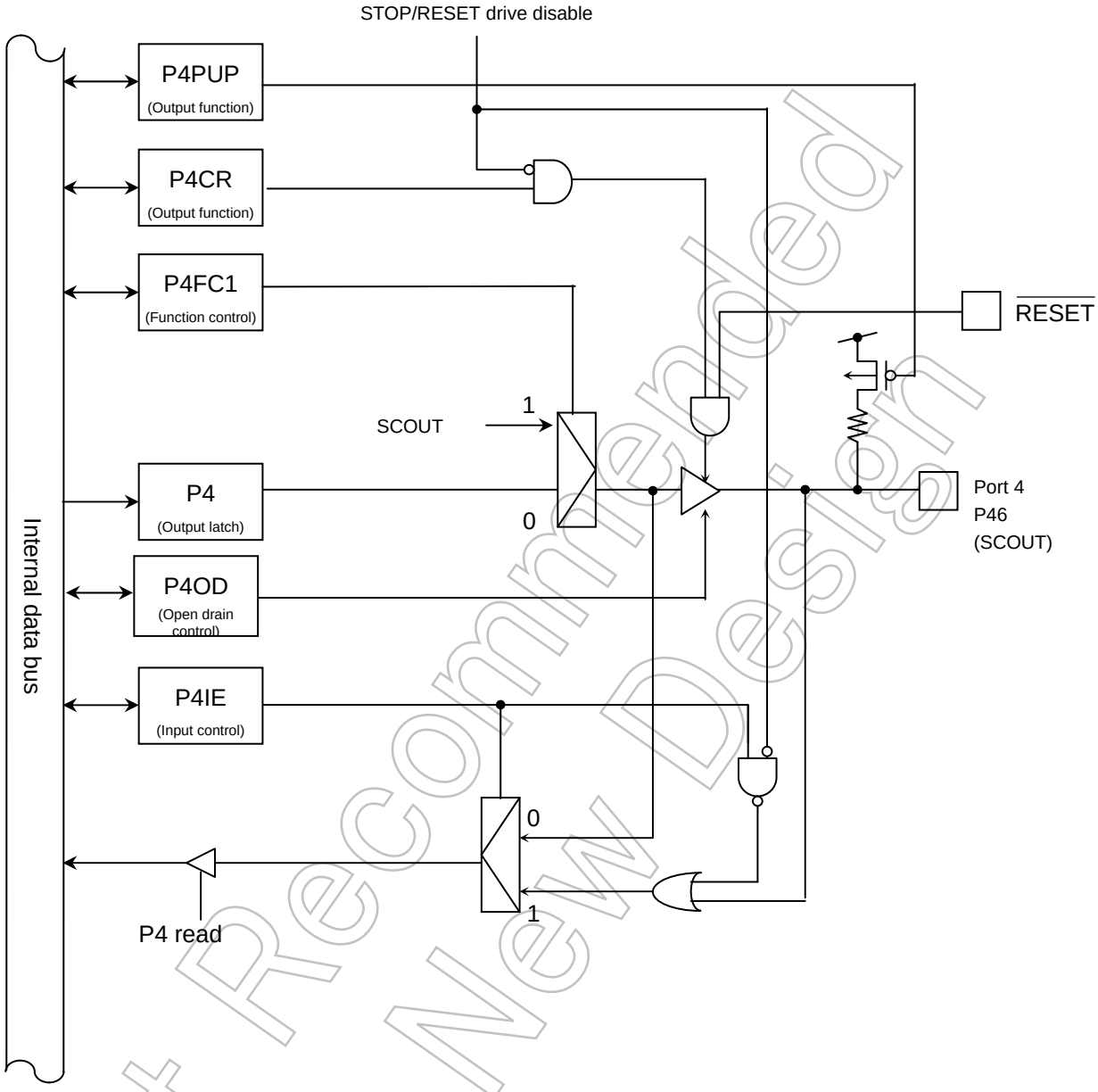


Fig. 7-9 Port 4 (P46)

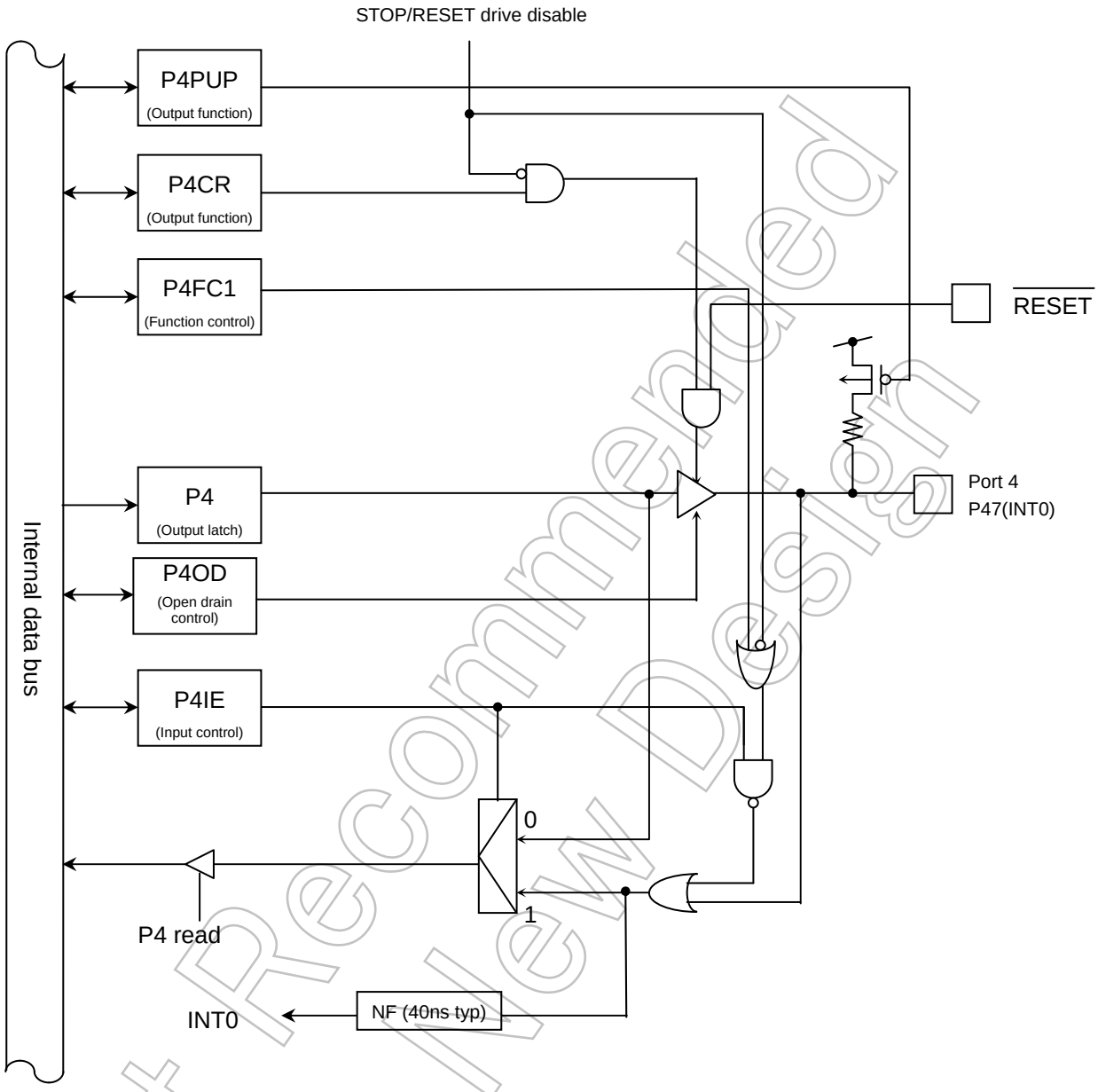


Fig. 7-10 Port 4 (P47)

Port 4 register

		7	6	5	4	3	2	1	0
P4	Bit Symbol	P47	P46	P45	P44	P43	P42	P41	P40
(0xFFFF_F040)	Read/Write	R/W							
	After reset	Input mode (output latch register is set to “1.”)							

Port 4 control register

		7	6	5	4	3	2	1	0
P4CR	Bit Symbol	P47C	P46C	P45C	P44C	P43C	P42C	P41C	P40C
(0xFFFF_F041)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
		0: Output disable 1: Output enable							

Port 4 function register 1

		7	6	5	4	3	2	1	0
P4FC1	Bit Symbol	P47F1	P46F1			P43F1	P42F1	P41F1	P40F1
(0xFFFF_F042)	Read/Write	R/W		R		R/W			
	After reset	0	0	0		0	0	0	0
	Function	0: PORT 1:INT0	0: PORT 1: SCOUT	“0” is read.		0: PORT 1: CS3	0: PORT 1: CS2	0: PORT 1: CS1	0: PORT 1: CS0

Port 4 Open drain control register

		7	6	5	4	3	2	1	0
P4OD	Bit Symbol	P47ODE	P46ODE	P45ODE	P44ODE	P43ODE	P42ODE	P41ODE	P40ODE
(0xFFFF_ F04A)	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	0:CMOS 1: Open drain	0:CMOS 1: Open drain	0:CMOS 1: Open drain	0:CMOS 1: Open drain	0:CMOS 1: Open drain	0:CMOS 1: Open drain	0:CMOS 1: Open drain	0:CMOS 1: Open drain

Port 4 pull up control register

		7	6	5	4	3	2	1	0
P4PUP	Bit Symbol	PE47	PE46	PE45	PE44	PE43	PE42	PE41	PE40
(0xFFFF_F04B)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up

Port 4 Input enable control register

		7	6	5	4	3	2	1	0
P4IE	Bit Symbol	PIE47	PIE46	PIE45	PIE44	PIE43	PIE42	PIE41	PIE40
(0xFFFF_F04E)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable

7.7 Port 5 (P50 through P57)

The port 5 is a general-purpose, 8-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Outputs can be set by using the function register P5FC1 ,P5FC2, P5FC3 and the control register P5CR. A reset allows all bits of the output latch P5 to be set to "1," all bits of P5CR, P5FC1, P5FC2 and P5FC3 to be cleared to "0," and the port 5 to be put in output disable mode.

Besides the input/output port function, the port 5 performs other functions: P50 through P53 output a 16-bit timer, P54 through P57 input a 16-bit timer, and P56 and P57 have DREQ and DACK function respectively. These functions are enabled by setting the corresponding bit of P5FC1, P5FC2 and P5FC3 to "1."

The port 5 also functions as an address bus (A0 through A7). To access external memory, P5CR, P5FC1,P5FC2 and P5FC3 must be provisioned to allow the port 5 to function as an address bus. This address bus function can be used only in separate bus mode. To put the port 5 in separate bus mode, the BUSMD pin (port 44) must be set to "L" level during a reset.

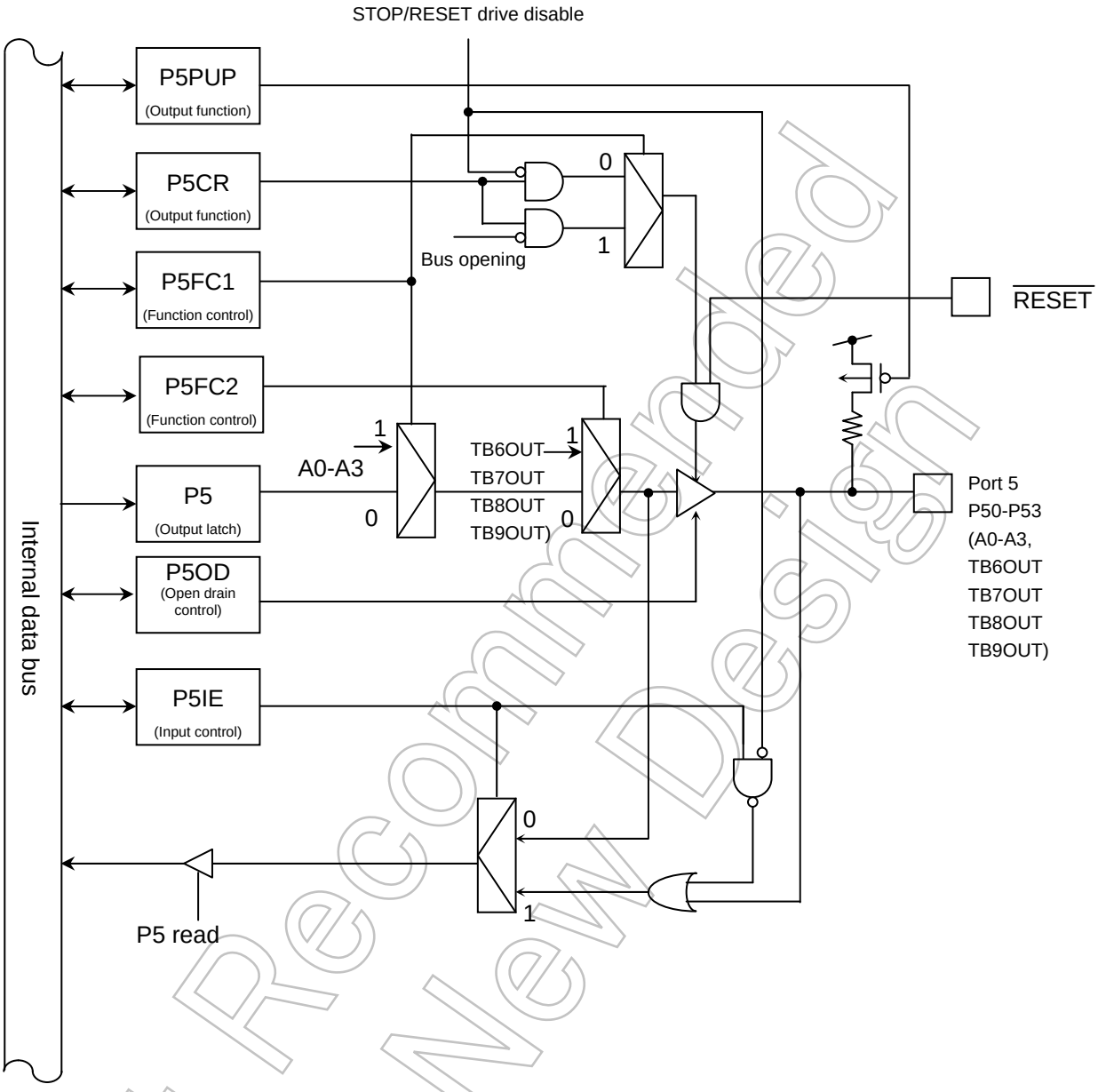


Fig. 7-11 Port 5(P50-P53)

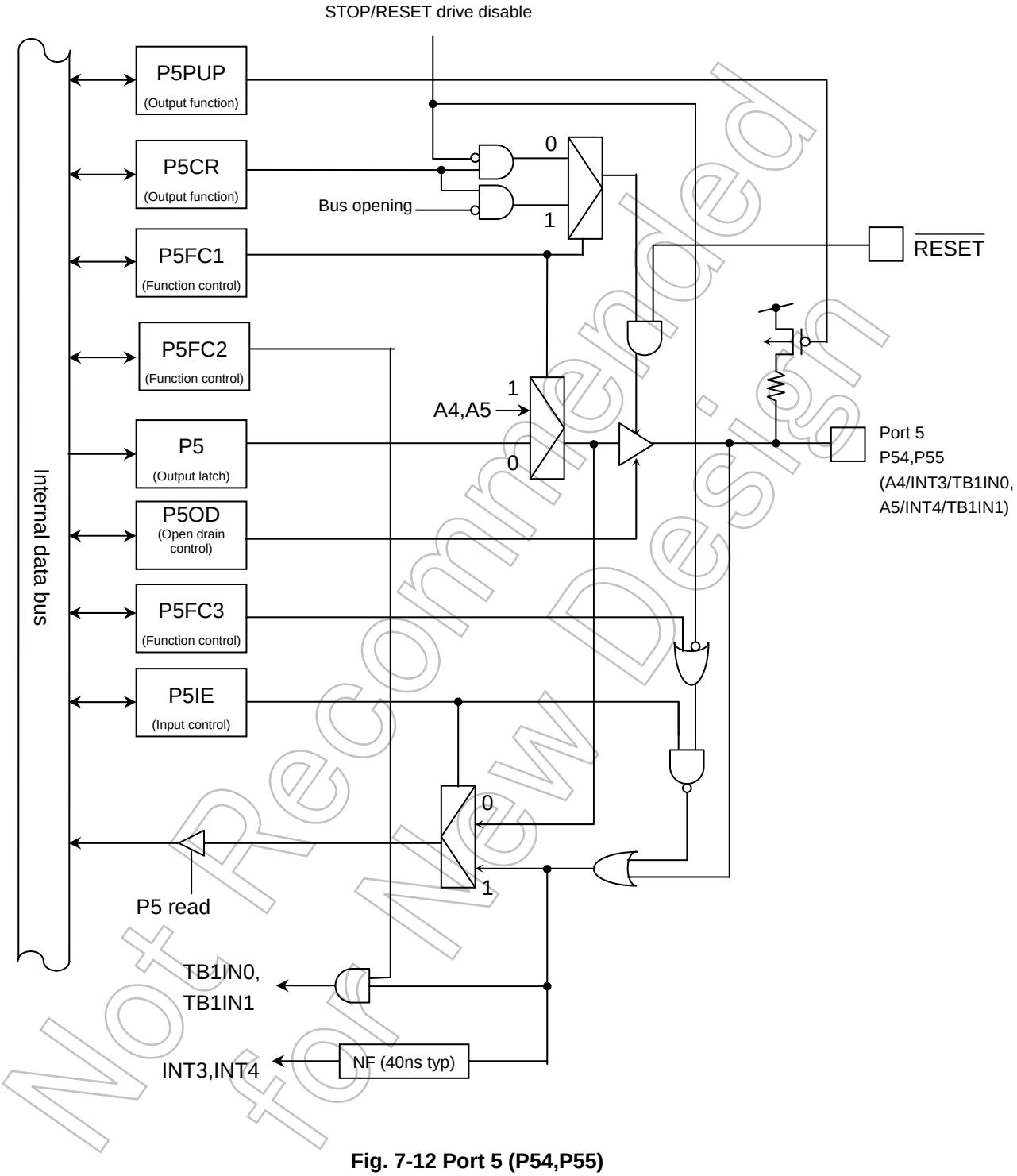


Fig. 7-12 Port 5 (P54,P55)

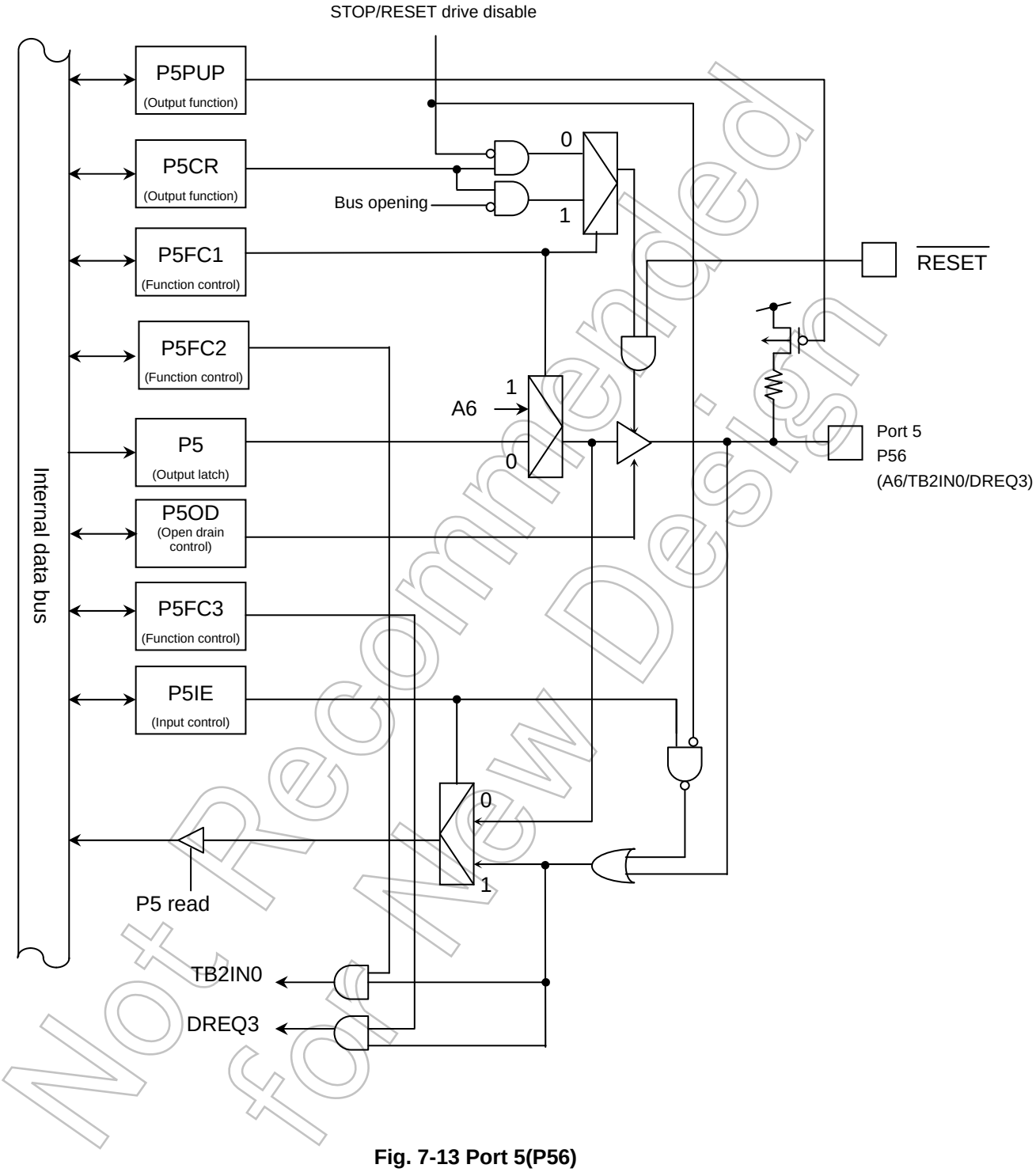


Fig. 7-13 Port 5(P56)

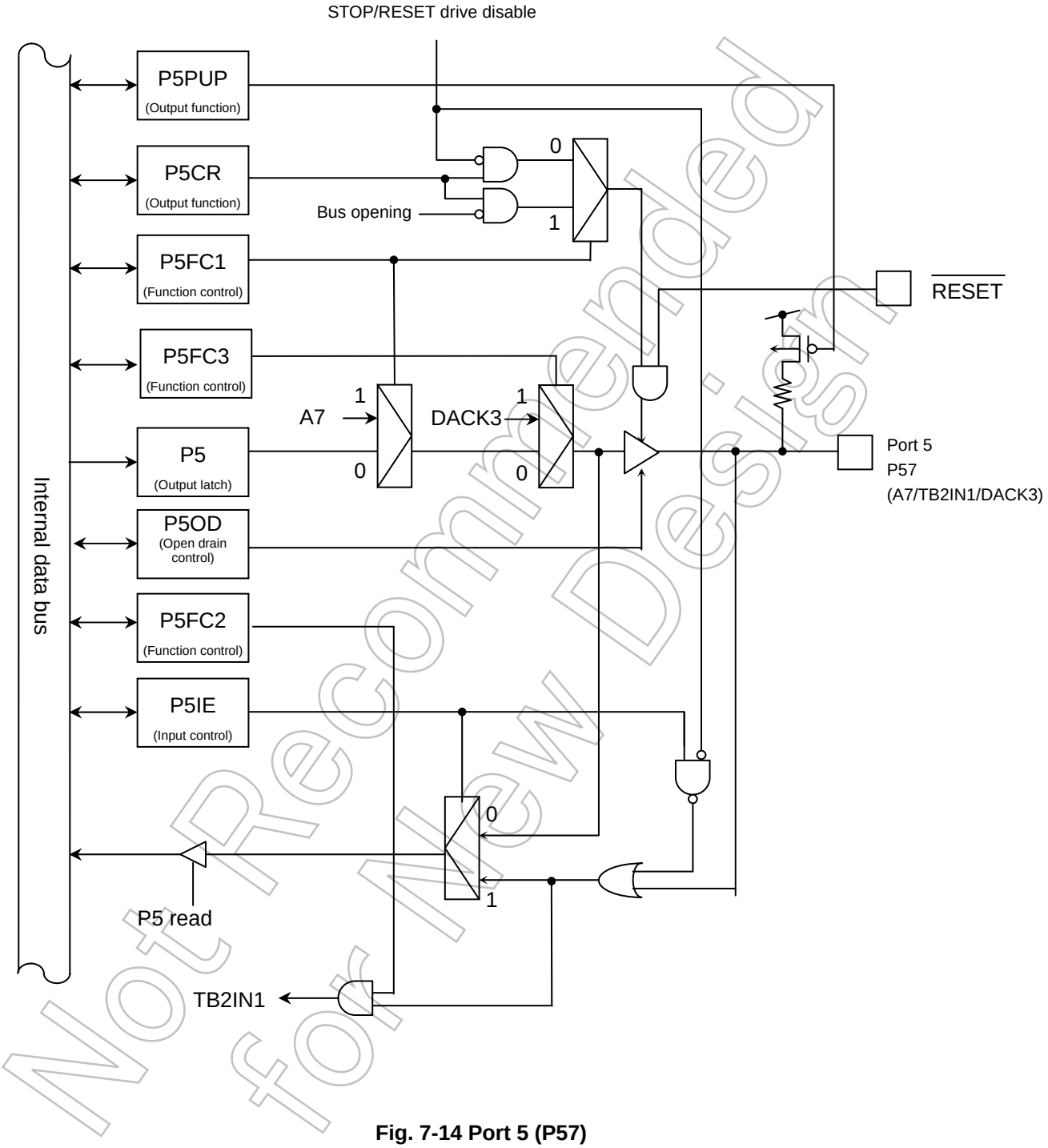


Fig. 7-14 Port 5 (P57)

Port 5 register

P5 (0xFFFF_F050)

	7	6	5	4	3	2	1	0
Bit Symbol	P57	P56	P55	P54	P53	P52	P51	P50
Read/Write	R/W							
After reset	Input mode (output latch register is set to “1.”)							

Port 5 control register

P5CR (0xFFFF_F051)

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

Port 5 function register 1

P5FC1 (0xFFFF_F052)

	7	6	5	4	3	2	1	0
Bit Symbol	P57F1	P56F1	P55F1	P54F1	P53F1	P52F1	P51F1	P50F1
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0: PORT 1:External bus setting							

Port 5 function register 2

P5FC2 (0xFFFF_F053)

	7	6	5	4	3	2	1	0
Bit Symbol	P57F2	P56F2	P55F2	P54F2	P53F2	P52F2	P51F2	P50F2
Read/Write	R/W				R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	0	0	0
Function	0:PORT 1:TB2IN1	0:PORT 1:TB2IN0	0:PORT 1:TB1IN1	0:PORT 1:TB1IN0	0:PORT 1:TB9OUT	0:PORT 1:TB8OUT	0:PORT 1:TB7OUT	0:PORT 1:TB6OUT

Port 5 function register 3

P5FC3 (0xFFFF_F054)

	7	6	5	4	3	2	1	0
Bit Symbol	P57F3	P56F3	P55F3	P54F3				
Read/Write	R/W				R			
After reset	0	0	0	0	0			
Function	0:PORT 1:DACK	0:PORT 1:DREQ	0:PORT 1:INT4	0:PORT 1:INT3	“0” is read.			

Port 5 Open drain control register

P5OD (0xFFFF_ F05A)

	7	6	5	4	3	2	1	0
Bit Symbol					P53ODE	P52ODE	P51ODE	P50ODE
Read/Write	R				R/W	R/W	R/W	R/W
After reset	0				0	0	0	0
Function	“0” is read.				0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain

Port 5 pull up control register

		7	6	5	4	3	2	1	0
P5PUP	Bit Symbol	PE57	PE56	PE55	PE54	PE53	PE52	PE51	PE50
(0xFFFF_F05B)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	0: Off 1:Pull up	0: Off 1:Pull up	0: Off 1:Pull up	0: Off 1:Pull up	0: Off 1:Pull up	0: Off 1:Pull up	0: Off 1:Pull up	0: Off 1:Pull up

Port 5 Input enable control register

		7	6	5	4	3	2	1	0
P5IE	Bit Symbol	PIE57	PIE56	PIE55	PIE54	PIE53	PIE52	PIE51	PIE50
(0xFFFF_F05E)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable

7.8 Port 6 (P60 through P67)

The port 6 is a general-purpose, 8-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Outputs can be set by using the function register P6FC1 through P6FC4 and the control register P6CR. A reset allows all bits of the output latch P6 to be set to "1," all bits of P6CR and P6FC1 through P6FC4 to be cleared to "0," and the port 6 to be put in output disable mode. Besides the input/output port function, the port 6 performs other functions: P60 through P67 input 16 bit timer, P60 and P61 output 16 bit timer, P62 and P63 input external interrupts, and P64 through P67 output high accuracy PPG.

The port 6 also functions as an address bus (A8 through A15). To access external memory, P6CR and P6FC1 through P6FC4 must be provisioned to allow the port 6 to function as an address bus. The address bus function can be used only in separate bus mode. To put the port 6 in separate bus mode, the BUSMD pin (port 44) must be set to "L" level during a reset.

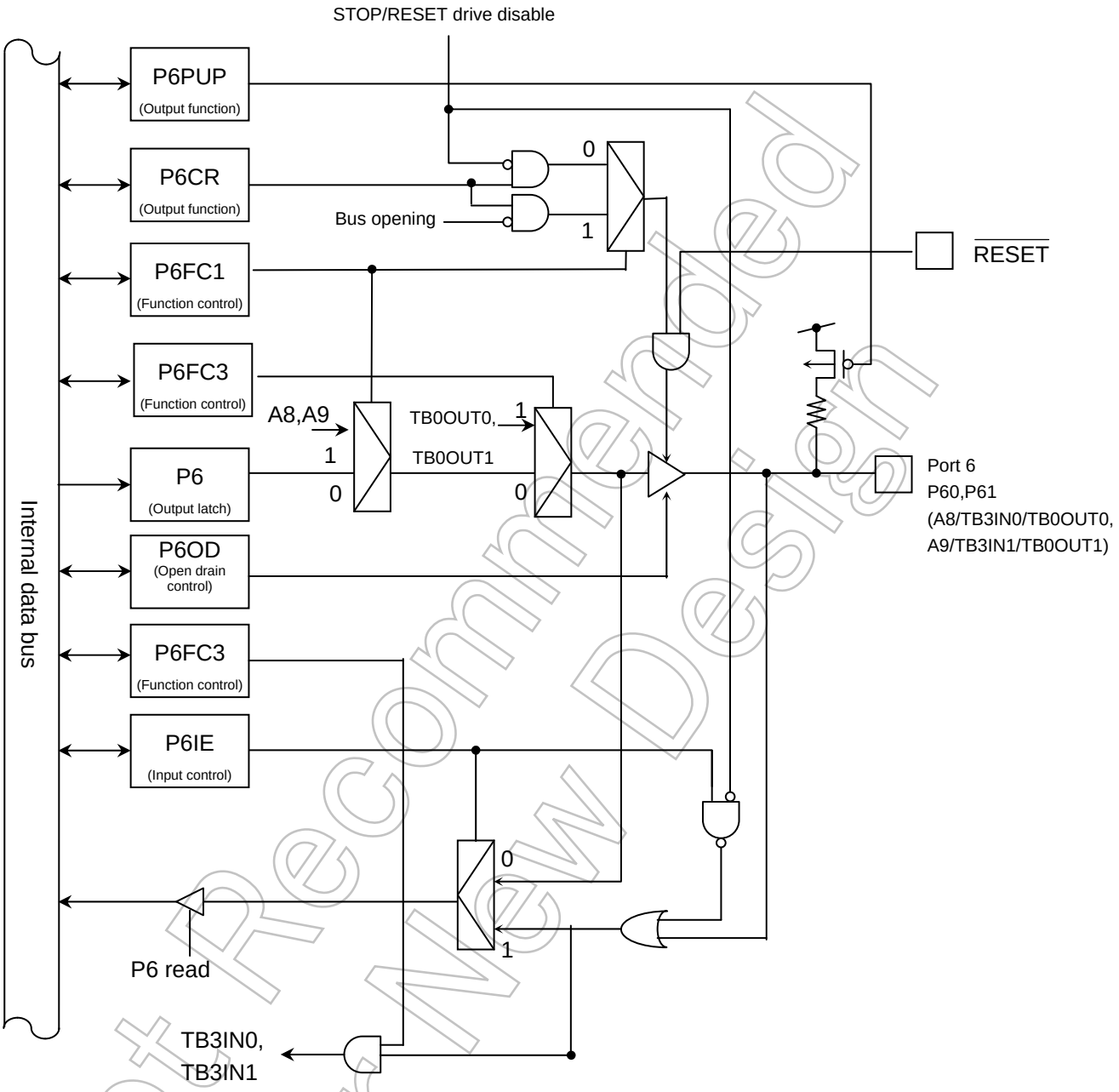


Fig. 7-15 Port 6 (P60,P61)

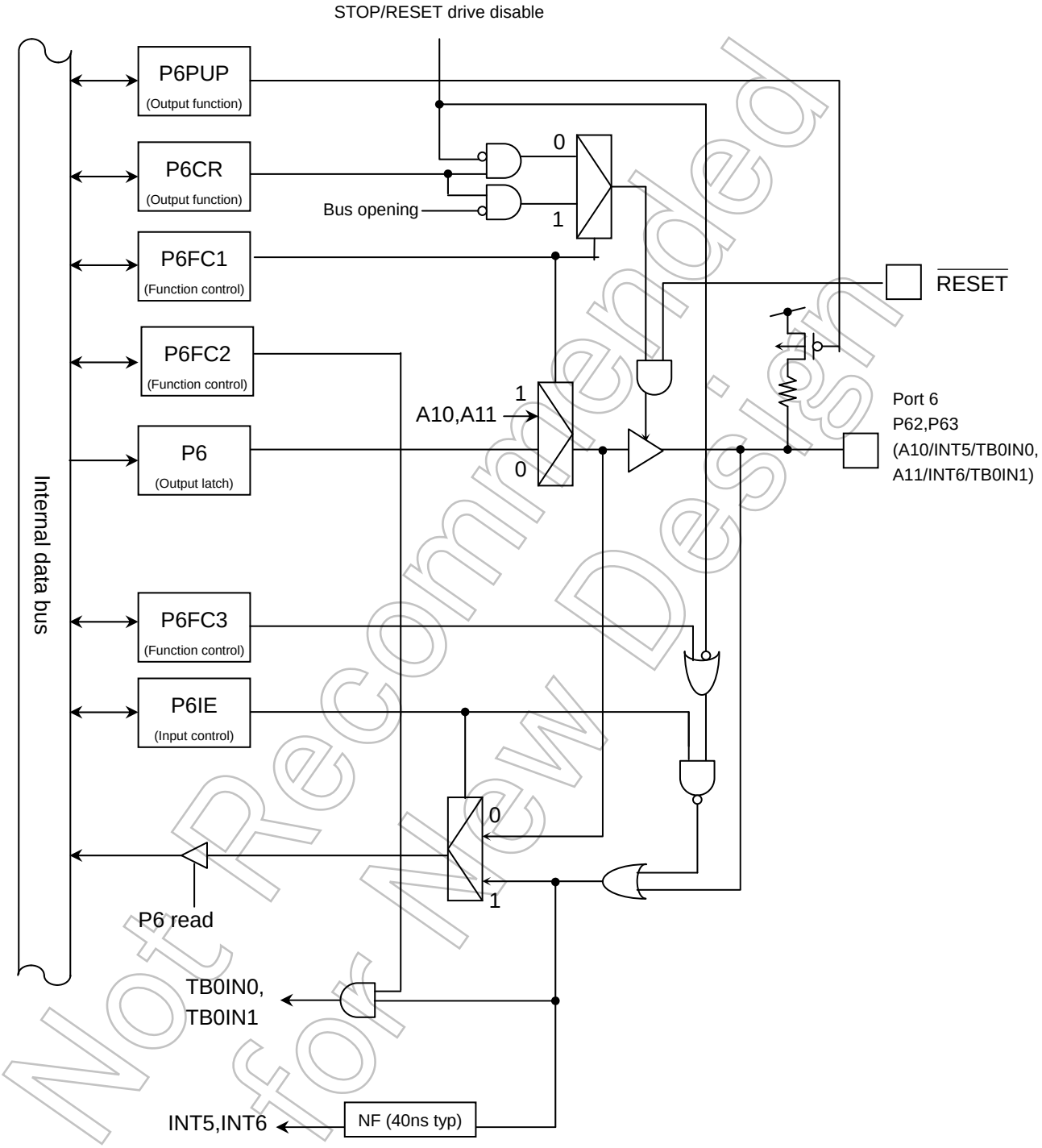


Fig. 7-16 Port 6 (P62, P63)

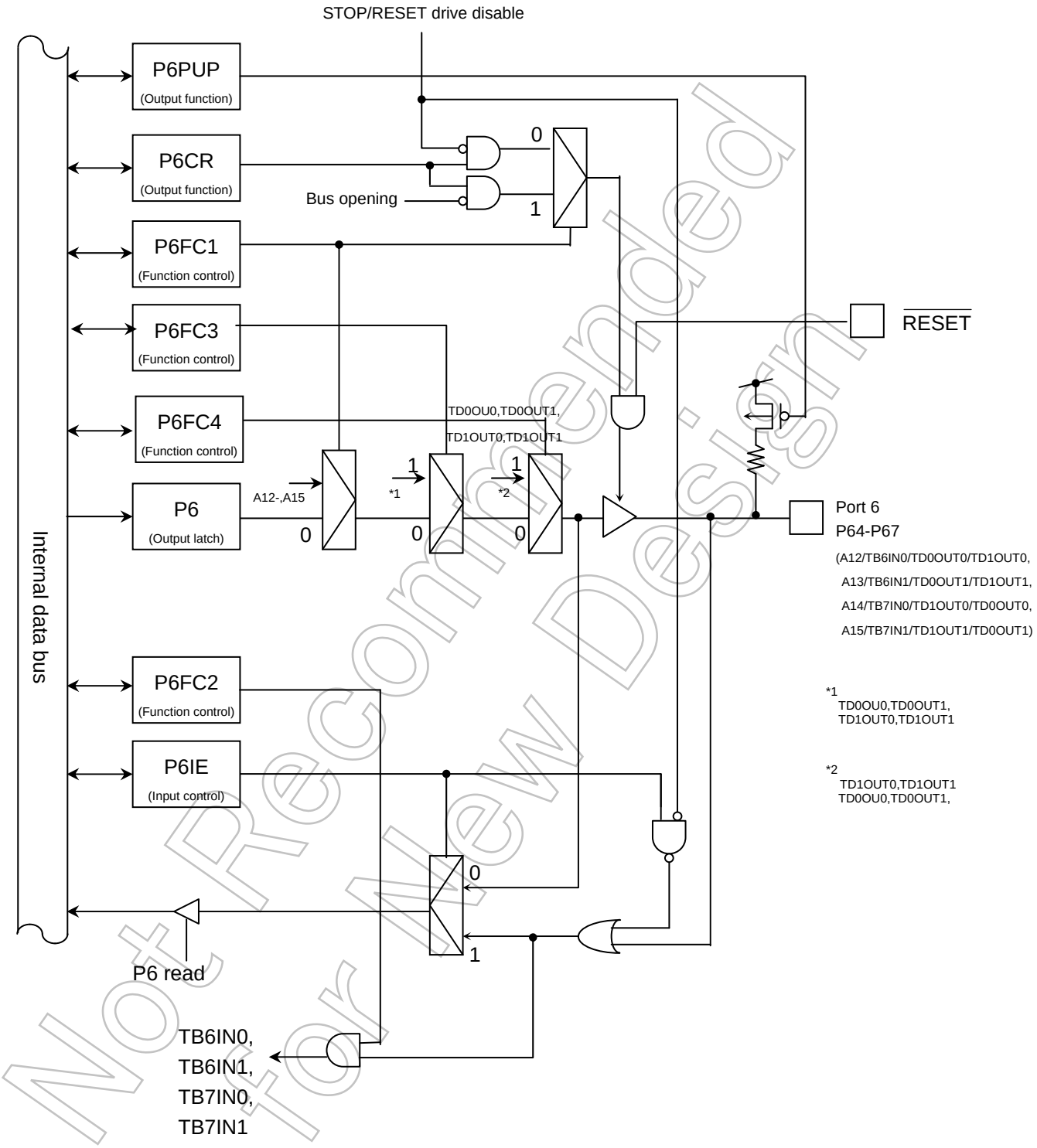


Fig. 7-17 Port 6 (P64-P67)

Port 6 register

P6 (0xFFFF_F060)

	7	6	5	4	3	2	1	0
Bit Symbol	P67	P66	P65	P64	P63	P62	P61	P60
Read/Write	R/W							
After reset	Input mode (output latch register is set to “1.”)							

Port 6 control register

P6CR (0xFFFF_F061)

	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: Output disable 1: Output enable							

Port 6 function register 1

P6FC1 (0xFFFF_F062)

	7	6	5	4	3	2	1	0
Bit Symbol	P67F1	P66F1	P65F1	P64F1	P63F1	P62F1	P61F1	P60F1
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0: PORT 1:External bus setting							

Port 6 function register 2

P6FC2 (0xFFFF_F063)

	7	6	5	4	3	2	1	0
Bit Symbol	P67F2	P66F2	P65F2	P64F2	P63F2	P62F2	P61F2	P60F2
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0:PORT 1:TB7IN1	0:PORT 1:TB7IN0	0:PORT 1:TB6IN1	0:PORT 1:TB6IN0	0:PORT 1:TB0IN1	0:PORT 1:TB0IN0	0:PORT 1:TB3IN1	0:PORT 1:TB3IN0

Port 6 function register 3

P6FC3 (0xFFFF_F064)

	7	6	5	4	3	2	1	0
Bit Symbol	P67F3	P66F3	P65F3	P64F3	P63F3	P62F3	P61F3	P60F3
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0:PORT 1:TD1OUT1	0:PORT 1:TD1OUT0	0:PORT 1:TD0OUT1	0:PORT 1:TD0OUT0	0:PORT 1:INT6	0:PORT 1:INT5	0:PORT 1:TB0OUT1	0:PORT 1:TB0OUT0

Port 6 function register 4

P6FC4 (0xFFFF_F065)

	7	6	5	4	3	2	1	0
Bit Symbol	P67F4	P66F4	P65F4	P64F4	P63F4	P62F4	P61F4	P60F4
Read/Write	R/W				R			
After reset	0	0	0	0	0	0	0	0
Function	0:PORT 1:TD0OUT1	0:PORT 1:TD0OUT0	0:PORT 1:TD1OUT1	0:PORT 1:TD1OUT0	“0” is read.			

Port 6 Open drain control register

P6OD (0xFFFF_ F06A)

	7	6	5	4	3	2	1	0
Bit Symbol							P61ODE	P60ODE
Read/Write	R						R/W	R/W
After reset	0						0	0
Function	"0" is read.						0:CMOS 1: Open drain	0:CMOS 1: Open drain

Port 6 pull up control register

P6PUP (0xFFFF_F06B)

	7	6	5	4	3	2	1	0
Bit Symbol	PE67	PE66	PE65	PE64	PE63	PE62	PE61	PE60
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up

Port 6 Input enable control register

P6IE (0xFFFF_F06E)

	7	6	5	4	3	2	1	0
Bit Symbol	PIE67	PIE66	PIE65	PIE64	PIE63	PIE62	PIE61	PIE60
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable

7.9 Port 7 (P70 through P77)

The port 7 is an 8-bit, analog input port for the A/D converter. Although the port 7 is an input port during a reset, any inputs are disabled.

Set the corresponding input enable control register when you use the port 7 as an input port.

The setting is not required when you use it as an AD function port.

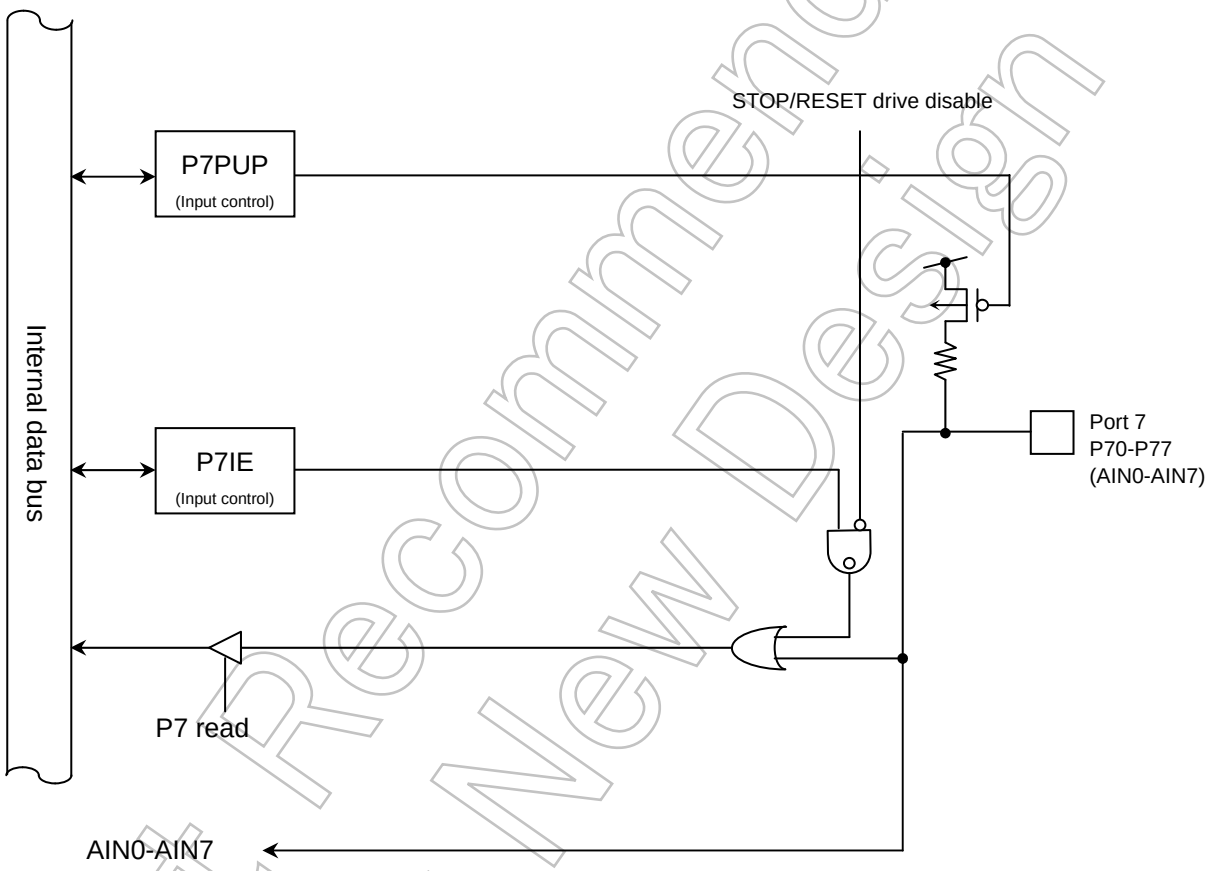


Fig. 7-18 Port 7(P70-P77)

Port 7 register

		7	6	5	4	3	2	1	0
P7	Bit Symbol	P77	P76	P75	P74	P73	P72	P71	P70
(0xFFFF_F070)	Read/Write	R							
	After reset	Input mode							

Port 7 pull up control register

		7	6	5	4	3	2	1	0
P7PUP	Bit Symbol	PE77	PE76	PE75	PE74	PE73	PE72	PE71	PE70
(0xFFFF_F07B)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up

Port 7 Input enable control register

		7	6	5	4	3	2	1	0
P7IE	Bit Symbol	PIE77	PIE76	PIE75	PIE74	PIE73	PIE72	PIE71	PIE70
(0xFFFF_F07E)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable

7.10 Port 8 (P80 through P84)

The port 8 is a 5-bit, analog input port for the A/D converter. Besides this analog input port function, P80 through P83 input the key-on wake-up, and P84 through P87 input external interrupts.

Although the port 8 is an input port during a reset, any inputs are disabled. Set the corresponding input enable control register when you use the port 8 as an input port.

The setting is not required when you use it as an AD function port.

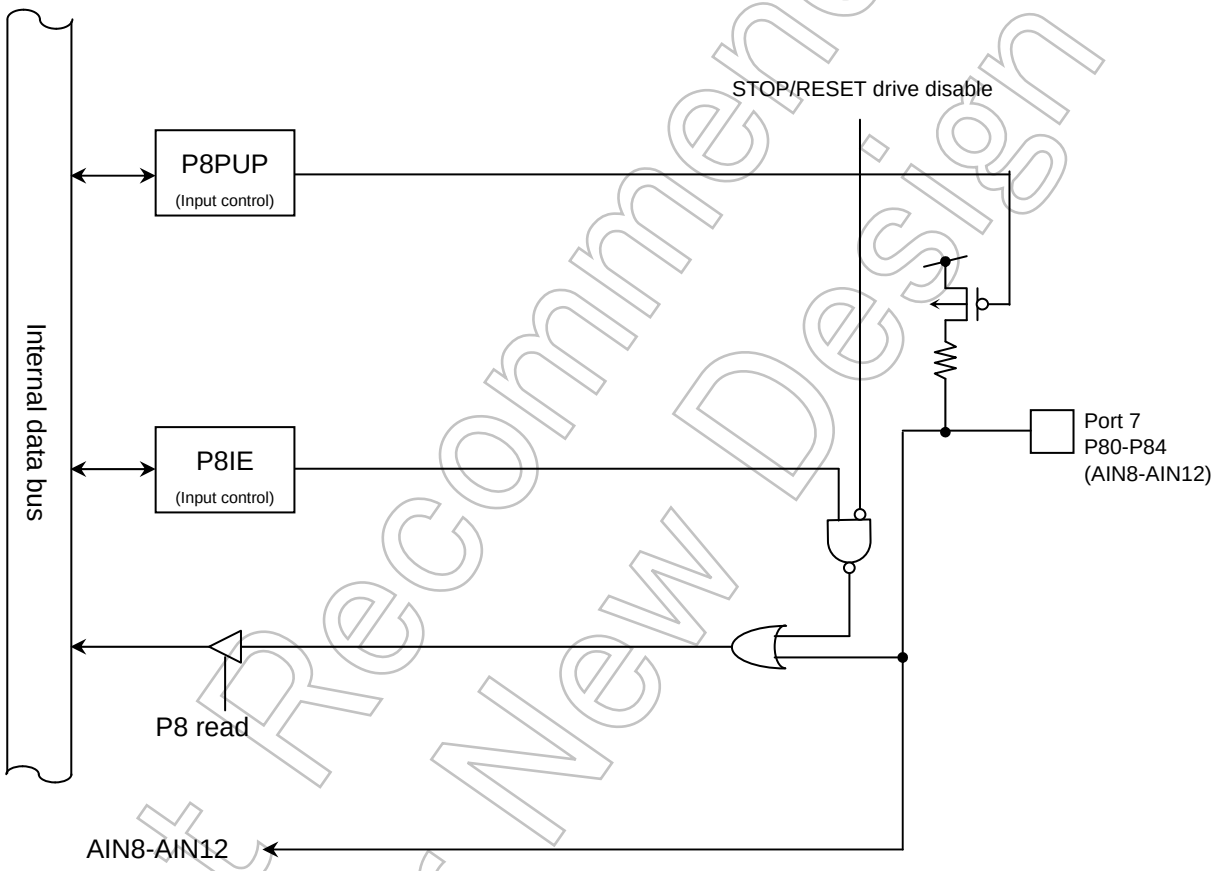


Fig. 7-19 Port 8 (P80-P84)

Port 8 register

		7	6	5	4	3	2	1	0
P8	Bit Symbol				P84	P83	P82	P81	P80
(0xFFFF_F080)	Read/Write	R							
	After reset	Input mode							

Port 8 pull up control register

		7	6	5	4	3	2	1	0
P8PUP	Bit Symbol				PE84	PE83	PE82	PE81	PE80
(0xFFFF_F08B)	Read/Write	R			R/W				
	After reset	0			0	0	0	0	0
	Function	"0" is read.			Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up

Port 8 Input enable control register

		7	6	5	4	3	2	1	0
P8IE	Bit Symbol				PIE84	PIE83	PIE82	PIE81	PIE80
(0xFFFF_F08E)	Read/Write	R			R/W				
	After reset	0			0	0	0	0	0
	Function	"0" is read.			Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable

7.11 Port 9 (P90 through P97)

The port 9 is a general-purpose, 8-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Outputs can be set by using the function register P9FC1 thorough P9FC3 and the control register P9CR. A reset allows all bits of the output latch P9 to be set to "1," all bits of P9CR and P9FC1 through P9FC3 to be cleared to "0," and the port 9 to be put in output disable mode.

Besides the input/output port function, the port 9 performs other functions: P90 and P94 outputs SI0 data, P91 and P95 inputs SI0 data, P92 and P96 inputs and outputs SI0 CLK or inputs CTS, and P91, P92, P93, P96 and P97 input external interrupts.

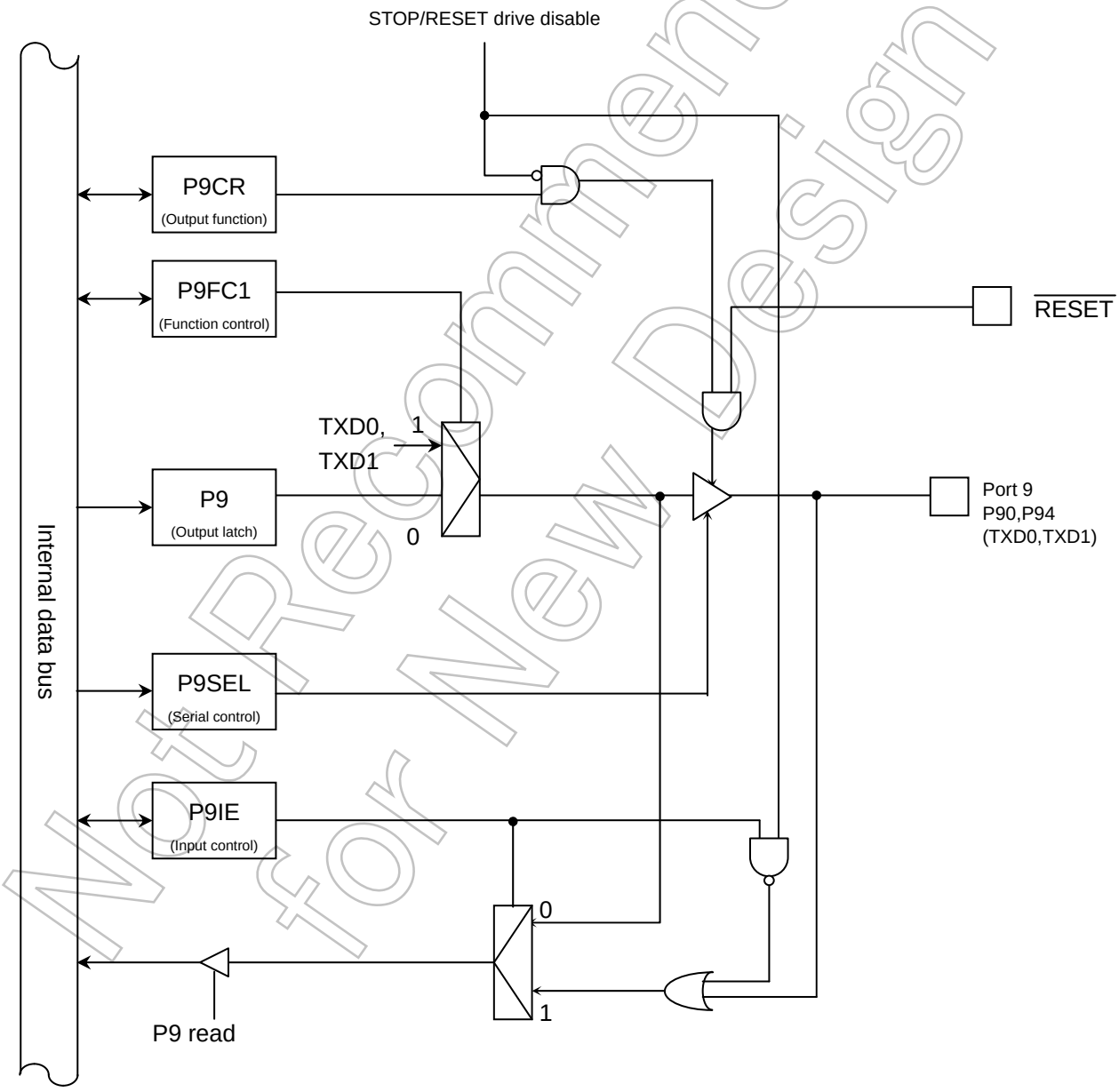


Fig. 7-20 Port F (P90,P94)

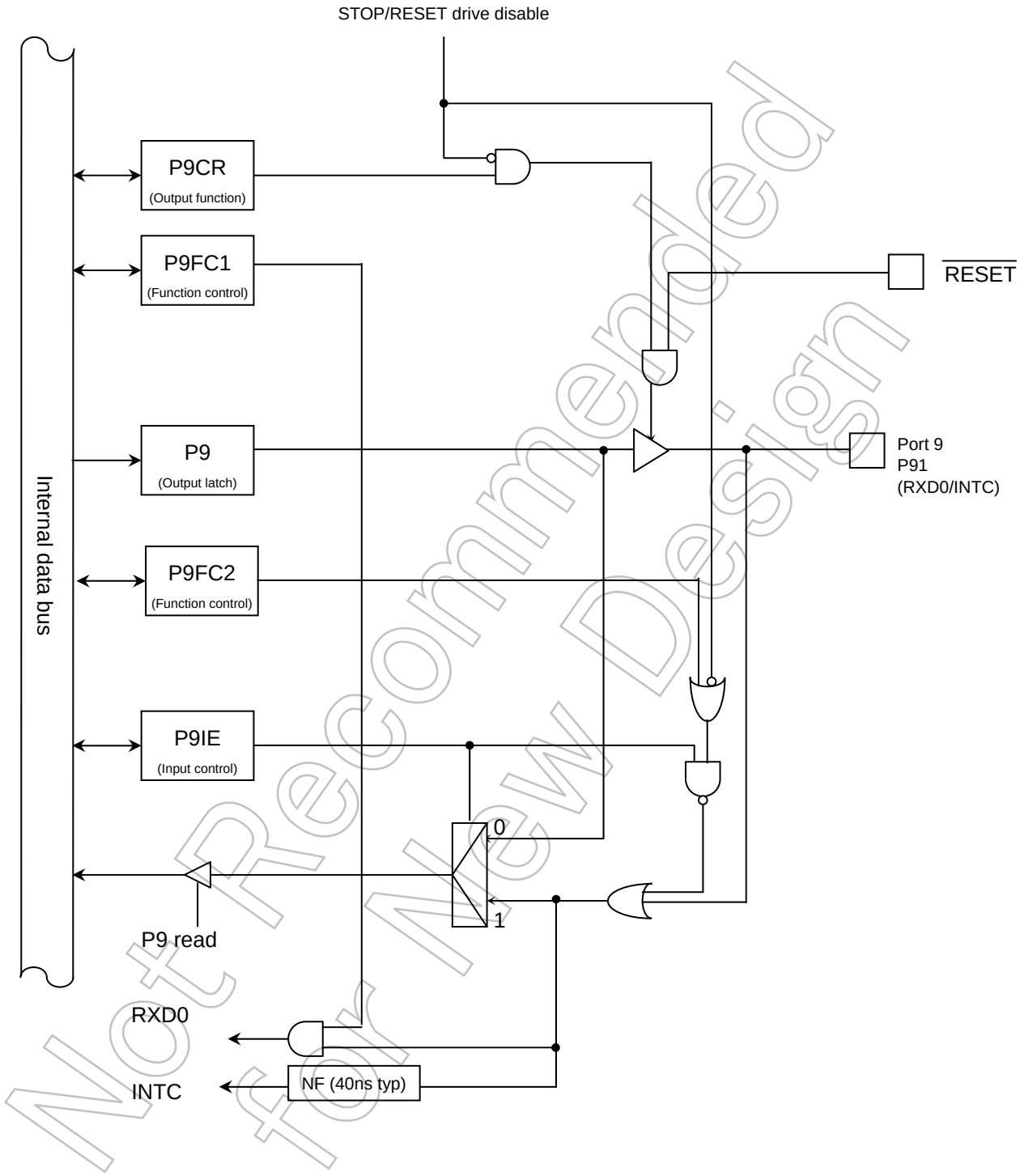


Fig. 7-21 Port 9 (P91)

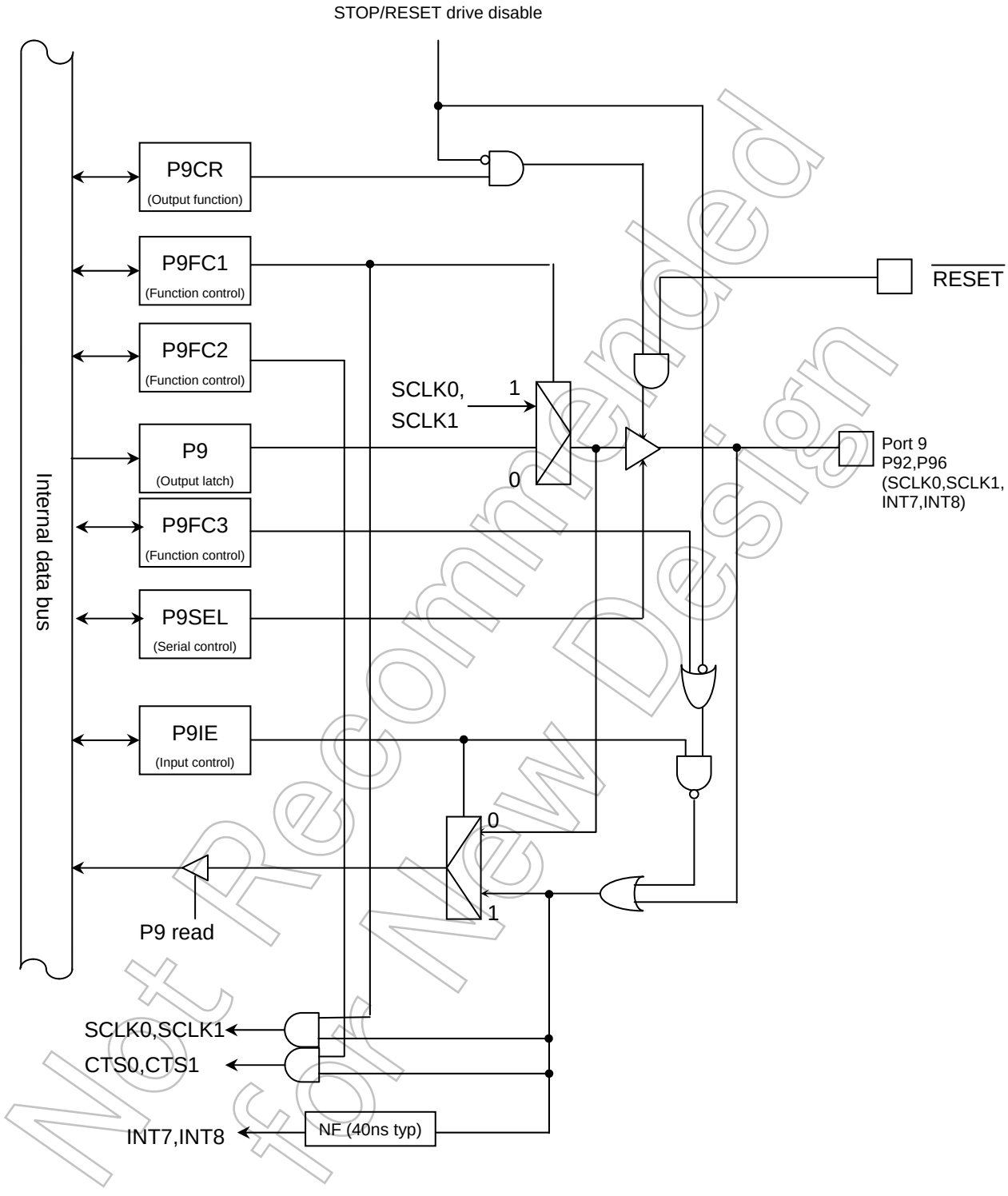


Fig. 7-22 Port 9(P92,P96)

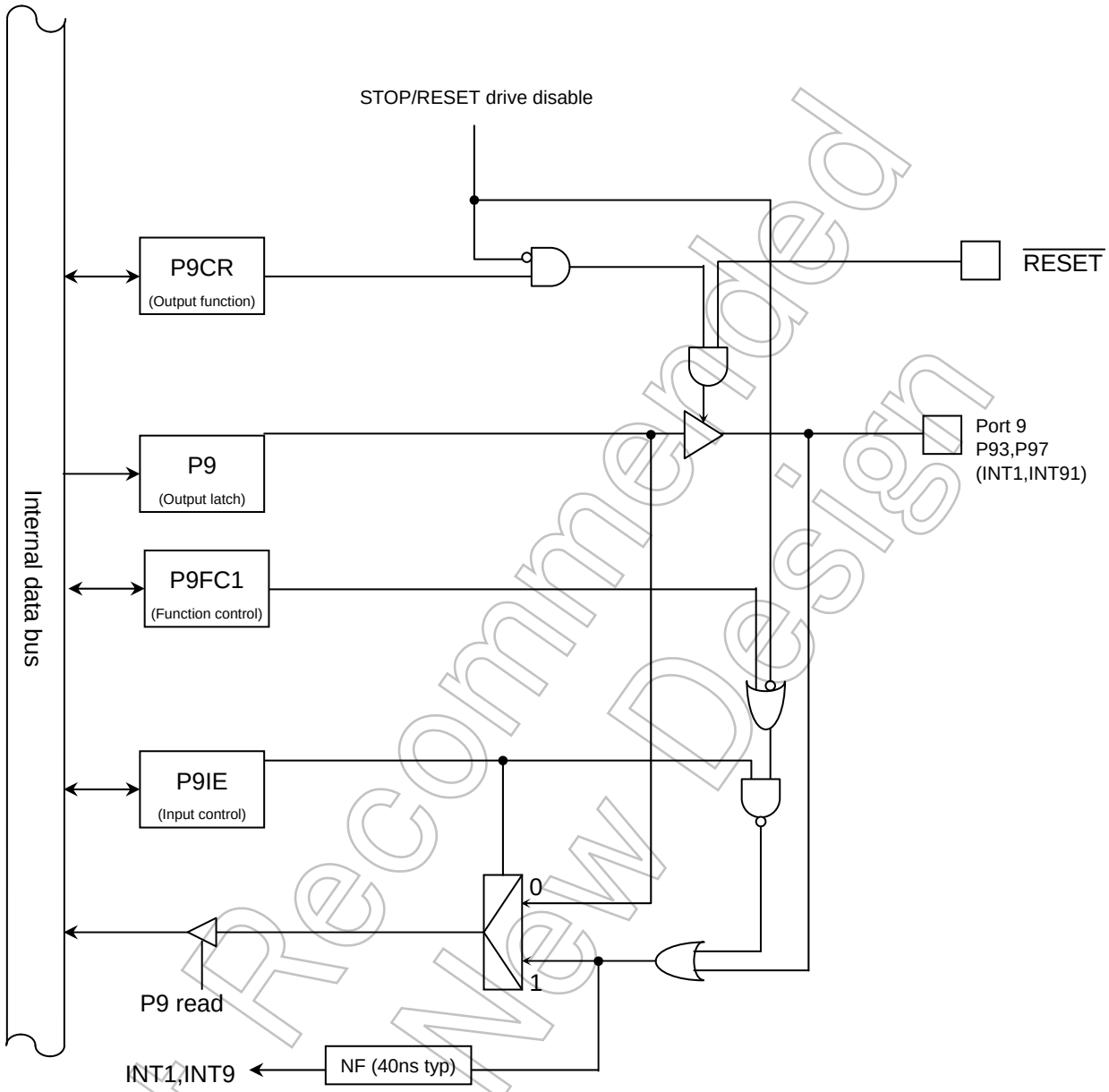


Fig. 7-23 Port 9 (P93,P97)

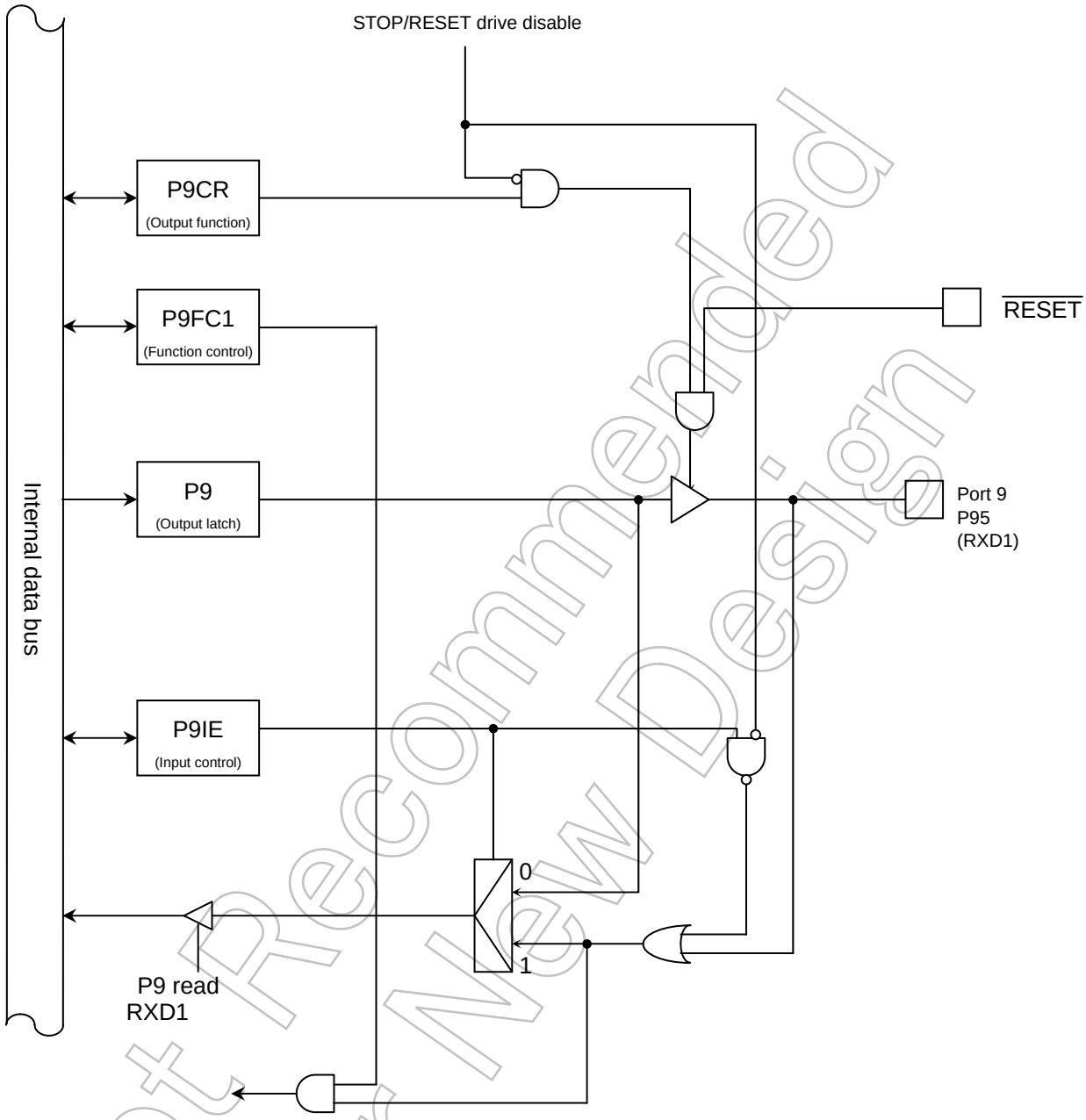


Fig. 7-24 Port 9 (P95)

Port 9 register

		7	6	5	4	3	2	1	0
P9	Bit Symbol	P97	P96	P95	P94	P93	P92	P91	P90
(0xFFFF_F090)	Read/Write	R/W							
	After reset	Input mode (output latch register is set to "1.")							

Port 9 control register

		7	6	5	4	3	2	1	0
P9CR	Bit Symbol	P97C	P96C	P95C	P94C	P93C	P92C	P91C	P90C
(0xFFFF_F091)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	0: Output disable 1: Output enable							

Port 9 function register 1

		7	6	5	4	3	2	1	0
P9FC1	Bit Symbol	P97F1	P96F1	P95F1	P94F1	P93F1	P92F1	P91F1	P90F1
(0xFFFF_F092)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	0:PORT 1:INT9	0:PORT 1:SCLK1	0:PORT 1RXD1	0:PORT 1:TXD1	0:PORT 1:INT1	0:PORT 1:SCLK0	0:PORT 1RXD0	0:PORT 1:TXD0

Port 9 function register 2

		7	6	5	4	3	2	1	0
P9FC2	Bit Symbol		P96F2				P92F2	P91F2	
(0xFFFF_F093)	Read/Write	R	R/W	R			R/W		R
	After reset	0	0	0			0	0	0
	Function	"0" is read.	0:PORT 1:CTS1	"0" is read.			0:PORT 1:CTS0	0:PORT 1:INTC	"0" is read.

Port 9 function register 3

		7	6	5	4	3	2	1	0
P9FC3	Bit Symbol		P96F				P92F		
(0xFFFF_F094)	Read/Write	R	R/W	R			R/W	R	
	After reset	0	0	0			0	0	
	Function	"0" is read.	0:PORT 1:INT8	"0" is read.			0:PORT 1:INT7	"0" is read.	

Port 9 Serial setting register

		7	6	5	4	3	2	1	0
P9SEL	Bit Symbol		PSEL96		PSEL94		PSEL92		PSEL90
(0xFFFF_F09D)	Read/Write	R	R/W	R	R/W	R	R/W	R	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	“0” is read.	0:PORT 1:Serial	“0” is read.	0:PORT 1:Serial	“0” is read.	0:PORT 1:Serial	“0” is read.	0:PORT 1:Serial

Port 9 Input enable control register

		7	6	5	4	3	2	1	0
P9IE	Bit Symbol	PIE97	PIE96	PIE95	PIE94	PIE93	PIE92	PIE91	PIE90
(0xFFFF_F09E)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable

7.12 Port A (PA0 through PA7)

The port A is a general-purpose, 8-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Outputs can be set by using the control register PACR and the function register PAFC. A reset allows all bits of the output latch PA to be set to "1," all bits of PACR, PAFC1 and PAFC2 to be cleared to "0," and the port A to be put in output disable mode.

Besides the input/output port function, the port A performs other functions: PA0 through PA2 and PA4 through PA6 have I2C function, PA0,PA1,PA3 and PA7 output a 16-bit timer, PA2 and PA6 input external interrupts, PA3 has DREQ function and PA7 has DACK function.

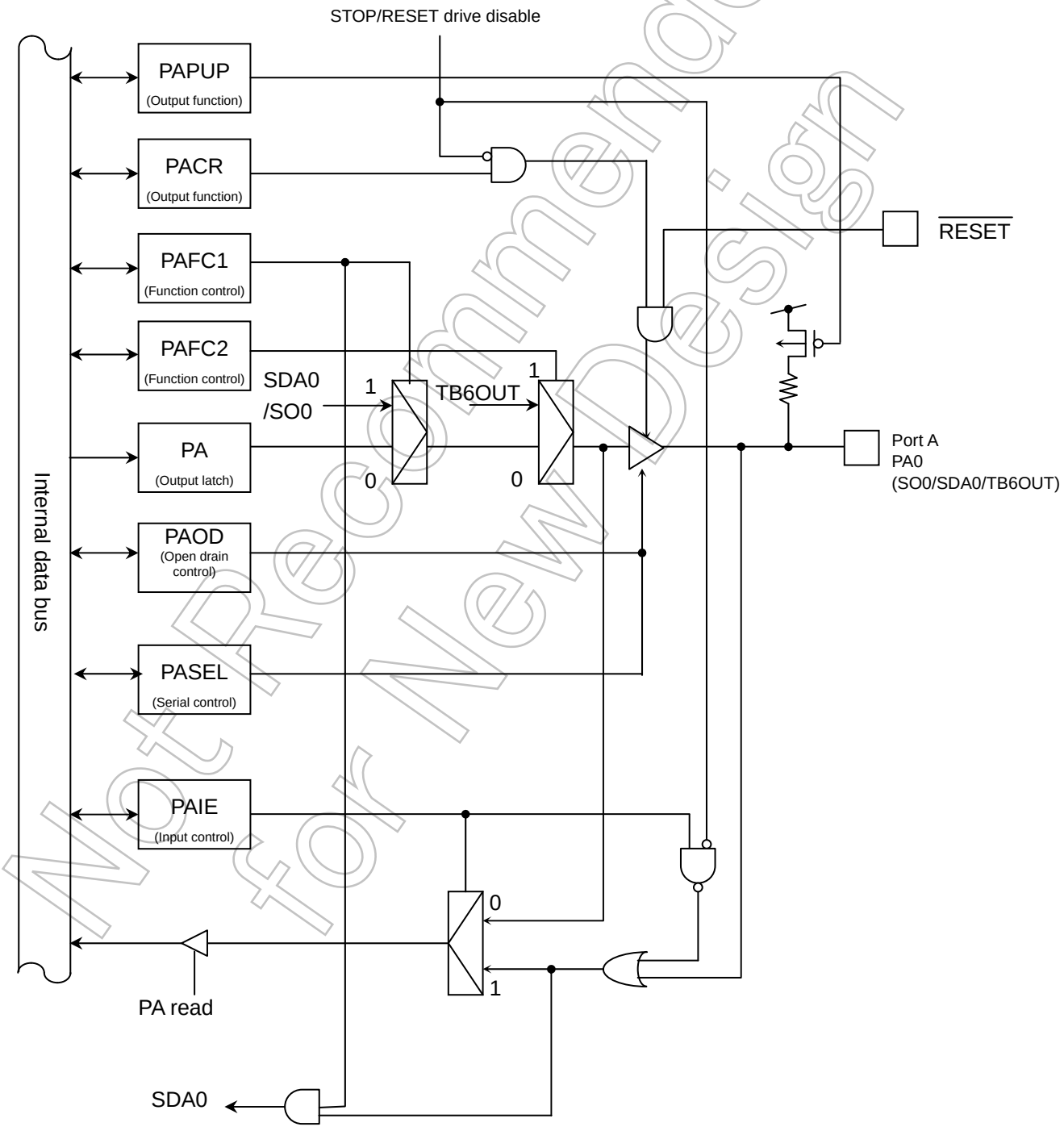


Fig. 7-25 Port A (PA0)

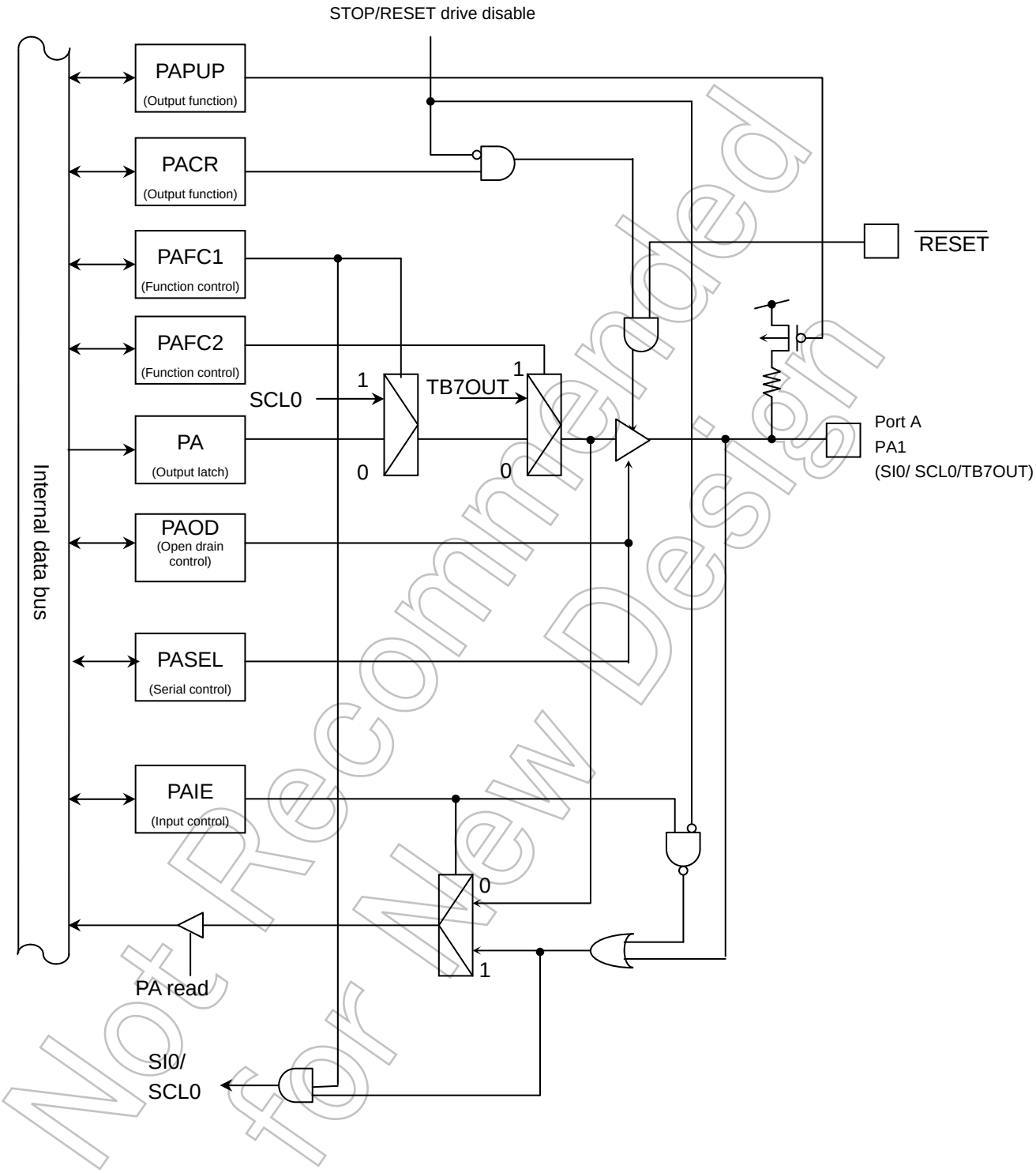


Fig. 7-26 Port A (PA1)

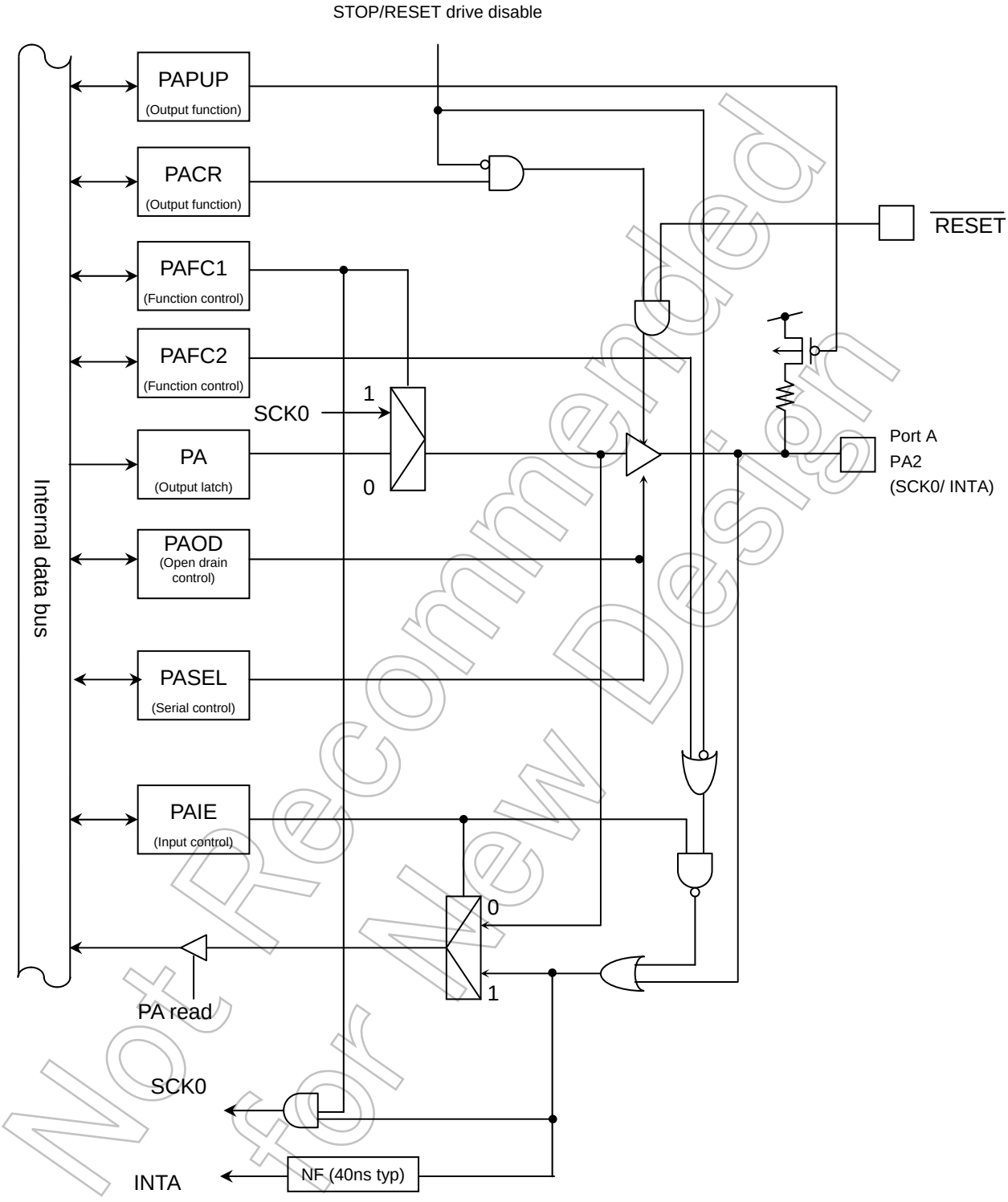


Fig. 7-27 Port A (PA2)

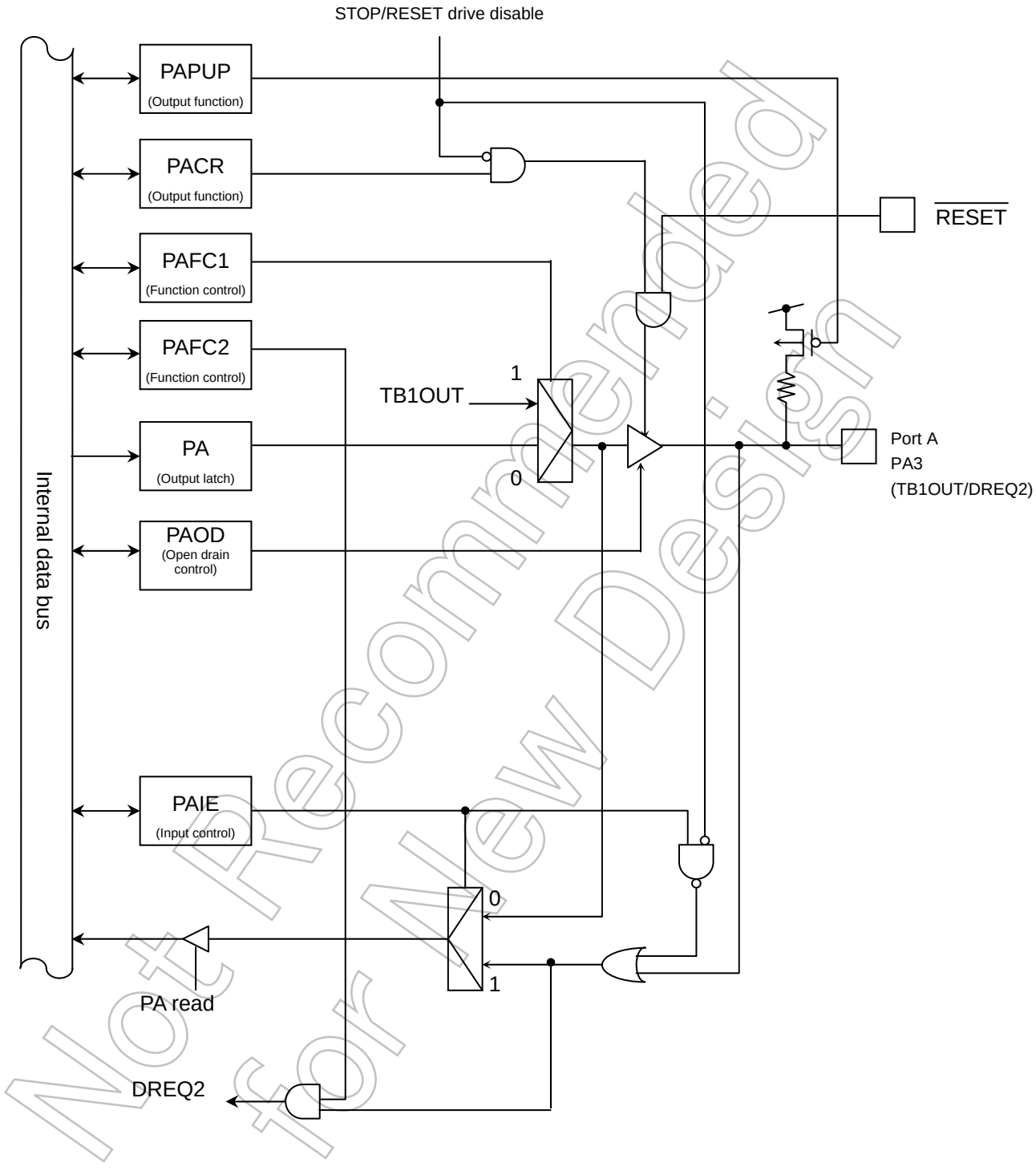


Fig. 7-28 Port A (PA3)

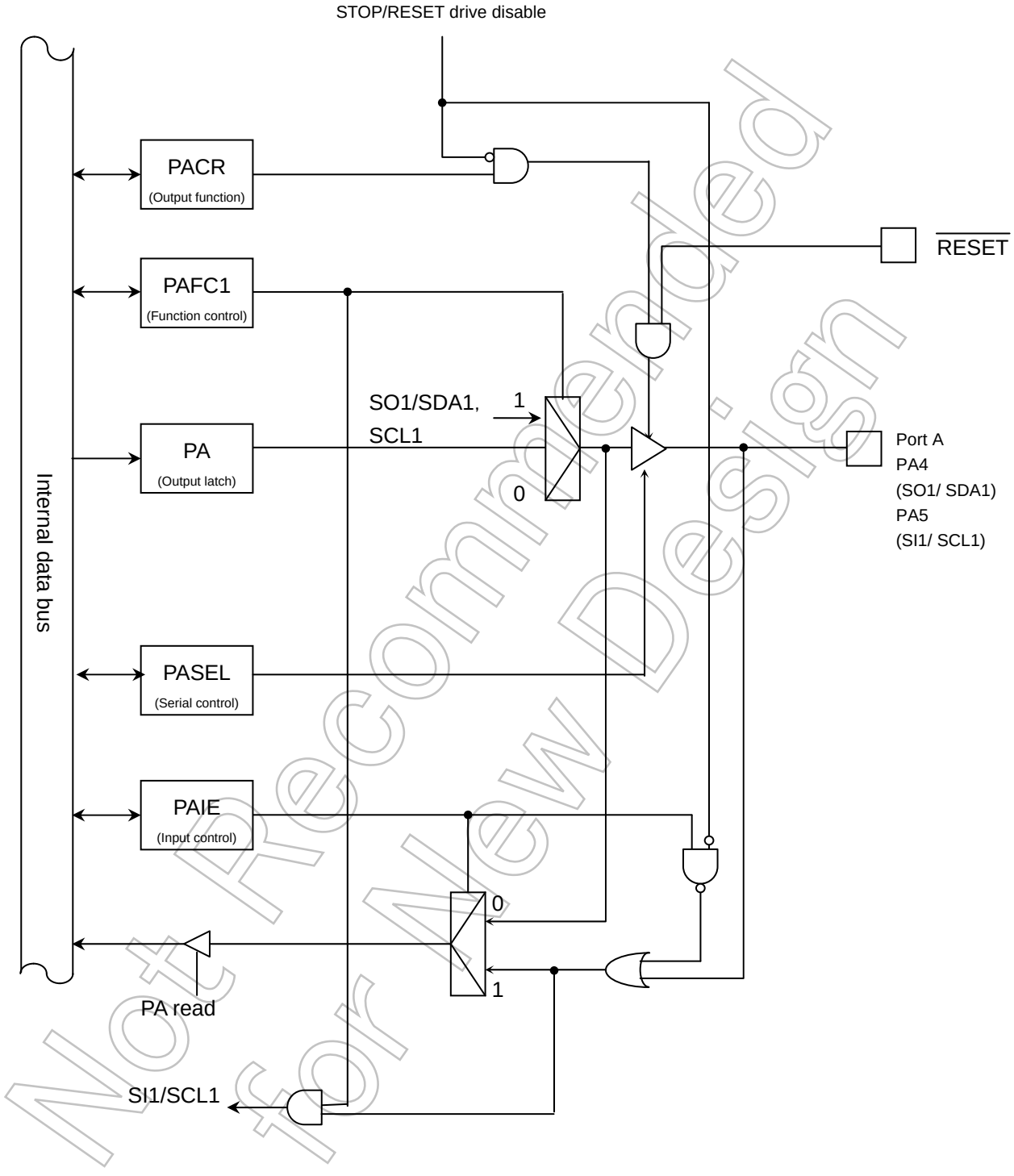


Fig. 7-29 Port A (PA4,5)

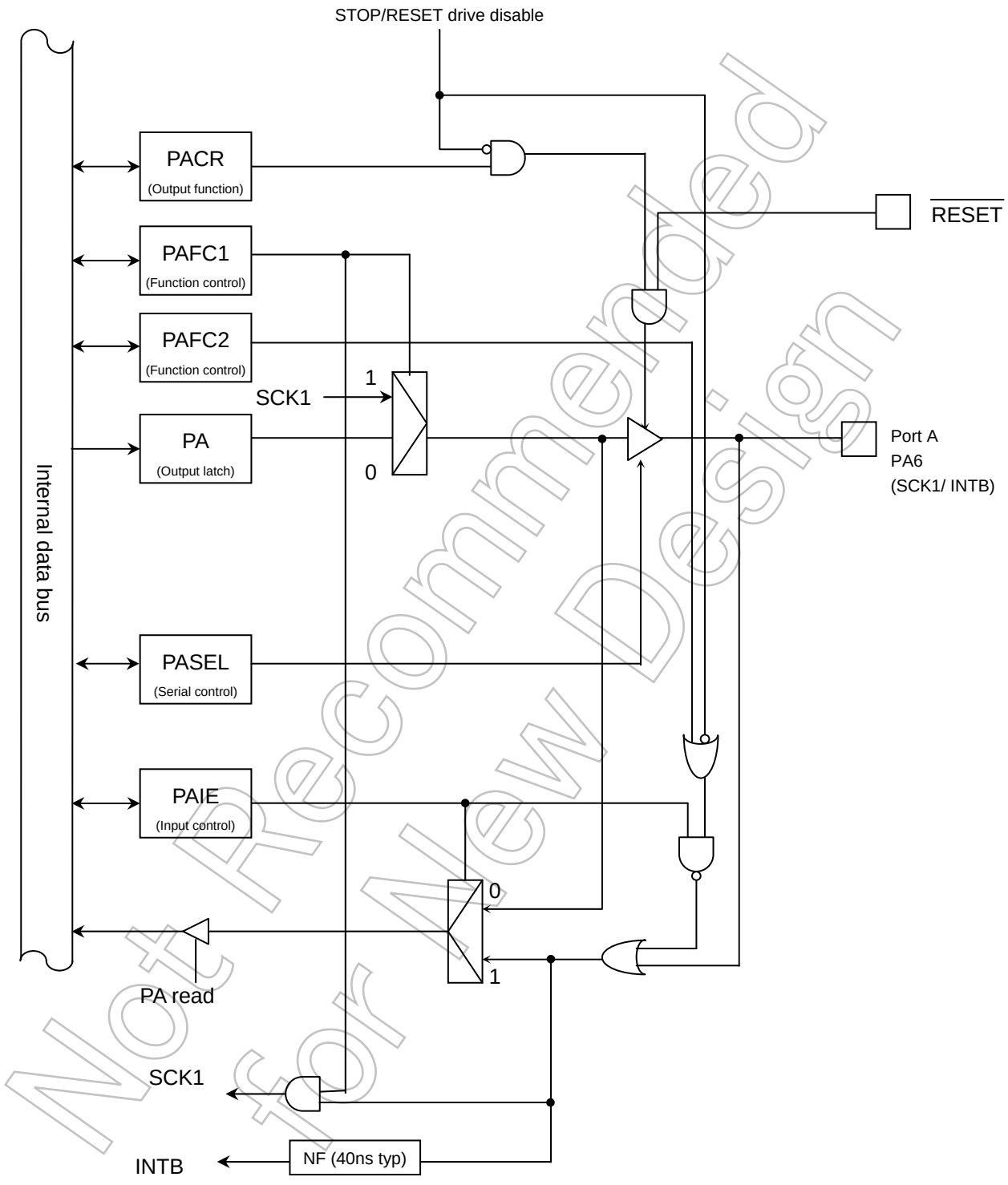


Fig. 7-30 Port A (PA6)

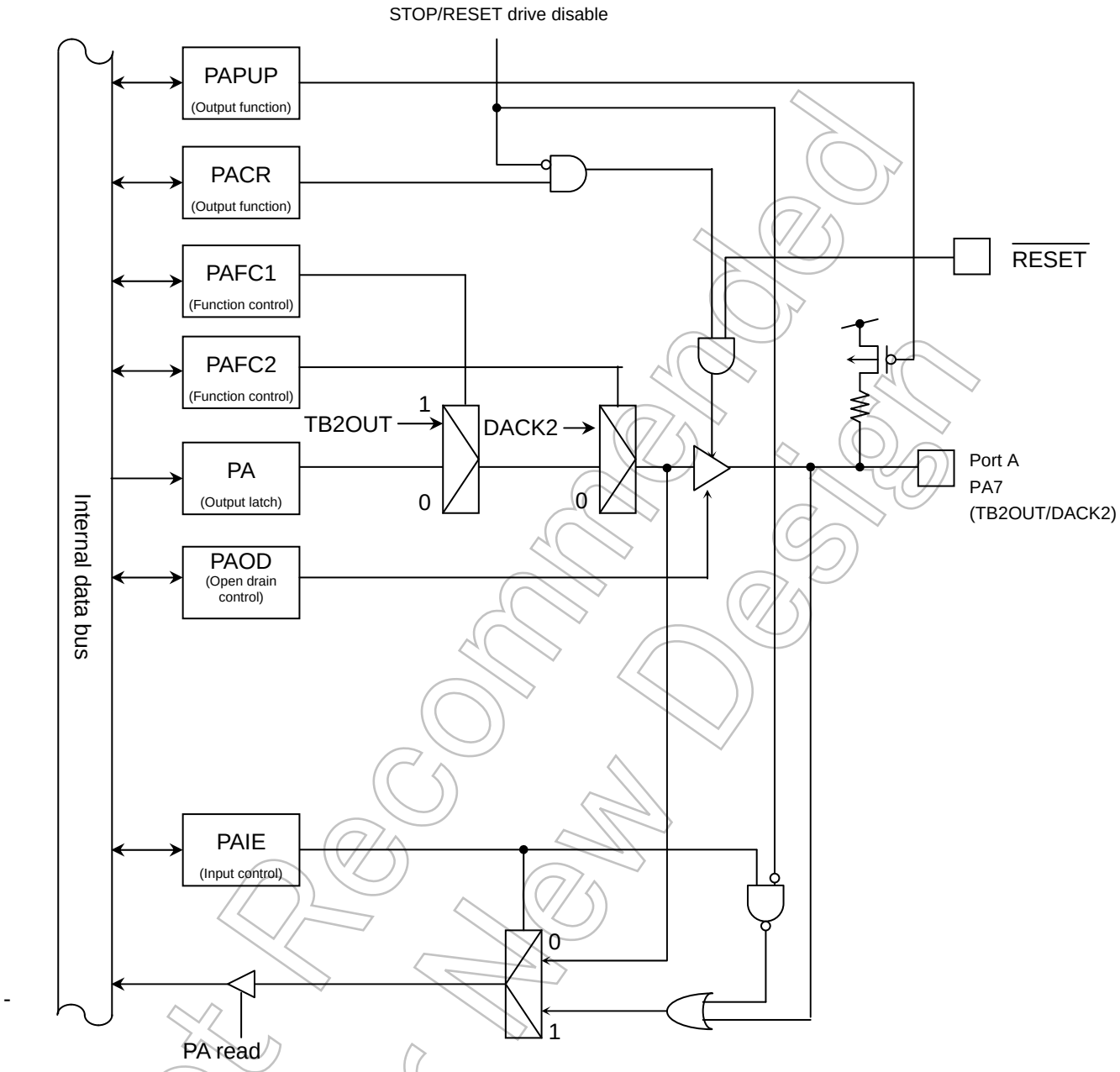


Fig. 7-31 Port A (PA7)

Port A register

PA (0xFFFF_F0A0)

	7	6	5	4	3	2	1	0
Bit Symbol	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
Read/Write	R/W							
After reset	Input mode (output latch register is set to “1.”)							

Port A control register

PACR (0xFFFF_F0A1)

	7	6	5	4	3	2	1	0
Bit Symbol	PA7C	PA6C	PA5C	PA4C	PA3C	PA2C	PA1C	PA0C
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0: Output disable 1: Output enable							

Port A function register 1

PAFC1 (0xFFFF_F0A2)

	7	6	5	4	3	2	1	0
Bit Symbol	PA7F1	PA6F1	PA5F1	PA4F1	PA3F1	PA2F1	PA1F1	PA0F1
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0:PORT 1:TB2OUT	0:PORT 1:SCK1	0:PORT 1:SCL1	0:PORT 1:SDA1	0:PORT 1:TB1OUT	0:PORT 1:SCK0	0:PORT 1:SCL0	0:PORT 1:SDA0

Port A function register 2

PAFC2 (0xFFFF_F0A3)

	7	6	5	4	3	2	1	0
Bit Symbol	PA7F2	PA6F2	PA5F2	PA4F2	PA3F2	PA2F2	PA1F2	PA0F2
Read/Write	R/W		R		R/W			
After reset	0	0	0		0	0	0	0
Function	0:PORT 1:DACK2	0:PORT 1:INTB	“0” is read.		0:PORT 1:DREQ2	0:PORT 1:INTA	0:PORT 1:TB7OUT	0:PORT 1:TB6OUT

Port A Open drain control register

PAOD (0xFFFF_ F0AA)

	7	6	5	4	3	2	1	0
Bit Symbol						PA2ODE	PA1ODE	PA0ODE
Read/Write	R	R			R	R/W	R/W	R/W
After reset	0	0			0	0	0	0
Function	“0” is read.	“0” is read. The output from these bits is in the open drain state. Connect a pull up resister externally when outputting “1”.			“0” is read.	0:CMOS 1:Open drain	0:CMOS 1:Open drain	0:CMOS 1:Open drain

Port A pull up control register

PAPUP (0xFFFF_F0AB)

	7	6	5	4	3	2	1	0
Bit Symbol	PEA7				PEA3	PEA2	PEA1	PEA0
Read/Write	R/W	R			R/W			
After reset	0	0			0	0	0	0
Function	Pull up 0: Off 1:Pull up	“0” is read. The output from these bits is in the open drain state. Connect a pull up resister externally when outputting “1”.			Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up

Port A Serial setting register

		7	6	5	4	3	2	1	0
PASEL	Bit Symbol		PSELA6	PSELA5	PSELA4		PSELA2	PSELA1	PSELA0
(0xFFFF_F0AD)	Read/Write	R	R/W			R	R/W		
	After reset	0	0	0	0	0	0	0	0
	Function	"0" is read.	0:PORT 1:Serial	0:PORT 1:Serial	0:PORT 1:Serial	"0" is read.	0:PORT 1:Serial	0:PORT 1:Serial	0:PORT 1:Serial

Port A Input enable control register

		7	6	5	4	3	2	1	0
PAIE	Bit Symbol	PIEA7	PIEA6	PIEA5	PIEA4	PIEA3	PIEA2	PIEA1	PIEA0
(0xFFFF_F0AE)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable

7.13 Port B (PB0 through PB7)

Port B is a general-purpose, 8-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Outputs can be set by using the function register PBFC1 through PBFC3 and the control register PBCR. A reset allows all bits of the output latch PB to be set to "1," all bits of PBCR and PBFC1 through PBFC3 to be cleared to "0," and the port B to be put in output disable mode.

Besides the input/output port function, the port B performs other functions: PB0 and PB4 output HSIO/SIO data, PB1 and PB5 input HSIO/SIO data, PB2 and PB6 input and output HSIO/SIO HCLK/CLK or input HCTS/CTS, PB3 and PB6 have external interrupts function, PB3 and PB4 perform a 16-bit capture input function and PB7 output 16-bit timer.

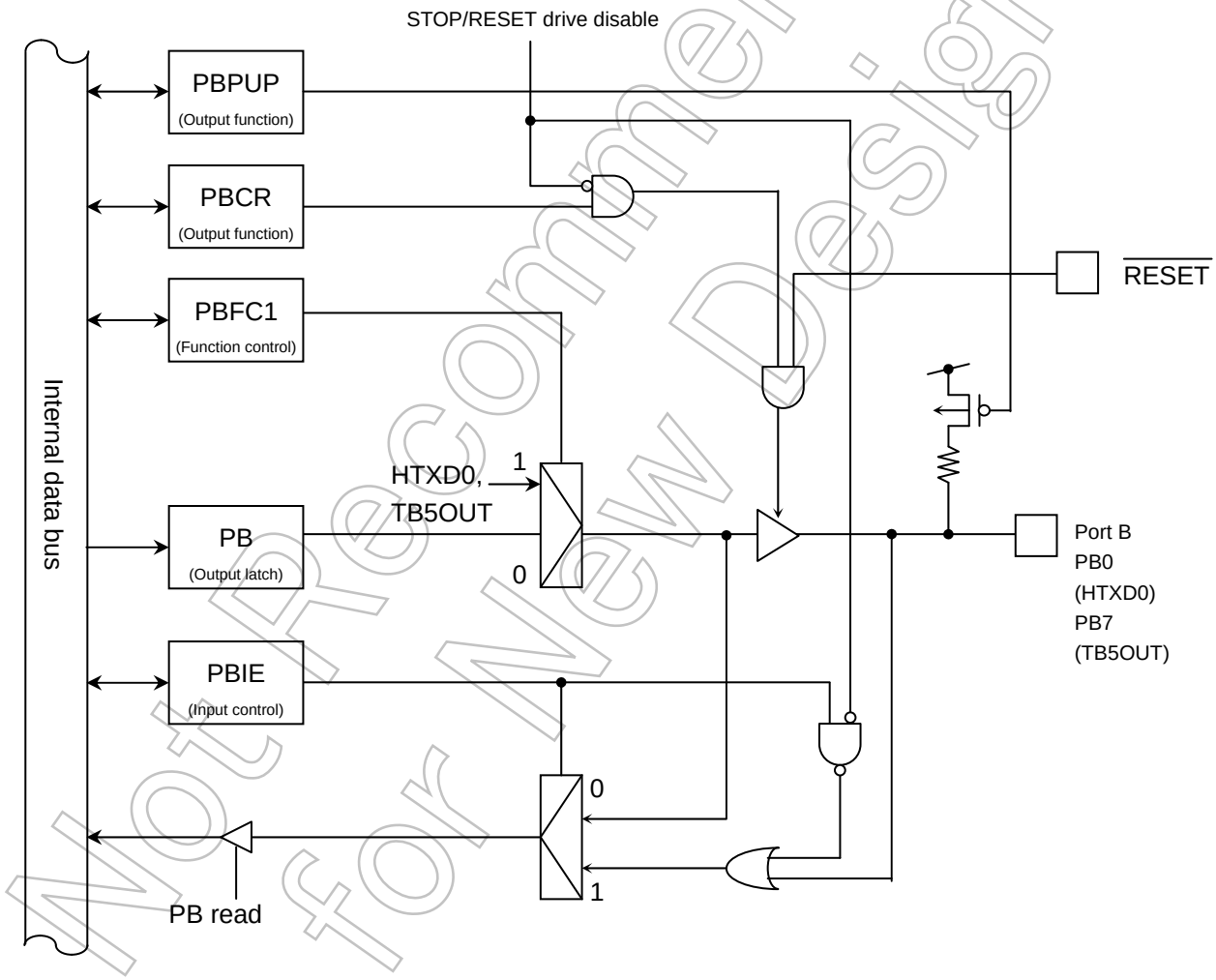


Fig. 7-32 Port B (PB0,PB7)

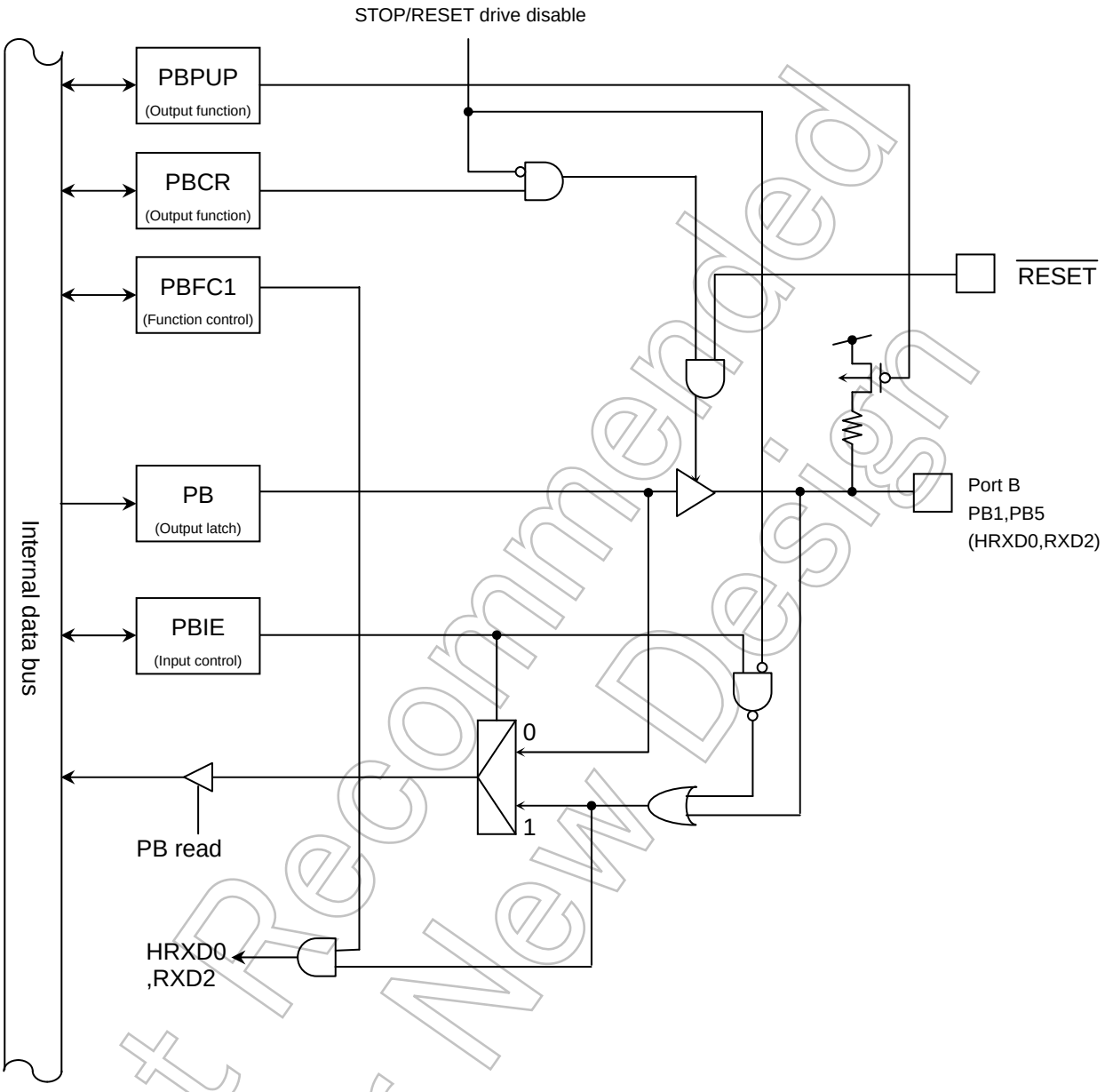


Fig. 7-33 Port B (PB1,PB5)

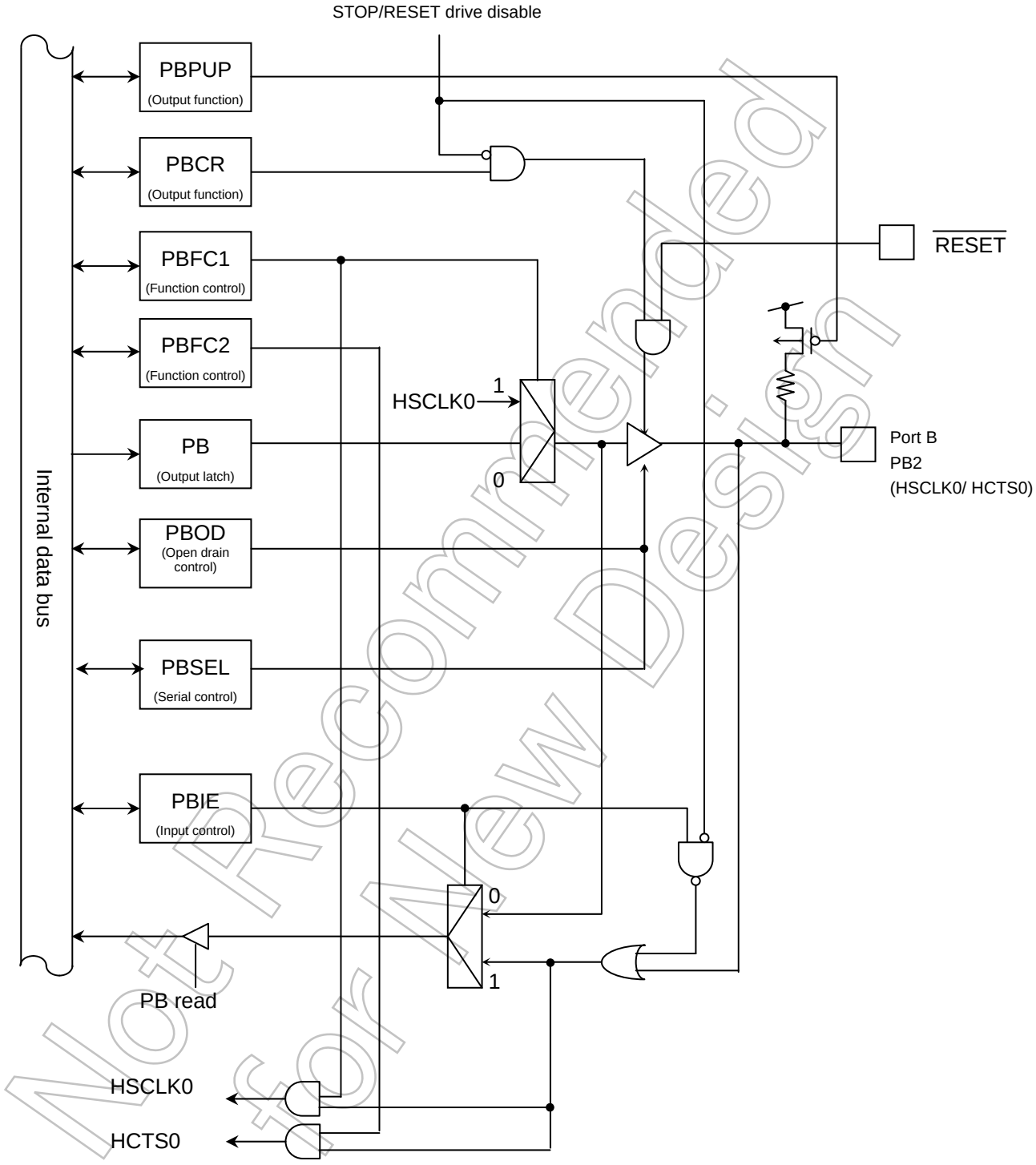


Fig. 7-34 Port B (PB2)

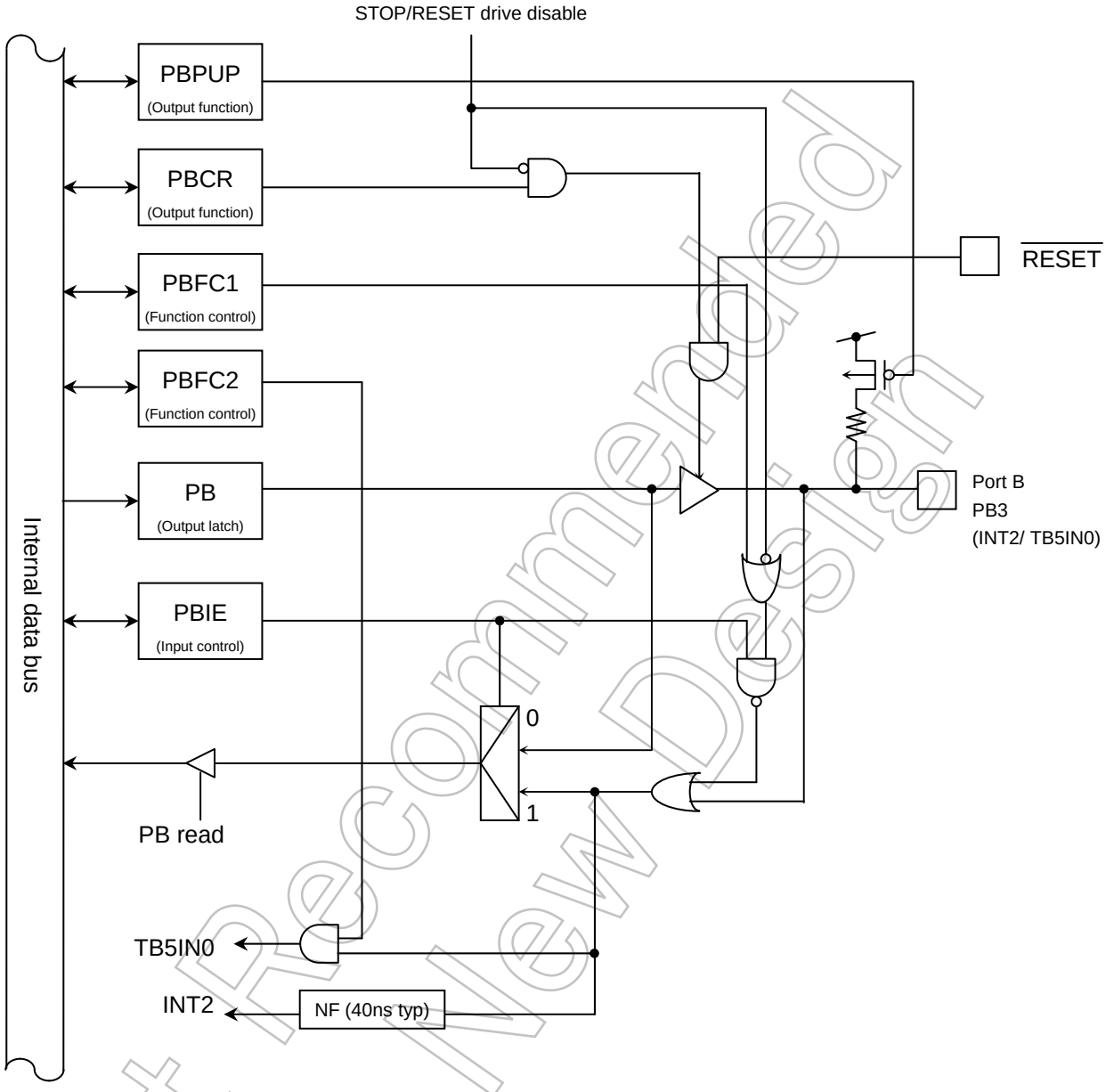


Fig. 7-35 Port B (PB3)

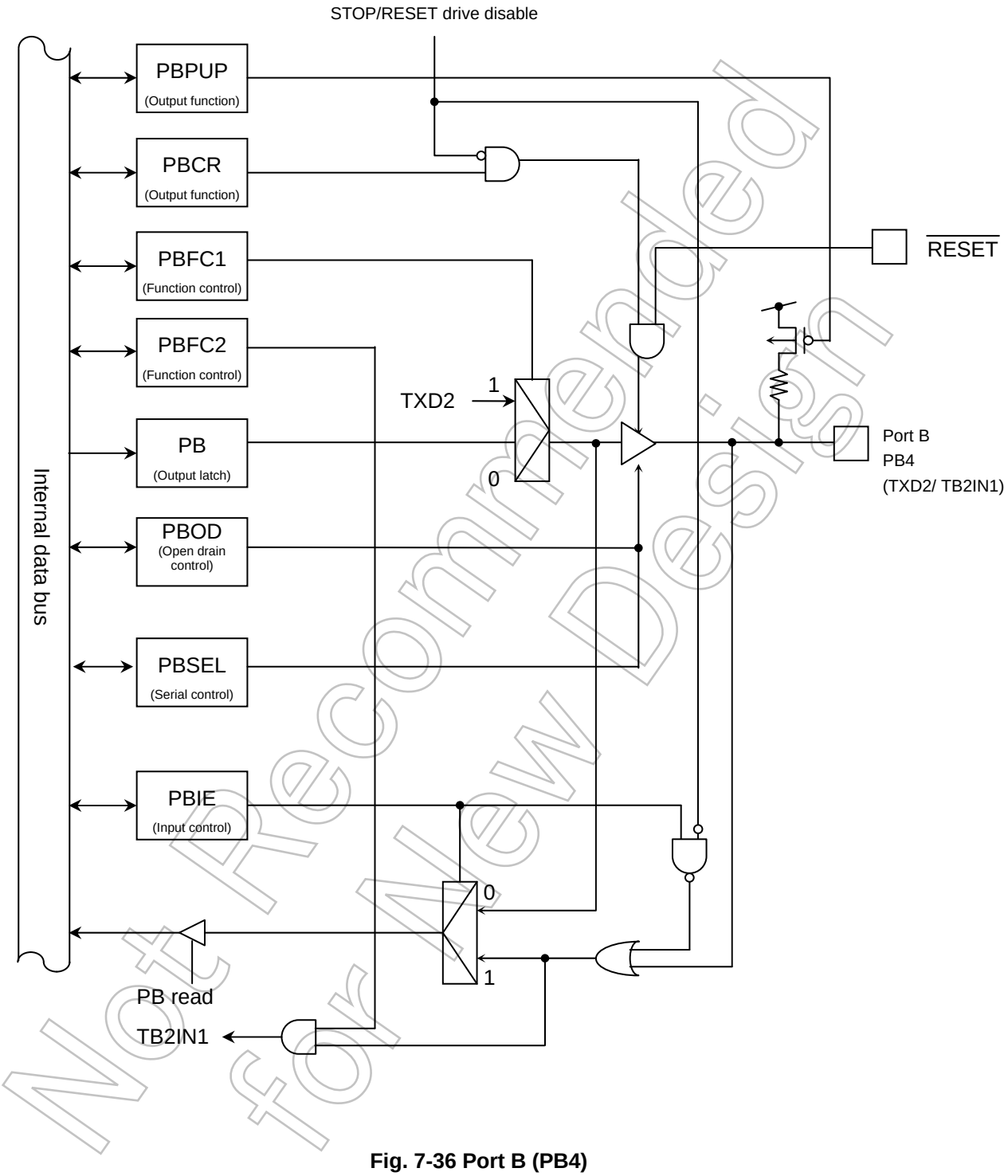


Fig. 7-36 Port B (PB4)

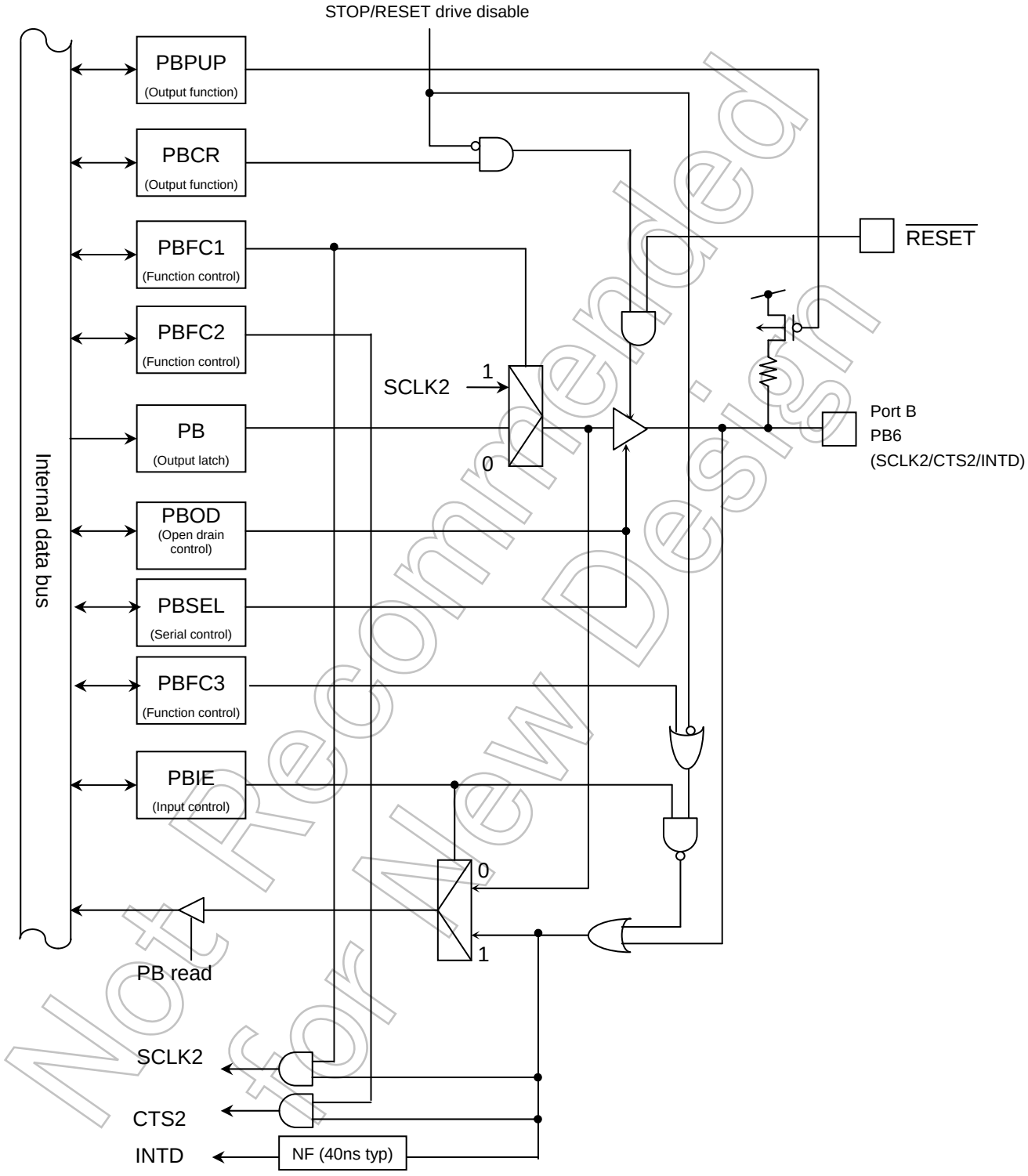


Fig. 7-37 Port B (PB6)

Port B register

		7	6	5	4	3	2	1	0
PB	Bit Symbol	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
(0xFFFF_F0B0)	Read/Write	R/W							
	After reset	Input mode (output latch register is set to “1.”)							

Port B control register

		7	6	5	4	3	2	1	0
PBCR	Bit Symbol	PB7C	PB6C	PB5C	PB4C	PB3C	PB2C	PB1C	PB0C
(0xFFFF_F0B1)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	0:Output disable 1:Output enable							

Port B function register 1

		7	6	5	4	3	2	1	0
PBFC1	Bit Symbol	PB7F1	PB6F1	PB5F1	PB4F1	PB3F1	PB2F1	PB1F1	PB0F1
(0xFFFF_F0B2)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	0:PORT 1:TB5OUT	0:PORT 1:SCLK2	0:PORT 1:RXD2	0:PORT 1:TXD2	0:PORT 1:INT2	0:PORT 1:HSCLK	0:PORT 1:HRXD	0:PORT 1: HTXD

Port B function register 2

		7	6	5	4	3	2	1	0
PBFC2	Bit Symbol		PB6F2		PB4F2	PB3F2	PB2F2		
(0xFFFF_F0B3)	Read/Write	R	R/W	R	R/W			R	
	After reset	0	0	0	0	0	0	0	
	Function	“0” is read.	0:PORT 1:CTS2	“0” is read.	0:PORT 1:TB5IN1	0:PORT 1:TB5IN0	0:PORT 1:HCTS	“0” is read.	

Port B function register 3

		7	6	5	4	3	2	1	0
PBFC3	Bit Symbol		PB6F3						
(0xFFFF_F0B4)	Read/Write	R	R/W	R					
	After reset	0	0	0					
	Function	“0” is read.	0:PORT 1:INTD	“0” is read.					

Port B Open drain control register

		7	6	5	4	3	2	1	0
PBODE	Bit Symbol		PB6ODE		PB4ODE		PB2ODE		PB0ODE
(0xFFFF_F0BA)	Read/Write	R	R/W	R	R/W	R	R/W	R	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	“0” is read.	0:CMOS 1: Open drain	“0” is read.	0:CMOS 1: Open drain	“0” is read.	0:CMOS 1: Open drain	“0” is read.	0:CMOS 1: Open drain

Port B pull up control register

		7	6	5	4	3	2	1	0
PBPUP	Bit Symbol	PEB7	PEB6	PEB5	PEB4	PEB3	PEB2	PEB1	PEB0
(0xFFFF_F0BB)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up

Port B Serial setting register

		7	6	5	4	3	2	1	0
PBSEL	Bit Symbol	PSELB7	PSELB6		PSELB4		PSELB2		PSELB0
(0xFFFF_F0BD)	Read/Write	R/W		R	R/W	R	R/W	R	R/W
	After reset	0	0	0	0	0	0	0	0
	Function	0:PORT 1:Serial	0:PORT 1:Serial	“0” is read.	0:PORT 1:Serial	“0” is read.	0:PORT 1:Serial	“0” is read.	0:PORT 1:Serial

Port B Input enable control register

		7	6	5	4	3	2	1	0
PBIE	Bit Symbol	PIEB7	PIEB6	PIEB5	PIEB4	PIEB3	PIEB2	PIEB1	PIEB0
(0xFFFF_F0BE)	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable

7.14 Port C (PC0 through PC3)

Port C is a general-purpose, 8-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Outputs can be set by using the function register PCFC1 and PCFC2 and the control register PCCR. A reset allows all bits of the output latch PC to be set to "1," all bits of PCCR and PCFC1 and PCFC2 to be cleared to "0," and the port C to be put in output disable mode.

Besides the input/output port function, the port C performs other functions: PC0 has AD trigger function, PC0 and PC1 output a 16-bit timer and PC2 and PC3 input external interrupts.

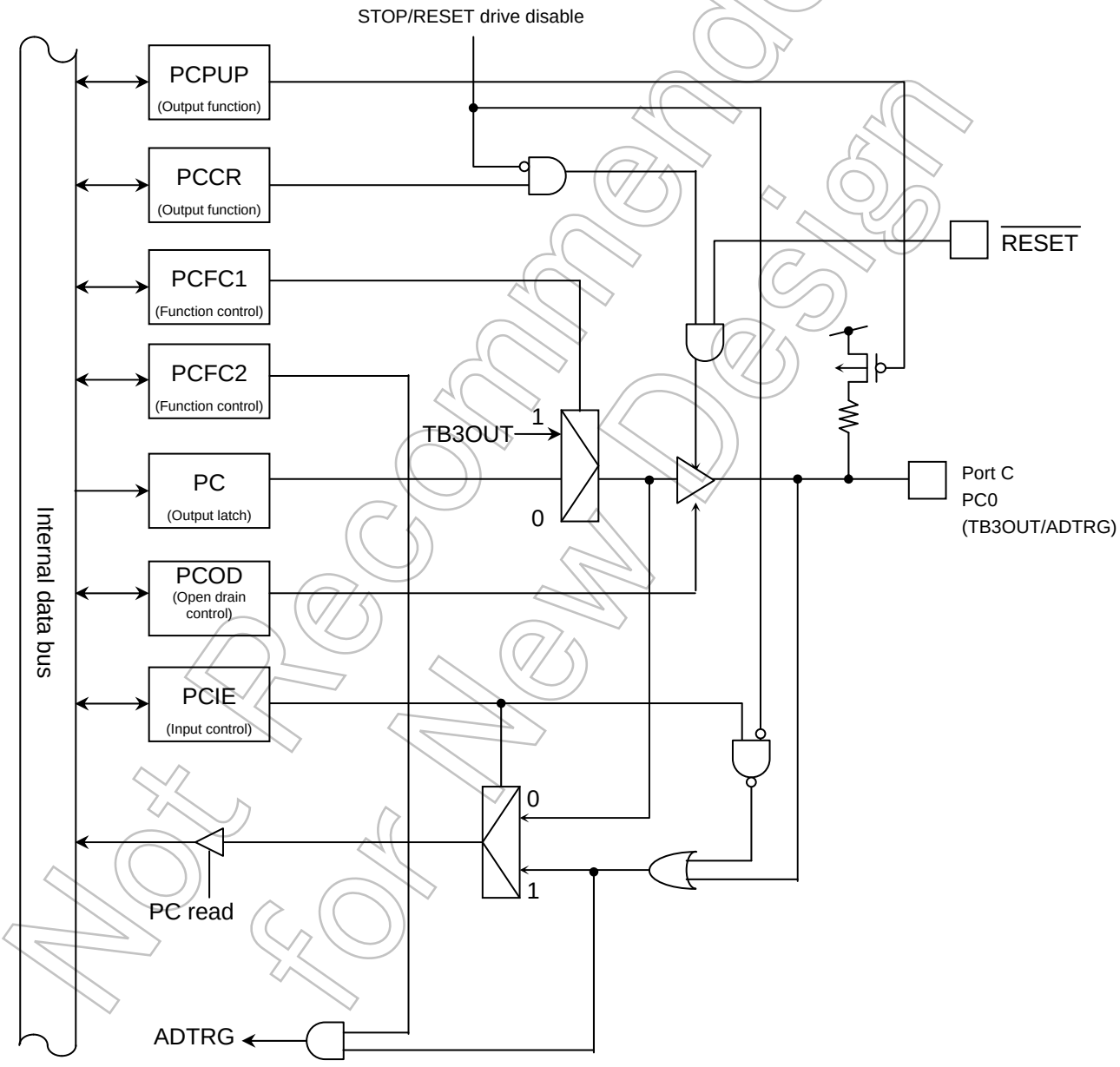


Fig. 7-38 Port C (PC0)

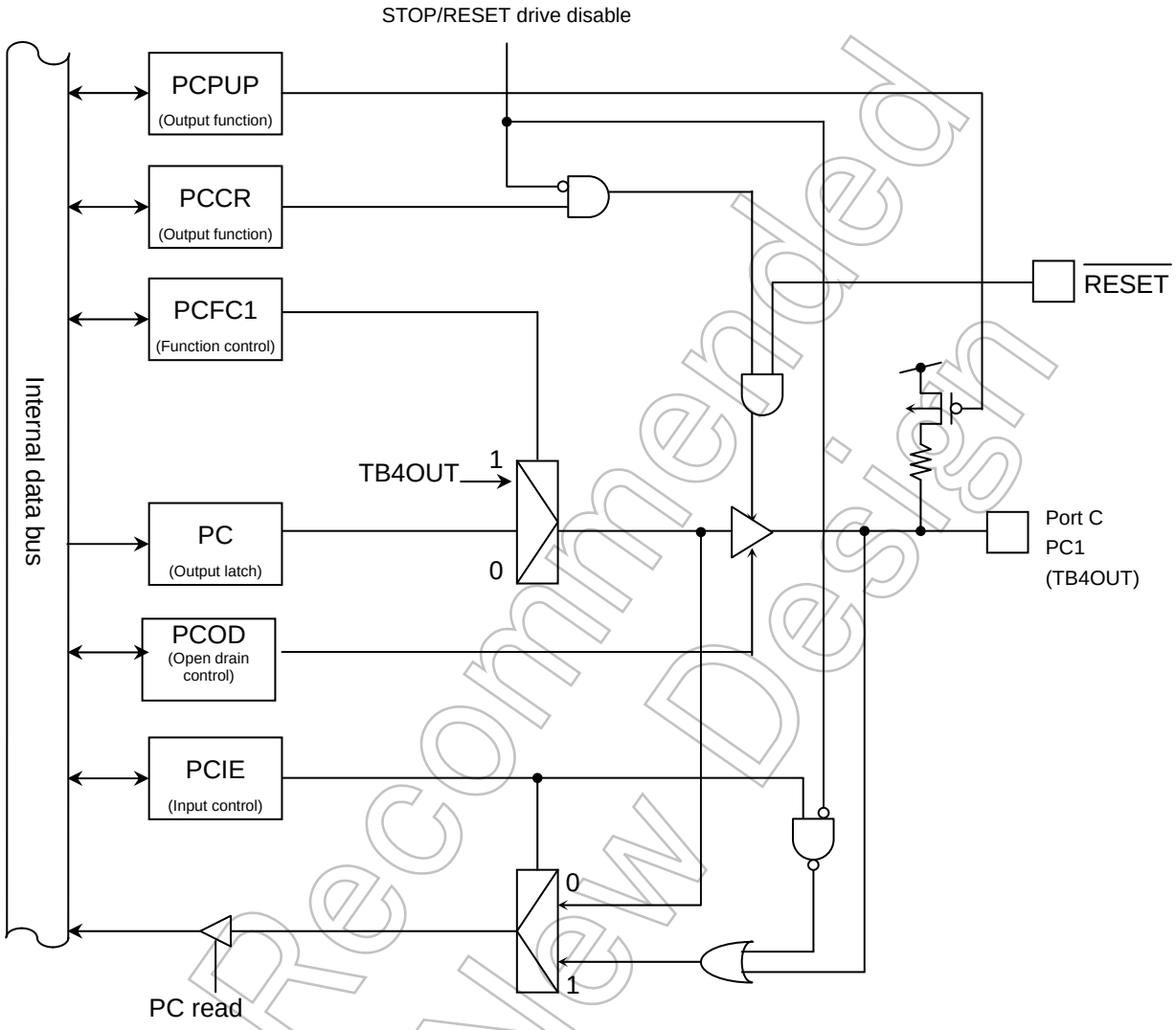


Fig. 7-39 Port C (PC1)

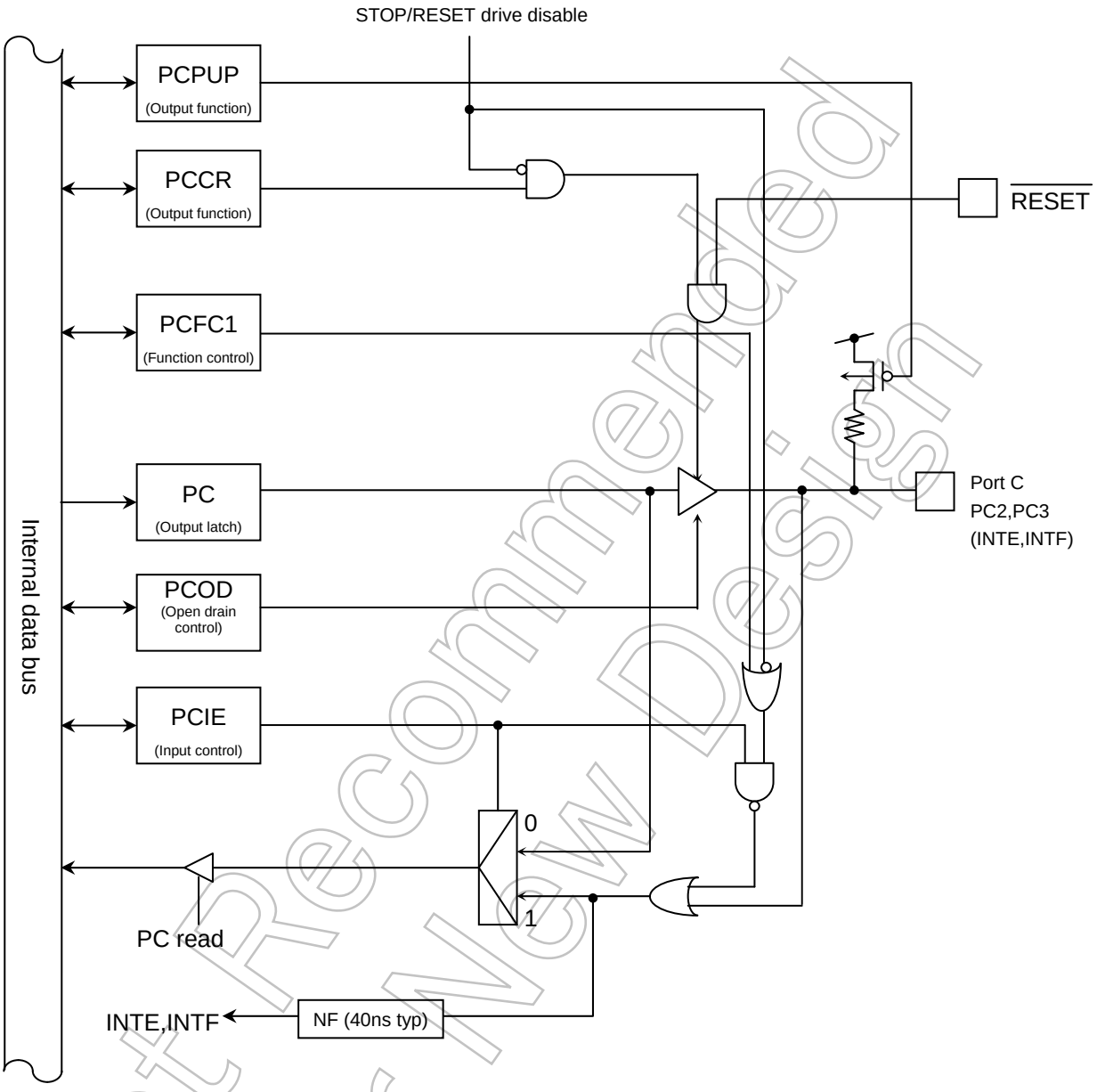


Fig. 7-40 Port C (PC2,PC3)

*Fig. 7-40 does not illustrate debug function.

Port C register

		7	6	5	4	3	2	1	0
PC	Bit Symbol					PC3	PC2	PC1	PC0
(0xFFFF_F0C0)	Read/Write	R				R/W			
	After reset	"0" is read.				Input mode (output latch register is set to "1.")			

Port C control register

		7	6	5	4	3	2	1	0
PCCR	Bit Symbol					PC3C	PC2C	PC1C	PC0C
(0xFFFF_F0C1)	Read/Write	R				R/W			
	After reset	0				0	0	0	0
	Function	"0" is read.				0:Output disable 1:Output enable			

Port C function register 1

		7	6	5	4	3	2	1	0
PCFC1	Bit Symbol					PC3F1	PC2F1	PC1F1	PC0F1
(0xFFFF_F0C2)	Read/Write	R				R/W			
	After reset	0				0	0	0	0
	Function	"0" is read.				0:PORT 1:INTF	0:PORT 1:INTE	0:PORT 1:TB4OUT	0:PORT 1:TB3OUT

Port C function register 2

		7	6	5	4	3	2	1	0
PCFC2	Bit Symbol								PC0F2
(0xFFFF_F0C3)	Read/Write	R							R/W
	After reset	0							0
	Function	"0" is read.							0:PORT 1:ADTRG

Port C Open drain control register

		7	6	5	4	3	2	1	0
PCODE	Bit Symbol					PC3ODE	PC2ODE	PC1ODE	PC0ODE
(0xFFFF_F0CA)	Read/Write	R				R/W	R/W	R/W	R/W
	After reset	0				0	0	0	0
	Function	"0" is read.				0:CMOS 1: Open drain	0:CMOS 1: Open drain	0:CMOS 1: Open drain	0:CMOS 1: Open drain

Port C pull up control register

		7	6	5	4	3	2	1	0
PCPUP	Bit Symbol					PEC3	PEC2	PEC1	PEC0
(0xFFFF_F0CB)	Read/Write	R				R/W			
	After reset	0				0	0	0	0
	Function	"0" is read.				Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up

Port C Input enable control register

		7	6	5	4	3	2	1	0
PCIE	Bit Symbol					PIEC3	PIEC2	PIEC1	PIEC0
(0xFFFF_F0CE)	Read/Write	R				R/W			
	After reset	0				0	0	0	0
	Function	"0" is read.				Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable

7.15 Port D(PD0 through P7) (not embedded in FBGA)

The port D is a general-purpose, 8-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Outputs can be set by using the control register PDCR and the function register PDFC1. A reset allows all bits of the output latch PD to be set to "1," all bits of PDCR and PDFC1 to be cleared to "0," and the port D to be put in output disable mode.

Besides the input port function, the port D performs other functions: PD4 through PD7 capture a 16-bit timer.

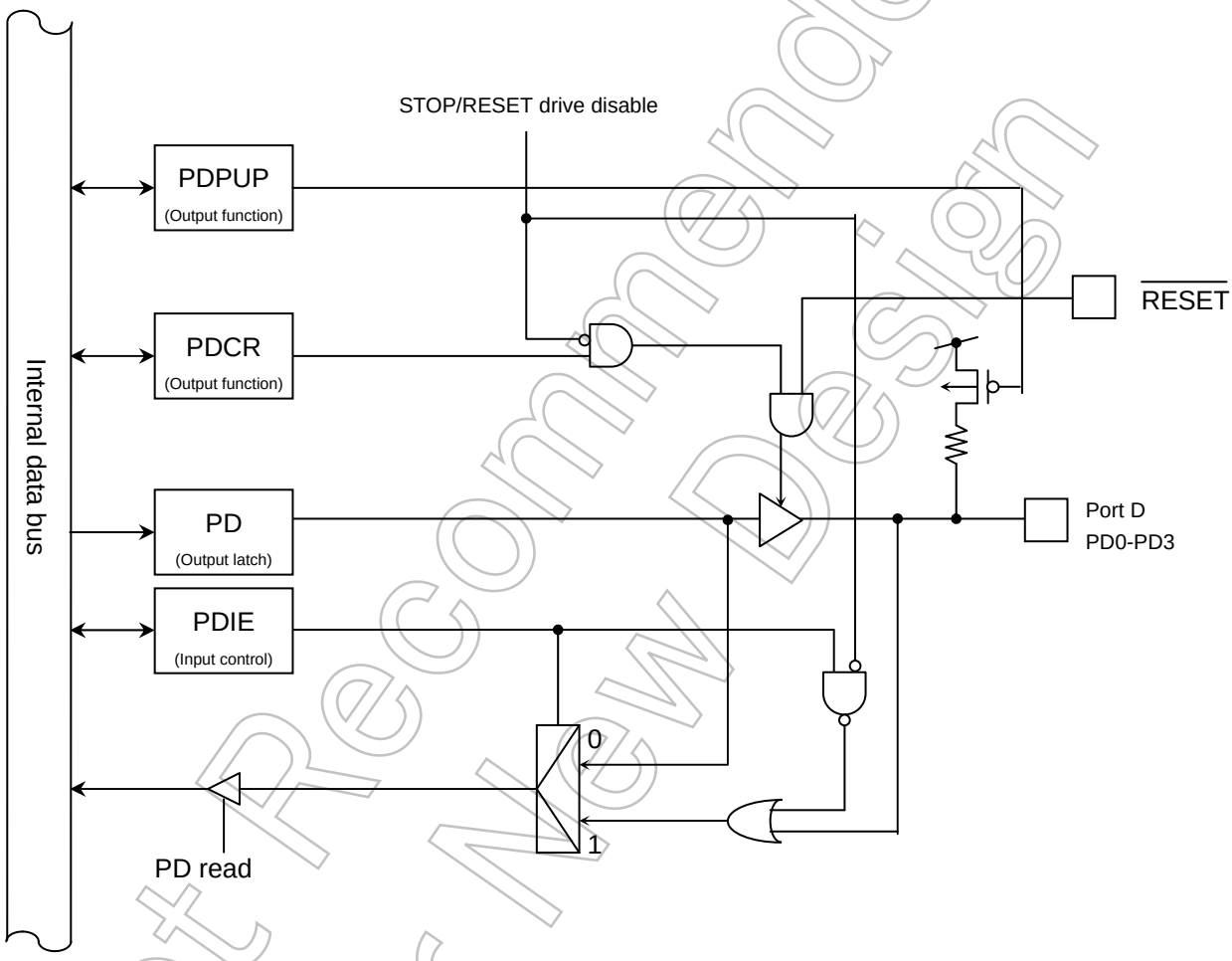


Fig. 7-41 Port D (PD0-PD3)

*Fig. 7-41 does not illustrate debug function.

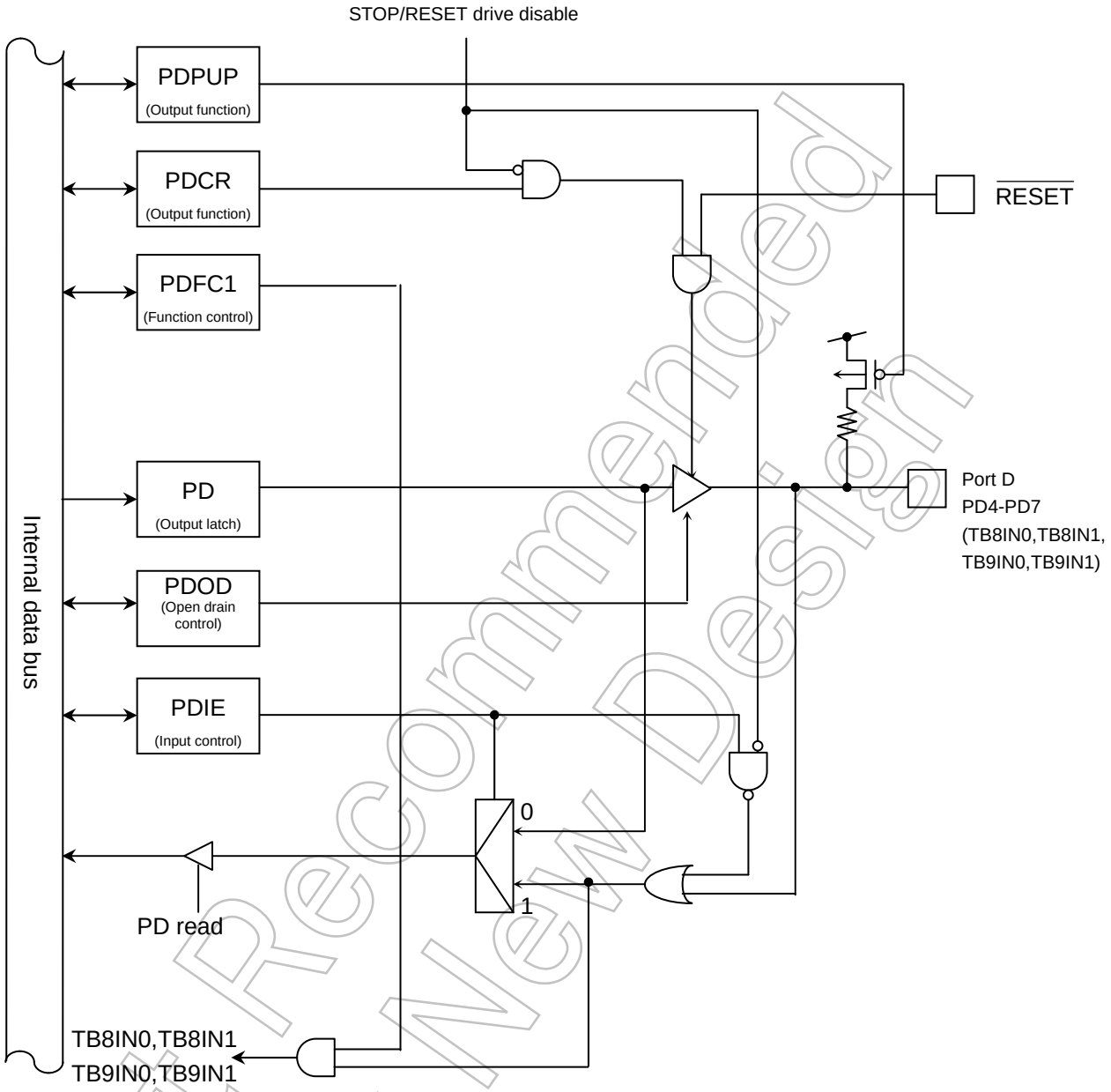


Fig. 7-42 Port D (PD4-PD7)

*Fig. 7-42 does not illustrate debug function.

Port D register

PD (0xFFFF_F0E0)

	7	6	5	4	3	2	1	0
Bit Symbol	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD0
Read/Write	R/W							
After reset	Input mode (output latch register is set to "0.")							

Port D control register

PDCR (0xFFFF_F0E1)

	7	6	5	4	3	2	1	0
Bit Symbol	PD7C	PD6C	PD5C	PD4C	PD3C	PD2C	PD1C	PD0C
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	0:Output disable 1:Output enable							

Port D function register 1

PDFC1 (0xFFFF_F0E2)

	7	6	5	4	3	2	1	0
Bit Symbol	PD7F1	PD6F1	PD5F1	PD4F1				
Read/Write	R/W				R			
After reset	0	0	0	0	0			
Function	0:PORT 1:TB9IN1	0:PORT 1:TB9IN0	0:PORT 1:TB8IN1	0:PORT 1:TB8IN0	"0" is read.			

Port D pull up control register

PDPUP (0xFFFF_F0EB)

	7	6	5	4	3	2	1	0
Bit Symbol	PED7	PED6	PED5	PED4	PED3	PED2	PED1	PED0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up

Port D Input enable control register

PDIE (0xFFFF_F0EE)

	7	6	5	4	3	2	1	0
Bit Symbol	PIED7	PIED6	PIED5	PIED4	PIED3	PIED2	PIED1	PIED0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable

7.16 Port E (PE0 through PE5)

The port E is a general-purpose, 5-bit input/output port. For this port, inputs and outputs can be specified in units of bits. Outputs can be set by using the control register PECR. A reset allows all bits of the output latch PE to be set to "1," all bits of PECR to be cleared to "0," and the port E to be put in output disable mode.

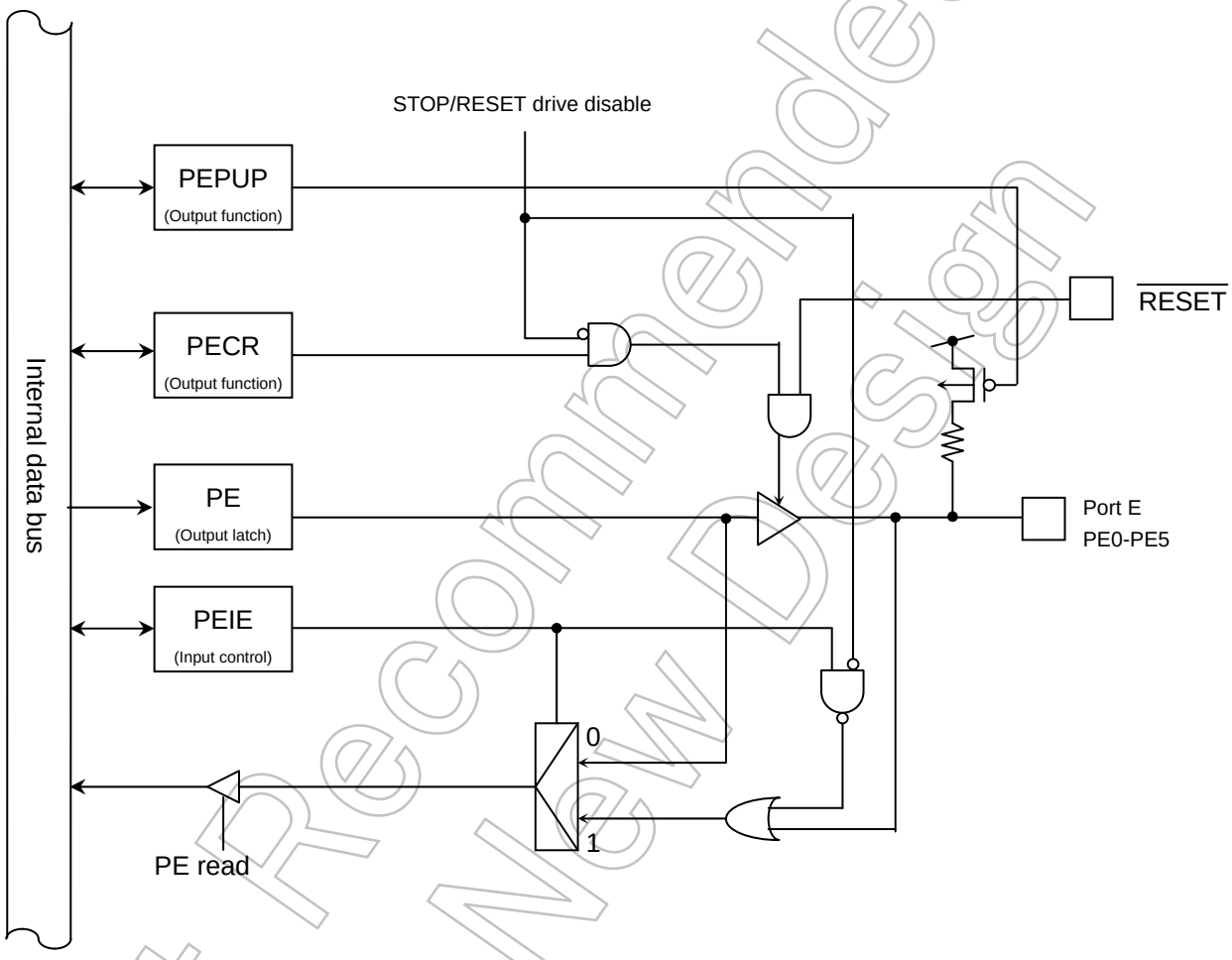


Fig. 7-43 Port E (PE0-PE5)

*Fig. 7-43 does not illustrate debug function.

Port E register

		7	6	5	4	3	2	1	0
PE	Bit Symbol			PE5	PE4	PE3	PE2	PE1	PE0
(0xFFFF_F0E0)	Read/Write	R		R/W					
	After reset	0		Input mode (output latch register is set to “0.”)					

Port E control register

		7	6	5	4	3	2	1	0
PECR	Bit Symbol			PE5C	PE4C	PE3C	PE2C	PE1C	PE0C
(0xFFFF_F0E1)	Read/Write	R		R/W					
	After reset	0		0	0	0	0	0	0
	Function	“0” is read.		0:Output disable 1:Output enable					

Port E pull up control register

		7	6	5	4	3	2	1	0
PEPUP	Bit Symbol			PEE5	PEE4	PEE3	PEE2	PEE1	PEE0
(0xFFFF_F0EB)	Read/Write	R		R/W					
	After reset	0		0	0	0	0	0	0
	Function	“0” is read.		Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up	Pull up 0: Off 1:Pull up

Port E Input enable control register

		7	6	5	4	3	2	1	0
PEIE	Bit Symbol			PIEE5	PIEE4	PIEE3	PIEE2	PIEE1	PIEE0
(0xFFFF_F0EE)	Read/Write	R		R/W					
	After reset	0		0	0	0	0	0	0
	Function	“0” is read.		Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable	Input 0:Disable 1:Enable

8. External Bus Interface

The TMP19A23 has a built-in external bus interface function to connect to external memory, I/Os, etc. This interface consists of an external bus interface circuit (EBIF), a chip selector (CS) and a wait controller.

The chip selector and wait controller designate mapping addresses in a 4-block address space and also control wait states and data bus widths (8- or 16-bit) in these and other external address spaces.

The external bus interface circuit (EBIF) controls the timing of external buses based on the chip selector and wait controller settings. The EBIF also controls the dynamic bus sizing and the bus arbitration with the external bus master.

●External bus mode

Selectable address, data separator bus mode and multiplex mode

●Wait function

This function can be enabled for each block.

- A wait of up to 7 clocks can be automatically inserted.
- A wait can be inserted via the $\overline{\text{WAIT}}$ / $\overline{\text{RDY}}$ pin.

●Data bus width

Either an 8- or 16-bit width can be set for each block.

●Recovery cycle (read/write)

If external bus cycles occur continuously, a dummy cycle of up to 2 clocks can be inserted and this dummy cycle can be specified for each block.

●Recovery cycle (chip selector)

When an external bus is selected, a dummy cycle of up to 3 clocks can be inserted and this dummy cycle can be specified for each block.

●Bus arbitration function

8.1 Address and Data Pins

(1) Address and data pin settings

The TMP19A23 can be set to either separate bus or multiplexed bus mode. Setting the BUSMD pin (port P44) to the "L" level at a reset activates the separate bus mode, and setting the pin to the "H" level activates the multiplexed bus mode. Port pins 0, 1, 2, 5 and 6, which are to be connected to external devices (memory), are used as address buses, data buses and address/data buses (see Table 8-1).

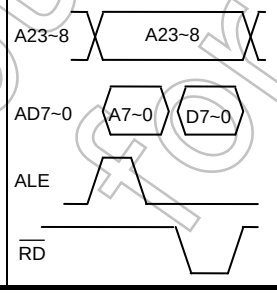
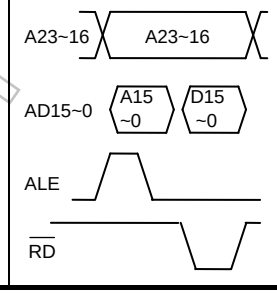
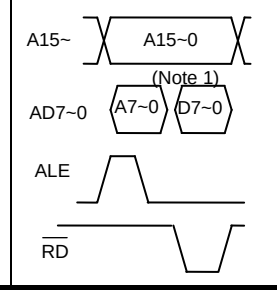
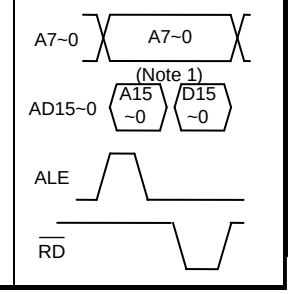
Table 8-1 Bus Mode, Address and Data Pins

	Separate BUSMD="L"	Multiplex BUSMD="H"
Port 0（P00-P07）	D0-D7	AD0-AD7
Port 1（P10-P17）	D8-D15	AD8-AD15／A8-A15
Port 2（P20-P27）	A16-A23	A0-A7／A16-A23
Port 5（P50-P57）	A0-A7	General-purpose port
Port 6（P60-P67）	A8-A15	General-purpose port
Port 37（P37）	General-purpose port	ALE

Each port is put into input mode after a reset. To access an external device, set the address and data bus functions by using the port control register (PnCR) and the port function register (PnFC).

In the multiplex mode, the four types of functions can be selected, as shown in Table 8-2, by setting the port registers (PnCR and PnFCx).

Table 8-2 Address and Data Pins in the Multiplex Mode

		(1)	(2)	(3)	(4)
Number of address buses		max.24 (~16MB)	max.24 (~16MB)	max.16 (~64KB)	max.8 (~256B)
Number of data buses		8	16	8	16
Number of address/data multiplexed buses		8	16	0	0
Port function	Port 0	AD0~AD7	AD0~AD7	AD0~AD7	AD0~AD7
	Port 1	A8~A15	AD8~AD15	A8~A15	AD8~AD15
	Port 2	A16~A23	A16~A23	A0~A7	A0~A7
Timing Diagram					

(Note 1): Even in cases of (3) and (4), address outputs are available as the data bus pins are also used for address buses.

(Note 2): Ports 0 to 2 are put into input modes after a reset, and they do not serve as address or data bus pins.

(Note 3): Any of (1) to (4) can be selected by setting the P1CR, P1FC, P2CR and P2FC registers.

(2) Address HOLD when an internal area is accessed

When an internal area is being accessed, the address bus maintains the address output of the previously accessed external area and doesn't change it. Also, the data bus is in a state of high impedance.

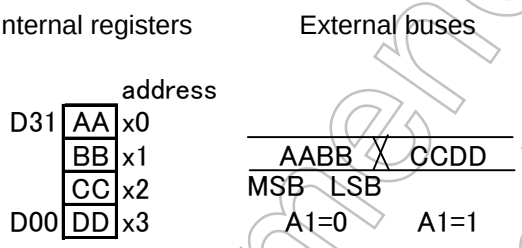
8.2 Data Format

Internal registers and external bus interfaces of the TMP19A23 are configured as described below.

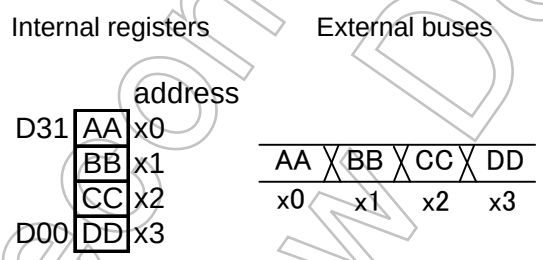
(1) Big-endian mode

① Word access

- 16-bit bus width

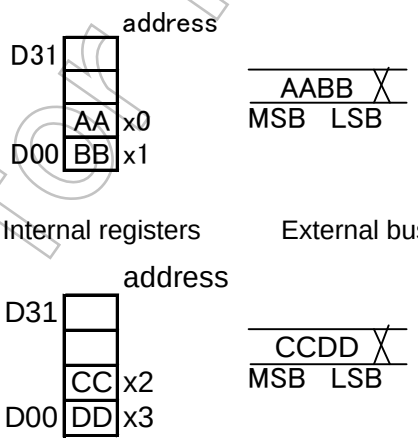


- 8-bit bus width

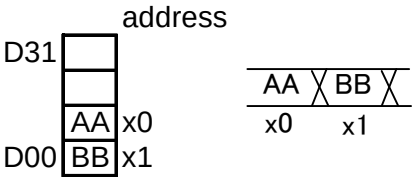


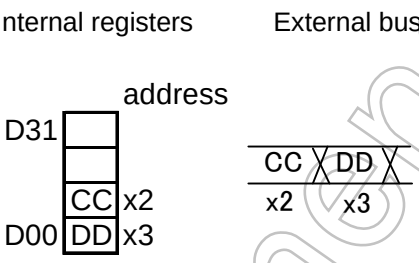
② Half word access

- 16-bit bus width



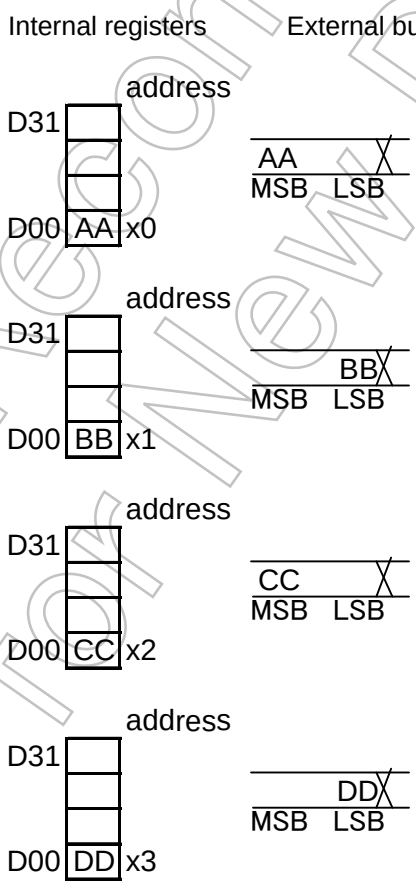
- 8-bit bus width



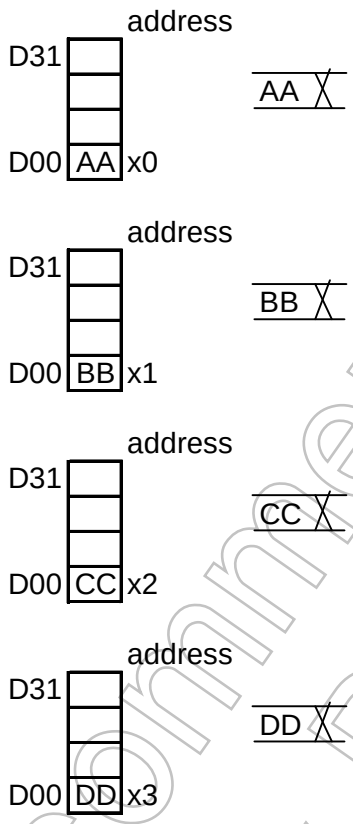


③ Byte access

- 16-bit bus width



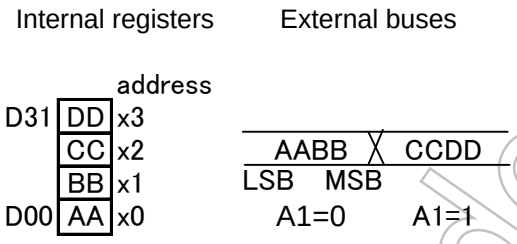
- 8-bit bus width



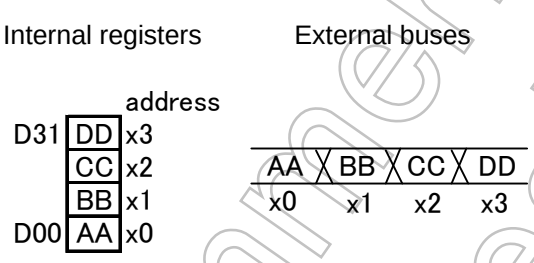
(2) Little-endian mode

① Word access

- 16-bit bus width

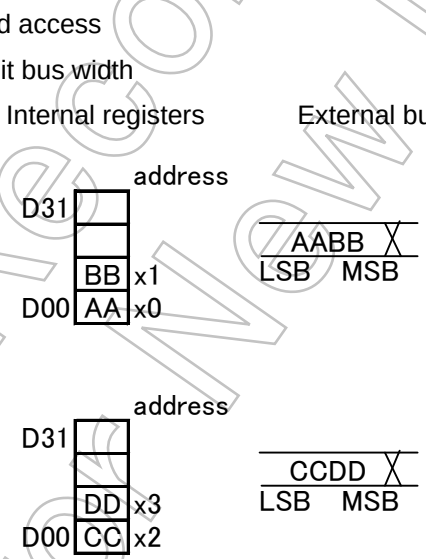


- 8-bit bus width

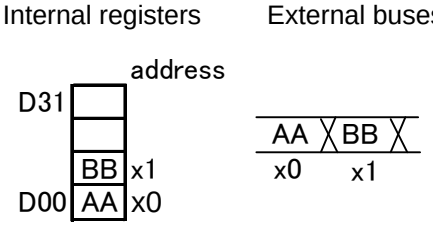


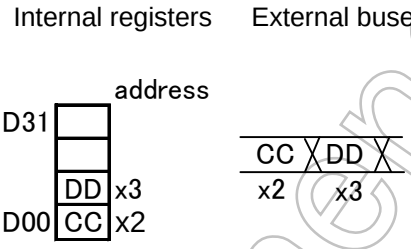
② Half word access

- 16-bit bus width



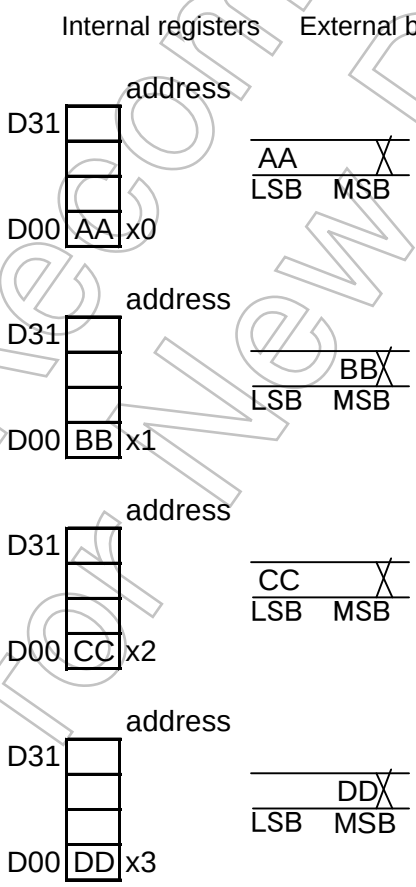
- 8-bit bus width



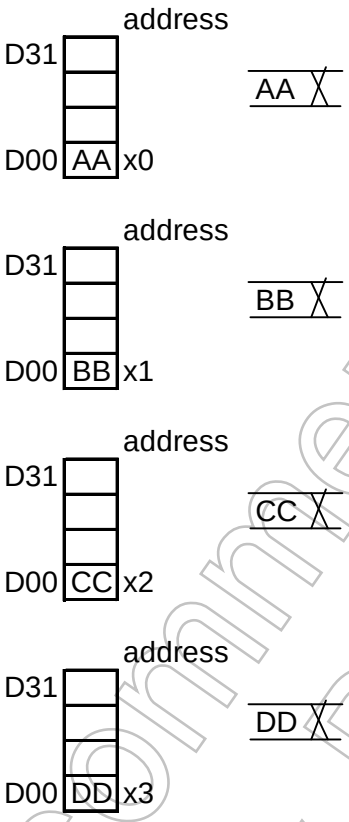


③ Byte access

- 16-bit bus width



- 8-bit bus width



8.3 External Bus Operations (Separate Bus Mode)

This section describes various bus timing values. The timing diagram shown below assumes that the address buses are A23 through A0 and that the data buses are D15 through D0.

(1) Basic bus operation

The external bus cycle of the TMP19A23 basically consists of three clock pulses and a wait can be inserted as mentioned later. The basic clock of an external bus cycle is the same as the internal system clock.

Fig. 8-1 shows read bus timing and Fig. 8-2 shows write bus timing. If internal areas are accessed, address buses remain unchanged as shown in these figures. Additionally, data buses are in a state of high impedance and control signals such as $\overline{RD}$ and $\overline{WR}$ do not become active.

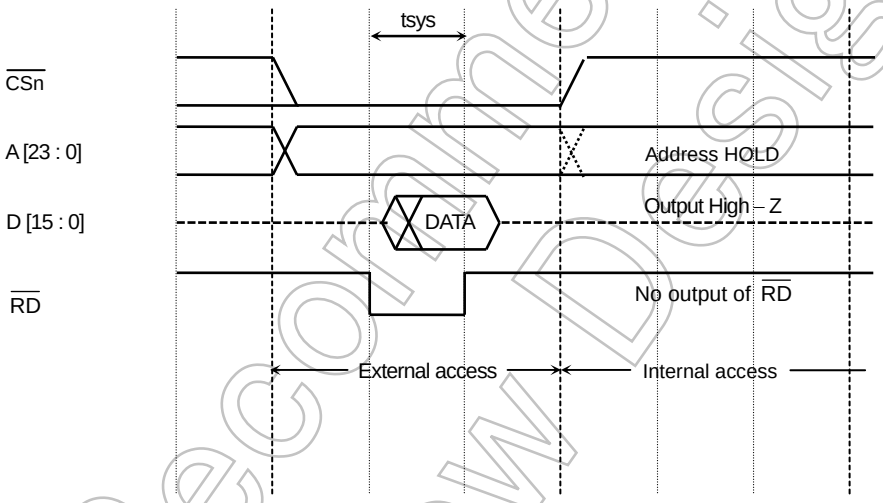


Fig. 8-1 Read Operation Timing Diagram

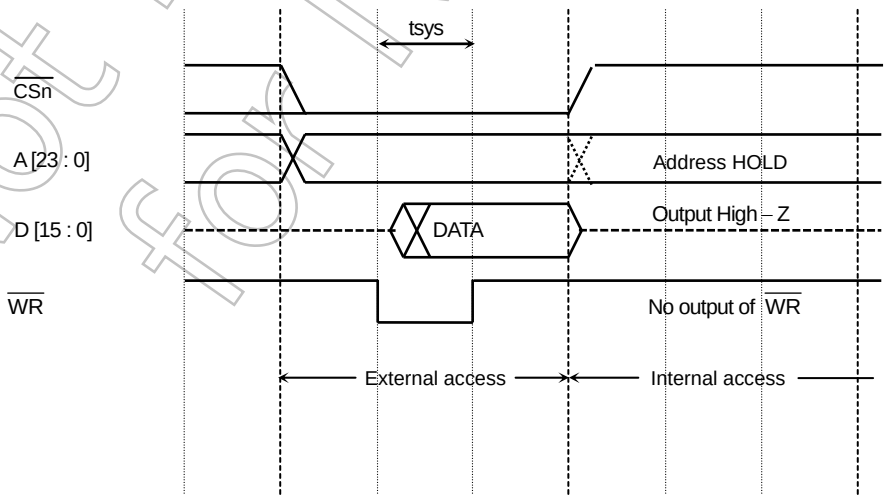


Fig. 8-2 Write Operation Timing Diagram

(2) Wait timing

A wait cycle can be inserted for each block by using the chip selector (CS) and wait controller.

The following three types of wait can be inserted:

① A wait of up to 7 clocks can be automatically inserted.

② A wait can be inserted via the $\overline{\text{WAIT}}$ pin (2+2N, 3+2N, 4+2N, 5+2N, 6+2N, 7+2N). Note: 2N is the number of external waits that can be inserted.

③ A wait can be inserted via the $\overline{\text{RDY}}$ pin (2+2N, 3+2N, 4+2N, 5+2N, 6+2N, 7+2N). Note: 2N is the number of external waits that can be inserted.

The setting of the number of waits to be automatically inserted and the setting of the external wait input can be made using the chip selector and wait controller registers, BmnCS<BnW>.

Fig. 8-3 through Fig. 8-102 show the timing diagrams in which waits have been inserted.

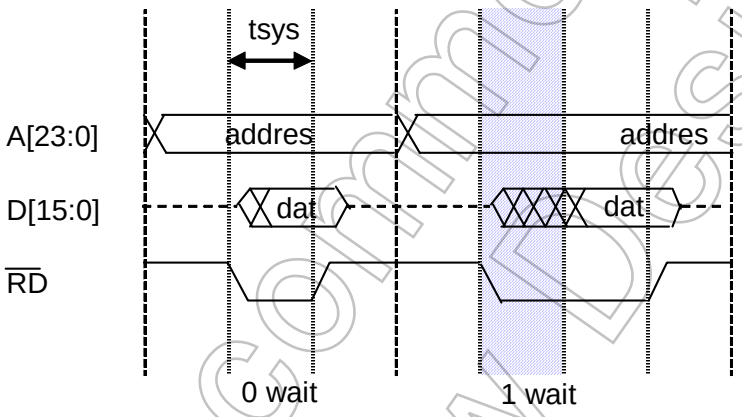


Fig. 8-3 Read Operation Timing Diagram (0 Wait and 1 Wait Automatically Inserted)

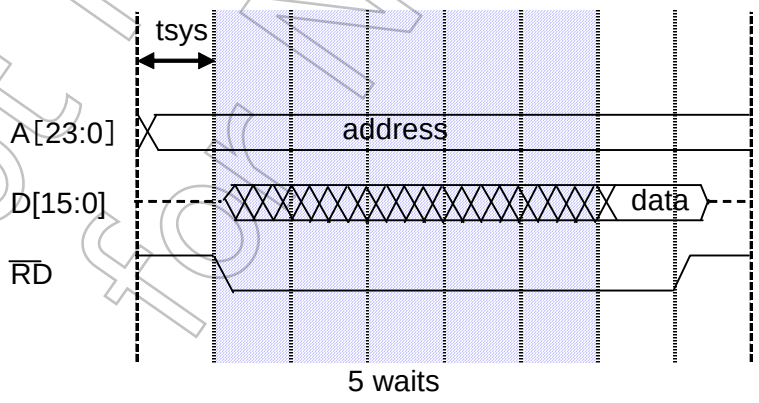
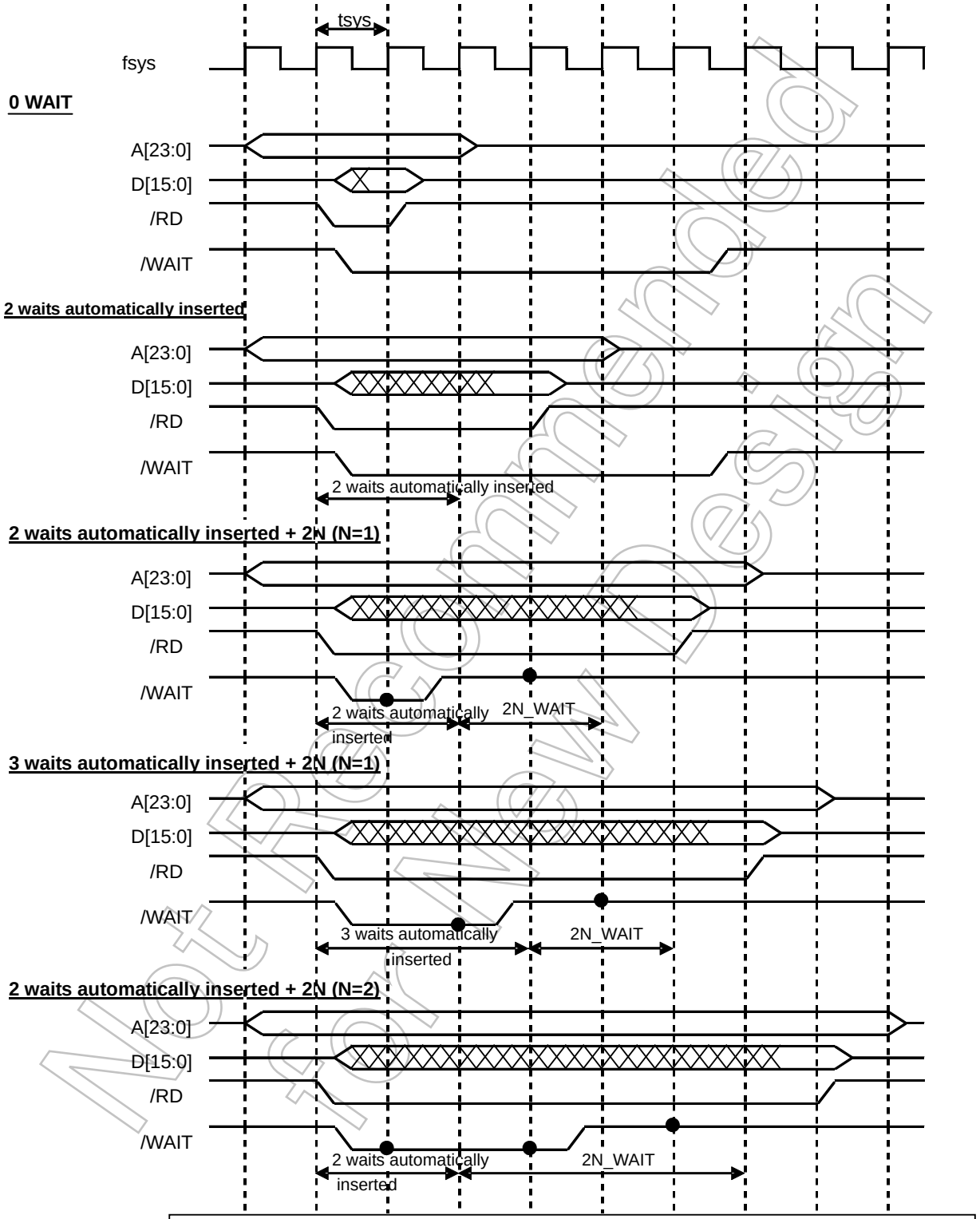


Fig. 8-4 Read Operation Timing Diagram (5 Waits Automatically Inserted)

Fig. 8-5 shows the read operation timing when 0 wait, waits automatically inserted, and waits automatically inserted + external waits are inserted in the separate bus mode.



● --- External wait sampling point

External wait sampling points take place before a cycle of waits automatically inserted is finished and before a 2N_wait cycle is finished as shown above. The same applies to combinations of other numbers of waits.

Fig. 8-5 Read Operation Timing Diagram

Fig. 8-6 shows the write operation timing when 0 wait, waits automatically inserted, and waits automatically inserted + external waits are inserted in the separate bus mode.

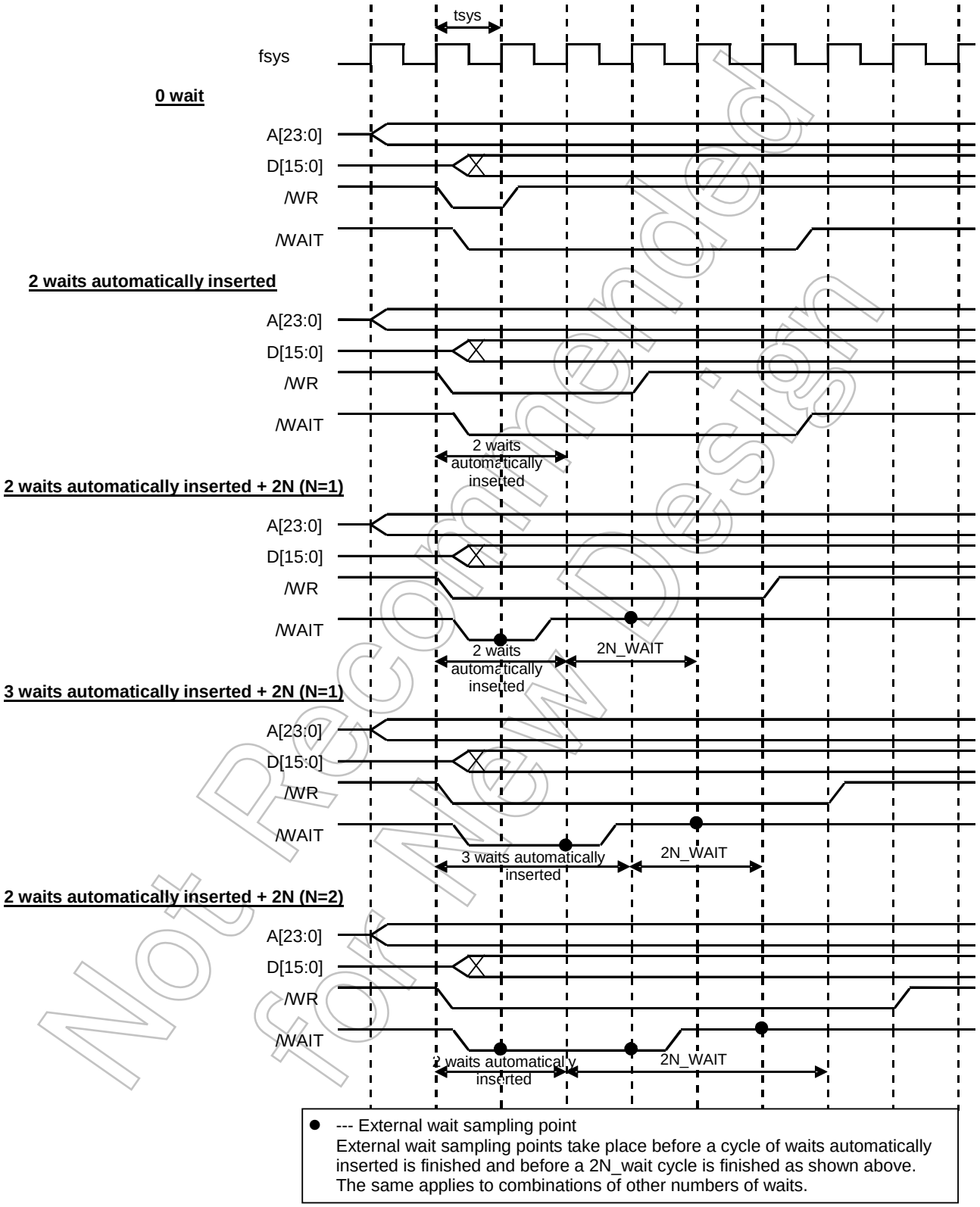


Fig. 8-6 Write Operation Timing Diagram

By setting the bit 3<P33F> of port 3 function register P3FC to "1," the $\overline{\text{WAIT}}$ input pin (P33) can also serve as the $\overline{\text{RDY}}$ input pin.

The $\overline{\text{RDY}}$ input is input to the external bus interface circuit as the logical reverse of the $\overline{\text{WAIT}}$ input. The number of waits is specified by the chip selector and a wait controller register, BmnCS<BnW>.

Fig. 8-7 shows the $\overline{\text{RDY}}$ inputs and the number of waits.

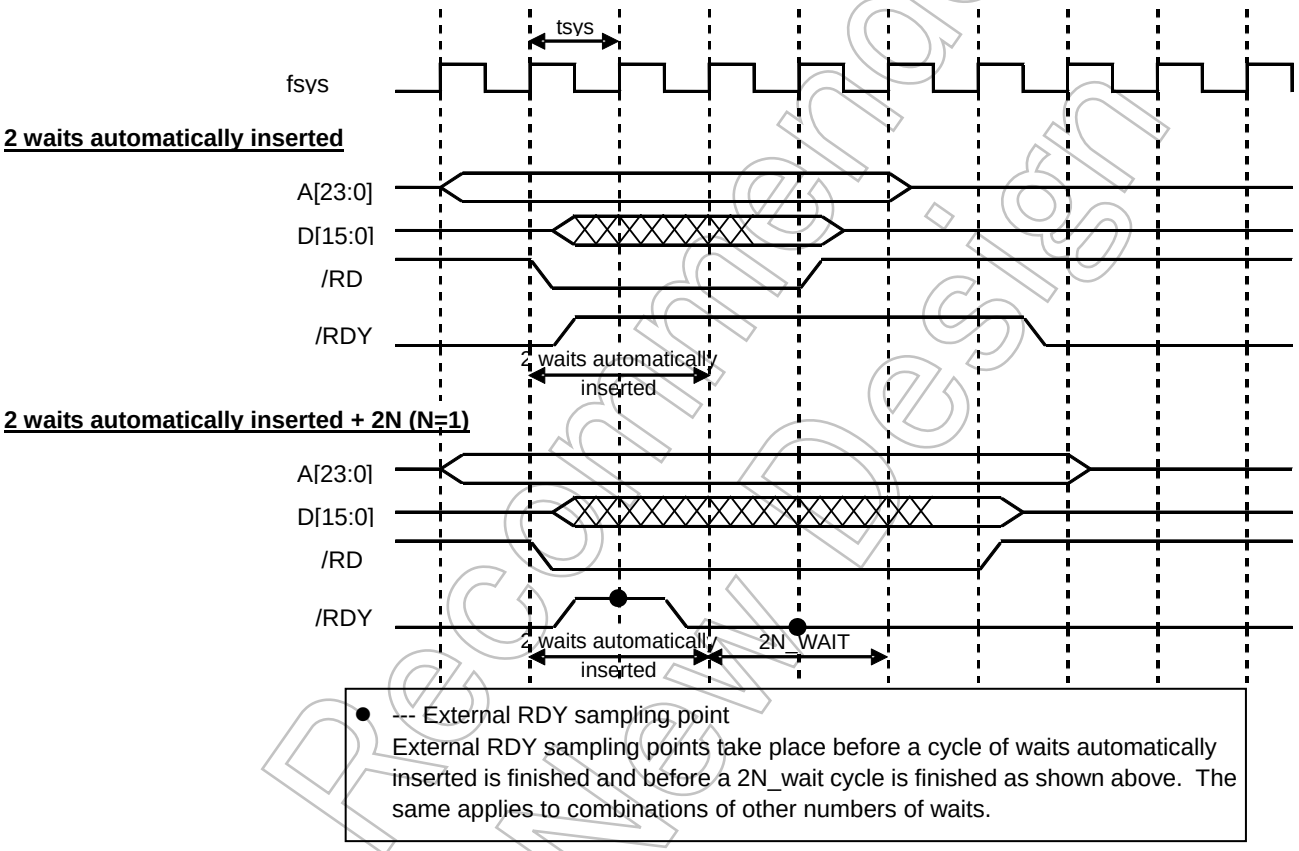


Fig. 8-7 $\overline{\text{RDY}}$ Input and Wait Operation Timing Diagram

(3) Time that it takes before ALE is asserted

When the external bus of the TMP19A23 is used as a multiplexed bus, the ALE width (assert time) can be specified by using the system control register SYSCR3 <ALESEL> in the CG. In the case of a separate bus mode, ALE is not output, but the time from when an address is established to the assertion of the $\overline{RD}$ or $\overline{WR}$ signal is different depending on the SYSCR3<ALESEL>.

During a reset, <ALESEL> = "1" is set and the $\overline{RD}$ or $\overline{WR}$ signal is asserted at a point of two system (internal) clocks after an address is established. If <ALESEL> is cleared to "0," the $\overline{RD}$ or $\overline{WR}$ signal is asserted at a point of one system (internal) clock after an address is established. This assert setting cannot be established for each block in an external area and the same setting is commonly used in an external address space.

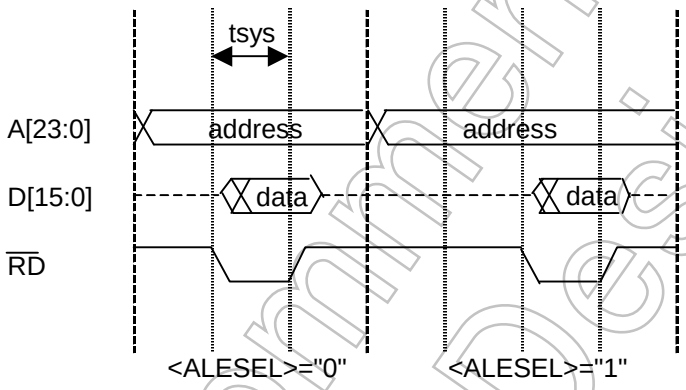


Fig. 8-8 SYSCR3 <ALESEL> Set Value and External Bus Operation

(4) Recovery time

If access to external areas occurs consecutively, a dummy cycle can be inserted for recovery time.

A dummy cycle can be inserted in both a read and a write cycle. The dummy cycle insertion setting can be made in the chip selector and wait controller registers, BmnCS<BnWCV> (write recovery cycle) and <BnRCV> (read recovery cycle). As for the number of dummy cycles, one or two system clocks (internal) can be specified for each block. Fig. 8-9 shows the timing of recovery time insertion.

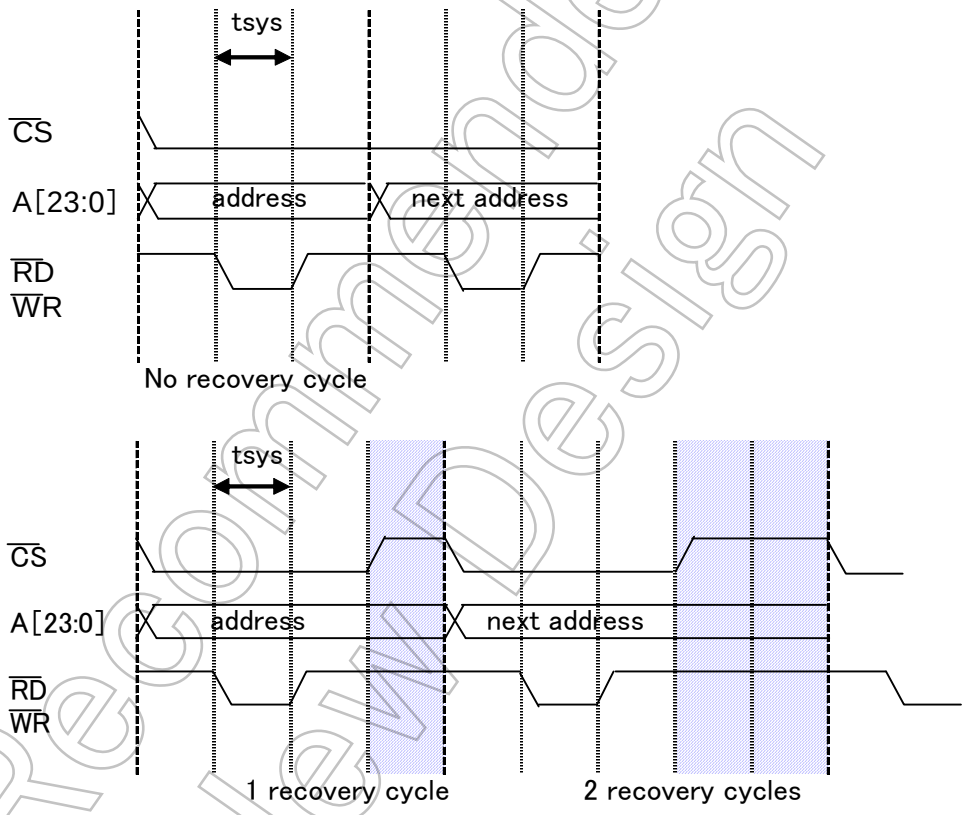


Fig. 8-9 Timing of Recovery Time Insertion

(5) Chip selector recovery time

If access to external areas occurs consecutively, a dummy cycle can be inserted for recovery time.

The dummy cycle insertion setting can be made in the chip selector and wait controller registers, BmnCS<BnCSCV>. As for the number of dummy cycles, one, two or three system clock(s) (internal) can be specified for each block. Fig. 8-12 shows the timing of recovery time insertion.

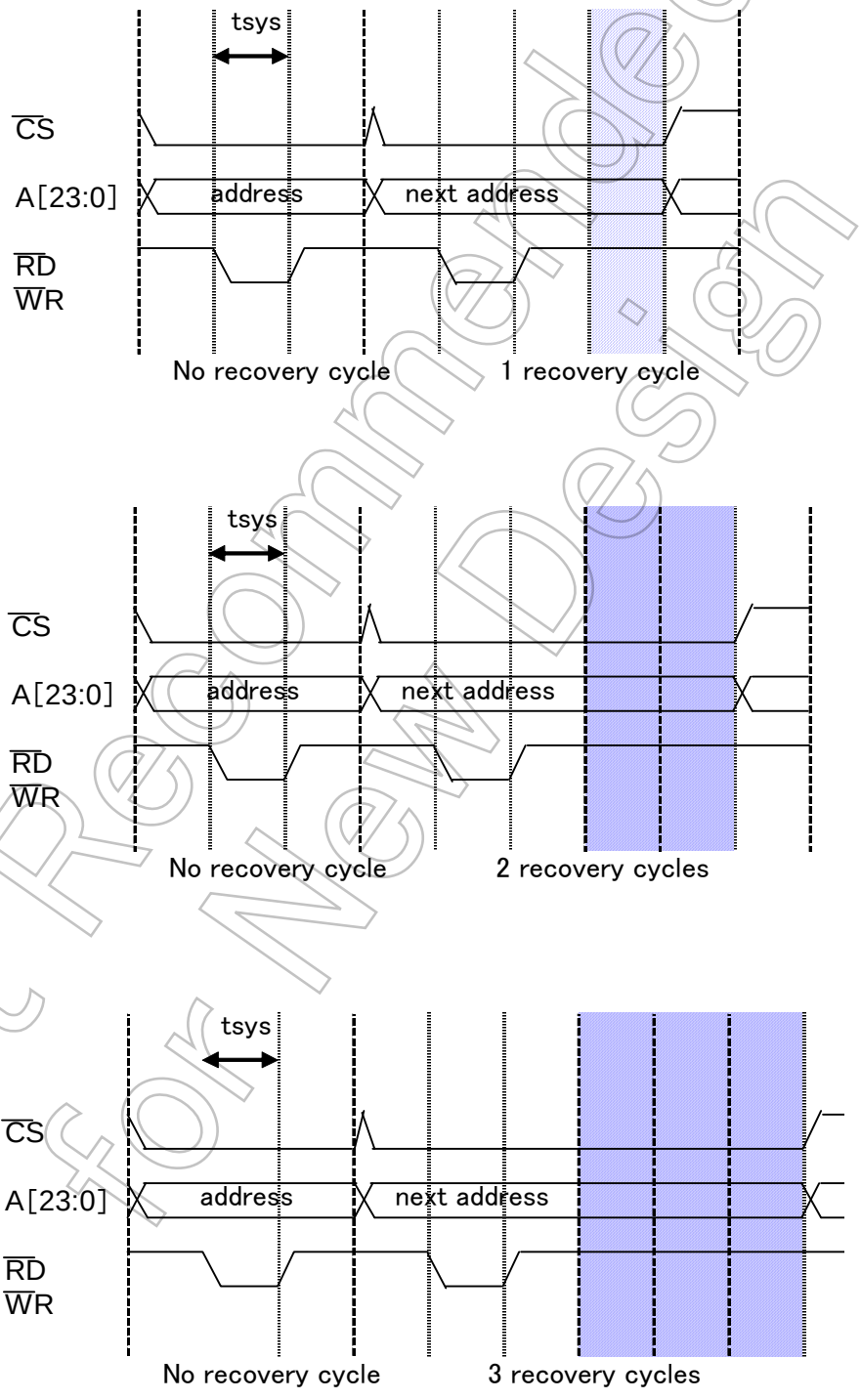


Fig. 8-12 Timing of Chip Selector Recovery Time Insertion

8.4 External Bus Operations (Multiplexed Bus Mode)

This section describes various bus timing values. The timing diagram shown below assumes that the address buses are A23 through A16 and that the address/data buses are AD15 through AD0.

(1) Basic bus operation

The external bus cycle of the TMP19A23 basically consists of three clock pulses and a wait can be inserted as mentioned later. The basic clock of an external bus cycle is the same as the internal system clock.

Fig. 8-13 shows read bus timing and Fig. 8-14 shows write bus timing. If internal areas are accessed, address buses remain unchanged and the ALE does not output latch pulse as shown in these figures. Additionally, address/data buses are in a state of high impedance and control signals such as $\overline{RD}$ and $\overline{WR}$ do not become active.

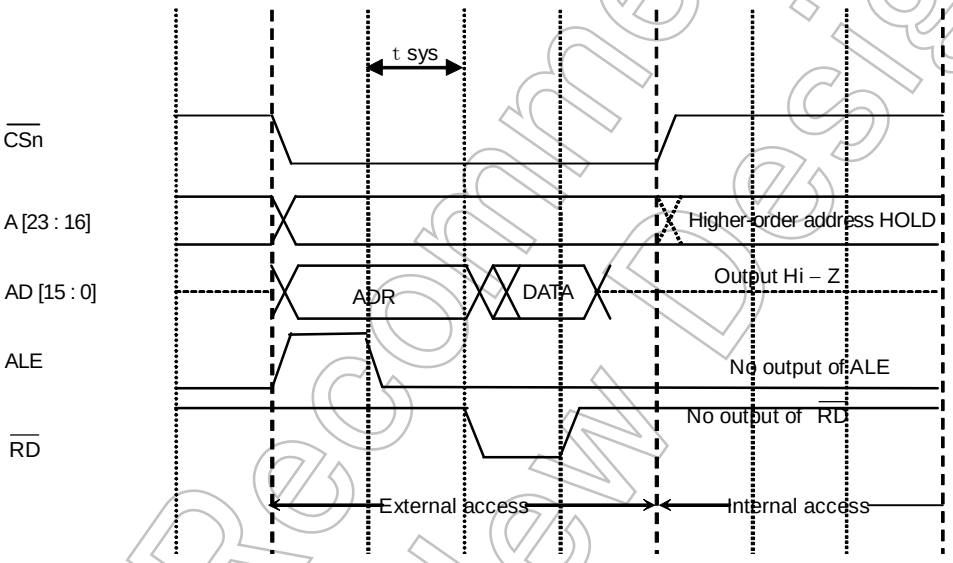


Fig. 8-13 Read Operation Timing Diagram

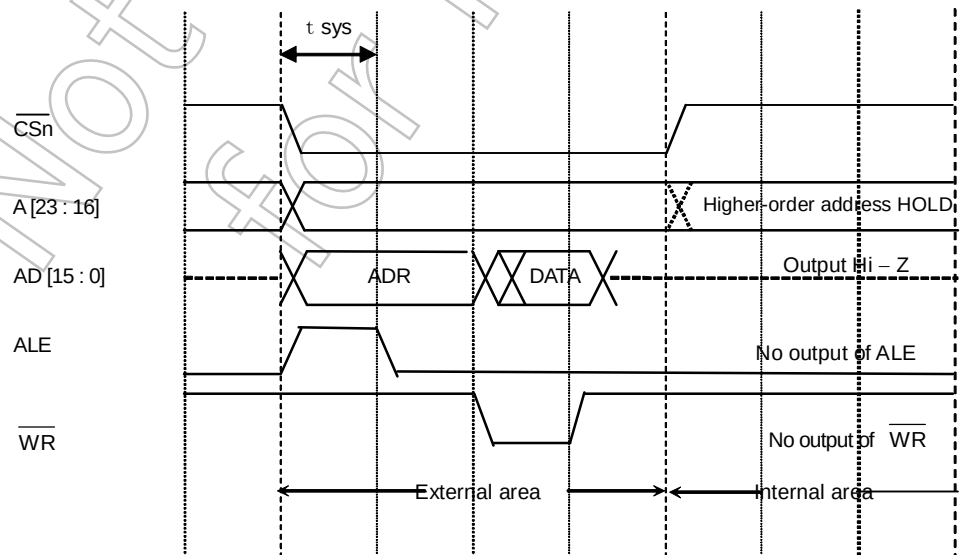


Fig. 8-14 Write Operation Timing Diagram

(2) Wait Timing

A wait cycle can be inserted for each block by using the chip selector (CS) and wait controller.

The following three types of wait can be inserted:

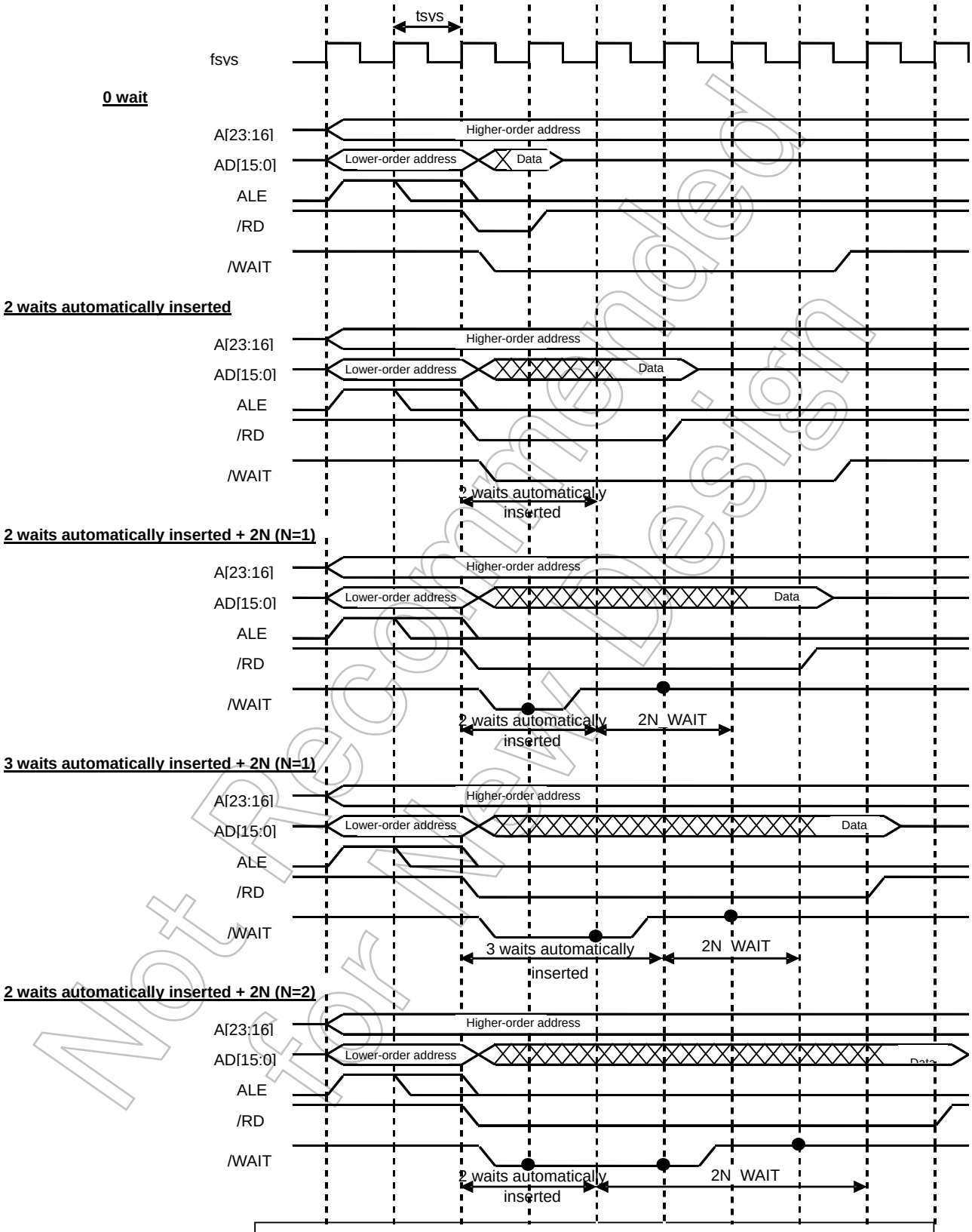
① A wait of up to 7 clocks can be automatically inserted.

② A wait can be inserted via the $\overline{\text{WAIT}}$ pin (2+2N, 3+2N, 4+2N, 5+2N, 6+2N, 7+2N).
Note: 2N is the number of external waits that can be inserted.

③ A wait can be inserted via the $\overline{\text{RDY}}$ pin (2+2N, 3+2N, 4+2N, 5+2N, 6+2N, 7+2N).
Note: 2N is the number of external waits that can be inserted.

The setting of the number of waits to be automatically inserted and the setting of the external wait input can be made using the chip selector and wait controller registers, BmnCS<BnW>.

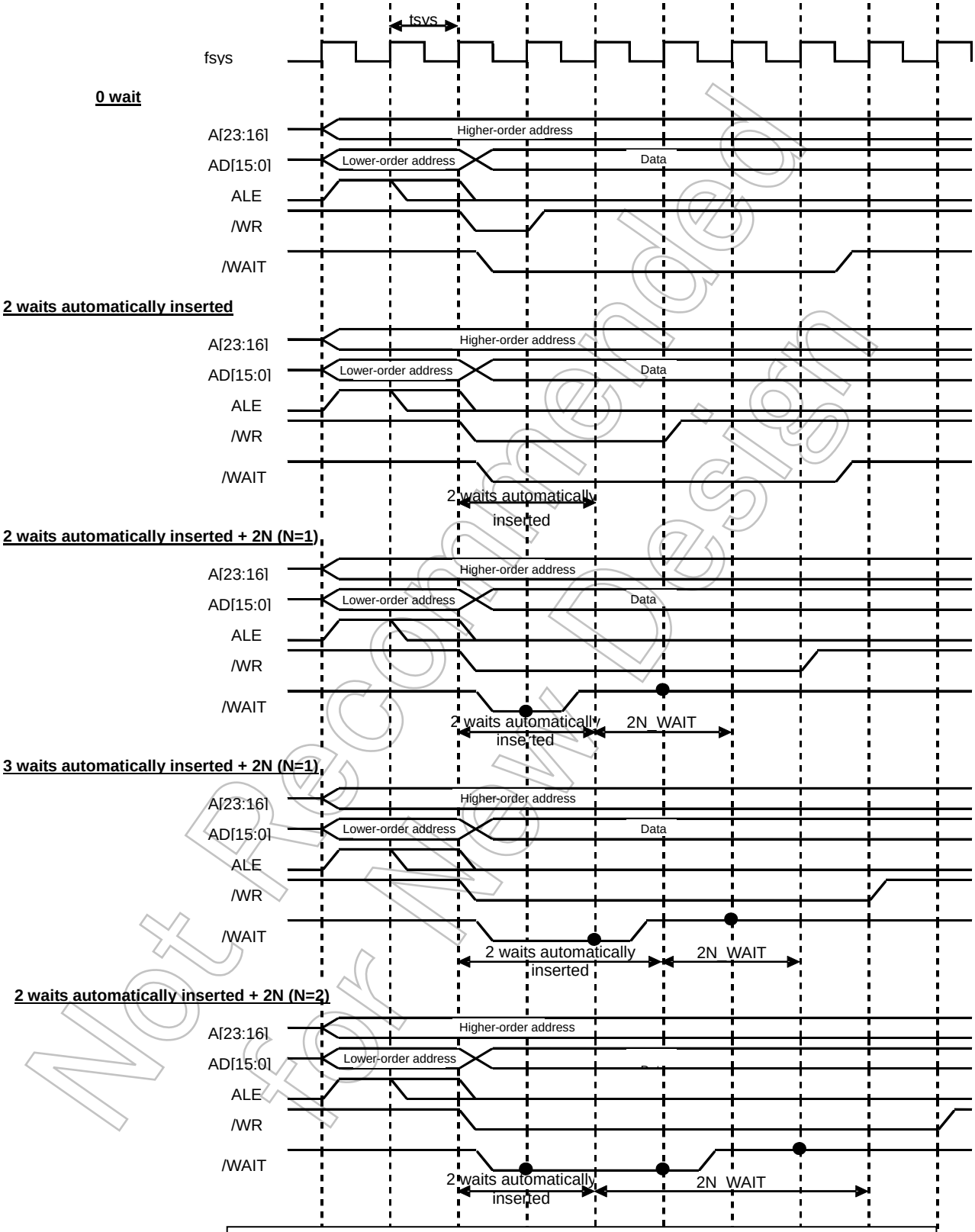
Fig. 8-15 shows the read operation timing when 0 wait, waits automatically inserted, and waits automatically inserted + external waits are inserted in the multiplexed bus mode.



- --- External wait sampling point
  External wait sampling points take place before a cycle of waits automatically inserted is finished and before a 2N_wait cycle is finished as shown above. The same applies to combinations of other numbers of waits.

Fig. 8-15 Read Operation Timing Diagram

Fig. 8-16 shows the write operation timing when 0 wait, waits automatically inserted, and waits automatically inserted + external waits are inserted in the multiplexed bus mode.



● --- External wait sampling point
External wait sampling points take place before a cycle of waits automatically inserted is finished and before a 2N_wait cycle is finished as shown above. The same applies to combinations of other numbers of waits.

Fig. 8-16 Write Operation Timing Diagram

(3) Time for ALE to be asserted

As an ALE assertion time, either 1 clock or 2 clocks can be selected. The setting bit is located in the system clock control register. The default is 2 clocks. This assert setting cannot be established for each block in an external area and the same setting is commonly used in an external address space.

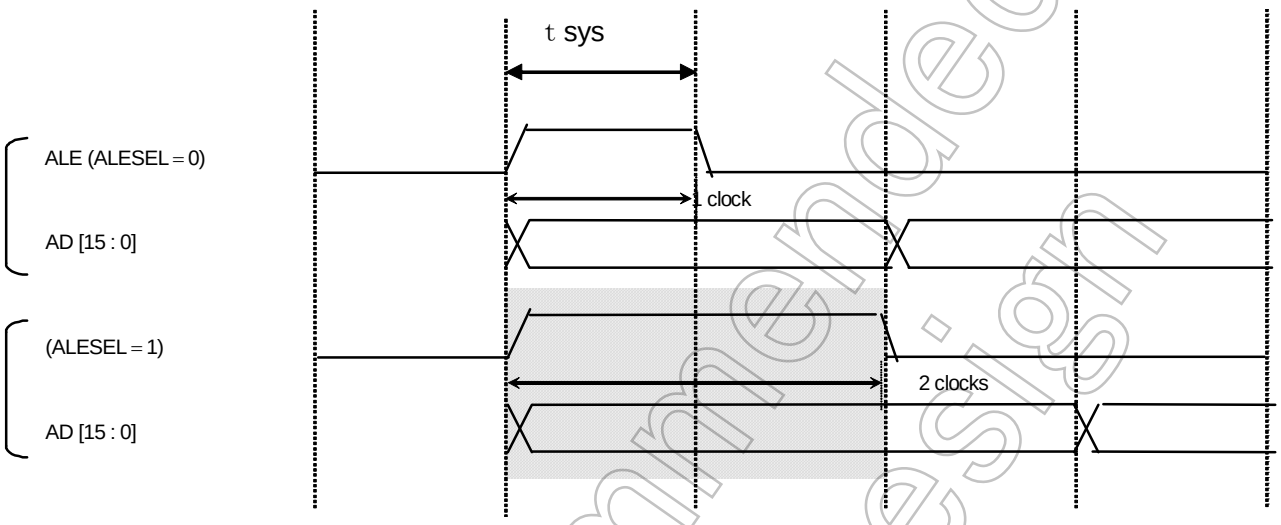


Fig. 8-17 Time for ALE to be asserted

Fig. 8-18 shows the ALE timings with 1 clock or 2 clocks.

The ALE timings with 1 clock or 2 clocks

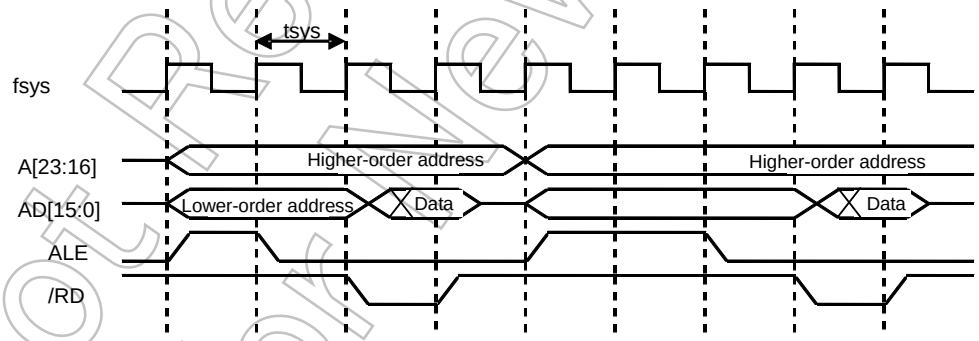


Fig. 8-18 Read Operation Timing (the ALE timings with 1 clock or 2 clocks)

(4) Read and Write Recovery Time

If access to external areas occurs consecutively, a dummy cycle can be inserted for recovery time.

A dummy cycle can be inserted in both a read and a write cycle. The dummy cycle insertion setting can be made in the chip selector and wait controller registers, BmnCS<BnWCV> (write recovery cycle) and <BnRCV> (read recovery cycle). As for the number of dummy cycles, one or two system clocks (internal) can be specified for each block. Fig. 8-17 shows the timing of recovery time insertion.

When read/write recovery is inserted (ALE width:1fsys)

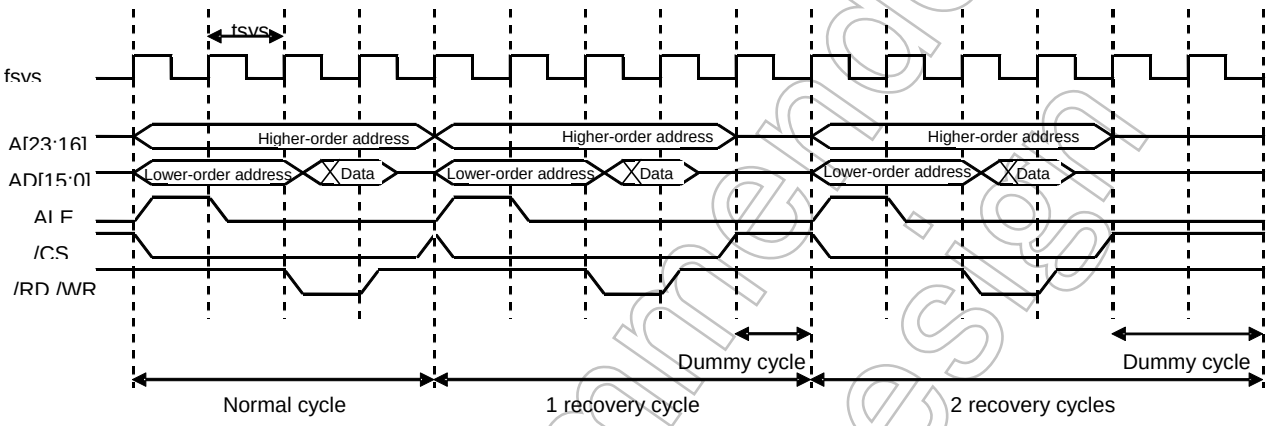


Fig. 8-19 Timing of Recovery Time Insertion

(5) Chip selector recovery time

If access to external areas occurs consecutively, a dummy cycle can be inserted for recovery time.

The dummy cycle insertion setting can be made in the chip selector and wait controller registers, BmnCS<BnCSCV>. The number of dummy cycles can be specified by one, two or three internal system clocks for each block. Fig. 8-20 shows the timing of recovery time insertion.

When chip selector recovery is inserted (ALE width:1fsys)

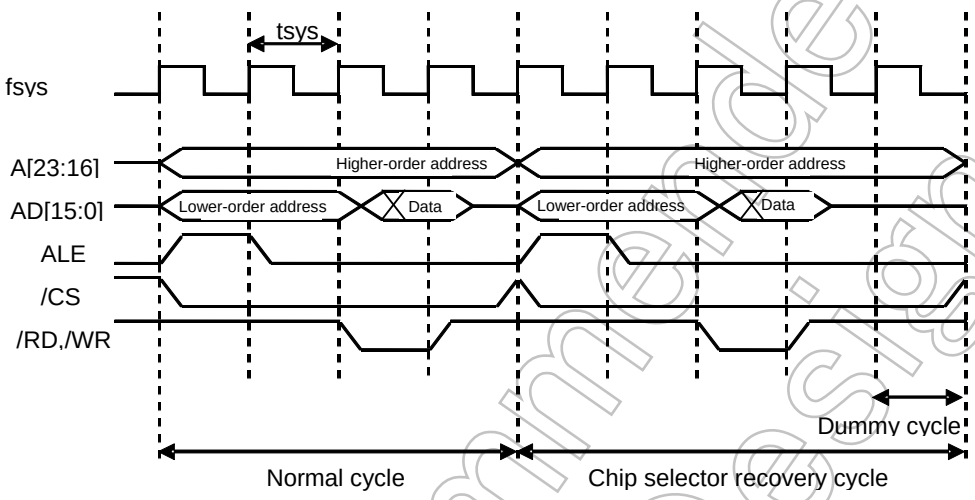


Fig. 8-20 Timing of Chip Selector Recovery Time Insertion

8.5 Bus Arbitration

The TMP19A23 can be connected to an external bus master. The arbitration of bus control authority with the external bus master is executed by using the two signals, $\overline{\text{BUSRQ}}$ and $\overline{\text{BUSAK}}$. The external bus master can acquire control authority for TMP19A23 external buses only, and cannot acquire control authority for internal buses.

(1) Accessible range of external bus master

The external bus master can acquire control authority only for TMP19A23 external buses, and cannot acquire control authority for internal buses (G-BUS). Therefore, the external bus master cannot access the internal memories or the internal I/O. The arbitration of bus control authority for external buses is executed by the external bus interface circuit (EBIF), and this is independent of the CPU and the internal DMAC. Even when the external bus master holds the external bus control authority, the CPU and the internal DMAC can access the internal ROM, RAM and registers. On the other hand, if the CPU or the internal DMAC tries to access an external memory when the external bus master holds the external bus control authority, the CPU or the internal DMAC bus cycle has to wait until the external bus master releases the bus. For this reason, if the $\overline{\text{BUSRQ}}$ remains active, the TMP19A23 can lock.

(2) Acquisition of bus control authority

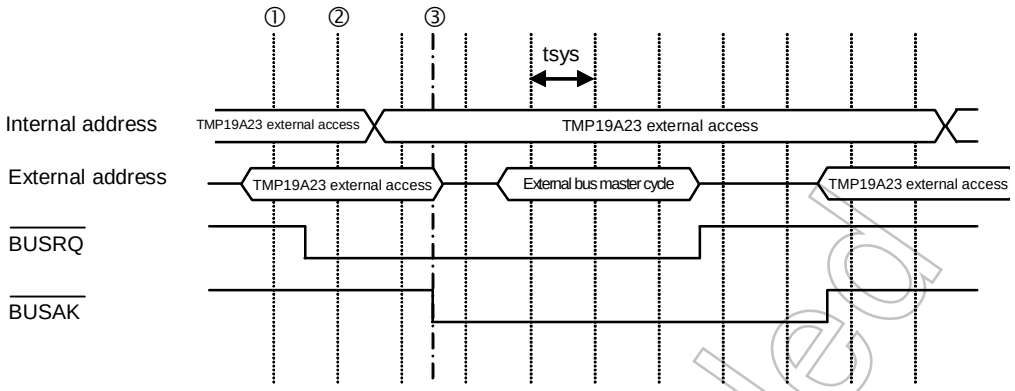
The external bus master requests the TMP19A23 for bus control authority by asserting the $\overline{\text{BUSRQ}}$ signal. The TMP19A23 samples the $\overline{\text{BUSRQ}}$ signal at the break of external bus cycles on the internal buses (G-BUS) and determines whether or not to give the bus control authority to the external bus master. When it gives the bus control authority to the external bus master, it asserts the $\overline{\text{BUSAK}}$ signal. At the same time, it makes address buses, data buses and bus control signals ($\overline{\text{RD}}$ and $\overline{\text{WR}}$) in a state of high impedance. (The internal pull up is enabled for the R/$\overline{\text{W}}$, $\overline{\text{HWR}}$ and $\overline{\text{CSx}}$.)

Depending on the relationship between the size of data to be loaded or stored and the external memory bus width, two or more bus cycles can occur in response to a single data transfer (bus sizing). In this case, the end of the last bus cycle is the break of external bus cycles.

If access to external areas occurs consecutively on the TMP19A23, a dummy cycle can be inserted. Again, requests for buses are accepted at the break of external bus cycles on the internal buses (G-BUS). During a dummy cycle, the next external bus cycle is already started on the internal buses. Therefore, even if the $\overline{\text{BUSRQ}}$ signal is asserted during a dummy cycle, the bus is not released until the next external bus cycle is completed.

Keep asserting the $\overline{\text{BUSRQ}}$ signal until the bus control authority is released.

Fig. 8-21 shows the timing of acquiring bus control authority by the external bus master.



① $\overline{\text{BUSRQ}}$ is at the "H" level.

② The TMP19A23 recognizes that the $\overline{\text{BUSRQ}}$ is at the "L" level, and releases the bus at the end of the bus cycle.

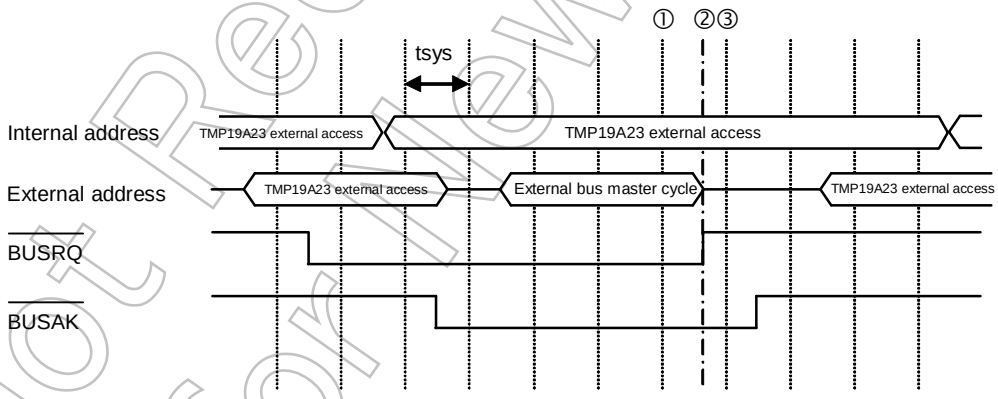
③ When the bus is completed, the TMP19A23 asserts $\overline{\text{BUSAK}}$. The external bus master recognizes that the $\overline{\text{BUSAK}}$ is at the "L" level, and acquires the bus control authority to start bus operations.

Fig. 8-21 Bus Control Authority Acquisition Timing

(3) Release of bus control authority

The external bus master releases the bus control authority when it becomes unnecessary. If the external bus master no longer needs the bus control authority having been obtained already, it negates the $\overline{\text{BUSRQ}}$ signal and returns the bus control authority to the TMP19A23.

Fig. 8-22 shows the timing of releasing unnecessary bus control authority.



① The external bus master has the bus control authority.

② The external bus master negates the $\overline{\text{BUSRQ}}$, as it no longer requires the bus control authority.

③ The TMP19A23 recognizes that the $\overline{\text{BUSRQ}}$ is at the "H" level, and deasserts the $\overline{\text{BUSAK}}$.

Fig. 8-22 Timing of Releasing Bus Control Authority

9. The Chip Selector and Wait Controller

The TMP19A23 can be connected to external devices (I/O devices, ROM and SRAM).

4-block address spaces ($\overline{CS0}$ through $\overline{CS3}$) can be established in the TMP19A23 and three parameters can be specified for each 4-block address and other address spaces: data bus width, the number of waits and the number of dummy cycles.

$\overline{CS0}$ through $\overline{CS3}$ (also used as P40 through P43) are the output pins corresponding to spaces $\overline{CS0}$ through $\overline{CS3}$. These pins generate chip selector signals (for ROM and SRAM) to each space when the CPU designates an address in which spaces $\overline{CS0}$ through $\overline{CS3}$ are selected. For chip selector signals to be generated, however, the port 4 controller register (P4CR) and the port 4 function register (P4FC) must be set appropriately.

The specification of the spaces $\overline{CS0}$ through $\overline{CS3}$ is to be performed with a combination of base addresses (BAn, n=0 to 3) and mask addresses (MAn, n=0 to 3) using the base and mask address setting registers (BMA0 through BMA3).

Meanwhile, master enable, data bus width, the number of waits and the number of dummy cycles for each address space are specified in the chip selector and wait controller registers (B01CS and B23CS).

A bus wait request pin ($\overline{WAIT}$) is provided as an input pin to control the status of these settings.

9.1 Specifying Address Spaces

Spaces $\overline{CS0}$ through $\overline{CS3}$ are specified using the base and mask address setting registers (BMA0 through BMA3).

In each bus cycle, a comparison is made to see if each address on the bus is located in the space $\overline{CS0}$ through $\overline{CS3}$. If the result of a comparison is a match, it is considered that the designated CS space has been accessed. Then chip selector signals are output from pins $\overline{CS0}$ through $\overline{CS3}$. The operations specified by the chip selector and wait controller registers (B01CS and B23CS) are executed (refer to "0 The Chip Selector and Wait Controller Register").

9.1.1 Base and Mask Address Setting Registers

Fig. 9-1 and Fig. 9-2 show base and mask address setting registers. For base addresses (BA0 through BA3), a start address in the space $\overline{CS0}$ through $\overline{CS3}$ is specified. In each bus cycle, the chip selector and wait controller compare values in their registers with addresses and those addresses with address bits masked by the mask address (MA0 through MA3) are not compared. The size of an address space is determined by the mask address setting.

(1) Base addresses

Base address BAn specifies the higher-order 16 bits (A31 through A16) of the start address. The lower-order 16 bits (A15 to A0) of the start address are always set to "0." Therefore, the start address begins with 0x0000_0000 and increases in 64 kilobyte units.

Fig. 9-3 shows the relationship between the start address and the BAn value.

(2) Mask addresses

Mask address (MAn) specifies which address bit value is to be compared. The address on the bus that corresponds to the bit for which "0" is written on the address mask MAn is to be included in address comparison to determine if the address is in the area of the $\overline{CS0}$ to $\overline{CS3}$ spaces. The bit in which "1" is written is not included in address comparison.

$\overline{CS0}$ to $\overline{CS3}$ spaces have different address bits that can be masked by MA0 to MA3.

$\overline{CS0}$ space and $\overline{CS1}$ space: A29 through A14

$\overline{CS2}$ space and $\overline{CS3}$ space: A30 through A15 (*Note)

(Note)	**Address settings must be made using physical addresses.**

Base and mask address setting registers BMA0 (0xFFFF_E400) to BMA3 (0xFFFF_E40C)

BMA0 (0xFFFF_E400)

	31	30	29	28	27	26	25	24
Bit symbol	BA0							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	A31 to A24 to be set as a start address.							
	23	22	21	20	19	18	17	16
Bit symbol	BA0							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	A23 to A16 to be set as a start address.							
	15	14	13	12	11	10	9	8
Bit symbol	MA0							
Read/Write	R/W							
After reset	0	0	0	0	0	0	1	1
Function	Make sure to write “0.”							
	7	6	5	4	3	2	1	0
Bit symbol	MA0							
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	CS0 space size setting 0: Address for comparison							

BMA1 (0xFFFF_E404)

	31	30	29	28	27	26	25	24
Bit symbol	BA1							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	A31 to A24 to be set as a start address.							
	23	22	21	20	19	18	17	16
Bit symbol	BA1							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	A23 to A16 to be set as a start address.							
	15	14	13	12	11	10	9	8
Bit symbol	MA1							
Read/Write	R/W							
After reset	0	0	0	0	0	0	1	1
Function	Make sure to write “0.”							
	7	6	5	4	3	2	1	0
Bit symbol	MA1							
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	CS1 space size setting 0: Address for comparison							

(Note) Make sure to write "0" for bits 10 through 15 for BMA0 and BMA1.
The size of both the CS0 and CS1 spaces can be a minimum of 16 KB to a maximum of 1 GB.
The external address space of the TMP19A23 is 16 MB and so bits 10 through 15 must be set to "0" as addresses A24 through A29 are not masked.

Fig. 9-1 Base and Mask Address Setting Registers (BMA0, BMA1)

BMA2 (0xFFFF_E408)

	31	30	29	28	27	26	25	24
Bit symbol	BA2							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	A31 to A24 to be set as a start address.							
	23	22	21	20	19	18	17	16
Bit symbol	BA2							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	A23 to A16 to be set as a start address.							
	15	14	13	12	11	10	9	8
Bit symbol	MA2							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	1
Function	Make sure to write “0.”							
	7	6	5	4	3	2	1	0
Bit symbol	MA2							
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	CS2 space size setting 0: Address for comparison							

BMA3 (0xFFFF_E40C)

	31	30	29	28	27	26	25	24
Bit symbol	BA3							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	A31 to A24 to be set as a start address.							
	23	22	21	20	19	18	17	16
Bit symbol	BA3							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	A23 to A16 to be set as a start address.							
	15	14	13	12	11	10	9	8
Bit symbol	MA3							
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	1
Function	Make sure to write “0.”							
	7	6	5	4	3	2	1	0
Bit symbol	MA3							
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	CS3 space size setting 0: Address for comparison							

(Note) Make sure to write "0" for bits 9 through 15 for BMA2 and BMA3.
The size of both the CS2 and CS3 spaces can be a minimum of 32 KB to a maximum of 2 GB. The external address space of the TMP19A23 is 16 MB and so bits 9 through 15 must be set to "0" as addresses A24 through A30 are not masked.

Fig. 9-2 Base and Mask Address Setting Registers (BMA2, BMA3)

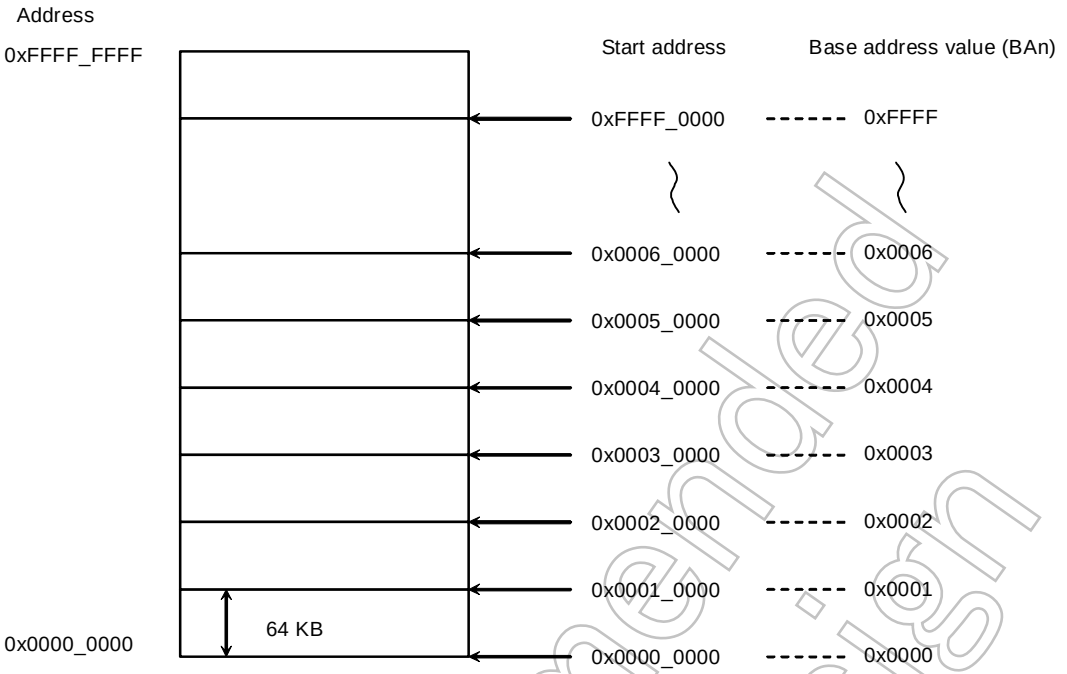
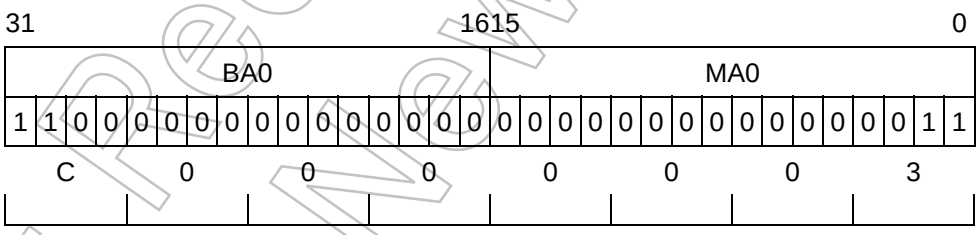


Fig. 9-3 Start and Base Address Register Values

9.1.2 How to Define Start Addresses and Address Spaces

- To specify a space of 64 KB starting at 0xC000_0000 in the $\overline{CS0}$ space, the base and mask address registers must be programmed as shown below.

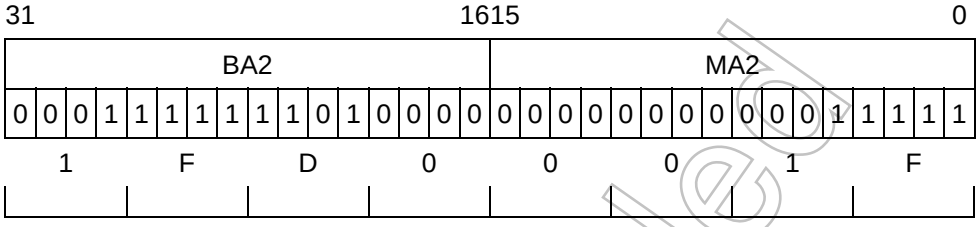


Values to be set in the base and mask address registers (BMA0)

In the base address (BA0), specify "0xC000" that corresponds to higher 16 bits of a start address, while in the mask address (MA0), specify whether a comparison of addresses in the space A29 through A14 is to be made or not. A comparison of A31 and A30 will definitely be made and to ensure a comparison of A29 through A16, set bits 15 to 2 of the mask address (MA0) to "0".

This setting allows A31 through A16 to be compared with the value specified as a start address. Therefore, a space of 64 KB from 0xC000_0000 to 0xC000_FFFF is designated as a $\overline{CS0}$ space and the $\overline{CS0}$ signal is asserted if there is a match with an address on the bus.

- To specify a space of 1 MB starting at 0x1FD0_0000 in the $\overline{CS2}$ space, the base and mask address registers must be programmed as shown below.



Values to be set in the base and mask address registers (BMA2)

In the base address (BA2), specify "0x1FD0" that corresponds to higher 16 bits of a start address, while in the mask address (MA2), specify whether a comparison of addresses in the space A30 through A15 is to be made or not. A comparison of A31 will definitely be made and to ensure a comparison of A30 through A20, set bits 15 to 5 of the mask address (MA2) to "0."

This setting allows A31 through A20 to be compared with the value specified as a start address. As A19 through A0 are masked, a space of 1 MB from 0x1FD0_0000 to 0x1FDF_FFFF is designated as a $\overline{CS2}$ space.

(注) TMP19A23 does not assert $\overline{CSn}$ signal in the following address spaces.
0x1FC0_0000 - 0x1FCF_FFFF
0x4000_0000 - 0x400F_FFFF
0xFFFD_6000 - 0xFFFD_FFFF、0xFFFF_6000 - 0xFFFF_DFFF

After a reset, the $\overline{CS0}$, $\overline{CS1}$ and $\overline{CS3}$ spaces are disabled.

Table 9-1 shows the relationship between CS space and space sizes. If two or more address spaces are specified simultaneously, a space or spaces with a smaller space number will be selected by priority.

Example: 0xC000_0000 as a start address of the CS0 space with a space size of 16 KB
0xC000_0000 as a start address of the CS1 space with a space size of 64 KB

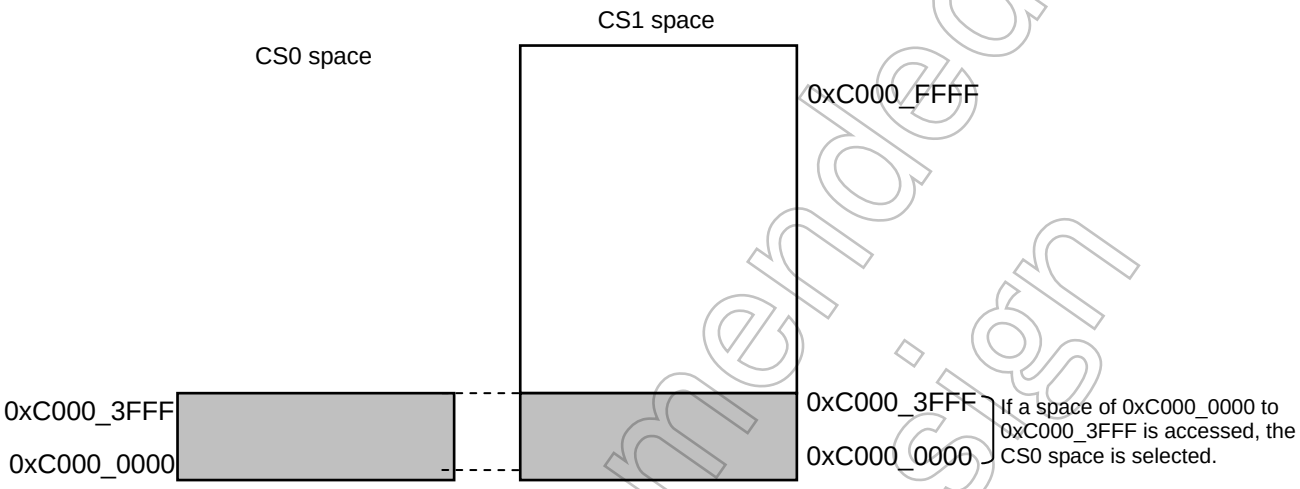


Table 9-1 CS Space and Space Sizes

CS space \ Size (bytes)	16 K	32 K	64 K	128 K	256 K	512 K	1 M	2 M	4 M	8 M	16 M
CS0	○	○	○	○	○	○	○	○	○	○	○
CS1	○	○	○	○	○	○	○	○	○	○	○
CS2		○	○	○	○	○	○	○	○	○	○
CS3		○	○	○	○	○	○	○	○	○	○

9.2 The Chip Selector and Wait Controller

Fig. 9-4 and Fig. 9-5 show the chip selector and wait controller registers. For each address space (spaces $\overline{CS0}$ through $\overline{CS3}$ and other address spaces), each chip selector and wait controller register (B01CS through B23CS) can be programmed to set master enable or disable, to select data bus width, to specify the number of waits and to insert dummy cycles.

If two or more address spaces are specified simultaneously, a space or spaces with a smaller space number will be selected by priority (priority order: $\overline{CS0} > \overline{CS1} > \overline{CS2} > \overline{CS3}$).

B01CS (0xFFFF_E480)

	7	6	5	4	3	2	1	0
Bit symbol	B0OM			B0BUS	B0W			
Read/Write	R/W		R	R/W	R/W			
After reset	0	0	0	0	0	1	0	1
Function	Select the chip selector output waveform. 00: ROM/RAM Do not make any other settings.		Always reads “0.”	Select data bus width. 0: 16bit 1: 8bit	Specify the number of waits. (Automatic wait insertion) 0000: 0WAIT 0001: 1WAIT 0010: 2WAIT 0011: 3WAIT 0100: 4WAIT 0101: 5WAIT 0110: 6WAIT 0111: 7WAIT (External wait input) 1010: (2+2N) WAIT 1011: (3+2N) WAIT 1100: (4+2N) WAIT 1101: (5+2N) WAIT 1110: (6+2N) WAIT 1111: (7+2N) WAIT 1000,1001: reserved			

	15	14	13	12	11	10	9	8
Bit symbol	B0CSCV		B0WCV		B0E		B0RCV	
Read/Write	R/W		R/W		R/W	R	R/W	
After reset	0	0	0	0	0	0	0	0
Function	Specify the number of dummy cycles to be inserted. (CS0 recovery time) 11: Disable 10: Disable 01: 1 cycle 00: None		Specify the number of dummy cycles to be inserted. (write, recovery time) 00: 2 cycles 01: 1 cycle 10: None 11: Disable		CS0 Enable 0: Disable 1: Enable	Always reads “0.”	Specify the number of dummy cycles to be inserted. (read, recovery time) 00: 2 cycles 01: 1 cycle 10: None 11: Disable	

	23	22	21	20	19	18	17	16
Bit symbol	B1OM			B1BUS	B1W			
Read/Write	R/W		R	R/W	R/W			
After reset	0	0	0	0	0	1	0	1
Function	Select the chip selector output waveform. 00: ROM/RAM Do not make any other settings.		Always reads “0.”	Select data bus width. 0: 16bit 1: 8bit	Specify the number of waits. (Automatic wait insertion) 0000: 0WAIT 0001: 1WAIT 0010: 2WAIT 0011: 3WAIT 0100: 4WAIT 0101: 5WAIT 0110: 6WAIT 0111: 7WAIT (External wait input) 1010: (2+2N) WAIT 1011: (3+2N) WAIT 1100: (4+2N) WAIT 1101: (5+2N) WAIT 1110: (6+2N) WAIT 1111: (7+2N) WAIT 1000,1001: reserved			

	31	30	29	28	27	26	25	24
Bit symbol	B1CSCV		B1WCV		B1E		B1RCV	
Read/Write	R/W		R/W		R/W	R	R/W	
After reset	0	0	0	0	0	0	0	0
Function	Specify the number of dummy cycles to be inserted. (CS1 recovery time) 11: Disable 10: Disable 01: 1 cycle 00: None		Specify the number of dummy cycles to be inserted. (write, recovery time) 00: 2 cycles 01: 1 cycle 10: None 11: Disable		CS1Enable 0: Disable 1: Enable	Always reads “0.”	Specify the number of dummy cycles to be inserted. (read, recovery time) 00: 2 cycles 01: 1 cycle 10: None 11: Disable	

Fig. 9-4 Chip Selector and Wait Controller Registers

B23CS (0xFFFF_E484)

	7	6	5	4	3	2	1	0
Bit symbol	B2OM			B2BUS	B2W			
Read/Write	R/W		R	R/W	R/W			
After reset	0	0	0	0	0	1	0	1
Function	Select the chip selector output waveform. 00: ROM/RAM Do not make any other settings.		Always reads “0.”	Select data bus width. 0: 16bit 1: 8bit	Specify the number of waits. (Automatic wait insertion) 0000: 0WAIT 0001: 1WAIT 0010: 2WAIT 0011: 3WAIT 0100: 4WAIT 0101: 5WAIT 0110: 6WAIT 0111: 7WAIT (External wait input) 1010: (2+2N) WAIT 1011: (3+2N) WAIT 1100: (4+2N) WAIT 1101: (5+2N) WAIT 1110: (6+2N) WAIT 1111: (7+2N) WAIT 1000,1001: reserved			

	15	14	13	12	11	10	9	8
Bit symbol	B2CSCV		B2WCV		B2E		B2RCV	
Read/Write	R/W		R/W		R/W	R	R/W	
After reset	0	0	0	0	0	0	0	0
Function	Specify the number of dummy cycles to be inserted. (CS2 recovery time) 11: Disable 10: Disable 01: 1 cycle 00: None		Specify the number of dummy cycles to be inserted. (write, recovery time) 00: 2 cycles 01: 1 cycle 10: None 11: Disable		CS2Enable 0: Disable 1: Enable	Always reads “0.”	Specify the number of dummy cycles to be inserted. (read, recovery time) 00: 2 cycles 01: 1 cycle 10: None 11: Disable	

	23	22	21	20	19	18	17	16
Bit symbol	B3OM			B3BUS	B3W			
Read/Write	R/W		R	R/W	R/W			
After reset	0	0	0	0	0	1	0	1
Function	Select the chip selector output waveform. 00: ROM/RAM Do not make any other settings.		Always reads “0.”	Select data bus width. 0: 16bit 1: 8bit	Specify the number of waits. (Automatic wait insertion) 0000: 0WAIT 0001: 1WAIT 0010: 2WAIT 0011: 3WAIT 0100: 4WAIT 0101: 5WAIT 0110: 6WAIT 0111: 7WAIT (External wait input) 1010: (2+2N) WAIT 1011: (3+2N) WAIT 1100: (4+2N) WAIT 1101: (5+2N) WAIT 1110: (6+2N) WAIT 1111: (7+2N) WAIT 1000,1001: reserved			

	31	30	29	28	27	26	25	24
Bit symbol	B3CSCV		B3WCV		B3E		B3RCV	
Read/Write	R/W		R/W		R/W	R	R/W	
After reset	0	0	0	0	0	0	0	0
Function	Specify the number of dummy cycles to be inserted. (CS3 recovery time) 11: Disable 10: Disable 01: 1 cycle 00: None		Specify the number of dummy cycles to be inserted. (write, recovery time) 00: 2 cycles 01: 1 cycle 10: None 11: Disable		CS3Enable 0: Disable 1: Enable	Always reads “0.”	Specify the number of dummy cycles to be inserted. (read, recovery time) 00: 2 cycles 01: 1 cycle 10: None 11: Disable	

Fig. 9-5 Chip Selector and Wait Controller Registers

A reset of the TMP19A23 allows the port 4 controller register (P4CR) and the port 4 function register (P4FC) to be cleared to "0," and the CS signal output is disabled. To output the CS signals, set the corresponding bits to "1" at the P4FC and the P4CR in that order.

The CS recovery time can be configured in any other areas than the CS setting areas, but CS signals will not be output.

10. DMA Controller (DMAC)

The TMP19A23 has a built-in 4-channel DMA Controller (DMAC).

10.1 Features

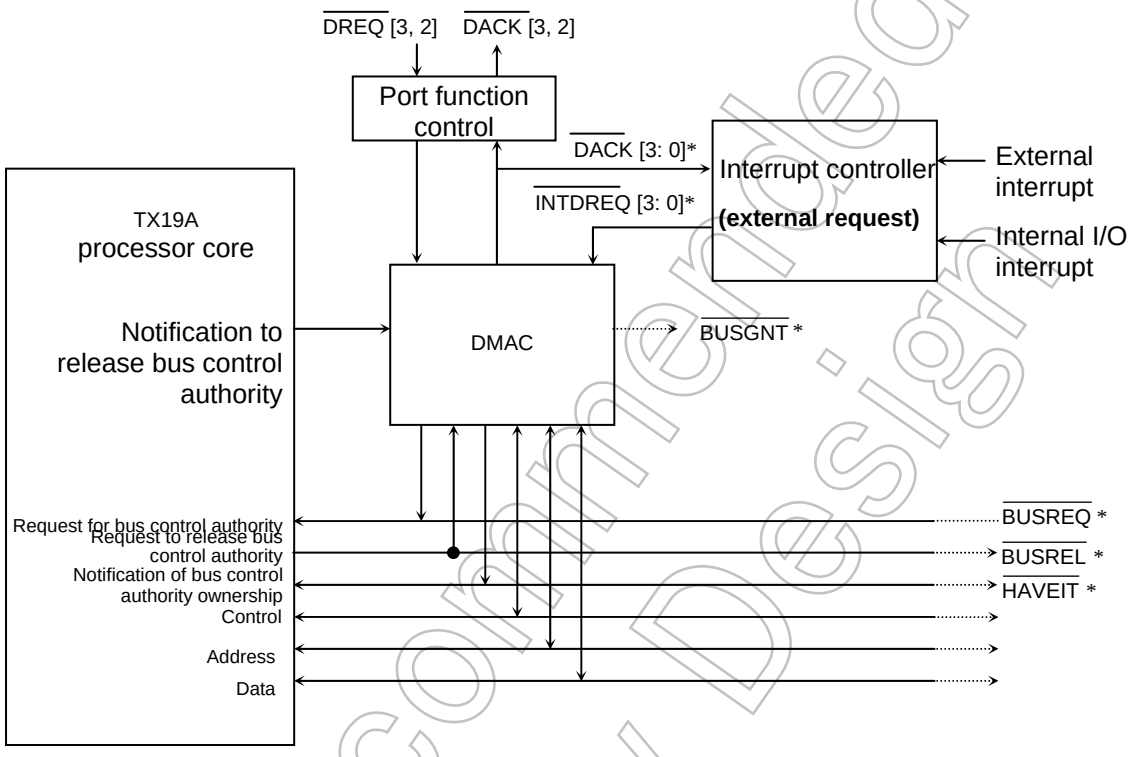
The DMAC of the TMP19A23 has the following features:

(1) DMA with 4 independent channels
(four interrupt factors, 0: INTDMA0 through INTDMA3)
(2) Two types of requests for bus control authority: With and without snoop requests
(3) Transfer requests: Internal requests (software initiated)/external requests (external interrupts, interrupt requests given by internal peripheral I/Os, and requests given by the $\overline{\text{DREQ}}$ pin)
Requests given by the $\overline{\text{DREQ}}$ pin: Level mode
(4) Transfer mode: Dual address mode
(5) Transfer devices: Memory space transfer
(6) Device size: 32-bit memory (8 or 16 bits can be specified using the CS/WAIT controller); I/O of 8, 16 or 32 bits
(7) Address changes: Increase, decrease, fixed, irregular increase, irregular decrease
(8) Channel priority: Fixed (in ascending order of channel numbers)
(9) Endian switchover function

10.2 Configuration

10.2.1 Internal Connections of the TMP19A23

Fig. 10-1 shows the internal connections with the DMAC in the TMP19A23.



(Note) In Fig. 10.1, signals indicated by * are internal signals.

Fig. 10-1 DMAC Connections in the TMP19A23

The DMAC has four DMA channels. Each of these channels handles the data transfer request signal (INTDREQn) from the interrupt controller and the acknowledgment signal (DACKn) generated in response to INTDREQn ("n" is a channel number from 0 to 7). External pins (DREQ3 and DREQ2) are internally wired to allow them to function as pins of the port 5 and the port A. To use them as pins of the port 5 or the port A, they must be selected by setting the function control register PFFC to an appropriate setting.

Pins handle the data transfer request from external pins DREQ3 and DREQ2 and acknowledge signal output supplied through external pins, DACK3 and DACK2. Channel 0 is given higher priority than channel 1. Channel 1 is given higher priority than channel 2. Channel 2 is given higher priority than channel 3. Subsequent channels are given priority in the same manner.

The TX19A processor core has a snoop function. Using the snoop function, the TX19A processor core opens the core's data bus to the DMAC, thus allowing the DMAC to access the internal ROM and RAM linked to the core. The DMAC is capable of determining whether or not to use this snoop function. For further information on the snoop function, refer to 10.2.3 “Snoop Function”.

In the DMAC, bus control authority can be select from SREQ and GREQ depend on the use or nonuse of the snoop function. GREQ is a request for bus control authority if the DMAC does not use the snoop function, while SREQ is a request for bus control authority if the DMAC uses the snoop function. SREQ is given higher priority than GREQ.

10.2.2 DMAC Internal Blocks

Fig. 10-2 shows the internal blocks of the DMAC.

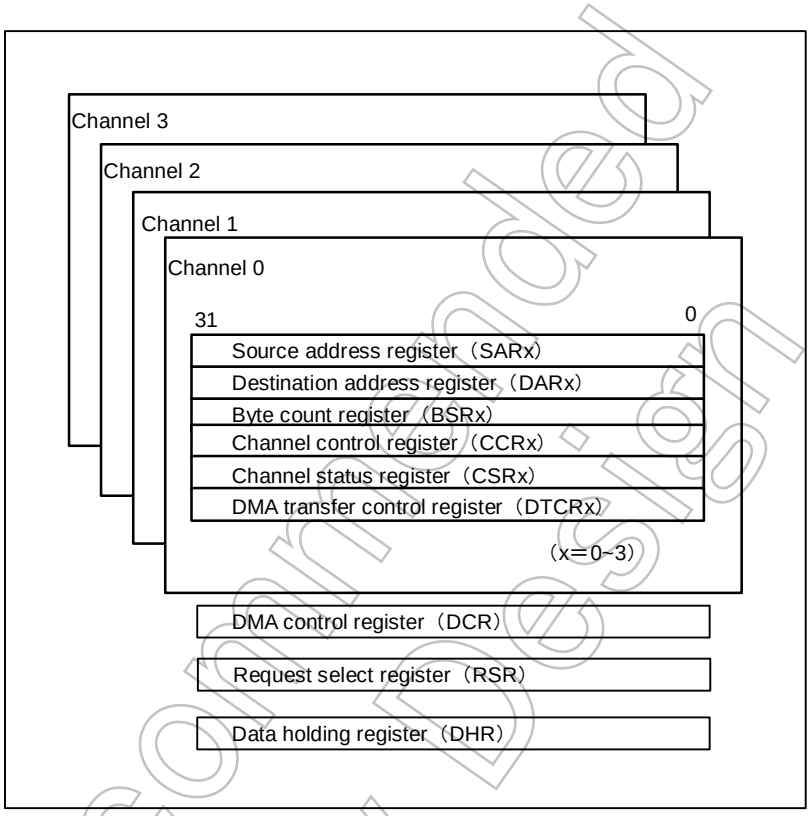


Fig. 10-2 DMAC Internal Blocks

10.2.3 Snoop Function

The TX19A processor core has a snoop function. If the snoop function is activated, the TX19A processor core opens the core's data bus to the DMAC and suspends its own operation until the DMAC withdraws a request for bus control authority. If the snoop function is enabled, the DMAC can access the internal RAM and ROM and therefore designate the RAM or ROM as a source or destination.

If the snoop function is not used, the DMAC cannot access the internal RAM or ROM. However, the G-Bus is opened to the DMAC. If the TX19A processor core attempts to access memory by way of the G-Bus, bus operations cannot be executed and, as a result, the pipeline stalls unless the DMAC accept a bus control release request.

(Note) If the snoop function is not used, the TX19A processor core does not open the data bus to the DMAC. If the data bus is closed and the internal RAM or ROM is designated as a DMAC source or destination, an acknowledgment signal will not be returned in response to a DMAC transfer bus cycle and, as a result, the bus will lock.

10.3 Registers

The DMAC has twenty-seven 32-bit registers. Table 10-1 shows the register map of the DMAC.

Table 10-1 DMAC Registers

Address	Register symbol	Register name
0xFFFF_E200	CCR0	Channel control register (ch. 0)
0xFFFF_E204	CSR0	Channel status register (ch. 0)
0xFFFF_E208	SAR0	Source address register (ch. 0)
0xFFFF_E20C	DAR0	Destination address register (ch. 0)
0xFFFF_E210	BCR0	Byte count register (ch. 0)
0xFFFF_E218	DTCR0	DMA transfer control register (ch. 0)
0xFFFF_E220	CCR1	Channel control register (ch. 1)
0xFFFF_E224	CSR1	Channel status register (ch. 1)
0xFFFF_E228	SAR1	Source address register (ch. 1)
0xFFFF_E22C	DAR1	Destination address register (ch. 1)
0xFFFF_E230	BCR1	Byte count register (ch. 1)
0xFFFF_E238	DTCR1	DMA transfer control register (ch. 1)
0xFFFF_E240	CCR2	Channel control register (ch. 2)
0xFFFF_E244	CSR2	Channel status register (ch. 2)
0xFFFF_E248	SAR2	Source address register (ch. 2)
0xFFFF_E24C	DAR2	Destination address register (ch. 2)
0xFFFF_E250	BCR2	Byte count register (ch. 2)
0xFFFF_E258	DTCR2	DMA transfer control register (ch. 2)
0xFFFF_E260	CCR3	Channel control register (ch. 3)
0xFFFF_E264	CSR3	Channel status register (ch. 3)
0xFFFF_E268	SAR3	Source address register (ch. 3)
0xFFFF_E26C	DAR3	Destination address register (ch. 3)
0xFFFF_E270	BCR3	Byte count register (ch. 3)
0xFFFF_E278	DTCR3	DMA transfer control register (ch. 3)
0xFFFF_E300	DCR	DMA control register (DMAC)
0xFFFF_E304	RSR	Request select register(DMAC)
0xFFFF_E30C	DHR	Data holding register (DMAC)

10.3.1 DMA Control Register (DCR)

DCR
(0xFFFF_E300)

	7	6	5	4	3	2	1	0
Bit symbol					Rst3	Rst2	Rst1	Rst0
Read/Write	W							
After reset	0							
Function	See detailed description.							

	15	14	13	12	11	10	9	8
Bit symbol								
Read/Write	W							
After reset	0							
Function								

	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write	W							
After reset	0							
Function								

	31	30	29	28	27	26	25	24
Bit symbol	Rstall							
Read/Write	W							
After reset	0							
Function	See detailed description.							

Bit	Mnemonic	Field name	Description
31	Rstall	Reset all	Performs a software reset of the DMAC. If the Rstall bit is set to 1, the values of all the internal registers of the DMAC are reset to their initial values. All transfer requests are canceled and all four channels go into an idle state. 0: Don't care 1: Initializes the DMAC
3	Rst3	Reset 3	Performs a software reset of the DMAC channel 3. If the Rst3 bit is set to 1, internal registers of the DMAC channel 3 and a corresponding bit of the channel 3 of the RSR register are reset to their initial values. The transfer request of the channel 3 is canceled and the channel 3 goes into an idle state. 0: Don't care 1: Initializes the DMAC channel 3
2	Rst2	Reset 2	Performs a software reset of the DMAC channel 2. If the Rst2 bit is set to 1, internal registers of the DMAC channel 2 and a corresponding bit of the channel 2 of the RSR register are reset to their initial values. The transfer request of the channel 2 is canceled and the channel 2 goes into an idle state. 0: Don't care 1: Initializes the DMAC channel 2
1	Rst1	Reset 1	Performs a software reset of the DMAC channel 1. If the Rst1 bit is set to 1, internal registers of the DMAC channel 1 and a corresponding bit of the channel 1 of the RSR register are reset to their initial values. The transfer request of the channel 1 is canceled and the channel 1 goes into an idle state. 0: Don't care 1: Initializes the DMAC channel 1
0	Rst0	Reset 0	Performs a software reset of the DMAC channel 0. If the Rst0 bit is set to 1, internal registers of the DMAC channel 0 and a corresponding bit of the channel 0 of the RSR register are reset to their initial values. The transfer request of the channel 0 is canceled and the channel 0 goes into an idle state. 0: Don't care 1: Initializes the DMAC channel 0

Fig. 10-3 DMA Control Register (DCR)

(Note 1) If a write to the DCR register occurs during software reset right after the last round of DMA transfer is completed, the interrupt to stop DMA transfer is not canceled although the channel register is initialized.

(Note 2) An attempt to execute a write (software reset) to the DCR register by DMA transfer is strictly prohibited.

10.3.2 Channel Control Registers (CCRn)

CCRn
(0xFFFF_E200)
(0xFFFF_E220)
(0xFFFF_E240)
(0xFFFF_E260)

	7	6	5	4	3	2	1	0
Bit symbol	SAC		DAC		TrSiz		DPS	
Read/Write	R/W	R/W	R/W		R/W		R/W	
After reset	0							
Function	See detailed description.							
	15	14	13	12	11	10	9	8
Bit symbol		ExR	PosE	Lev	SReq	RelEn	SIO	SAC
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0							
Function	Always set this bit to “0”.	See detailed description.						
	23	22	21	20	19	18	17	16
Bit symbol	NIEn	AblEn					Big	
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	1			0			1	0
Function	See detailed description.		Always set this bit to “0”.				See detailed description.	Always set this bit to “0”.
	31	30	29	28	27	26	25	24
Bit symbol	Str							
Read/Write	W							W
After reset	0							
Function	See detailed description.							Always set this bit to “0”.

Fig. 10-4 Channel Control Register (CCRn) (1/2)

Bit	Mnemonic	Field name	Description
31	Str	Channel start	Start (initial value:–) Starts channel operation. If this bit is set to 1, the channel goes into a standby mode and starts to transfer data in response to a transfer request. Only a write of 1 is valid to the Str bit and a write of 0 is ignored. A read always returns 0. 1: Starts channel operation
24	—	(Reserved)	This is a reserved bit. Always set this bit to "0."
23	NIEn	Normal completion interrupt enable	Normal Completion Interrupt Enable (initial value: 1) 1: Normal completion interrupt enable 0: Normal completion interrupt disable
22	AbIEn	Abnormal completion interrupt enable	Abnormal Completion Interrupt Enable (initial value: 1) 1: Abnormal completion interrupt enable 0: Abnormal completion interrupt disable
21	—	(Reserved)	This is a reserved bit. Although it's initial value is "1," always set this bit to "0".
20	—	(Reserved)	This is a reserved bit. Always set this bit to "0".
19	—	(Reserved)	This is a reserved bit. Always set this bit to "0".
18	—	(Reserved)	This is a reserved bit. Always set this bit to "0".
17	Big	Big-endian	Big Endian (initial value: 1) 1: A channel operates by big-endian 0: A channel operates by little-endian
16	—	(Reserved)	This is a reserved bit. Always set this bit to "0".
15	—	(Reserved)	This is a reserved bit. Always set this bit to "0".
14	ExR	External request mode	External Request Mode (initial value: 0) Selects a transfer request mode. (only for 2ch and 3ch) 1: External transfer request (interrupt request or external $\overline{\text{DREQn}}$ request) 0: Internal transfer request (software initiated)
13	PosE	Positive edge	Positive Edge (initial value: 0) The effective level of the transfer request signal $\overline{\text{INTDREQn}}$ or $\overline{\text{DREQn}}$ is specified. This function is valid only if the transfer request is an external transfer request (if the ExR bit is 1). If it is an internal transfer request (if the ExR bit is 0), the PosE value is ignored. Because the $\overline{\text{INTDREQn}}$ and $\overline{\text{DREQn}}$ signals are active at "L" level, make sure that this PosE bit is set to "0." 1: Setting prohibited 0: The falling edge of the $\overline{\text{INTDREQn}}$ or $\overline{\text{DREQn}}$ signal or the "L" level is effective. The $\overline{\text{DACKn}}$ is active at "L" level.
12	Lev	Level mode	Level Mode (initial value: 0) Specifies signal level or signal change for recognizing the external transfer request. This setting is valid only if a transfer request is the external transfer request (if the ExR bit is 1). If the internal transfer request is specified as a transfer request (if the ExR bit is 0), the value of the Lev bit is ignored. Because the $\overline{\text{INTDREQn}}$ signal is active at "L" level, make sure that you set the Lev bit to "1." The state of active $\overline{\text{DREQn}}$ is determined by the Lev bit setting. 1: Level mode The level of the $\overline{\text{DREQn}}$ signal is recognized as a data transfer request. (The "L" level is recognized if the PosE bit is 0.) 0: Edge mode A change in the $\overline{\text{DREQn}}$ signal is recognized as a data transfer request. (A falling edge is recognized if the PosE bit is 0.)
11	SReq	Snoop request	Snoop Request (initial value: 0) The use of the snoop function is specified by asserting the bus control request mode. If the snoop function is used, the snoop function of the TX19A processor core is enabled and the DMAC can use the data bus of the TX19A processor core. If the snoop function is not used, the snoop function of the TX19A processor core does not work. 1: Use snoop function (SREQ) 0: Do not use snoop function (GREQ)

Bit	Mnemonic	Field name	Description
10	RelEn	Bus control release request enable	Release Request Enable (initial value: 0) Acknowledgment of the bus control release request made by the TX19A processor core is specified. This function is valid only if GREQ is generated. This function cannot be used if SREQ is generated since the TX19A processor core cannot make a bus control release request. 1: The bus control release request is acknowledged if the DMAC has control of the bus. If the TX19A processor core issues a bus control release request, the DMAC relinquishes control of the bus to the TX19A processor core at the break of bus operation. 0: The bus control release request is not acknowledged.
9	SIO	Transfer type selection	Source Type: I/O (initial value: 0) 1: Single transfer 0: Continuous transfer (Data is transferred successively until BCRx becomes "0").
8 : 7	SAC	Source Address Count	Source Address Count (initial value: 00) Specifies the manner of change in a source address. 1x: Address fixed 01: Address decrease 00: Address increase
6	—	(Reserved)	This is a reserved bit. Always set this bit to “0”.
5 : 4	DAC	Destination address count	Destination Address Count (initial value: 00) Specifies the manner of change in a destination address. 1x: Address fixed 01: Address decrease 00: Address increase
3 : 2	TrSiz	Transfer unit	Transfer Size (initial value: 00) Specifies the amount of data to be transferred in response to one transfer request. 11: 8 bits (1 byte) 10: 16 bits (2 bytes) 0x: 32 bits (4bytes) *Make sure to set the same size as the device port size (DPS).
1 : 0	DPS	Device port size	Device Port Size (initial value: 00) Specifies the bus width of an I/O device designated as a source or destination device. 11: 8 bits (1 byte) 10: 16 bits (2bytes) 0x: 32 bits (4 bytes) *Make sure to set the same size as the transfer unit (TrSiz).

Fig. 10-4 Channel Control Register (CCRn) (2/2)

(Note 1) The CCRn register setting must be completed before the DMAC is put into a standby mode.

(Note 2) When accessing the internal I/O or transferring data by DMA in response to the DREQ pin request, make sure that you set the transfer unit <TrSiz> size to be the same as the device port size <DPS>.

(Note 3) In executing memory-to-memory data transfer, a value set in DPS becomes invalid.

(Note 4) Please specify the mode first and then specify the <STR> bit.

10.3.3 Request Select Register (RSR)

RSR (0xFFFF_E304H)

	7	6	5	4	3	2	1	0
Bit symbol					ReqS3	ReqS2		
Read/Write					R/W	R/W		
After reset	0							
Function	Always set this bit to “0”.				See detailed description.		Always set this bit to “0”.	
	15	14	13	12	11	10	9	8
Bit symbol								
Read/Write								
After reset	0							
Function								
	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write								
After reset	0							
Function								
	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write								
After reset	0							
Function								

Bit	Mnemonic	Field name	Description
3	ReqS3	Request select (ch.3)	Request Select (initial value: 0) Selects a source of the external transfer request for the DMA channel 3. 1: Request made by $\overline{\text{DREQ4}}$ 0: Request made by the interrupt controller (INTC)
2	ReqS2	Request select (ch.2)	Request Select (initial value: 0) Selects a source of the external transfer request for the DMA channel 2. 1: Request made by $\overline{\text{DREQ2}}$ 0: Request made by the interrupt controller (INTC)

(Note) Make sure to write "0" to bits 0, 1 and 4 through 7 of the RSR register.

Fig. 10-5 DMA Control Register (RSR)

10.3.4 Channel Status Registers (CSRn)

CSRn
(0xFFFF_E204H)
(0xFFFF_E224H)
(0xFFFF_E244H)
(0xFFFF_E264H)

	7	6	5	4	3	2	1	0
Bit symbol								
Read/Write						R/W		
After reset	0							
Function						Always set this bit to “0”.		

	15	14	13	12	11	10	9	8
Bit symbol								
Read/Write								
After reset	0							
Function								

	23	22	21	20	19	18	17	16
Bit symbol	NC	AbC		BES	BED	Conf		
Read/Write	R/W	R/W	R/W	R	R	R		
After reset	0							
Function	See detailed description.		Always set this bit to “0”.	See detailed description.				

	31	30	29	28	27	26	25	24
Bit symbol	Act							
Read/Write	R							
After reset	0							
Function	See detailed description.							

Fig. 10-6 Channel Status Register (CSRn) (1/2)

Bit	Mnemonic	Field name	Description
31	Act	Channel active	Channel Active (initial value: 0) Channel Active (initial value: 0) Indicates whether the channel is in a standby mode: 1: In a standby mode 0: Not in a standby mode
23	NC	Normal completion	Normal Completion (initial value: 0) Indicates normal completion of channel operation. If an interrupt at normal completion is permitted by the CCR register, the DMAC requests an interrupt when the NC bit becomes 1. This setting can be cleared by writing 0 to the NC bit. If a request for an interrupt at normal completion was previously issued, the request is canceled when the NC bit becomes 0. If an attempt is made to set the Str bit to 1 when the NC bit is 1, an error occurs. To start the next transfer, the NC bit must be cleared to 0. A write of 1 will be ignored. 1: Channel operation has been completed normally. 0: Channel operation has not been completed normally.
22	AbC	Abnormal completion	Indicates abnormal completion of channel operation. If an interrupt at abnormal completion is permitted by the CCR register, the DMAC requests an interrupt when the AbC bit becomes 1. This setting can be cleared by writing 0 to the AbC bit. If a request for an interrupt at abnormal completion was previously issued, the request is canceled when the AbC bit becomes 0. Additionally, if the AbC bit is cleared to 0, each of the BES, BED and Conf bits are cleared to 0. If an attempt is made to set the Str bit to 1 when the AbC bit is 1, an error occurs. To start the next transfer, the AbC bit must be cleared to 0. A write of 1 will be ignored. 1: Channel operation has been completed abnormally. 0: Channel operation has not been completed abnormally.
21	—	(Reserved)	This is a reserved bit. Always set this bit to “0”.
20	BES	Source bus error	Source Bus Error (initial value: 0) 1: A bus error has occurred when the source was accessed. 0: A bus error has not occurred when the source was accessed.
19	BED	Destination bus error	Destination Bus Error (initial value: 0) 1: A bus error has occurred when the destination was accessed. 0: A bus error has not occurred when the destination was accessed.
18	Conf	Configuration error	Configuration Error (initial value: 0) 1: A configuration error has occurred. 0: A configuration error has not occurred.
2 : 0	—	(Reserved)	These three bits are reserved bits. Always set them to "0."

Fig. 10-6 Channel Status Register (CSRn) (2/2)

10.3.5 Source Address Registers (SARn)

SARn		7	6	5	4	3	2	1	0
(0xFFFF_E208H)	Bit symbol	SAddr7	SAddr6	SAddr5	SAddr4	SAddr3	SAddr2	SAddr1	SAddr0
(0xFFFF_E228H)	Read/Write	R/W							
(0xFFFF_E248H)	After reset	Indeterminate							
(0xFFFF_E268H)	Function	See detailed description.							
		15	14	13	12	11	10	9	8
	Bit symbol	SAddr15	SAddr14	SAddr13	SAddr12	SAddr11	SAddr10	SAddr9	SAddr8
	Read/Write	R/W							
	After reset	Indeterminate							
	Function	See detailed description.							
		23	22	21	20	19	18	17	16
	Bit symbol	SAddr23	SAddr22	SAddr21	SAddr20	SAddr19	SAddr18	SAddr17	SAddr16
	Read/Write	R/W							
	After reset	Destination Address Register (DARn)							
	Function	See detailed description.							
		31	30	29	28	27	26	25	24
	Bit symbol	SAddr31	SAddr30	SAddr29	SAddr28	SAddr27	SAddr26	SAddr25	SAddr24
	Read/Write	R/W							
	After reset	Indeterminate							
	Function	See detailed description.							

Bit	Mnemonic	Field name	Description
31 : 0	SAddr	Source address	Source Address (initial value: –) Specifies the address of the source from which data is transferred using a physical address. This address changes according to the SAC and TrSiz settings of CCRn and the SACM setting of DTCRn.

Fig. 10-7 Source Address Register (SARn)

10.3.6 Destination Address Register (DARn)

DARn		7	6	5	4	3	2	1	0
(0xFFFF_E20CH)	Bit symbol	DAddr7	DAddr6	DAddr5	DAddr4	DAddr3	DAddr2	DAddr1	DAddr0
	Read/ Write	R/W							
(0xFFFF_E22CH)	After reset	Indeterminate							
(0xFFFF_E24CH)	Function	See detailed description.							
(0xFFFF_E26CH)		15	14	13	12	11	10	9	8
	Bit symbol	DAddr15	DAddr14	DAddr13	DAddr12	DAddr11	DAddr10	DAddr9	DAddr8
	Read/ Write	R/W							
	After reset	Indeterminate							
	Function	See detailed description.							
		23	22	21	20	19	18	17	16
	Bit symbol	DAddr23	DAddr22	DAddr21	DAddr20	DAddr19	DAddr18	DAddr17	DAddr16
	Read/ Write	R/W							
	After reset	Indeterminate							
	Function	See detailed description.							
		31	30	29	28	27	26	25	24
	Bit symbol	DAddr31	DAddr30	DAddr29	DAddr28	DAddr27	DAddr26	DAddr25	DAddr24
	Read/ Write	R/W							
	After reset	Indeterminate							
	Function	See detailed description.							

Bit	Mnemonic	Field name	Description
31 : 0	DAddr	Destination address	Destination Address (initial value: –) Specifies the address of the destination to which data is transferred using a physical address. This address changes according to the DAC and TrSiz settings of CCRn and the DACM setting of DTCRn.

Fig. 10-8 Destination Address Register (DARn)

10.3.7 Byte Count Registers (BCRn)

BCRn
(0xFFFF_E210)
(0xFFFF_E230)
(0xFFFF_E250)
(0xFFFF_E270)

	7	6	5	4	3	2	1	0
Bit symbol	BC7	BC6	BC5	BC4	BC3	BC2	BC1	BC0
Read/Write	R/W							
After reset	0							
Function	See detailed description.							
	15	14	13	12	11	10	9	8
Bit symbol	BC15	BC14	BC13	BC12	BC11	BC10	BC9	BC8
Read/Write	R/W							
After reset	0							
Function	See detailed description.							
	23	22	21	20	19	18	17	16
Bit symbol	BC23	BC22	BC21	BC20	BC19	BC18	BC17	BC16
Read/Write	R/W							
After reset	0							
Function	See detailed description.							
	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write								
After reset	0							
Function								

Bit	Mnemonic	Field name	Description
23 : 0	BC	Byte count	Byte Count (initial value: 0) Specifies the number of bytes of data to be transferred. The address decreases by the number of pieces of data transferred (a value specified by TrSiz of CCRn).

Fig. 10-9 Byte Count Register (BCRn)

10.3.8 DMA Transfer Control Register (DTCRn)

DTCRn
(0xFFFF_E218H)
(0xFFFF_E238H)
(0xFFFF_E258H)
(0xFFFF_E278H)

	7	6	5	4	3	2	1	0
Bit symbol			DACM			SACM		
Read/Write			R/W			R/W		
After reset	0							
Function			See detailed description.			See detailed description.		

	15	14	13	12	11	10	9	8
Bit symbol								
Read/Write								
After reset	0							
Function								

	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write								
After reset	0							
Function								

	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write								
After reset	0							
Function								

Bit	Mnemonic	Field name	Description
5 : 3	DACM	Destination address count mode	Destination Address Count Mode Specifies the count mode of the destination address. 000: Counting begins from bit 0 001: Counting begins from bit 4 010: Counting begins from bit 8 011: Counting begins from bit 12 100: Counting begins from bit 16 101: Setting prohibited 110: Setting prohibited 111: Setting prohibited
2 : 0	SACM	Source address count mode	Source Address Count Mode Specifies the count mode of the source address. 000: Counting begins from bit 0 001: Counting begins from bit 4 010: Counting begins from bit 8 011: Counting begins from bit 12 100: Counting begins from bit 16 101: Setting prohibited 110: Setting prohibited 111: Setting prohibited

Fig. 10-10 DMA Transfer Control Register (DTCRn)

10.3.9 Data Holding Register (DHR)

DHR (0xFFFF_E30CH)

	7	6	5	4	3	2	1	0
Bit symbol	DOT7	DOT6	DOT5	DOT4	DOT3	DOT2	DOT1	DOT0
Read/Write	R/W							
After reset	0							
Function	See detailed description.							
	15	14	13	12	11	10	9	8
Bit symbol	DOT15	DOT14	DOT13	DOT12	DOT11	DOT10	DOT9	DOT8
Read/Write	R/W							
After reset	0							
Function	See detailed description.							
	23	22	21	20	19	18	17	16
Bit symbol	DOT23	DOT22	DOT21	DOT20	DOT19	DOT18	DOT17	DOT16
Read/Write	R/W							
After reset	0							
Function	See detailed description.							
	31	30	29	28	27	26	25	24
Bit symbol	DOT31	DOT30	DOT29	DOT28	DOT27	DOT26	DOT25	DOT24
Read/Write	R/W							
After reset	0							
Function	See detailed description.							

Bit	Mnemonic	Field name	Description
31 : 0	DOT	Data on transfer	Data on Transfer (initial value: 0) Data that is read from the source in a dual-address data transfer mode.

Fig. 10-11 Data Holding Register (DHR)

10.4 Functions

The DMAC is a 32-bit DMA controller capable of transferring data in a system using the TX19A processor core at high speeds without routing data via the core.

10.4.1 Overview

(1) Source and destination

The DMAC handles data transferred within memory space. A device where the data is output is called a source device and a device where the data is input is called a destination device. The memory device can be designated as a source or destination device.

An interrupt factor can be attached to a transfer request to be sent to the DMAC. If an interrupt factor is generated, the interrupt controller (INTC) issues a request to the DMAC (the TX19A processor core is not notified of the interrupt request. For details, see description on Interrupts.). The request issued by the INTC is cleared by the $\overline{\text{DACKn}}$ signal. Therefore, a request made to the DMAC is cleared after completion of each data transfer (transfer of the amount of data specified by TrSiz) if a single transfer is designated to select a transfer type (SIO BIT). On the other hand, the $\overline{\text{DACKn}}$ signal is asserted only when the number of bytes transferred (value set in the BCRn register) becomes "0" at a continuous transfer. Therefore, one transfer request allows data to be transferred successively without a pause.

For example, if data is transferred between an internal I/O and the internal (external) memory of the TMP19A23, a request made by the internal I/O to the DMAC is cleared after completion of each data transfer. The transfer operation is always put in a standby mode for the next transfer request unless the number of bytes transferred (value set in the BCRn register) becomes "0." Therefore, the DMA transfer operation continues until the value of the BCRn register becomes "0."

(2) Bus control arbitration (bus arbitration)

In response to a transfer request made inside the DMAC, the DMAC requests the TX19A processor core to arbitrate bus control authority. When a response signal is returned from the core, the DMAC acquires bus control authority and executes a data transfer bus cycle.

In acquiring bus control for the DMAC, use or nonuse of the data bus of the TX19A processor core can be specified; specifically either snoop mode or non-snoop mode can be specified for each channel by using bit 11 (SReq) of the CCRn register.

In acquiring bus control for the DMAC, you can select use or non-use of snoop that can use TX19A processor core data bus. This selection can be made in bit 11 (SReq) of the CCRn register by channel.

There are cases in which the TX19A processor core requests the release of bus control authority. Whether or not to respond to this request can be specified for each channel by using the bit 10 (RelEn) of the CCRn register. However, this function can only be used in non-snoop mode (GREQ). In snoop mode (SREQ), the TX19A processor core cannot request the release of bus control; therefore this function cannot be used.

When there are no more transfer requests, the DMAC releases the bus control.

(Note 1) Do not bring the TX19A to a halt when the DMAC is in operation.

(Note 2) Stop the DMAC before putting the TX19A into IDLE (doze) mode while snoop function is active.

(3) Transfer request modes

Two transfer request modes are used for the DMAC: an internal transfer request mode and an external transfer request mode.

In the internal transfer request mode, a transfer request is generated inside the DMAC. Setting a start bit (Str bit of the channel control register CCRn) in the internal register of the DMAC to "1" generates a transfer request, and the DMAC starts to transfer data.

In the external transfer request mode, after a start bit is set to "1," a transfer request is generated when a transfer request signal $\overline{\text{INTDREQn}}$ output by the INTC is input, or when a transfer request signal $\overline{\text{DREQn}}$ output by an external device is input. For the DMAC, two modes are provided: the level mode in which a transfer request is generated when the "L" level of the $\overline{\text{INTDREQn}}$ signal is detected and a mode in which a transfer request is generated when the falling edge or "L" level of the $\overline{\text{DREQn}}$ signal is detected.

(4) Address mode

The DMAC of the TMP19A23 provides only one address mode, a dual address mode. A single address mode is not available. In the dual address mode, data can be transferred within memory space. Source and destination device addresses are output by the DMAC. To access an I/O device, the DMAC asserts the $\overline{\text{DACKn}}$ signal. In the dual address mode, two bus operations, a read and a write, are executed. Data that is read from a source device for transfer is first put into the data holding register (DHR) inside the DMAC and then written to a destination device.

(5) Channel operation

The DMAC has four channels (channels 0 through 3). A channel is activated and put into a standby mode by setting a start (Str) bit in the channel control register (CCRn) to "1."

If a transfer request is generated when a channel is in a standby mode, the DMAC acquires bus control authority and transfers data. If there is no transfer request, the DMAC releases bus control authority and goes into a standby mode. If data transfer has been completed, a channel is put in an idle state. Data transfer is completed either normally or abnormally (e.g. error occurrence). An interrupt signal can be generated upon completion of data transfer.

Fig. 10-15 shows the state transitions of channel operation.

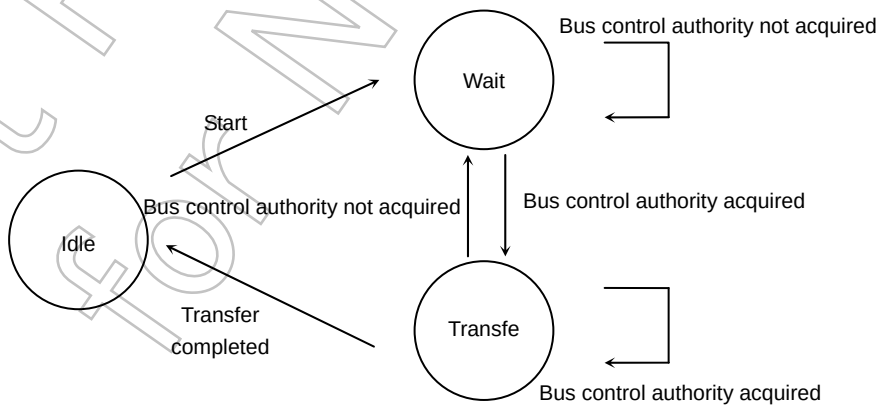


Fig. 10-12 Channel Operation State Transition

(6) Combinations of transfer modes

The DMAC can transfer data by combining each transfer mode as follows:

Transfer request	Edge/level	Address mode	Transfer type
Internal	—	Dual	Continuous transfer
External	"L" level ($\overline{\text{INTDREQn}}$)		Continuous transfer
			Single transfer
External	"L" level ($\overline{\text{DREQn}}$)		Continuous transfer
	Falling edge ($\overline{\text{DREQn}}$)		Single transfer

(7) Address changes

Address changes are broadly classified into three types: increases, decreases and fixed. The type of address change can be specified for each source and destination address by using SAC and DAC in the CCRn register. If a single transfer is selected as a source or destination device, SAC or DAC in the CCRn register must be set to "fixed".

If address increase or decrease is selected, the bit position for counting can be specified using SACM or DACM in the DTCRn register. To specify the bit position for counting a source address, any of the bits 0, 4, 8, 12 and 16 can be specified as the bit position for address counting. If 0 is selected, an address increases or decreases as per normal. By selecting bits 4, 8, 12 or 16, it is possible to increase or decrease an address irregularly.

Examples of address changes are shown below.

Example 1: Monotonic increase for a source device and irregular increase for a destination device

SAC: Address increase
DAC: Address increase
TrSiz: Transfer unit 32 bits
Source address: 0xA000_1000
Destination address: 0xB000_0000
SACM: 000 → counting to begin from bit 0 of the address counter
DACM: 001 → counting to begin from bit 4 of the address counter

	Source	Destination
1st	0xA000_1000	0xB000_0000
2nd	0xA000_1004	0xB000_0010
3rd	0xA000_1008	0xB000_0020
4th	0xA000_100C	0xB000_0030
	…	…

Example 2: Irregular decrease for a source device and monotonic decrease for a destination device

SAC: Address decrease
DAC: Address decrease
TrSiz: Transfer unit 16 bits
Source address: Initial value 0xA000_1000
Destination address: 0xB000_0000
SACM: 010 → counting to begin from bit 8 of the address counter
DACM: 000 → counting to begin from bit 0 of the address counter

	Source	Destination
1st	0xA000_1000	0xB000_0000
2nd	0x9FFF_FF00	0xAFFF_FFFE
3rd	0x9FFF_FE00	0xAFFF_FFFC
4th	0x9FFF_FD00	0xAFFF_FFFA
	…	…

10.4.2 Transfer Request

For the DMAC to transfer data, a transfer request must be issued to the DMAC. There are two types of transfer request: an internal transfer request and an external transfer request. Either of these transfer requests can be selected and specified for each channel.

Whichever is selected, the DMAC acquires the bus control authority and starts to transfer data if the transfer request is generated after the start of channel operation.

- Internal transfer request

  A transfer request is generated immediately if the Str bit of CCR is set to "1" when the ExR bit of CCRn is "0". This transfer request is called an internal transfer request.

  The internal transfer request is valid until the channel operation is completed. Therefore, data can be transferred continuously unless transition to a channel with higher priority or transition of the bus control authority to a bus master with higher priority occurs.

- External transfer request

  Setting the Str bit of CCR to "1" allows a channel to go into a standby mode if the ExR bit of CCRn is "1". The INTC or an external device generates the $\overline{\text{INTDREQn}}$ or $\overline{\text{DREQn}}$ signal for this channel to notify the DMAC of a transfer request, and then a transfer request is generated. This transfer request is called an external transfer request.

  The TMP19A23 recognizes the transfer request signal by detecting the "L" level of the $\overline{\text{INTDREQn}}$ signal or by detecting the falling edge or "L" level of the $\overline{\text{DREQn}}$ signal.

  The unit of data to be transferred in response to one transfer request is specified in the TrSiz field of CCRn. 32, 16 or 8 bits can be selected.

  See the next page for the detailed descriptions on transfer requests using $\overline{\text{INTDREQn}}$ and $\overline{\text{DREQn}}$.

① A transfer request made by the interrupt controller (INTC)

The $\overline{\text{DACKn}}$ signal can clear a transfer request made by the interrupt controller. This $\overline{\text{DACKn}}$ signal is asserted only if a bus cycle for a single transfer or the number of bytes (value set in the BCRn register) transferred at continuous transfer becomes "0." Therefore, at the single transfer, the amount of data specified by TrSiz is transferred only once because $\overline{\text{INTDREQn}}$ is cleared upon completion of one data transfer from one transfer request. On the other hand, at the continuous transfer, it can be transferred successively in response to a transfer request because $\overline{\text{INTDREQn}}$ is not cleared until the number of bytes transferred (value set in the BCRn register) becomes "0."

Note that there is a possibility that the DMA transfer might be executed once after the interrupt is cleared depending on the timing if the DMAC acknowledges an interrupt set in $\overline{\text{INTDREQn}}$ and this interrupt is cleared by the INTC before the DMA transfer begins.

② A transfer request made by an external device

External pins ($\overline{\text{DREQ3}}$ and $\overline{\text{DREQ2}}$) are internally wired to allow them to function as pins of the port 5 and the port A. These pins can be selected by setting the function control register PFFC to an appropriate setting.

In the edge mode, the $\overline{\text{DREQn}}$ signal must be negated and then asserted for each transfer request to create an effective edge. In the level mode, however, successive transfer requests can be recognized by maintaining an effective level. At continuous transfer, only the "L" level mode can be used. At a single transfer, only the falling edge mode can be used.

– Level mode

In the level mode, the DMAC detects the "L" level of the $\overline{\text{DREQn}}$ signal upon the rising of the internal system clock. If it detects the "L" level of the $\overline{\text{DREQn}}$ signal when a channel is in a standby mode, it goes into transfer mode and starts to transfer data. To use the $\overline{\text{DREQn}}$ signal at an active level, the PosE bit (bit 13) of the CCRn register must be set to "0." The $\overline{\text{DACKn}}$ signal is active at the "L" level, as in the case of the $\overline{\text{DREQn}}$ signal.

If an external circuit asserts the $\overline{\text{DREQn}}$ signal, the $\overline{\text{DREQn}}$ signal must be maintained at the "L" level until the $\overline{\text{DACKn}}$ signal is asserted. If the $\overline{\text{DREQn}}$ signal is negated before the $\overline{\text{DACKn}}$ signal is asserted, a transfer request may not be recognized.

If the $\overline{\text{DREQn}}$ signal is not at the "L" level, the DMAC judges that there is no transfer request, and starts a transfer operation for other channels or releases bus the control authority and goes into a standby mode.

The unit of a transfer request is specified in the TrSiz field (<bit3:2>) of the CCRn register.

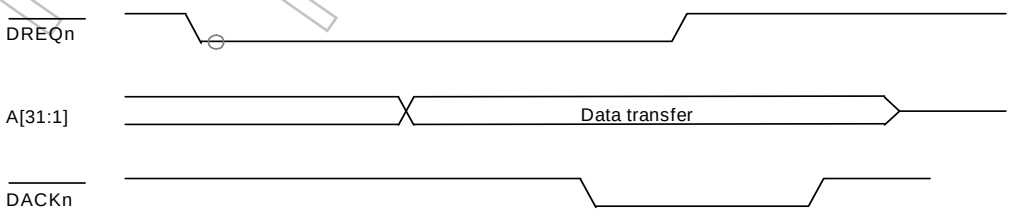


Fig. 10-13 Transfer Request Timing (Level Mode)

– Edge mode

In the edge mode, the DMAC detects the falling edge of the $\overline{\text{DREQn}}$ signal. If it detects the falling edge of the $\overline{\text{DREQn}}$ signal upon the rising of the internal system clock (the case in which the "L" level is detected upon the rising of the system clock although it was not detected upon the rising of the previous system clock) when a channel is in a standby mode, it judges that there is a transfer request, goes into transfer mode, and starts a transfer operation. To detect the falling edge of the $\overline{\text{DREQn}}$ signal, the PosE bit (bit 13) of the CCRn register must be set to "0," and the Lev bit (bit 12) must also be set to "0." The $\overline{\text{DACKn}}$ signal is active at the "L" level.

If the falling edge of the $\overline{\text{DREQn}}$ signal is detected after the $\overline{\text{DACKn}}$ signal is asserted, the next data is transferred without a pause.

If there is no falling edge of the $\overline{\text{DREQn}}$ signal after the $\overline{\text{DACKn}}$ signal is asserted, the DMAC judges that there is no transfer request, and starts a transfer operation for other channels or goes into a standby mode after releasing bus control authority.

The unit of a transfer request is specified in the TrSiz field (<bit3:2>) of the CCRn register.

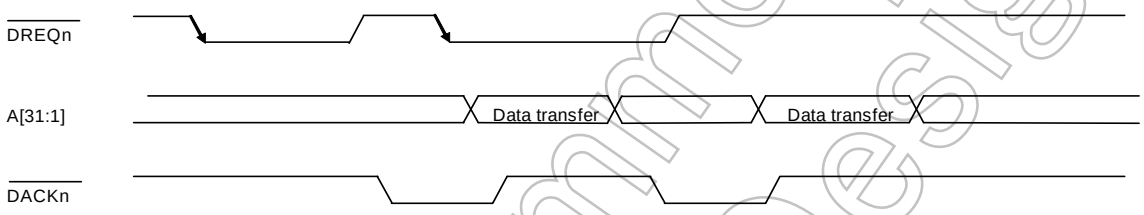


Fig. 10-14 Transfer Request Timing (Edge Mode)

10.4.3 Address Mode

In the address mode, you can specify whether the DMAC executes data transfers by outputting addresses to both source and destination devices or it does by outputting addresses to either a source device or a destination device. The former is called as the dual address mode, and the latter is called as the single address mode. For TMP19A23, only the dual address mode is available.

In the dual address mode, the DMAC first performs a read of the source device by storing the data output by the source device in one of its registers (DHR). It then executes a write on the destination device by writing the stored data to the device, thereby completing the data transfer.

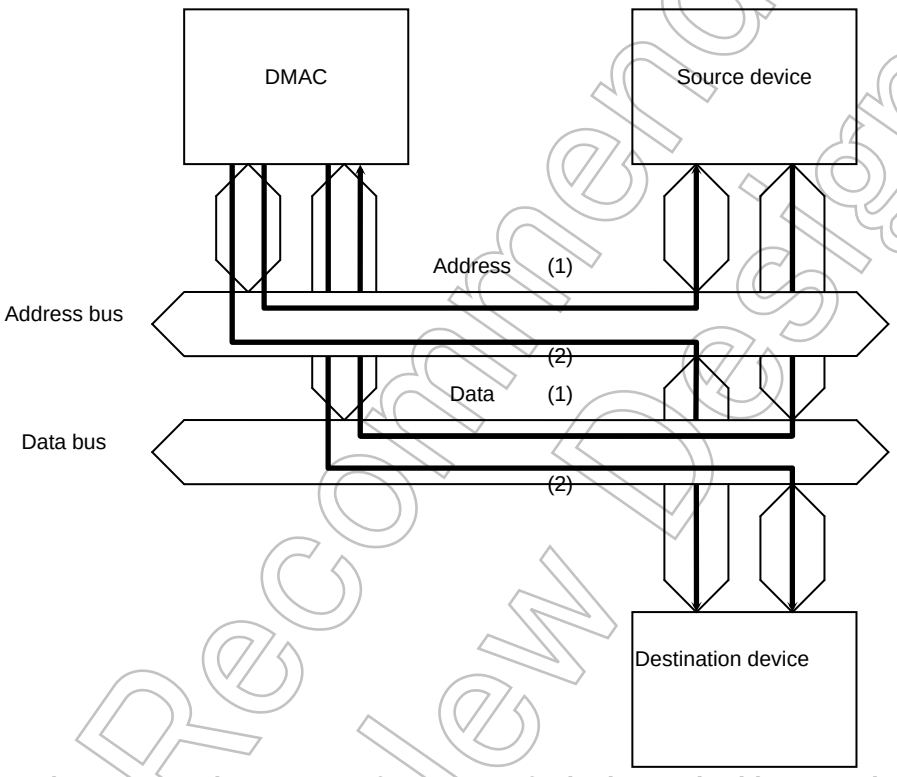


Fig. 10-15 Basic Concept of Data Transfer in the Dual Address Mode

The unit of data to be transferred by the DMAC is the amount of data (32, 16 or 8 bits) specified in the TrSiz field of the CCRn. One unit of data is transferred each time a transfer request is acknowledged.

In the dual address mode, the unit of data is read from the source device, put into the DHR and written to the destination device.

Access to memory takes place when the specified unit of data is transferred. If access to external memory takes place, 16-bit access takes place twice if the unit of data is set to 32 bits and the bus width set in the CS wait controller is 16 bits. Likewise, if the unit of data is set to 32 bits and the bus width set in the CS wait controller is 8 bits, 8-bit access takes place four times.

10.4.4 Channel Operation

A channel is activated if the Str bit of the CCRn in each channel is set to "1." If a channel is activated, an activation check is conducted and the channel is put into a standby mode if no error is detected.

The DMAC acquires bus control authority and starts to transfer data if a transfer request is generated when a channel is in a standby mode.

Channel operation is completed either normally or abnormally (forced termination or occurrence of an error). One of the conditions is indicated to the CSRn.

Start of channel operation

A channel is activated if the Str bit of the CCRn is set to "1".

When a channel is activated, a configuration error check is conducted and the channel is put into a standby mode if no error is detected. If an error is detected, the channel is gone into the abnormal completion. When a channel goes into a standby mode, the Act bit of the CSRn of that channel becomes "1".

A transfer request is generated immediately if a channel is programmed to start operation in response to an internal transfer request. Then the DMAC acquires bus control authority and starts to transfer data. The DMAC acquires bus control authority after INTDREQn or DREQn is asserted and starts to transfer data if a channel is programmed to start operation in response to an external transfer request.

Completion of channel operation

A channel completes operation either normally or abnormally and one of these states is indicated to the CSRn.

Channel operation does not start and the completion of operation is considered to be abnormal completion if "1" is set to the Str bit of the CCRn register when the NC or AbC bit of the CSRn register is "1,"

Normal completion

Channel operation is considered to have been completed normally in the case shown below. For the normally completed channel operation, it needs to be completed after the transfer of a unit of data (value specified in the TrSiz field of CCRn) is completed successfully.

- When the contents of BCRn become 0 and data transfer is completed.

Abnormal completion

Cases of abnormal completion of DMAC operation are as follows:

- Completion due to a configuration error

  A configuration error occurs if there is a mistake in the DMA transfer setting. Because a configuration error occurs before data transfer begins, values specified in SARn, DARn and BCRn remain the same as when they were initially specified. If channel operation is completed abnormally due to a configuration error, the AbC bit of the CSRn is set to "1" along with the Conf bit. Causes of a configuration error are as follows:

  – The Str bit of CCRn was set to "1" when the NC bit or AbC bit of CSRn was "1".

  – A value that is not an integer multiple of the unit of data was set for BCRn.

  – A value that is not an integer multiple of the unit of data was set for SARn or DARn.

  – The Str bit of CCRn was set to "1" when the BCRn value was "0".

- Completion due to a bus error

  The AbC bit of CSRn is set to "1" and the BES or BED bit of CSRn is set to "1" if the DMAC operation has been completed abnormally due to a bus error.

  – A bus error was detected during data transfer.

(Note) If the DMAC operation has been completed abnormally due to a bus error, BCR, SAR and DAR values cannot be guaranteed. If a bus error persists, refer to the chapter 21 "List of Functional Registers".

10.4.5 Order of Priority of Channels

Concerning the four channels of the DMAC, the smaller the channel number assigned to each channel, the higher the priority. If a transfer request is generated to channels 0 and 1 simultaneously, a transfer request for channel 0 is processed with higher priority and the transfer operation is performed accordingly. When the transfer request for channel 0 is cleared, the transfer operation for channel 1 is performed if the transfer request still exists (An internal transfer request is retained if it is not cleared. The interrupt controller retains an external transfer request if the active state for an interrupt request assigned to DMA requests in the interrupt controller is set to edge mode. However, the interrupt controller does not retain an external transfer request if the active state is set to level mode. If the active state for an interrupt request assigned to DMA requests in the interrupt controller is set to level mode, it is necessary to continue asserting the interrupt request signal).

If a transfer request is generated when data is being transferred through channel 1, a channel transition occurs at channel 0, that is, data transfer through channel 1 is temporarily suspended and data transfer through channel 0 is started. When the transfer request for channel 0 is cleared, data transfer through channel 1 resumes.

Channel transitions occur upon completion of the data transfers (when the writing of all data in the DHR has been completed).

Interrupts

Upon completion of a channel operation, the DMAC can generate interrupt requests (INTDMAn: DMA transfer completion interrupt) to the TX19A processor core with two types of interrupts available: a normal completion interrupt and an abnormal completion interrupt.

INTDMA0: 0ch, INTDMA1: 1ch, INTDMA2: 2ch, INTDMA3: 3ch,

- Normal completion interrupt

If a channel operation is completed normally, the NC bit of CSRn is set to "1." If a normal completion interrupt is authorized for the NIEn bit of the CCRn, the DMAC requests the TX19A processor core to authorize an interrupt.

- Abnormal completion interrupt

If a channel operation is completed abnormally, the AbC bit of CSRn is set to "1." If an abnormal completion interrupt is authorized for the AbIEn bit of the CCRn, the DMAC requests the TX19A processor core to authorize an interrupt.

10.5 Timing Diagrams

DMAC operations are synchronous to the rising edges of the internal system clock.

10.5.1 Dual Address Mode

- Continuous transfer

Fig. 10-16 shows an example of the timing with which 16-bit data is transferred from one external memory (16-bit width) to another (16-bit width). Data is actually transferred successively until BCRn becomes "0."

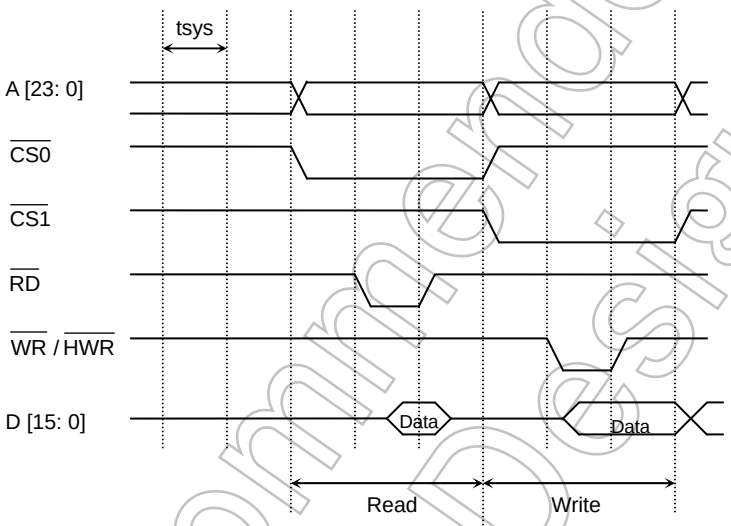


Fig. 10-16 Dual address mode (continuous transfer)

- Single transfer (1)

Fig. 10-17 shows an example of the timing if the unit of data to be transferred is set to 16 bits and the device port size is set to 16 bits.

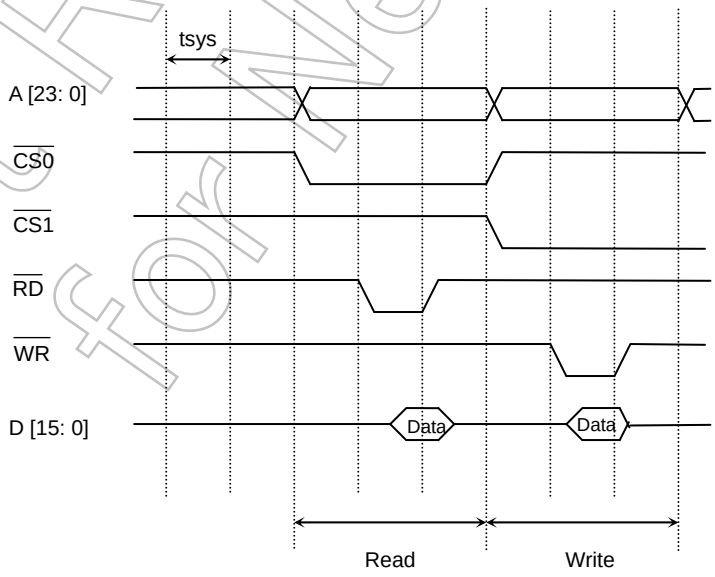


Fig. 10-17 Dual address mode (single transfer)

- Single transfer (2)

Fig. 10-18 shows an example of the timing with which data is transferred from an I/O device to memory if the unit of data to be transferred is set to 16 bits and the device port size is set to 16 bits.

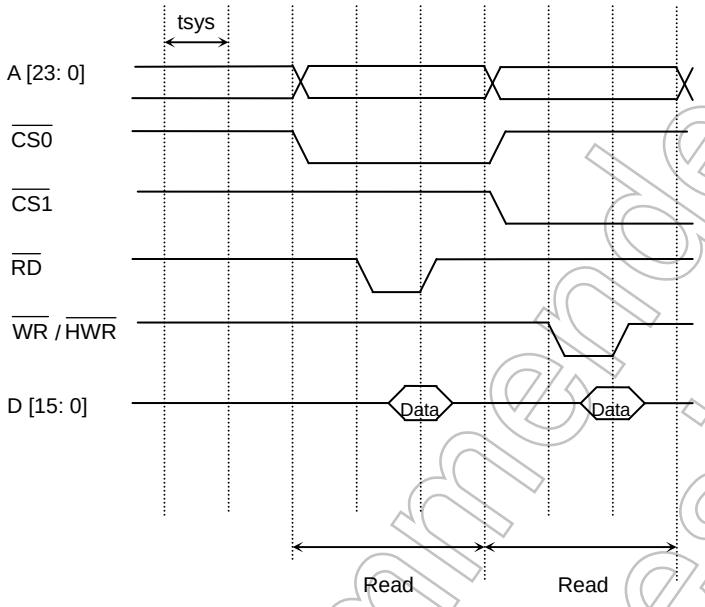


Fig. 10-18 Dual address mode (single transfer)

10.5.2 DREQn-Initiated Transfer Mode

- Data transfer from internal RAM to external memory (multiplexed bus, 5-wait insertion, level mode)

  Fig. 10-19 shows two timing cycles in which 16-bit data is transferred twice from internal RAM to external memory (16-bit width).

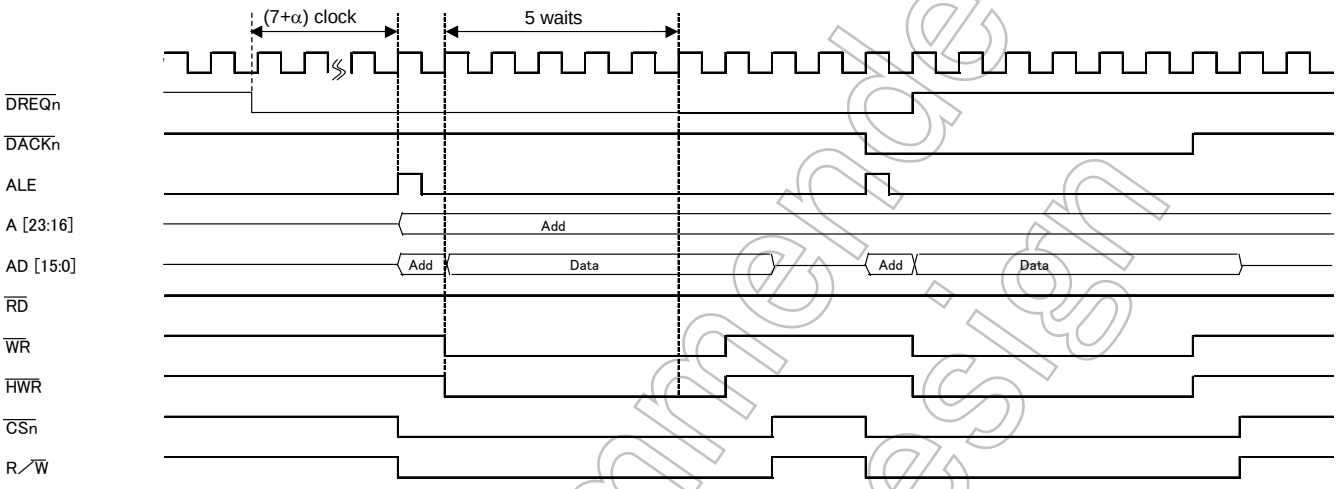


Fig. 10-19 Level Mode (from Internal RAM to External Memory)

- Data transfer from external memory to internal RAM (multiplexed bus, 5-wait insertion, level mode)

  Fig. 10-20 shows two timing cycles in which 16-bit data is transferred twice from external memory (16-bit width) to internal RAM.

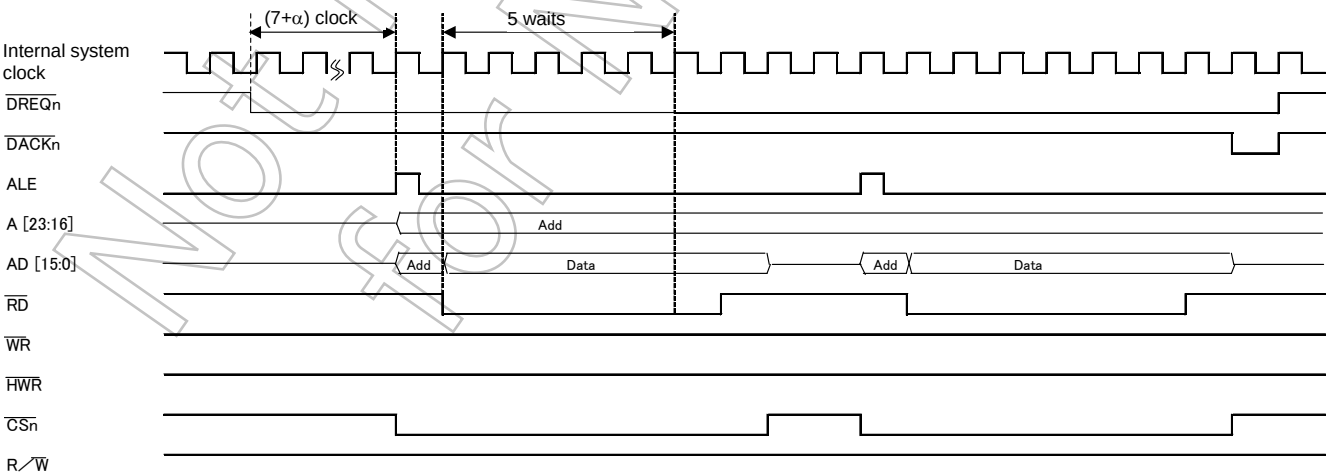


Fig. 10-20 Level Mode (from External Memory to Internal RAM)

- Data transfer from internal RAM to external memory (separate bus, 5-wait insertion, level mode)

  Fig. 10-21 shows two timing cycles in which 16-bit data is transferred twice from internal RAM to external memory (16-bit width).

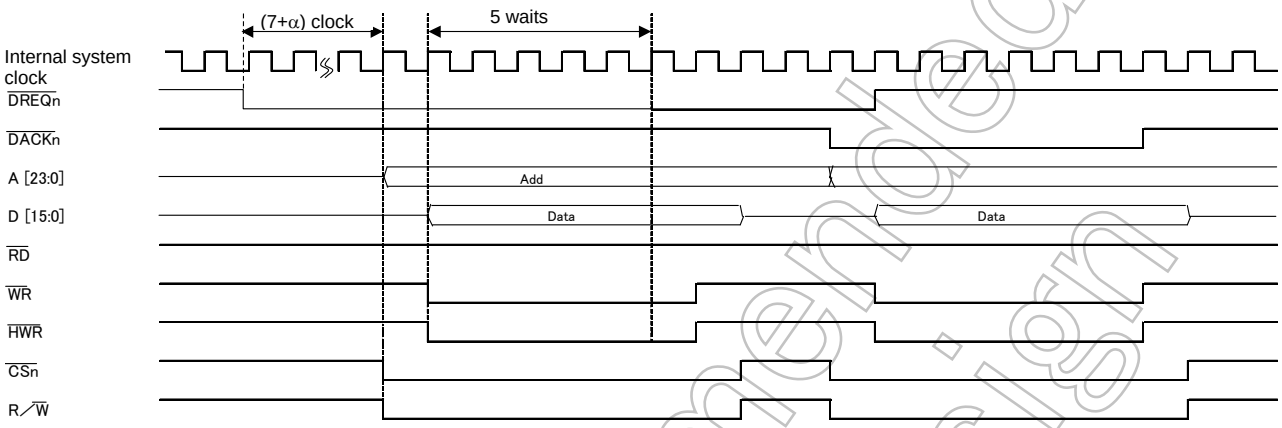


Fig. 10-21 Level Mode (Internal RAM to External Memory)

- Data transfer from external memory to internal RAM (separate bus, 5-wait insertion, level mode)

  Fig. 10-22 shows two timing cycles in which 16-bit data is transferred twice from external memory (16-bit width) to internal RAM.

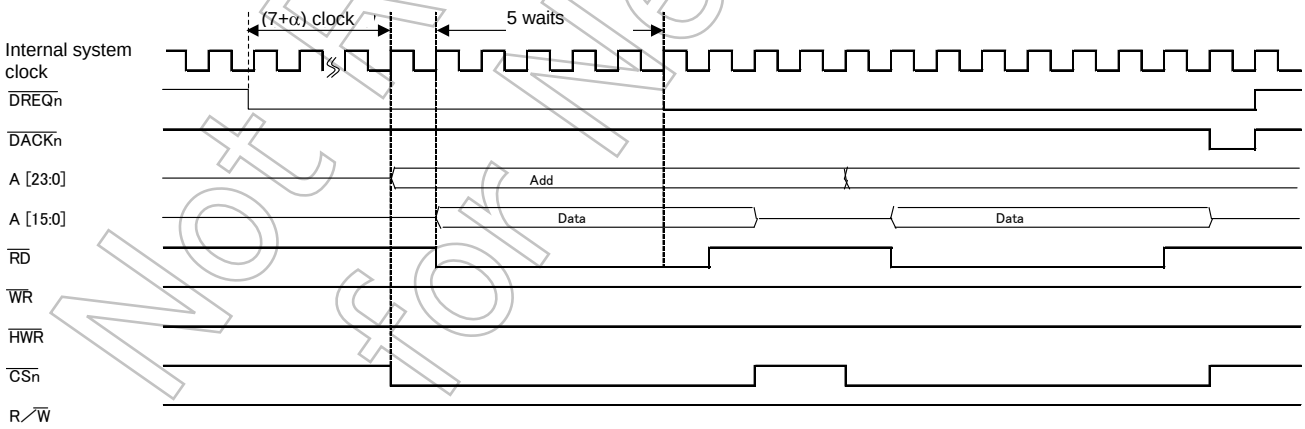


Fig. 10-22 Level Mode (from External Memory to Internal RAM)

- Data transfer from internal RAM to external memory (multiplexed bus, 5-wait insertion, edge mode)

  Fig. 10-23 shows one timing cycle in which 16-bit data is transferred once from internal RAM to external memory (16-bit width).

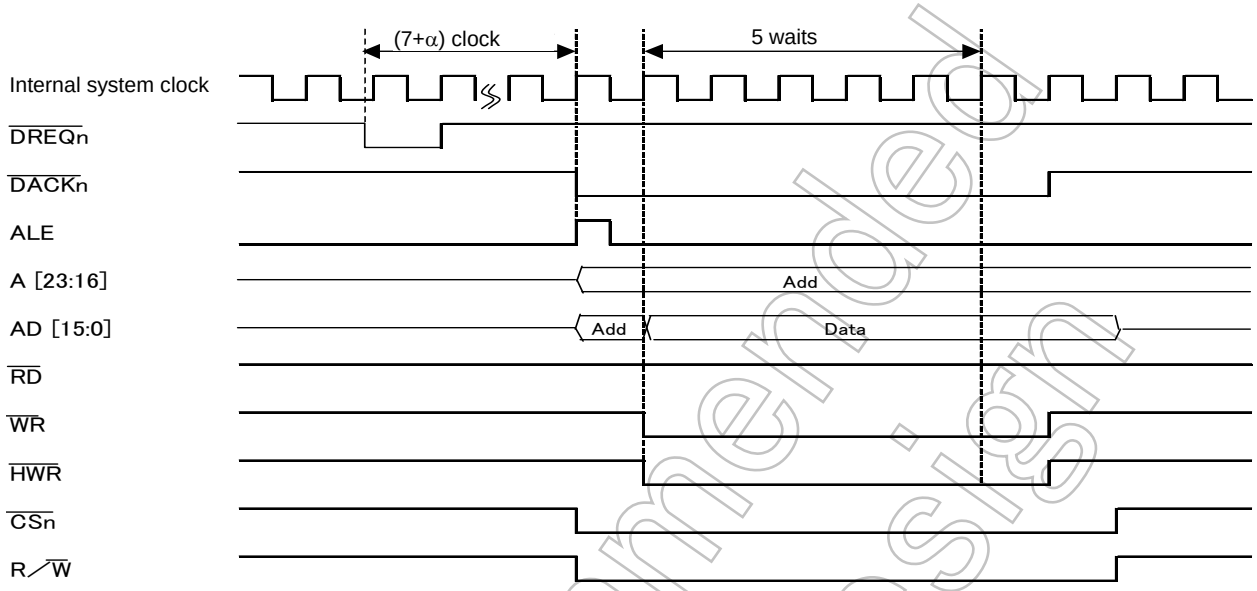


Fig. 10-23 Edge Mode (from Internal RAM to External Memory)

- Data transfer from external memory to internal RAM (multiplexed bus, 5-wait insertion, edge mode)

  Fig. 10-24 shows one timing cycle in which 16-bit data is transferred once from external memory (16-bit width) to internal RAM.

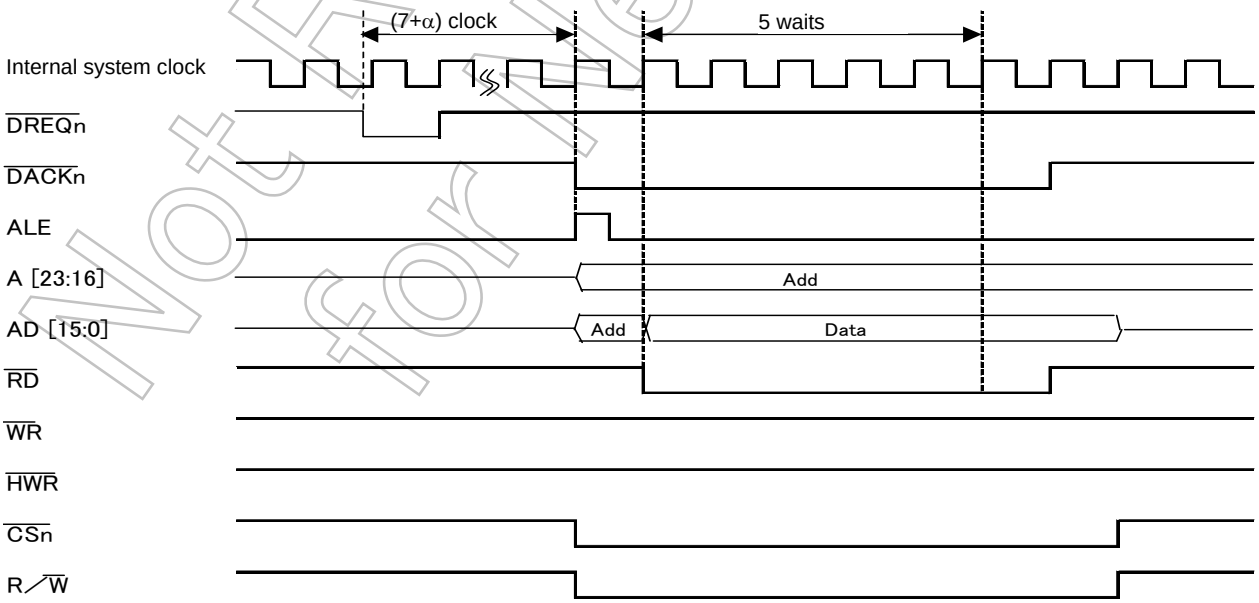


Fig. 10-24 Edge Mode (from External Memory to Internal RAM)

- Data transfer from internal RAM to external memory (separate bus, 5-wait insertion, edge mode)

  Fig. 10-25 shows one timing cycle in which 16-bit data is transferred once from internal RAM to external memory (16-bit width).

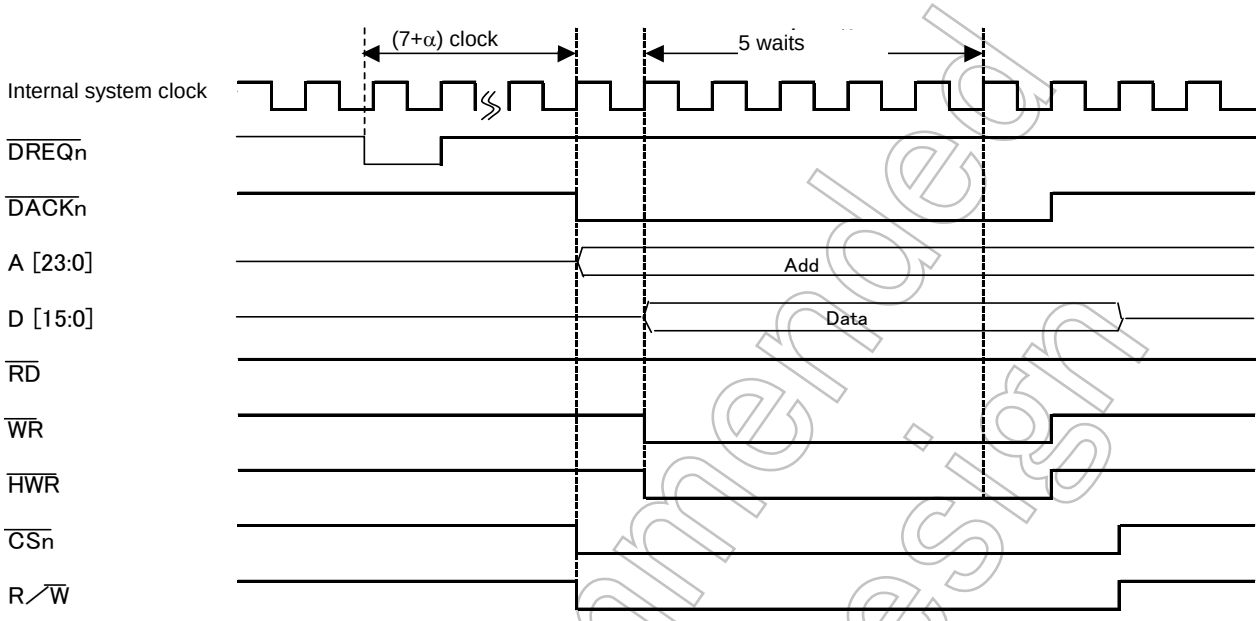


Fig. 10-25 Edge Mode (from Internal RAM to External Memory)

- Data transfer from external memory to internal RAM (separate bus, 5-wait insertion, edge mode)

  Fig. 10-26 shows one timing cycle in which 16-bit data is transferred once from external memory (16-bit width) to internal RAM.

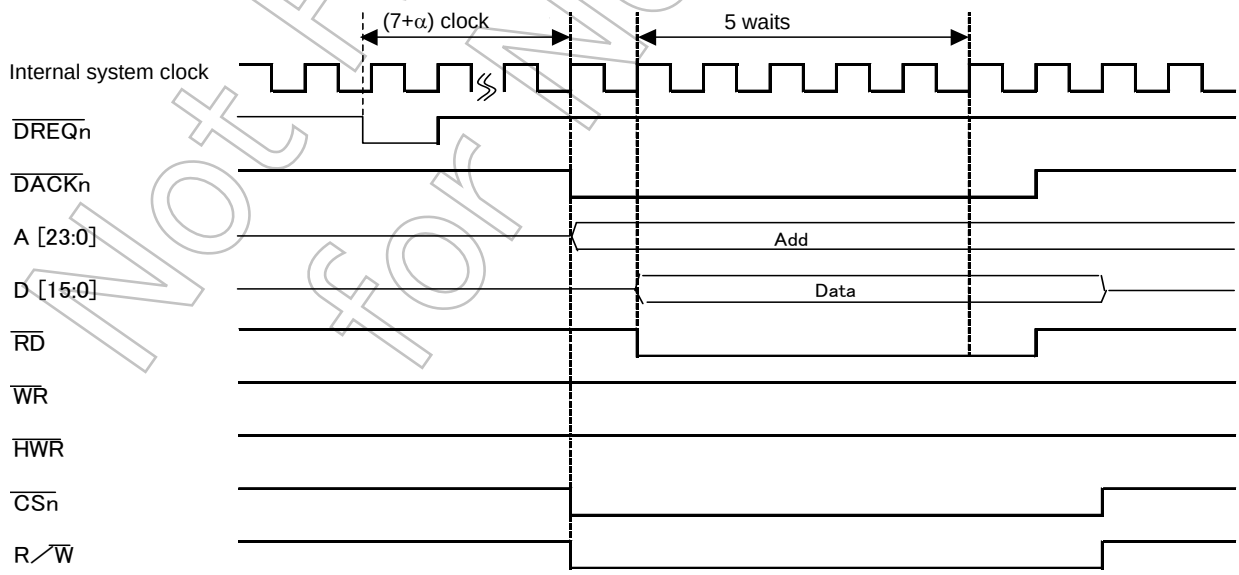


Fig. 10-26 Edge Mode (from External Memory to Internal RAM)

10.6 Case of Data Transfer:

The following case describes how to DMA transfer the serial received data (SCnBUF) to internal RAM.

DMA (ch.0) is used to transfer data. The DMA0 is activated by a receive interrupt generated by SIO1.

<DMA setting>

- Channel used: 0
- Source address: SC1BUF
- Destination: (Physical address) 0xFFFF_9800
- Number of bytes transferred: 256 bytes

<Serial channel setting>

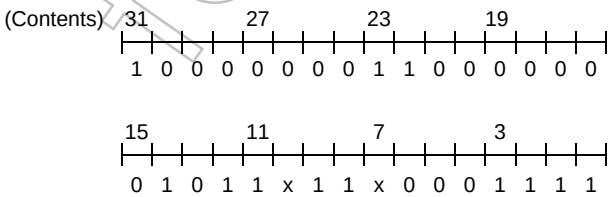
- Data length 8 bits: UART
- Serial channel: ch 1
- Transfer rate: 9600 bps

<SIO ch.1 setting>

```
IMC4HH  ←  x111, x000          /* assigned to DMC0 activation factor * /
INTCLR  ←  0x04C               /* IVR [8:0], INTRX1 interrupt factor * /
SC1MOD0 ←  0x29                /* UART mode, 8-bit length, baud rate generator * /
SC1CR   ←  0x00
BR1CR   ←  0x1F                /* φT4,   N=15 */
```

<DMA0 setting >

```
DCR     ←  0x8000_0000         /* DMA reset * /
IMCCHH  ←  x000, x000          /* disable interrupt * /
INTCLR  ←  0x0CC               /* IVR [8:0] value * /
IMCFHL  ←  x000, x100          /* level = 4 (any given value) */
DTCR0   ←  0x0000_0000         /* DACM = 000 * /
                               /* SACM = 000 * /
SAR0    ←  0xFFFF_F710         /* physical address of SC1BUF */
DAR0    ←  0xFFFF_9800         /* physical address of destination to which data is transferred */
BCR0    ←  0x0000_00FF         /* 256 (number of bytes transferred) /
CCR0    ←  0x80C0_5B0F         /* DMA ch.0 setting */
```



11. 16-bit Timer/Event Counters (TMRBs)

Each of the ten channels (TMRB0 through TMR9) has a multi-functional 16-bit timer/event counter. TMRBs operate in the following four operation modes:

- 16-bit interval timer mode
- 16-bit event counter mode
- 16-bit programmable square-wave output mode (PPG)
- Only TMRB0 provides two-phase pulse input counter mode (quad/normal-speed) and two-phase pulse output mode TMRB0 only)

The use of the capture function allows TMRBs to operate in three other modes

- Frequency measurement mode
- Pulse width measurement mode
- Time difference measurement mode

Each channel consists of a 16-bit up-counter, two 16-bit timer registers (one of which is double-buffered, except for TBRB0 with one double-buffered 16-bit timer register), two 16-bit capture registers, two comparators, a capture input control, a timer flip-flop and its associated control circuit. Timer operation modes and the timer flip-flop are controlled by a register.

Each channel (TMRB0 through TMRB9) functions independently and the channels operate in the same way, except for the differences in their specifications as shown in Table 11-1 and the two-phase pulse count function. Therefore, the operational descriptions here are only for TMRB1 and for the two-phase pulse count function (TMRB0).

The flip-flop output of TMRB 4 or TMRB 9 can be used as the capture trigger of other channels.

・TB4OUT => available for TMRB 0 through TMRB 3
・TB9OUT => available for TMRB 5 through TMRB 8

Table 11-1 Differences in the Specifications of TMRB Modules

Specification \ Channel		TMRB0	TMRB1	TMRB2	TMRB3
External pins	External clock/ capture trigger input pins	TB0IN0 (shared with P62) TB0IN1 (shared with P63)	TB1IN0 (shared with P54) TB1IN1 (shared with P55)	TB2IN0 (shared with P56) TB2IN1 (shared with P57)	TB3IN0 (shared with P60) TB3IN1 (shared with P61)
	Timer flip-flop output pin	TB0OUT0(shared with P60) TB0OUT1(shared with P61)	TB1OUT (shared with PA3)	TB2OUT(shared with PA7)	TB3OUT (shared with PC0)
Internal signals	Timer for capture triggers	TB4OUT	TB4OUT	TB4OUT	TB4OUT
Register names (addresses)	Timer RUN register	TB0RUN (0xFFFF_F200)	TB1RUN (0xFFFF_F210)	TB2RUN (0xFFFF_F220)	TB3RUN (0xFFFF_F230)
	Timer control register	TB0CR (0xFFFF_F201)	TB1CR (0xFFFF_F211)	TB2CR (0xFFFF_F221)	TB3CR (0xFFFF_F231)
	Timer mode register	TB0MOD (0xFFFF_F202)	TB1MOD (0xFFFF_F212)	TB2MOD (0xFFFF_F221)	TB3MOD (0xFFFF_F232)
	Timer flip-flop control register	TB0FFCR (0xFFFF_F203)	TB1FFCR (0xFFFF_F213)	TB2FFCR(0xFFFF_F223)	TB3FFCR (0xFFFF_F233)
	Timer status register	TB0ST (0xFFFF_F204)	TB1ST (0xFFFF_F214)	TB2ST (0xFFFF_F224)	TB3ST (0xFFFF_F234)
	two-phase output control	TB0OUTCR (0xFFFF_F205)	—	—	—
	Timer up counter register	TB0UCL(0xFFFF_F206) TB0UCH(0xFFFF_F207)	TB1UCL(0xFFFF_F216) TB1UCH(0xFFFF_F217)	TB3UCL(0xFFFF_F226) TB3UCH(0xFFFF_F227)	TB3UCL(0xFFFF_F236) TB3UCH(0xFFFF_F237)
	Timer register	TB0RG0L (0xFFFF_F208) TB0RG0H (0xFFFF_F209) TB0RG1L (0xFFFF_F20A) TB0RG1H (0xFFFF_F20B)	TB1RG0L (0xFFFF_F218) TB1RG0H (0xFFFF_F219) TB1RG1L (0xFFFF_F21A) TB1RG1H (0xFFFF_F21B)	TB2RG0L (0xFFFF_F228) TB2RG0H (0xFFFF_F229) TB2RG1L (0xFFFF_F22A) TB2RG1H (0xFFFF_F22B)	TB3RG0L (0xFFFF_F238) TB3RG0H (0xFFFF_F239) TB3RG1L (0xFFFF_F23A) TB3RG1H (0xFFFF_F23B)
	Capture register	TB0CP0L (0xFFFF_F20C) TB0CP0H (0xFFFF_F20D) TB0CP1L (0xFFFF_F20E) TB0CP1H (0xFFFF_F20F)	TB1CP0L (0xFFFF_F21C) TB1CP0H (0xFFFF_F21D) TB1CP1L (0xFFFF_F21E) TB1CP1H (0xFFFF_F21F)	TB2CP0L (0xFFFF_F22C) TB2CP0H (0xFFFF_F22D) TB2CP1L (0xFFFF_F22E) TB2CP1H (0xFFFF_F22F)	TB3CP0L (0xFFFF_F23C) TB3CP0H (0xFFFF_F23D) TB3CP1L (0xFFFF_F23E) TB3CP1H (0xFFFF_F23F)

Specification \ Channel		TMRB4	TMRB5	TMRB6	TMRB7
External pins	External clock/ capture trigger input pins	-	TB5IN0 (shared with PB3) TB5IN1 (shared with PB4)	TB6IN0 (shared with P64) TB6IN1 (shared with P65)	TB7IN0 (shared with P66) TB7IN1 (shared with P67)
	Timer flip-flop output pin	TB4OUT (shared with PC1)	TB5OUT (shared with PB7)	TB6OUT(shared with P50)	TB7OUT (shared with P51)
Internal signals	Timer for capture triggers	-	TB9OUT	TB9OUT	TB9OUT
Register names (addresses)	Timer RUN register	TB4RUN (0xFFFF_F240)	TB5RUN (0xFFFF_F250)	TB6RUN (0xFFFF_F260)	TB7RUN (0xFFFF_F270)
	Timer control register	TB4CR (0xFFFF_F241)	TB5CR (0xFFFF_F251)	TB6CR (0xFFFF_F261)	TB7CR (0xFFFF_F271)
	Timer mode register	TB4MOD (0xFFFF_F242)	TB5MOD (0xFFFF_F252)	TB6MOD (0xFFFF_F262)	TB7MOD (0xFFFF_F272)
	Timer flip-flop control register	TB4FFCR (0xFFFF_F243)	TB5FFCR (0xFFFF_F253)	TB6FFCR(0xFFFF_F263)	TB7FFCR (0xFFFF_F273)
	Timer status register	TB4ST (0xFFFF_F244)	TB5ST (0xFFFF_F254)	TB6ST (0xFFFF_F264)	TB7ST (0xFFFF_F274)
	Timer up counter register	TB4UCL(0xFFFF_F246) TB4UCH(0xFFFF_F247)	TB5UCL(0xFFFF_F256) TB5UCH(0xFFFF_F257)	TB6UCL(0xFFFF_F266) TB6UCH(0xFFFF_F267)	TB7UCL(0xFFFF_F276) TB7UCH(0xFFFF_F277)
	Timer register	TB0RG0L (0xFFFF_F248) TB4RG0H (0xFFFF_F249) TB4RG1L (0xFFFF_F24A) TB4RG1H (0xFFFF_F24B)	TB5RG0L (0xFFFF_F258) TB5RG0H (0xFFFF_F259) TB5RG1L (0xFFFF_F25A) TB5RG1H (0xFFFF_F25B)	TB6RG0L (0xFFFF_F268) TB6RG0H (0xFFFF_F269) TB6RG1L (0xFFFF_F26A) TB6RG1H (0xFFFF_F26B)	TB7RG0L (0xFFFF_F278) TB7RG0H (0xFFFF_F279) TB7RG1L (0xFFFF_F27A) TB7RG1H (0xFFFF_F27B)
	Capture register	TB4CP0L (0xFFFF_F24C) TB4CP0H (0xFFFF_F24D) TB4CP1L (0xFFFF_F24E) TB4CP1H (0xFFFF_F24F)	TB5CP0L (0xFFFF_F25C) TB5CP0H (0xFFFF_F25D) TB5CP1L (0xFFFF_F25E) TB5CP1H (0xFFFF_F25F)	TB6CP0L (0xFFFF_F26C) TB6CP0H (0xFFFF_F26D) TB6CP1L (0xFFFF_F26E) TB6CP1H (0xFFFF_F26F)	TB7CP0L (0xFFFF_F27C) TB7CP0H (0xFFFF_F27D) TB7CP1L (0xFFFF_F27E) TB7CP1H (0xFFFF_F27F)

Specification \ Channel		TMRB8	TMRB9
External pins	External clock/ capture trigger input pins	TB8IN0 (shared with PD4) TB8IN1 (shared with PD5)	TB9IN0 (shared with PD6) TB9IN1 (shared with PD7)
	Timer flip-flop output pin	TB8OUT (shared with P52)	TB9OUT (shared with P53)
Internal signals	Timer for capture triggers	TB9OUT	−
Register names (addresses)	Timer RUN register	TB8RUN (0xFFFF_F280)	TB9RUN (0xFFFF_F290)
	Timer control register	TB8CR (0xFFFF_F281)	TB9CR (0xFFFF_F291)
	Timer mode register	TB8MOD (0xFFFF_F282)	TB9MOD (0xFFFF_F282)
	Timer flip-flop control register	TB8FFCR (0xFFFF_F283)	TB9FFCR (0xFFFF_F293)
	Timer status register	TB8ST (0xFFFF_F284)	TB9ST (0xFFFF_F294)
	Timer up counter register	TB8UCL(0xFFFF_F286) TB8UCH(0xFFFF_F287)	TB9UCL(0xFFFF_F296) TB9UCH(0xFFFF_F297)
	Timer register	TB8RG0L (0xFFFF_F288) TB8RG0H (0xFFFF_F289) TB8RG1L (0xFFFF_F28A) TB8RG1H (0xFFFF_F28B)	TB9RG0L (0xFFFF_F298) TB9RG0H (0xFFFF_F299) TB9RG1L (0xFFFF_F29A) TB9RG1H (0xFFFF_F29B)
	Capture register	TB8CP0L (0xFFFF_F28C) TB8CP0H (0xFFFF_F28D) TB8CP1L (0xFFFF_F28E) TB8CP1H (0xFFFF_F28F)	TB9CP0L (0xFFFF_F29C) TB9CP0H (0xFFFF_F29D) TB9CP1L (0xFFFF_F29E) TB9CP1H (0xFFFF_F29F)

Block Diagram of Each Channel

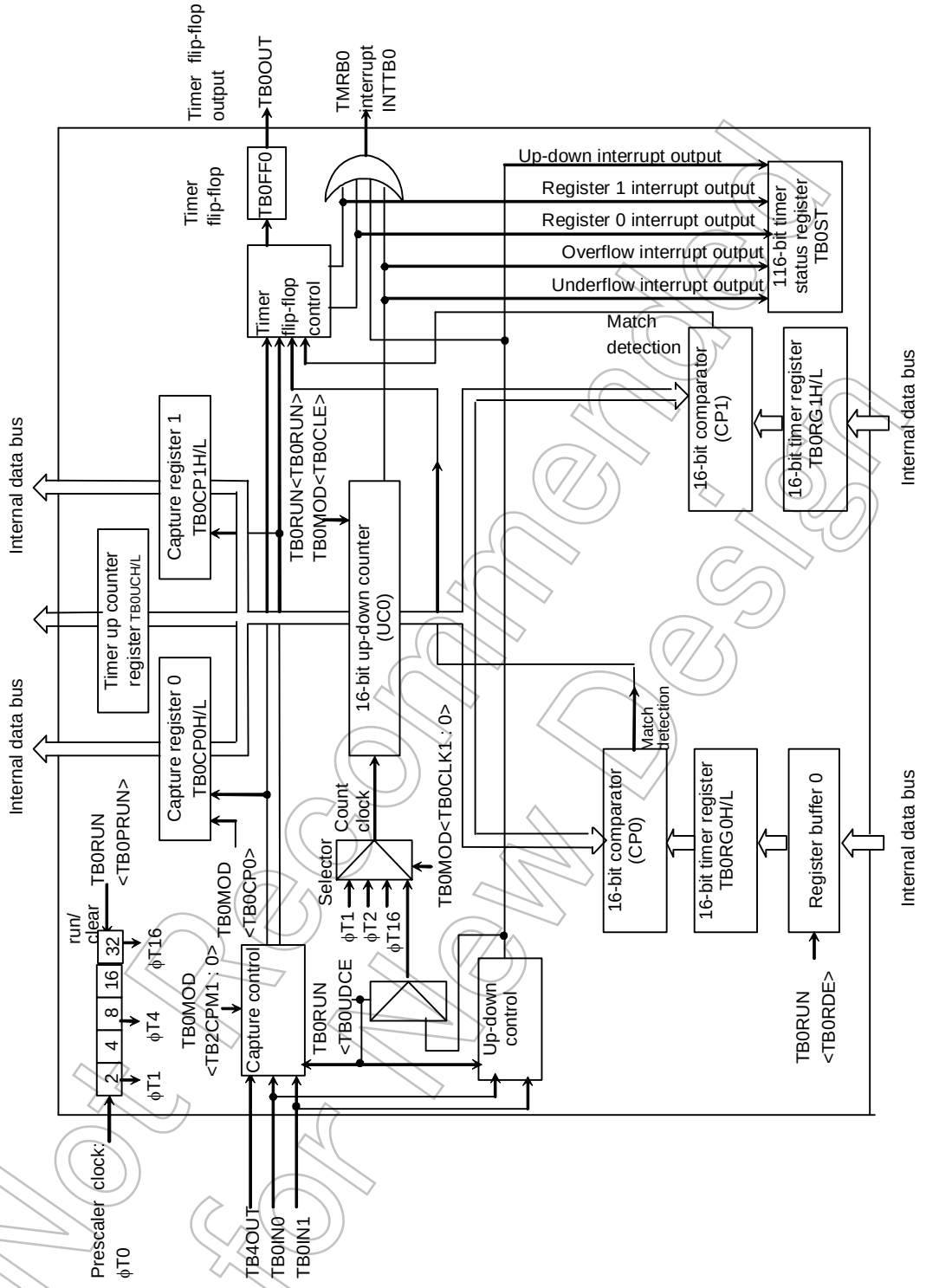
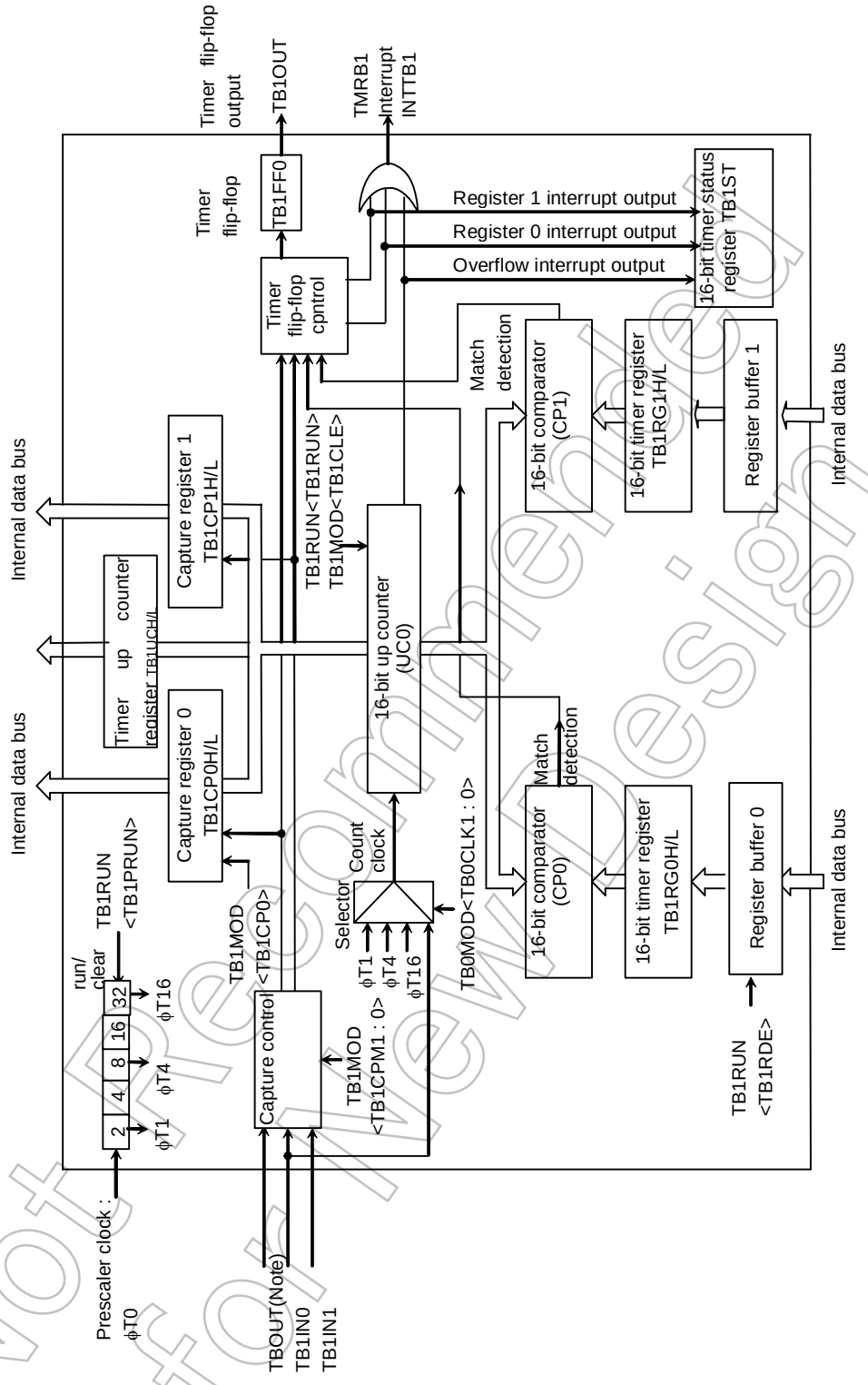


Fig. 11-1 TMRB0 Block Diagram



(Note) TB4OUT is input to channels 0 through 3. TB9OUT is input to channels 5 through 8. Channel 4 and channel 9 do not have input pin for TBOUT.

Fig. 11-2 TMRB1 Block Diagram (Same for Channels 2 through 9)

11.1 Description of Operations for Each Circuit

11.1.1 Prescaler

There is a 4-bit prescaler for acquiring the TMRB0 clock source. The prescaler input clock ϕT0 is fperiph/2, fperiph/4, fperiph/8 or fperiph/16 selected by SYSCR0<PRCK1:0> in the CG. The peripheral clock, fperiph, is either fgear, a clock selected by SYSCR1<FPSEL> in the CG, or fc, which is a clock before it is divided by the clock gear.

The operation or the stoppage of a prescaler is set with TB0RUN<TB0PRUN> where writing “1” starts counting and writing “0” clears and stops counting. Table 11-4 shows prescaler output clock resolutions.

Table 11-2 Prescaler Output Clock Resolutions @fc = 54MHz

Release peripheral clock <FPSEL>	Clock gear value <GEAR2:0>	Select prescaler clock <PRCK1:0>	Prescaler output clock resolutions		
			ϕT1	ϕT4	ϕT16
0 (fgear)	000 (fc)	00(fperiph/16)	fc/2^5(0.59μs)	fc/2^7(2.37μs)	fc/2^9(9.48μs)
		01(fperiph/8)	fc/2^4(0.30μs)	fc/2^6(1.19μs)	fc/2^8(4.74μs)
		10(fperiph/4)	fc/2^3(0.15μs)	fc/2^5(0.59μs)	fc/2^7(2.37μs)
		11(fperiph/2)	fc/2^2(0.07μs)	fc/2^4(0.30μs)	fc/2^6(1.19μs)
	100 (fc/2)	00(fperiph/16)	fc/2^6(1.19μs)	fc/2^8(4.74μs)	fc/2^{10}(18.96μs)
		01(fperiph/8)	fc/2^5(0.59μs)	fc/2^7(2.37μs)	fc/2^9(9.48μs)
		10(fperiph/4)	fc/2^4(0.30μs)	fc/2^6(1.19μs)	fc/2^8(4.74μs)
		11(fperiph/2)	fc/2^3(0.15μs)	fc/2^5(0.59μs)	fc/2^7(2.37μs)
	110 (fc/4)	00(fperiph/16)	fc/2^7(2.37μs)	fc/2^9(9.48μs)	fc/2^{11}(37.93μs)
		01(fperiph/8)	fc/2^6(1.19μs)	fc/2^8(4.74μs)	fc/2^{10}(18.96μs)
		10(fperiph/4)	fc/2^5(0.59μs)	fc/2^7(2.37μs)	fc/2^9(9.48μs)
		11(fperiph/2)	fc/2^4(0.30μs)	fc/2^6(1.19μs)	fc/2^8(4.74μs)
	111 (fc/8)	00(fperiph/16)	fc/2^8(4.74μs)	fc/2^{10}(18.96μs)	fc/2^{12}(75.85μs)
		01(fperiph/8)	fc/2^7(2.37μs)	fc/2^9(9.48μs)	fc/2^{11}(37.93μs)
		10(fperiph/4)	fc/2^6(1.19μs)	fc/2^8(4.74μs)	fc/2^{10}(18.96μs)
		11(fperiph/2)	fc/2^5(0.59μs)	fc/2^7(2.37μs)	fc/2^9(9.48μs)
1 (fc)	000 (fc)	00(fperiph/16)	fc/2^5(0.59μs)	fc/2^7(2.37μs)	fc/2^9(9.48μs)
		01(fperiph/8)	fc/2^4(0.30μs)	fc/2^6(1.19μs)	fc/2^8(4.74μs)
		10(fperiph/4)	fc/2^3(0.15μs)	fc/2^5(0.59μs)	fc/2^7(2.37μs)
		11(fperiph/2)	fc/2^2(0.07μs)	fc/2^4(0.30μs)	fc/2^6(1.19μs)
	100 (fc/2)	00(fperiph/16)	fc/2^5(0.59μs)	fc/2^7(2.37μs)	fc/2^9(9.48μs)
		01(fperiph/8)	fc/2^4(0.30μs)	fc/2^6(1.19μs)	fc/2^8(4.74μs)
		10(fperiph/4)	fc/2^3(0.15μs)	fc/2^5(0.59μs)	fc/2^7(2.37μs)
		11(fperiph/2)	—	fc/2^4(0.30μs)	fc/2^6(1.19μs)
	110 (fc/4)	00(fperiph/16)	fc/2^5(0.59μs)	fc/2^7(2.37μs)	fc/2^9(9.48μs)
		01(fperiph/8)	fc/2^4(0.30μs)	fc/2^6(1.19μs)	fc/2^8(4.74μs)
		10(fperiph/4)	—	fc/2^5(0.59μs)	fc/2^7(2.37μs)
		11(fperiph/2)	—	fc/2^4(0.30μs)	fc/2^6(1.19μs)
	111 (fc/8)	00(fperiph/16)	fc/2^5(0.59μs)	fc/2^7(2.37μs)	fc/2^9(9.48μs)
		01(fperiph/8)	—	fc/2^6(1.19μs)	fc/2^8(4.74μs)
		10(fperiph/4)	—	fc/2^5(0.59μs)	fc/2^7(2.37μs)
		11(fperiph/2)	—	—	fc/2^6(1.19μs)

(Note 1) The prescaler output clock ϕTn must be selected so that ϕTn<fsys/2 is satisfied (so that ϕTn is slower than fsys/2).

(Note 2) Do not change the clock gear while the timer is operating.

(Note 3) "—" denotes a setting prohibited.

Table 11-3 Prescaler Output Clock Resolutions @ = 40MHz

Release peripheral clock <FPSEL	Clock gear value <GEAR2:0>	Select prescaler clock <PRCK1:0>	Prescaler output clock resolutions		
			ϕT1	ϕT4	ϕT16
0 (fgear)	000 (fc)	00(fperiph/16)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		01(fperiph/8)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		10(fperiph/4)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2 μs)
		11(fperiph/2)	fc/2^2(0.1μs)	fc/2^4(0.4μs)	fc/2^6(1.6μs)
	100(fc/2)	00(fperiph/16)	fc/2^6(1.6μs)	fc/2^8 (6.4μs)	fc/2^{10}(25.6μs)
		01(fperiph/8)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		10(fperiph/4)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		11(fperiph/2)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)
	110(fc/4)	00(fperiph/16)	fc/2^7(3.2μs)	fc/2^9(12.8μs)	fc/2^{11} (51.2μs)
		01(fperiph/8)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
		10(fperiph/4)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		11(fperiph/2)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
	111(fc/8)	00(fperiph/16)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)	fc/2^{12}(102.4μs)
		01(fperiph/8)	fc/2^7(3.2μs)	fc/2^9 (12.8μs)	fc/2^{11} (51.2μs)
		10(fperiph/4)	fc/2^6(1.6μs)	fc/2^8(6.4μs)	fc/2^{10}(25.6μs)
		11(fperiph/2)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
1 (fc)	000 (fc)	00(fperiph/16)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		01(fperiph/8)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		10(fperiph/4)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)
		11(fperiph/2)	fc/2^2(0.1μs)	fc/2^4(0.4μs)	fc/2^6(1.6μs)
	100(fc/2)	00(fperiph/16)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		01(fperiph/8)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		10(fperiph/4)	fc/2^3(0.2μs)	fc/2^5(0.8μs)	fc/2^7(3.2μs)
		11(fperiph/2)	—	fc/2^4(0.4μs)	fc/2^6(1.6μs)
	110(fc/4)	00(fperiph/16)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		01(fperiph/8)	fc/2^4(0.4μs)	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		10(fperiph/4)	—	fc/2^5(0.8μs)	fc/2^7(3.2μs)
		11(fperiph/2)	—	fc/2^4(0.4μs)	fc/2^6(1.6μs)
	111(fc/8)	00(fperiph/16)	fc/2^5(0.8μs)	fc/2^7(3.2μs)	fc/2^9(12.8μs)
		01(fperiph/8)	—	fc/2^6(1.6μs)	fc/2^8(6.4μs)
		10(fperiph/4)	—	fc/2^5(0.8μs)	fc/2^7(3.2μs)
		11(fperiph/2)	—	—	fc/2^6(1.6μs)

(Note 1) The prescaler output clock ϕTn must be selected so that ϕTn<fsys/2 is satisfied (so that ϕTn is slower than fsys/2).

(Note 2) Do not change the clock gear while the timer is operating.

(Note 3) "—" denotes a setting prohibited.

11.1.2 Up-counter (UC0) and Up-counter Capture Registers (TB0UCL, TB0UCH)

This is the 16-bit binary counter that counts up in response to the input clock specified by TB0MOD<TB0CLK1:0>.

UC0 input clock can be selected from either three types - ϕT1, ϕT4 and ϕT16 - of prescaler output clock or the external clock of the TB0IN0 pin. For UC0, start, stop and clear are specified by TB0RUN<TB0RUN> and if UC0 matches the TB0RG1H/L timer register, it is cleared to "0" provided the setting is "clear enable." Clear enable/disable is specified by TB0MOD<TB0CLE>.

If the setting is "clear disable," the counter operates as a free-running counter.

The current count value of the UC0 can be captured by reading the TB0UCL and TB0UCH registers.

(Note) Make sure that reading is performed in the order of low-order bits followed by high-order bits. The counted value is not cleared when using the two-phase pulse input mode or the compare function.

If UC0 overflow occurs, the INTTB0 overflow interrupt is generated.
TMRB0 has the two-phase pulse input count function. The two-phase pulse count mode is activated by TB0RUN<TB0UDCE>. This counter serves as the up-down counter, and is initialized to 0x7FFF. If a counter overflow occurs, the initial value 0x0000 is reloaded. If a counter underflow occurs, the initial value 0xFFFF count is continued. When the two-phase pulse count mode is not active, the counter counts up only.

11.1.3 Timer Registers (TB0RG0H/L, TB0RG1H/L)

These are 16-bit registers for specifying counter values and two registers are built into each channel. If a value set on this timer register matches that on a UC0 up-counter, the match detection signal of the comparator becomes active.

To write data to the TB0RG0H/L and TB0RG1H/L timer registers, either a 2-byte data transfer instruction or a 1-byte data transfer instruction written twice in the order of low-order 8 bits followed by high-order 8 bits can be used.

TB0RG0 of this timer register is paired with register buffer 0 - a double-buffered configuration. TB0RG0 uses TB0RUN<TB0RDE> to control the enabling/disabling of double buffering. If <TB0RDE> = "0," double buffering is disabled and if <TB0RDE> = "1," it is enabled. If double buffering is enabled, data is transferred from register buffer 0 to the TB0RG0 timer register when there is a match between UC0 and TB0RG1.

The values of TB0RG0H/L and TB0RG1H/L become undefined after a reset; therefore it is necessary to write data to them beforehand in case of using a 16-bit timer. A reset initializes TB0RUN <TB0RDE> to "0" and sets double buffering to "disable." To use double buffering, write data to the timer register, set <TB0RDE> to "1" and then write the following data to the register buffers.

TB0RG0 and the register buffers are assigned to the same address: 0xFFFF_F208/0xFFFF_F209. If <TB0RDE> = "0," the same value is written to TB0RG0 and each register buffer; if <TB0RDE> = "1," the value is only written to each register buffer. To write an initial value to the timer register, therefore, the register buffers must be set to "disable."

(Note 1) Please rewrite neither TB0REG0 nor TB0RG0 of which a double buffer is unused while the timer is working.
(Note 2) When a double buffer is used, data is not updated while rewriting TB0REG0. As for TMRB1 through TMRB9, TBxREG1 and TBxREG0 have a double buffer configuration.

11.1.4 Capture Registers (TB0CP0H/L, TB0CP1H/L)

These are 16-bit registers for latching values from the UC0 up-counter. To read data from the capture register, use a 16-bit data transfer instruction or read in the order of low-order bits followed by high-order bits.

11.1.5 Capture

This is a circuit that controls the timing of latching values from the UC0 up-counter into the TB0CP0 and TB0CP1 capture registers. The timing with which to latch data is specified by TB0MOD <TB0CPM1:0>.

Software can also be used to import values from the UC0 up-counter into the capture register; specifically, UC0 values are taken into the TB0CP0 capture register each time "0" is written to TB0MOD<TB0CP0>. To use this capability, the prescaler must be running (TB0RUN<TB0PRUN> = "1").

In the two-phase pulse count mode (for the TMRB0 only), the counter value is captured by using software.

(Note 1) Although a read of low-order 8 bits in the capture register suspends the capture operation, it is resumed by successively reading high-order 8 bits.

(Note 2) If the timer stops after a read of low-order 8 bits, the capture operation remains suspended even after the timer restarts. Please do not stop the timer after a read of low-order 8 bits.

11.1.6 Comparators (CP0, CP1)

These are 16-bit comparators for detecting a match by comparing set values of the UC0 up-counter with set values of the TB0RG0 and TB0RG1 timer registers. If a match is detected, INTTB0 is generated.

11.1.7 Timer Flip-flop (TB0FF0)

The timer flip-flop (TB0FF0) is reversed by a match signal from the comparator and a latch signal to the capture registers. It can be enabled or disabled to reverse by setting the TB0FFCR<TB0C1T1, TB0C0T1, TB0E1T1, TB0E0T1>.

The value of TB0FF0 becomes undefined after a reset. The flip-flop can be reversed by writing "00" to TB0FFCR<TB0FF0C1:0>. It can be set to "1" by writing "01," and can be cleared to "0" by writing "10."

The value of TB0FF0 can be output to the timer output pin, TB0OUT (shared with P54). To enable timer output, the port 5 related registers P5CR and P5FC must be programmed beforehand.

11.2 Register Description

TMRBn RUN register (n=1 through 9)

TBnRUN (0xFFFF_F2x0)

	7	6	5	4	3	2	1	0
Bit symbol	TBnRDE				I2TBn	TBnPRUN		TBnRUN
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
After reset	0	0	0	0	0	0	0	0
Function	Double Buffer 0: Disable 1: Enable	Write “0”.	Write “0”.	Write “0”.	In the IDLE mode. 0: Stop 1: Operation	Timer Run/Stop Control 0: Stop & clear 1: Count * The first bit can be read as “0.”		

<TBnRUN>:Controls the TMRB0 count operation.

<TBnPRUN>:Controls the TMRB0 prescaler operation.

<I2TBn>:Controls the operation in the IDLE mode.

<TBnRDE>:Controls enabling/disabling of double buffering **(TBxREG0 and TBxREG 1 are configured with double buffer).**

(Note 1) The value read from bit 1 of TBnRUN is “0.”
(Note 2) Please specify the mode first and then specify the <TBnRUN> and <TBnPRUN> bits.

TMRB0 RUN register

TB0RUN (0xFFFF_F200)

	7	6	5	4	3	2	1	0
Bit symbol	TB0RDE		UD0CMP	TB0UDCE	I2TB0	TB0PRUN		TB0RUN
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
After reset	0	0	0	0	0	0	0	0
Function	Double Buffer 0: Disable 1: Enable	Write “0”.	Compare function 0: Enable 1: Disable	Enable/ disable two-phase counter 0: Disable 1: Enable	IDLE 0: Stop 1: Operation	Timer Run/Stop Control 0: Stop & clear 1: Count * The first bit can be read as “0.”		

<TB0RUN>:Controls the TMRB0 count operation.

<TB0PRUN>:Controls the TMRB0 prescaler operation.

<I2TB0>:Controls the operation in the IDLE mode.

<TB0UDCE>:Controls enabling/disabling of the two-phase pulse input count operation.

Enable: The counter counts up and counts down.

Disable: This is the normal timer mode and the counter counts up only.

<UD0CMP>:Selects for enabling or disabling interrupt generated when the comparator detects a match at the two-phase pulse input count mode.

Enable: Interrupt is generated if the value of 16-bit timer register matches that of the 16-bit up-counter.

Disable: Interrupt is generated even if the value of 16-bit timer register matches that of the 16-bit up-counter.

<TBmRDE>:Controls enabling/disabling of double buffering **(TB0REG0 is configured with double buffer).**

(Note 1) The value read from bit 1 of TBmRUN is “0.”
(Note 2) TB0RUN bit 5 is set to “0” after reset. Be sure to set it to “1” if you use two-phase pulse counter mode.
(Note 3) Please specify the mode first and then specify the <TBnRUN> and <TBnPRUN> bits.

Fig. 11-3 TMRB-related Registers

TMRB0 control register

TB0CR (0xFFFF_F201)

	7	6	5	4	3	2	1	0
Bit symbol	TB0EN					UD0NF	UD01CNT	UD00CNT
Read/Write	R/W		R			R/W		
After reset	0	0	0	0	0	0	0	0
Function	TMRBn operation 0: Disable 1: Enable	Write “0”.	This can be read as “0”.	This can be read as “0”.	This can be read as “0”.	Digital noise filter 0:No use 1:Use	Mode switch-over 00:Normal 01:TB0IN0 double 10:Quadruple 11:TB0IN1 double	

(Note) TB0CR <TB0EN> must be set to “1” (enabled) in order to activate TMRB0.

TMRBn control register (n=1 through 9)

TBnCR (0xFFFF_F2x1)

	7	6	5	4	3	2	1	0
Bit symbol	TBnEN							
Read/Write	R/W	R/W	R	R	R	R	R	R
After reset	0	0	0	0	0	0	0	0
Function	TMRBn operation 0: Disable 1: Enable	Write “0”.	This can be read as “0”.	This can be read as “0”.	This can be read as “0”.	This can be read as “0”.	This can be read as “0”.	This can be read as “0”.

<TBnEN>: Specifies the TMRB operation. When the operation is disabled, no clock is supplied to the other registers in the TMRB module. This can reduce power dissipation. (This disables reading from and writing to the other registers.) To use the TMRB, enable the TMRB operation (set to “1”) before programming each register in the TMRB module. If the TMRB operation is executed and then disabled, the settings will be maintained in each register.

TMRBn mode register (n=0 through 9)

TBnMOD (0xFFFF_F2x2)

	7	6	5	4	3	2	1	0
Bit symbol			TBnCP0	TBnCPM1	TBnCPM0	TBnCLE	TBnCLK1	TBnCLK0
Read/Write	R		W	R/W				
After reset	0	0	1	0	0	0	0	0
Function	This can be read as “00”.		Capture control by software 0: Capture by software 1: Don't care	Capture timing 00: Disable 01: TBnIN0 ↑ TBnIN1 ↑ 10: TBnIN0 ↑ TBnIN0 ↓ 11: TB4OUT ↑ TB4OUT ↓ (11: TB9OUT ↑ TB9OUT ↓)		Up-counter control 0: Clear/disable 1: clear/enable	Selects source clock 00: TB0IN0 pin input 01: ϕT1 10: ϕT4 11: ϕT16	

<TBnCLK1:0>: Selects the TMRBn timer count clock.

<TBnCLE>: Clears and controls the TMRBn up-counter.

“0”: Disables clearing of the up-counter.

“1”: Clears up-counter if there is a match with timer register 1 (TBnRG1).

<TBnCPM1:0>: Specifies TMRBn capture timing.

“00”: Capture disable

“01”:Takes count values into capture register 0 (TBnCP0) upon rising of TBnIN0 pin input. Takes count values into capture register 1 (TBnCP1) upon rising of TBnIN1 pin input.

“10”:Takes count values into capture register 0 (TBnCP0) upon rising of TBnIN0 pin input. Takes count values into capture register 1 (TBnCP1) upon falling of TBnIN0 pin input.

“11”:Takes count values into capture register 0 (TBnCP0) upon rising of 16-bit timer match output (TBxOUT) and into capture register 1 (TBnCP1) upon falling of TBxOUT (TMRB0 through TMRB3:TB4OUT, TMRB5 through TMRB8:TB9OUT).

<TBnCP0>:Captures count values by software and takes them into capture register 0 (TBnCP0).

(Note 1) The value read from bit 5 of TBnMOD is “1”.

(Note 2) Please specify the mode first and then specify the <TBnCP0> bit.

Fig. 11-4 TMRB-related Register

TMRBn flip-flop control register (n=0 through 9)

TBnFFCR (0xFFFF_F1x3)

	7	6	5	4	3	2	1	0
Bit symbol			TBnC1T1	TBnC0T1	TBnE1T1	TBnE0T1	TBnFF0C1	TBnFF0C0
Read/Write	R		R/W				W	
After reset	1	1	0	0	0	0	1	1
Function	This can be read as "11" .		TBnFF0 reverse trigger 0: Disable trigger 1: Enable trigger				TBnFF0 control 00: Invert 01: Set 10: Clear 11: Don't care * This is always read as "11."	
			When the up-counter value is taken into TBnCP1	When the up-counter value is taken into TBnCP0	When the up-counter matches TBnRG1	When the up-counter matches TBnRG0		

<TBnFF0C1:0>: Controls the timer flip-flop.

"00": Reverses the value of TBnFF0 (reverse by using software).

"01": Sets TBnFF0 to "1".

"10": Clears TBnFF0 to "0".

"11":Don't care

<TBnE1:0>:Reverses the timer flip-flop when the up-counter matches the timer register 0,1 (TBnRG0,1).

<TBnC1:0>:Reverses the timer flip-flop when the up-counter value is taken into the capture register 0,1 (TBnCP0,1).

(Note) The timer flip-flop function is not available when TB0RUN <UD0CMP> is "0" and using the two-phase pulse input mode or the compare function.

Fig. 11-5 TMRB-related Register

TMRBn status register (n=0 through 9)

TBnST (0xFFFF_F2x4)

	7	6	5	4	3	2	1	0
Bit symbol						INTTBOFn	INTTBn1	INTTBn0
Read/Write	R					R		
After reset	0					0	0	0
Function	This can be read as “0”.					0: Interrupt not generated 1: Interrupt generated	0: Interrupt not generated 1: Interrupt generated	0: Interrupt not generated 1: Interrupt generated

<INTTBn0>:Interrupt generated if there is a match with timer register 0 (TBnRG0)
<INTTBn1>: Interrupt generated if there is a match with timer register 1 (TBnRG1)
<INTTBOFn>: Interrupt generated if an up-counter overflow occurs

(Note) If any interrupt is generated, the flag that corresponds to the interrupt is set to TBnST and the generation of interrupt is notified to INTC. The flag is cleared by reading the TBnST register.

TMRB0 status register

① When TB0RUN <TB0UDCE> = 0: Normal timer mode

TB0ST (0xFFFF_F204)

	7	6	5	4	3	2	1	0
Bit symbol						INTTBOF0	INTTB01	INTTB00
Read/Write	R					R		
After reset	0					0	0	0
Function	This can be read as “0”.					0: Interrupt not generated 1: Interrupt generated	0: Interrupt not generated 1: Interrupt generated	0: Interrupt not generated 1: Interrupt generated

<INTTB00>: Interrupt generated if there is a match with timer register 0 (TB0RG0)
<INTTB01>: Interrupt generated if there is a match with timer register 1 (TB0RG1)
<INTTBOF0>: Interrupt generated if an up-counter overflow occurs

(Note) If any interrupt is generated, the flag that corresponds to the interrupt is set to TB0ST and the generation of interrupt is notified to INTC. The flag is cleared by reading the TB0ST register.

② When TB0RUN <TB0UDCE> = 1: Two-phase pulse input count mode

TB0ST (0xFFFF_F204)

	7	6	5	4	3	2	1	0
Bit symbol				INTTBUD0	INTTBUDF0	INTTBOUF0		
Read/Write	R			R			R	
After reset	0			0	0	0	0	
Function	This can be read as “0”.			Up-and-down count 0: Not generated 1: Generated	Underflow 0: Not generated 1: Generated	Overflow 0: Not generated 1: Generated	This can be read as “0”.	

<INTTB00>: Interrupt generated if there is a match with timer register 0 (TB0RG0)
<INTTB01>: Interrupt generated if there is a match with timer register 1 (TB0RG1)
<INTTBOUF0>: Interrupt generated if an up-and-down counter overflow occurs
<INTTBUDF0>: Interrupt generated if an up-and-down counter underflow occurs
<INTTBUD0>: Interrupt generated if an up- or down-count occurs

(Note) If any interrupt is generated, the flag that corresponds to the interrupt is set to TB0ST and the generation of interrupt is notified to INTC. The flag is cleared by reading the TB0ST register.

Fig. 11-6 TMRB-related Register

TBnRG0H/L and TBnRG1H/L timer registers

TBnRG0H/L timer registers (n=0 through 9)

TBnRG0L (0xFFFF_F2x8)

	7	6	5	4	3	2	1	0
Bit symbol	TBnRG0L 7	TBnRG0L 6	TBnRG0L 5	TBnRG0L 4	TBnRG0L 3	TBnRG0L 2	TBnRG0L 1	TBnRG0L 0
Read/Write	W							
After reset	Undefined							
Function	Timer count value, Data of low-order 8 bits							

TBnRG0H (0xFFFF_F2x9)

	7	6	5	4	3	2	1	0
Bit symbol	TBnRG0H 7	TBnRG0H 6	TBnRG0H 5	TBnRG0H 4	TBnRG0H 3	TBnRG0H 2	TBnRG0H 1	TBnRG0H 0
Read/Write	W							
After reset	Undefined							
Function	Timer count value, Data of high-order 8 bits							

(Note) To write data to the timer registers, use a 16-bit data transfer instruction or set the data in the order of low-order 8 bits followed by high-order.

TBnRG1H/L timer registers (n=0 through 9)

TBnRG1L (0xFFFF_F2xA)

	7	6	5	4	3	2	1	0
Bit symbol	TBnRG1L 7	TBnRG1L 6	TBnRG1L 5	TBnRG1L 4	TBnRG1L 3	TBnRG1L 2	TBnRG1L 1	TBnRG1L 0
Read/Write	W							
After reset	Undefined							
Function	Timer count value, Data of low-order 8 bits							

TBnRG1H (0xFFFF_F2xB)

	7	6	5	4	3	2	1	0
Bit symbol	TBnRG1H 7	TBnRG1H 6	TBnRG1H 5	TBnRG1H 4	TBnRG1H 3	TBnRG1H 2	TBnRG1H 1	TBnRG1H 0
Read/Write	W							
After reset	Undefined							
Function	Timer count value, Data of high-order 8 bits							

(Note) To write data to the timer registers, use a 16-bit data transfer instruction or set the data in the order of low-order 8 bits followed by high-order.

TBnCP0H/L and TBnCP1H/L capture registers

TBnCP0H/L capture registers (n=0 through 9)

TBnCP0L (0xFFFF_F2xC)

	7	6	5	4	3	2	1	0
Bit symbol	TBnCP0L 7	TBnCP0L 6	TBnCP0L 5	TBnCP0L 4	TBnCP0L 3	TBnCP0L 2	TBnCP0L 1	TBnCP0L 0
Read/Write	R							
After reset	Undefined							
Function	Timer capture value, Data of low-order 8 bits							

TBnCP0H (0xFFFF_F2xD)

	7	6	5	4	3	2	1	0
Bit symbol	TBnCP0H 7	TBnCP0H 6	TBnCP0H 5	TBnCP0H 4	TBnCP0H 3	TBnCP0H 2	TBnCP0H 1	TBnCP0H 0
Read/Write	R							
After reset	Undefined							
Function	Timer capture value, Data of high-order 8 bits							

(Note) To read data from the capture registers, use a 16-bit data transfer instruction or read out the data in the order of low-order 8 bits followed by high-order.

TBnCP1H/L capture registers (n=0 through 9)

TBnCP1L (0xFFFF_F2xE)

	7	6	5	4	3	2	1	0
Bit symbol	TBnCP1L 7	TBnCP1L 6	TBnCP1L 5	TBnCP1L 4	TBnCP1L 3	TBnCP1L 2	TBnCP1L 1	TBnCP1L 0
Read/Write	R							
After reset	Undefined							
Function	Timer capture value, Data of low-order 8 bits							

TBnCP1H (0xFFFF_F2xF)

	7	6	5	4	3	2	1	0
Bit symbol	TBnCP1H 7	TBnCP1H 6	TBnCP1H 5	TBnCP1H 4	TBnCP1H 3	TBnCP1H 2	TBnCP1H 1	TBnCP1H 0
Read/Write	R							
After reset	Undefined							
Function	Timer capture value, Data of high-order 8 bits							

(Note) To read data from the capture registers, use a 16-bit data transfer instruction or read out the data in the order of low-order 8 bits followed by high-order.

11.3 Description of Operations for Each Mode

11.3.1 16-bit Interval Timer Mode

Generating interrupts at periodic cycles

To generate the INTTB0 interrupt, specify a time interval in the TB0RG1 timer register.

		7	6	5	4	3	2	1	0	
TB0CR		1	0	X	X	X	X	X	X	Starts the TMRB0 module.
TB0RUN	←	0	0	0	0	–	0	X	0	Stops TMRB0.
IMC7	←	0	1	1	0	0	1	0	0	Enables INTTB0, and sets it to level 4. (Setting of INTTB0 only is shown here. This is a 32-bit register and requires settings of other interrupts as well.)
TB0FFCR	←	1	1	0	0	0	0	1	1	Disables the trigger.
TB0MOD	←	0	0	1	0	0	1	*	*	Designates the prescaler output clock as the input clock,
TB0RG1L	←	*	*	*	*	*	*	*	*	and specifies the time interval.
TB0RG1H		*	*	*	*	*	*	*	*	(16 bits)
TB0RUN	←	0	0	0	0	–	1	X	1	Starts TMRB0.

X; Don't care –; no change

11.3.2 16-bit Event Counter Mode

It is possible to make it the event counter by using an input clock as an external clock (TB0IN0 pin input).

The up-counter counts up on the rising edge of TB0IN0 pin input. It is possible to read the count value by capturing value using software and reading the captured value.

		7	6	5	4	3	2	1	0	
TB0CR	←	1	0	X	X	X	X	X	X	Starts the TMRB0 module.
TB0RUN	←	0	0	0	0	–	0	X	0	Stops TMRB0.
P6CR	←	–	–	–	–	0	–	–	–	Sets P63 to the input mode.
P6FC2	←	–	–	–	–	1	–	–	–	
P6IE	←	–	–	–	–	1	–	–	–	
IMC8	←	0	1	1	0	0	1	0	0	Enables INTTB0, and sets it to level 4. (Setting of INTTB0 only is shown here. This is a 32-bit register and requires settings of other interrupts as well.)
TB0FFCR	←	1	1	0	0	0	0	1	1	Disables the trigger.
TB0MOD	←	0	0	1	0	0	1	0	0	Designates the TB0IN0 pin input as the input clock.
TB0RUN	←	0	0	0	0	–	1	X	1	Starts TMRB0.
TB0MOD	←	X	X	0	0	0	1	0	0	Captures a value using software.
TB0RG1L	←	*	*	*	*	*	*	*	*	Specifies the time interval.
TB0RG1H		*	*	*	*	*	*	*	*	(16 bits)

X; Don't care –; no change

To be used as the event counter, put the prescaler in a “RUN” state (TB0RUN<TB0PRUN> = “1”).

11.3.3 16-bit Programmable Square Wave Output Mode (PPG)

Square waves with any frequency and any duty (programmable square waves) can be output. The output pulse can be either low-active or high-active.

Programmable square waves can be output from the TB0OUT pin by triggering the timer flip-flop (TB0FF) to reverse when the set value of the up-counter (UCO) matches the set values of the timer registers (TB0RG0H/L and TB0RG1H/L). Note that the set values of TB0RG0H/L and TB0RG1H/L must satisfy the following requirement:

(Set value of TB0RG0H/L) < (Set value of TB0RG1H/L)

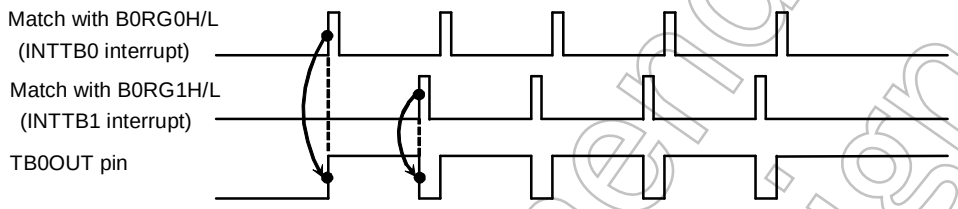


Fig. 11-7 Example of Output of Programmable Square Wave (PPG)

In this mode, by enabling the double buffering of TB0RG0H/L, the value of register buffer 0 is shifted into TB0RG0H/L when the set value of the up-counter matches the set value of TB0RG1H/L. This facilitates handling of small duties.

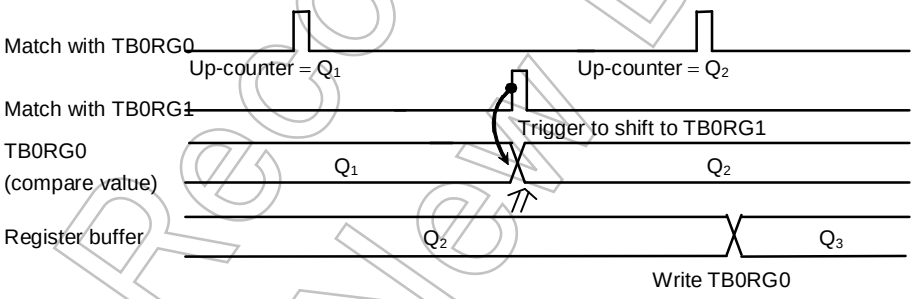


Fig. 11-8 Register Buffer Operation

Note: Double buffering is available for TB0RG0 only. TB0RG1 change must be completed before the next match. As for TMRB1 through TMRB9, TBxREG0 and TBxREG1 have a double buffer configuration.

The block diagram of this mode is shown below.

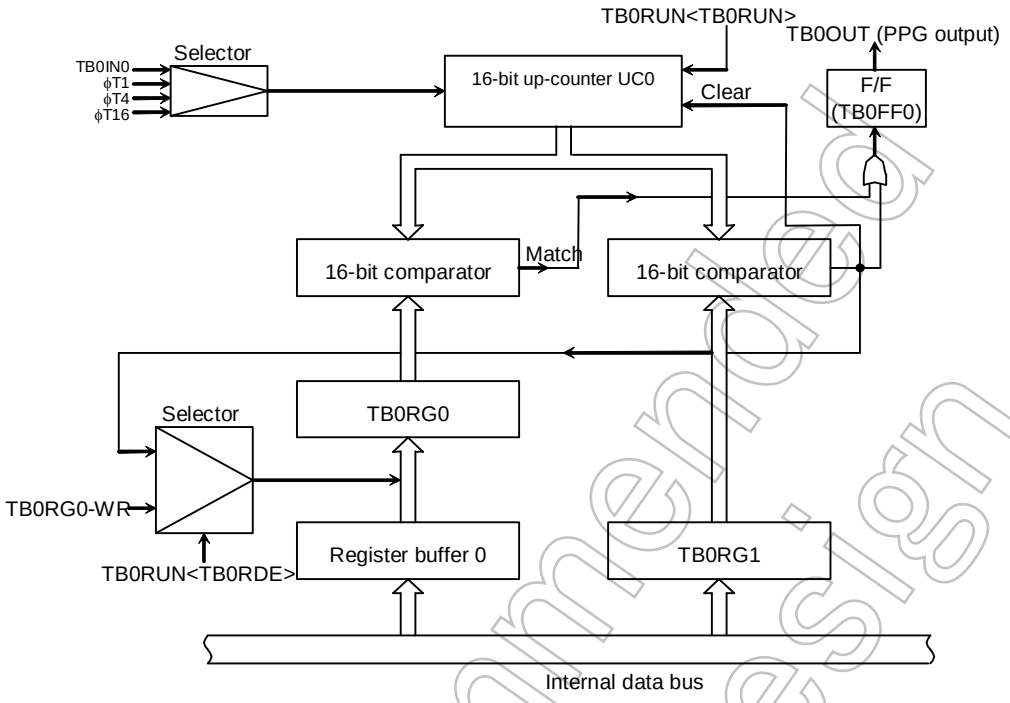


Fig. 11-9 Block Diagram of 16-bit PPG Mode

Each register in the 16-bit PPG output mode must be programmed as listed below.

		7	6	5	4	3	2	1	0	
TB0CR	←	1	0	X	X	X	X	X	X	Starts the TMRB0 module.
TB0RUN	←	0	0	0	0	–	0	X	0	Disables the TB0RG0 double buffering and stops TMRB0.
TB0RG0L	←	*	*	*	*	*	*	*	*	Specifies a duty. (16 bits)
TB0RG0H	←	*	*	*	*	*	*	*	*	
TB0RG1L	←	*	*	*	*	*	*	*	*	Specifies a cycle. (16 bits)
TB0RG1H	←	*	*	*	*	*	*	*	*	
TB0RUN	←	1	0	0	0	–	0	X	0	Enables the TB0RG0 double buffering. (Changes the duty/cycle when the INTTB0 interrupt is generated)
TB0FFCR	←	X	X	0	0	1	1	1	0	Specifies to trigger TB0FF0 to reverse when a match with TB0RG0 or TB0RG1 is detected, and sets the initial value of TB0FF0 to "0."
TB0MOD	←	0	0	1	0	0	1	*	*	Designates the prescaler output clock as the input clock, and disables the capture function. (** = 01, 10, 11)
P6CR	←	–	–	–	–	–	–	–	1	Assigns P54 to TB0OUT.
P6FC3	←	–	–	–	–	–	–	–	1	
TB0RUN	←	1	0	0	0	–	1	X	1	Starts TMRB0.

X; Don't care –; no change

11.4 Applications using the Capture Function

The capture function can be used to develop many applications, including those described below:

① One-shot pulse output triggered by an external pulse

② Frequency measurement

③ Pulse width measurement

④ Time difference measurement

① One-shot pulse output triggered by an external pulse

One-shot pulse output triggered by an external pulse is carried out as follows:

The 16-bit up-counter is made to count up by putting it in a free-running state using the prescaler output clock. An external pulse is input through the TB5IN0 pin. A trigger is generated at the rising of the external pulse by using the capture function and the value of the up-counter is taken into the capture registers (TB5CP0H/L).

The INTC must be programmed so that an interrupt INT2 is generated at the rising of an external trigger pulse. This interrupt is used to set the timer registers (TB5RG0H/L) to the sum of the TB5CP0 value (c) and the delay time (d), (c + d), and set the timer registers (TB5RG1H/L) to the sum of the TB5RG0H/L values and the pulse width (p) of one-shot pulse, (c + d + p).

TB5RG1 change must be completed before the next match.

In addition, the timer flip-flop control registers (TB5FFCR<TB5E1T1, TB5E0T1>) must be set to "11." This enables triggering the timer flip-flop (TB5FF0) to reverse when UC5 matches TB5RG0H/L and TB5RG1H/L. This trigger is disabled by the INTTB5 interrupt after a one-shot pulse is output.

Symbols (c), (d) and (p) used in the text correspond to symbols c, d and p in Fig. 11-10 One-shot Pulse Output (With Delay)."

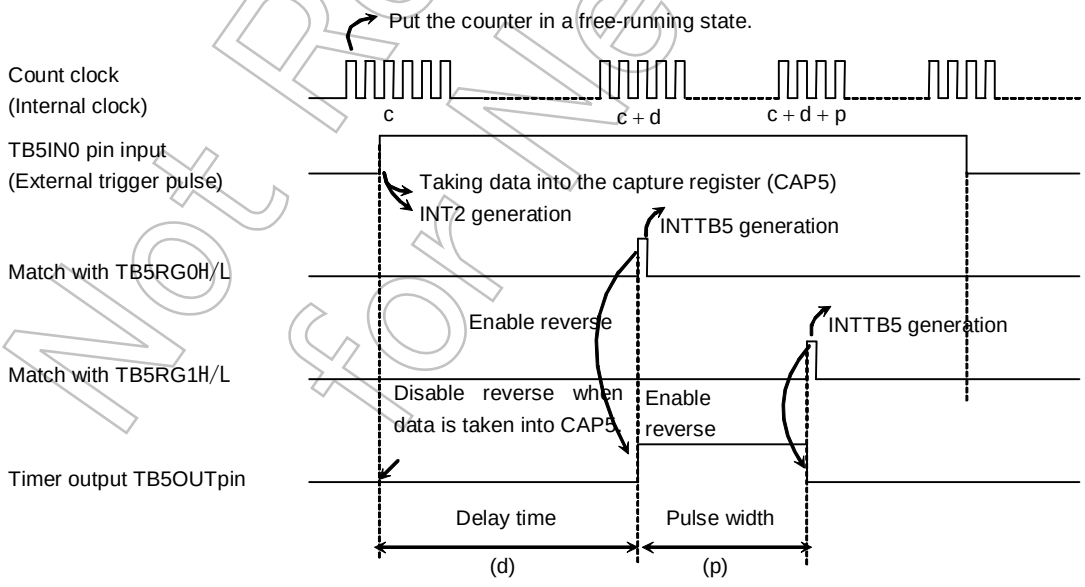


Fig. 11-10 One-shot Pulse Output (With Delay)

Programming example: Output a 2-ms one-shot pulse triggered by an external pulse from the TB5IN0 pin with a 3-ms delay

* Clock conditions

System clock : High speed (fc)
High-speed clock gear : 1X (fc)
Prescaler clock : fperiph/4 (fperiph · fsys)

Main programming

Puts to a free-running state.

	7	6	5	4	3	2	1	0	
PBCR	X	X	X	X	0	X	X	X	Set PB3 pin to TB5IN0.
PBFC1	X	X	X	X	1	X	X	X	
PBIE	X	X	X	X	1	X	X	X	

Uses φT1 for counting.

TB5MOD	←	X	X	1	0	1	0	0	1
TB5FFCR	←	X	X	0	0	0	0	1	0
PBCR	←	1	–	–	–	–	–	–	–
PBFC1	←	1	–	–	–	–	–	–	–
IMC0	←	X	1	1	0	0	1	0	0
IMC8	←	X	1	1	0	0	0	0	0
TB5RUN	←	–	0	X	0	–	1	X	1

Takes data into TB5CP0 at the rising of TB5IN0 input
Clears TB5FF0 to zero
Disables TB5FF0 to reverse
Assigns PB7 pin to TB5OUT
Enables INT2 and disables INTTB5.
These are 32-bit registers and must be all processed.
Starts TMRB5.

INT2 programming

TB5RG0	←	TB0CP0 + 3ms/φT1
TB5RG1	←	TB0RG0 + 2ms/φT1

TB5FFCR	←	X	X	–	–	1	1	–	–
IMC0	←	X	1	1	0	0	1	0	0

Enables TB2FF0 to reverse when there is a match with TB2RG0, TB2RG1.
Enables INTTB5.
These are 32-bit registers and must be all processed.

INTTB5 programming

TB5FFCR	←	X	X	–	–	0	0	–	–
IMC8	←	X	1	1	0	0	0	0	0

Disables TB5FF0 to reverse when there is a match with TB5RG0, 1
Disables INTTB5
These are 32-bit registers and must be all processed.

X; Don't care —;no change

If a delay is not required, TB5FF0 is reversed when data is taken into TB5CP0, and TB5RG1 is set to the sum of the TB5CPO value (c) and the one-shot pulse width (p), (c + p), by generating the INT0 interrupt. TB5RG1 change must be completed before the next match. TB5FF0 is enabled to reverse when UC5 matches with TB5RG1, and is disabled by generating the INTTB5 interrupt.

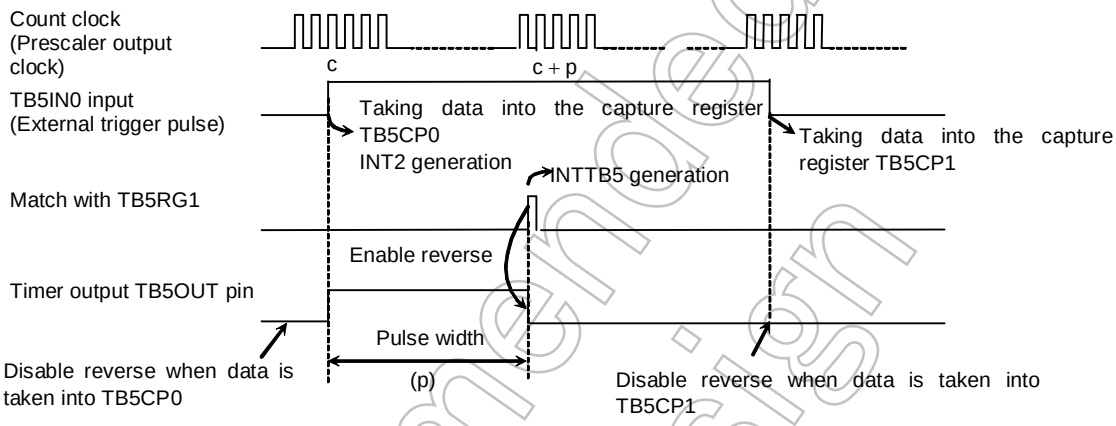


Fig. 11-11 One-shot Pulse Output Triggered by an External Pulse (Without Delay)

② Frequency measurement

The frequency of an external clock can be measured by using the capture function.

To measure frequency, another 16-bit timer (TMRB0) is used in combination with the 16-bit event counter mode (TMRB0 reverses TB0FFCR to specify the measurement time).

The TB3IN0 pin input is selected as the TMRB3 count clock to perform the count operation using an external input clock. TB3MOD<TB3CPM1:0> is set to "11." This setting allows a count value of the 16-bit up-counter UC3 to be taken into the capture register (TB3CP0) upon rising of a timer flip-flop (TB0FFCR) of the 16-bit timer (TMRB0), and an UC3 counter value to be taken into the capture register (TB3CP1) upon falling of TB0FF of the 16-bit timer (TMRB0).

A frequency is then obtained from the difference between TB3CP0 and TB3CP1 based on the measurement, by generating the INTTB0 16-bit timer interrupt.

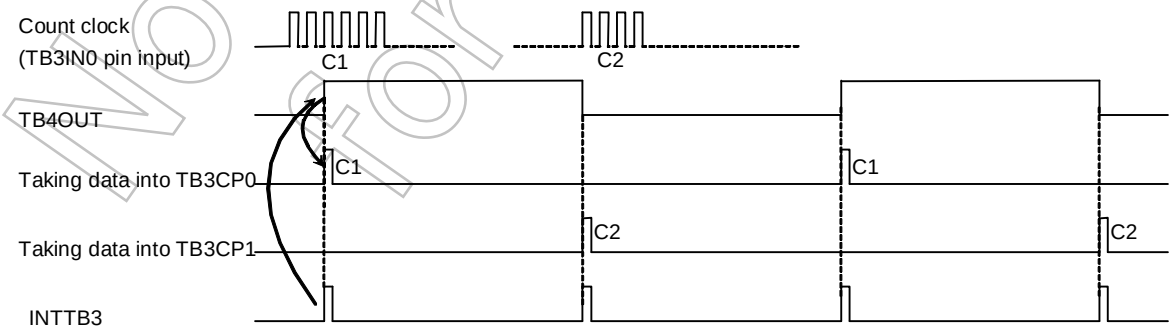


Fig. 11-12 Frequency Measurement

For example, if the set width of TB0FF level "1" of the 16-bit timer is 0.5 s and if the difference between TB3CP0 and TB3CP1 is 100, the frequency is 100 / 0.5 s = 200 Hz.

③ Pulse width measurement

By using the capture function, the “H” level width of an external pulse can be measured. Specifically, by putting it in a free-running state using the prescaler output clock, an external pulse is input through the TB5IN0 pin and the up-counter (UC5) is made to count up. A trigger is generated at each rising and falling edge of the external pulse by using the capture function and the value of the up-counter is taken into the capture registers (TB5CP0, TB5CP1). The INTC must be programmed so that INT2 is generated at the falling edge of an external pulse input through the TB5IN0 pin.

The “H” level pulse width can be calculated by multiplying the difference between TB5CP0 and TB5CP1 by the clock cycle of an internal clock.

For example, if the difference between TB5CP0 and TB5CP1 is 100 and the cycle of the prescaler output clock is 0.5 *u*s, the pulse width is 100 × 0.5 *u*s = 50 *u*s.

Caution must be exercised when measuring pulse widths exceeding the UC5 maximum count time which is dependant upon the source clock used. The measurement of such pulse widths must be made using software.

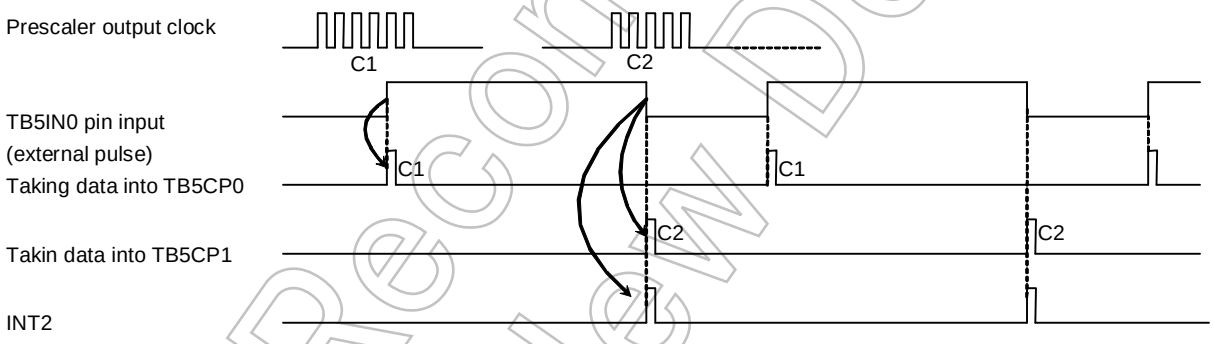


Fig. 11-13 Pulse Width Measurement

The “L” level width of an external pulse can also be measured. In such cases, the difference between C2 generated the first time and C1 generated the second time is initially obtained by performing the second stage of INT0 interrupt processing as shown in “Fig. 11-17 Time Difference Measurement” and this difference is multiplied by the cycle of the prescaler output clock to obtain the “L” level width.

④ Time Difference Measurement

The up-counter (UC5) is made to count up by putting it in a free-running state using the prescaler output clock. The value of UC5 is taken into the capture register (TB5CP0) at the rising edge of the TB5IN0 pin input pulse. The INTC must be programmed to generate INT2 interrupt at this time.

The value of UC5 is taken into the capture register TB5CP1 at the rising edge of the TB5CP1 pin input pulse. The INTC must be programmed to generate INT1 interrupt at this time.

The time difference can be calculated by multiplying the difference between TB5CP1 and TB5CP0 by the clock cycle of an internal clock.

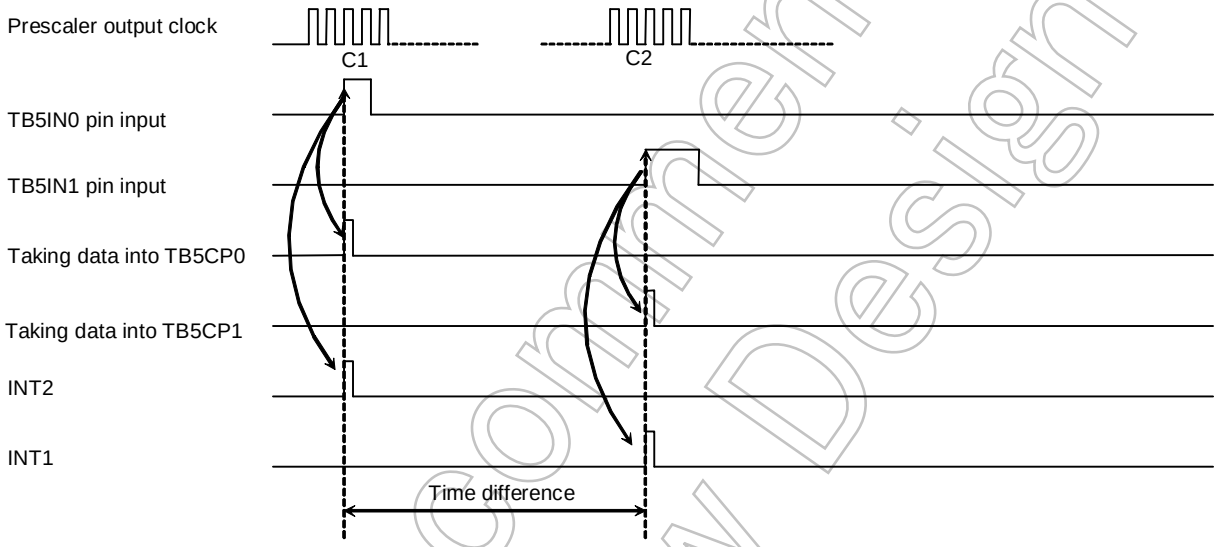


Fig. 11-14 Time Difference Measurement

11.5 Two-phase Pulse Input Count Mode (TMRB0)

In this mode, the counter is incremented or decremented by one depending on the state transition of the two-phase clock that is input through TB0IN0 and TB0IN1 and has phase difference. Interrupt is output in the ups and downs counter mode by the count operation.

There are two counting operation modes, which are switched by the register setting.

1) Normal operation mode (up/down at the fourth count)

2) Quadruple frequency mode (up/down at each count)

3) Double frequency mode (up at each count) Count-up is available by an input from a pin.

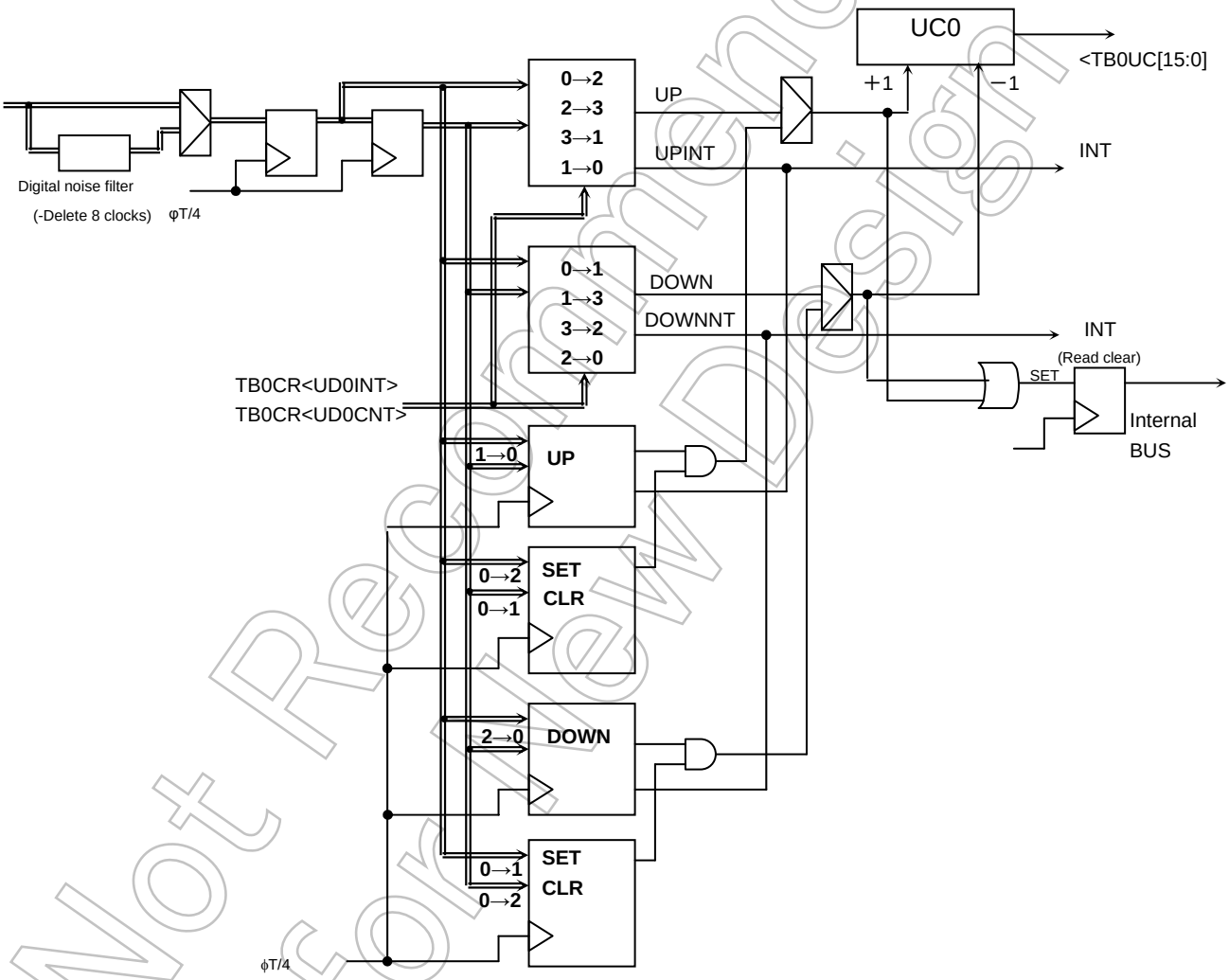
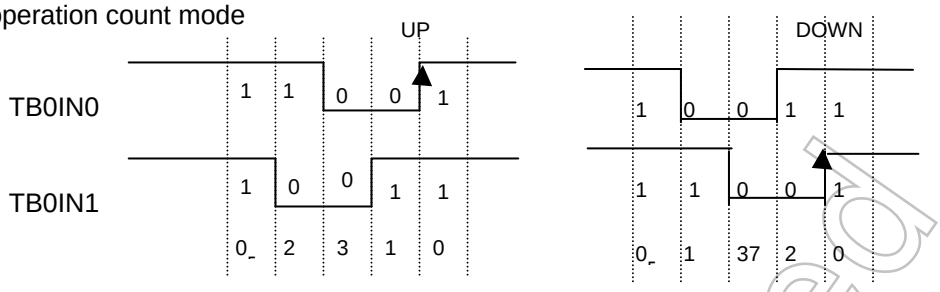


Fig. 11-15 Count Circuit of Two-phase Counter

・Normal operation count mode



	Pin state						
Count condition	UP			DOWN			
TB0IN0,TB0IN1	0	→	2	0	→	1	←SET
		↓					
TB0IN0,TB0IN1	1	→	0	2	→	0	←Interrupt generated
TB0IN0,TB0IN1	0	→	1	0	→	2	←CLR

Count conditions

・UP: COUNT UP is triggered by a change from 1 to 0 with a change history from 0 to 2 (UP SET).

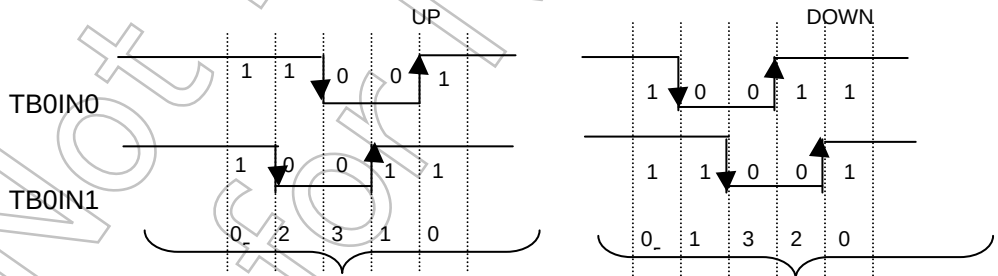
CLEAR is triggered by a change from 0 to 1 (reversed) with a change history from 0 to 2 (UP SET).

・DOWN: COUNT DOWN is triggered by a change from 2 to 0 with a change history from 0 to 1 (DOWN SET).

CLEAR is triggered by a change from 0 to 2 (reversed) while a change from 0 to 1 (DOWN SET) is set.

Note: Changes from 0 to 3 and from 3 to 0 are considered as irregular states and are not counted. Up and down state settings are cleared.

・Multiplication-by-4 operation count mode



① Count up at each edge **② Count down at each edge**

	Pin state					
Count condition	UP			DOWN		
TB0IN0,TB0IN1	0	→	2	0	→	1
	2		3	1		3
	3		1	3		2
	1		0	2		0

TMRB0RUN register (TB0RUN)

TB0RUN (0xFFFF_F200)

	7	6	5	4	3	2	1	0
Bit symbol	TB0RDE		UD0CMP	TB0UDCE	I2TB0	TB0PRUN		TB0RUN
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
After reset	0	0	0	0	0	0	0	0
Function	Double Buffer 0: Disable 1: Enable	Write "0".	Compare function 0: Enable 1: Disable.	Enable/ disable two-phase counter 0: Disable 1: Enable	IDLE 0: Stop 1: Operate	Timer Run/Stop Control 0: Stop & Clear 1: Run (Count Up)		

Fig. 11-16 Two-phase Pulse Input Count Mode Setting Register

① Operation mode

Register setting determines whether the external input signals from the TB0IN0 and TB0IN1 input pins are input to the normal 16-bit timer (capture input) or the up-and-down counter.

- In the up-and-down counter mode, capture is executed by the software only. Capture at the external clock timing does not work.
- In the up-and-down counter mode, the comparator is disabled and it does not execute comparison with timer registers (the comparator is activated when the compare function is being set).
- The input clock sampling is executed by a system clock. The maximum input frequency is ϕT0/4.

<< How to program the up-and-down counter >>

Set the TB0MOD register < TB0CLK0, TB0CLK1> to "00" (prescaler OFF). Then, program the fourth bit < TB0UDCE > of the TB0RUN register to determine whether to operate the counter as the up-and-down counter or as the conventional up-counter for external clock input.

TB0UDCE (Enable the up-and-down counter) = "0": Normal 16-bit timer operation
="1": Up-and-down counter operation

② Interrupt

- In the NORMAL or SLOW mode

The INTTB0 interrupt is enabled using the interrupt controller (INTC). The INTTB0 interrupt is generated by counting up or down. Reading the status register TB0ST during interrupt handling allows simultaneous check for occurrences of an overflow and an underflow. If TB0ST<INTTBOUF0> is "1," it indicates that an overflow has occurred. If <INTTBUDF0> is "1," it indicates that an underflow has occurred. This register is cleared after it is read. The counter becomes 0x0000 when an overflow occurs, and it becomes 0xFFFF when an underflow occurs. After that, the counter continues the counting operation.

- When the compare function is being used (TB0RUN <UD0CNP>)

An interrupt triggered by the counting is generated if counted value matches that of the comparator. Interrupt is not generated by each count-up/down or at an underflow/ overflow (flag is set at the underflow/ overflow).

(Note) Caution must be exercised in rewriting the value that is set in the register when an overflow/ underflow or a match is going to happen.

TMRB0 status register

TB0ST (0xFFFF_F204)

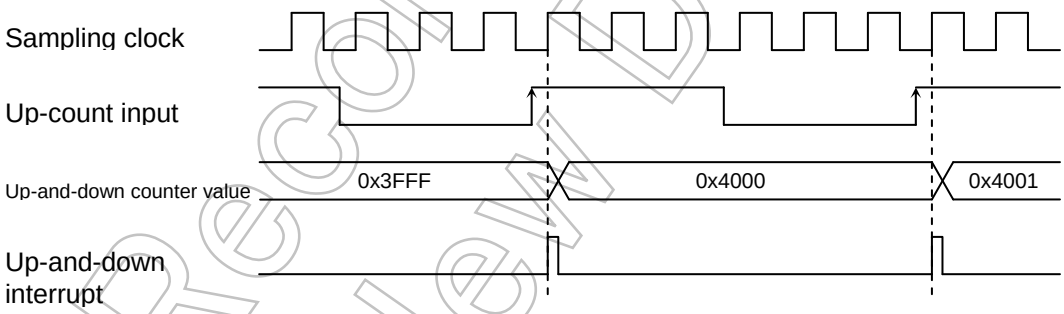
	7	6	5	4	3	2	1	0
Bit symbol				INTTBUD0	INTTBUDF0	INTTBOUF0	INTTB01	INTTB00
Read/Write	R			R			R	
After reset	0			0	0	0	0	
Function	This can be read as “0”.			Up-and-down count 0: Not occurred 1: Occurred	Underflow 0: Not occurred 1: Occurred	Overflow 0: Not occurred 1: Occurred	Interrupt triggered by a match 1: Not generated 2: Generated	Interrupt triggered by a match 1: Not generated 2: Generated

Fig. 11-17 TMRB0 status register

(Note): The status is cleared after the register is read.

③ Up-and-down counter

When the two-phase input count mode is selected (TB0RUN<TB0UDCE> = “1”), the up-counter becomes the up-and-down counter and it is initialized to 0x7FFF. If a counter overflow occurs, the counter returns to 0x0000. If a counter underflow occurs, the counter returns to 0xFFFF. After that, the counter continues the counting operation. Therefore, the state can be checked by reading the counter value and the status flag TB0ST after an interrupt is generated.



(Note 1) The up (down) count input must be set to the “H” level for the states before and after an input.

(Note 2) Reading of counter value must be executed during INTTB0 interrupt handling.

TMRB0 control register

TB0CR (0xFFFF_F201)

	7	6	5	4	3	2	1	0
Bit symbol	TB0EN					UD0NF	UD01CNT	UD01CNT
Read/Write	R/W		R			R/W		
After reset	0	0	0	0	0	0	0	0
Function	TMRB0 operation 0: Disable 1: Enable	Write “0”.	This can be read as “0”.	This can be read as “0”.	This can be read as “0”.	Digital noise filter 0:No use 1:Use	Mode switch-over 00:Normal 01:TB0IN0 x2 10:Quadruple 11:TB0IN1 x2	

UD0xCNT Mode switch-over bit

00: Normal mode

01:TB0IN0 double frequency mode Not affected by the input condition of TB0IN1.

10: Quadruple frequency mode

11:TB0IN1 double frequency mode Not affected by the input condition of TB0IN0.

UD0NF Controls noise removal.

If this is set to “1 (Use),” the TMRB0 pin input is removed when it is shorter than 8 system clocks. Pay close attention to the input signal frequency because an error of one system clock occurs due to synchronization with internal signals.

Fig. 11-18 TMRB0 control register

11.6 Two-phase Pulse Input Waveform Output Mode (TMRB0)

In this mode, TB0OUT0 and TB0OUT1 output the state of the two-phase clock that is input through TB0IN0 and TB0IN1 and has phase difference.

1)Normal output mode (output the input waveform)

2)Reversed output mode (output the reversed input waveform)

3)Double frequency output mode (Output the XORed waveform of TB0IN0 and TB0IN1 from T0OUT0)

TMRB0 output control register

TB0OUTCR (0xFFFF_F205)

	7	6	5	4	3	2	1	0
Bit symbol					UD1OUT1	UD1OUT0	UD0OUT1	UD0OUT0
Read/Write	R/W	R/W	R/W	R/W	R/W		R/W	
After reset	0	0	0	0	0	0	0	0
Function	This can be read as “0”.	This can be read as “0”.	This can be read as “0”.	This can be read as “0”.	TB0OUT1 output control 00: TB0IN1 01: Reverse TB0IN1 10: Double 11: Don’t care		TB0OUT0 output control 00: TB0IN0 01: Reverse TB0IN0 10: Double 11: Don’t care	

<UDxOUT1:0>:Output the input to the two-phase counter from TB0OUT0 and TB0OUT1.

“00”: Output the timer flip-flop.

“01”: Output the TB0INx input waveform.

“10”: Output the reversed TB0INx input waveform.

“11”: Output the XORed waveform of TB0IN0 and TB0IN1 from T0OUT0

Fig. 11-19 TMRB0 output control register

12. 16-Bit Timer/ Event Counter (TMRDs)

Both of the two channels (TMRB0 through TMR1) have a multi-functional 16-bit timer/event counter. TMRDs operate in the following four operation modes:

- 16-bit interval timer mode
- 16-bit event counter mode
- 16-bit programmable square-wave output mode (PPG)
- Only TMRB0 provides two-phase pulse input counter mode (quad/normal-speed) and two-phase pulse output mode TMRB0.
- Timer synchronous mode (2ch synchronous output available)

Each channel consists of a 16-bit up-counter, five 16-bit timer registers (one of which is double-buffered), five comparators, a capture input control, a timer flip-flop and its associated control circuit. Timer operation modes and the timer flip-flop are controlled by a register.

Each channel (TMRD0, TMRD1) functions independently and the channels operate in the same way, except for the differences in their specifications as shown in Table 12-1. Therefore, the operational descriptions here are only for TMRD0 and for the operation in the synchronous mode. This module is equipped with two channels of PPGs that can program the timings of the rising edge and the falling edge for one cycle.

The following channel can be used as the synchronous trigger.

Start trigger of the timer synchronous mode using TDxRUN
/Start TMRD0 => TMRD1at the same timing

A slave module outputs the PPG waveform which is in the same or the 90 degree delayed cycle of the master module by connecting the master module to the slave module. The master module is used to set the phase.

Table 12-1 PPG Output Phase Selection

		Operation mode				
		Timer	Single	Master	Slave S[*1]	Slave Q[*2]
Phase selection	0°	×	○		○[*3]	×
	90°	×				○[*3]

*1 Slave S: The synchronous mode of the master and the slave modules.
*2 Slave Q: The mode of the delayed slave output by the 90 degree cycle from the master output.
*3 The CPO of the master module is useful for setting cycles. CP1 through CP4 of the slave module are useful for setting PPG waveforms.

Table 12-2 Compare Register Function

		Operation mode			
		Timer	Single	Master	Slave (S/Q)
Compare registers	TDnCP0	CMPINT0	Cycles [*1]		×[*2]
	TDnCP1	CMPINT1	Set PPG0[*1]		
	TDnCP2	CMPINT2	Clear PPG0[*1]		
	TDnCP3	CMPINT3	Set PPG1[*1]		
	TDnCP4	CMPINT4	Clear PPG1[*1]		

*1: Each one causes an interruption due to CPx matching.
*2: The slave cycle is specified by the master CP0. No interrupt caused by the slave CP0 is generated.
*3: n=0, 1

Table 12-3 Differences in the Specifications of TMRD Modules

Specification \ Channel		TMRD0	TMRD1
External pins	Timer flip-flop output pin	TD0OUT0 (shared with Pxx) TD0OUT1 (shared with Pxx)	TD1OUT0 (shared with Pxx) TD1OUT1 (shared with Pxx)
Internal signals	Timer for synchronous start trigger	–	TMRD0
Register names (addresses)	Timer RUN register	TD0RUN (0xFFFF_E900)	TD1RUN (0xFFFF_E940)
	Timer control register	TD0CR (0xFFFF_E904)	TD1CR (0xFFFF_E944)
	Timer mode register	TD0MOD (0xFFFF_E908)	TD1MOD (0xFFFF_E948)
	Timer flip-flop control register	TD0FFCR (0xFFFF_E90C)	TD1FFCR (0xFFFF_E94C)
	Timer status register	TD0ST (0xFFFF_E910)	TD1ST (0xFFFF_E950)
	Timer up counter register	TD0UC	TD1UC
	Timer register	TD0RG0 (0xFFFF_ E914) TD0RG1 (0xFFFF_ E918) TD0RG2 (0xFFFF_ E91C) TD0RG3 (0xFFFF_ E920) TD0RG4 (0xFFFF_ E924)	TD1RG0 (0xFFFF_ E954) TD1RG1 (0xFFFF_ E958) TD1RG2 (0xFFFF_ E95C) TD1RG3 (0xFFFF_ E960) TD1RG4 (0xFFFF_ E964)

TMRD input clock setting

PWMCG Mode Register

PWMCG (0xFFFF_EE28)

	7	6	5	4	3	2	1	0
Bit symbol			PWMGEAR1	PWMGEAR0				PWMEN
Read/Write	R				R			
After reset	0	0	1	1	0	0	0	0
Function	This can be read as “0”.		Select source clock 00: PLL 16x 01: PLL octuple 10: PLL quadruple 11: GCLK		This can be read as “0”.			0:PWMCLK OFF 1:PWMCLK ON

Fig. 12-4 PWMCG Mode Register

One out of four clocks (16 times PPL, octuple PPL, quadruple PLL and GCLK) can be selected as a source clock to input TMRD block.

(Note) When using TMRD, set PWMCG register to PWMEN=”1” to provide clocks.

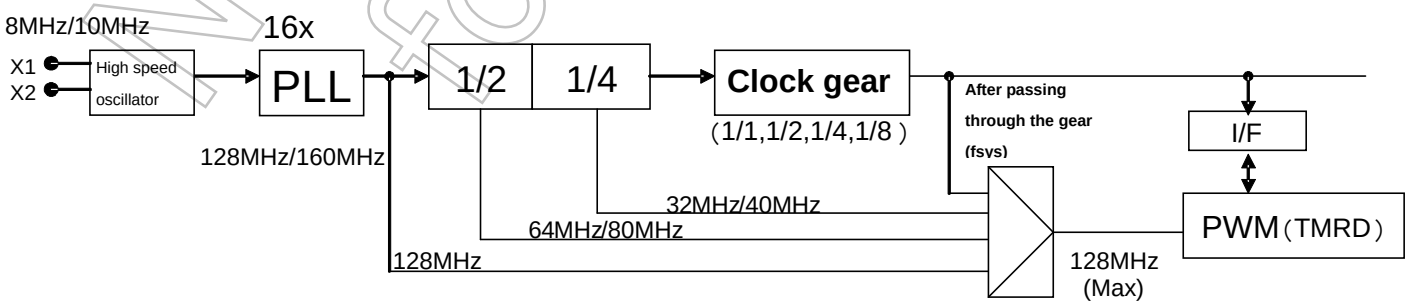


Fig. 12-5 PWM Clock Systematic Chart

The above can be handled by using prescaler after TMRD block is input.

12.1 Block Diagram

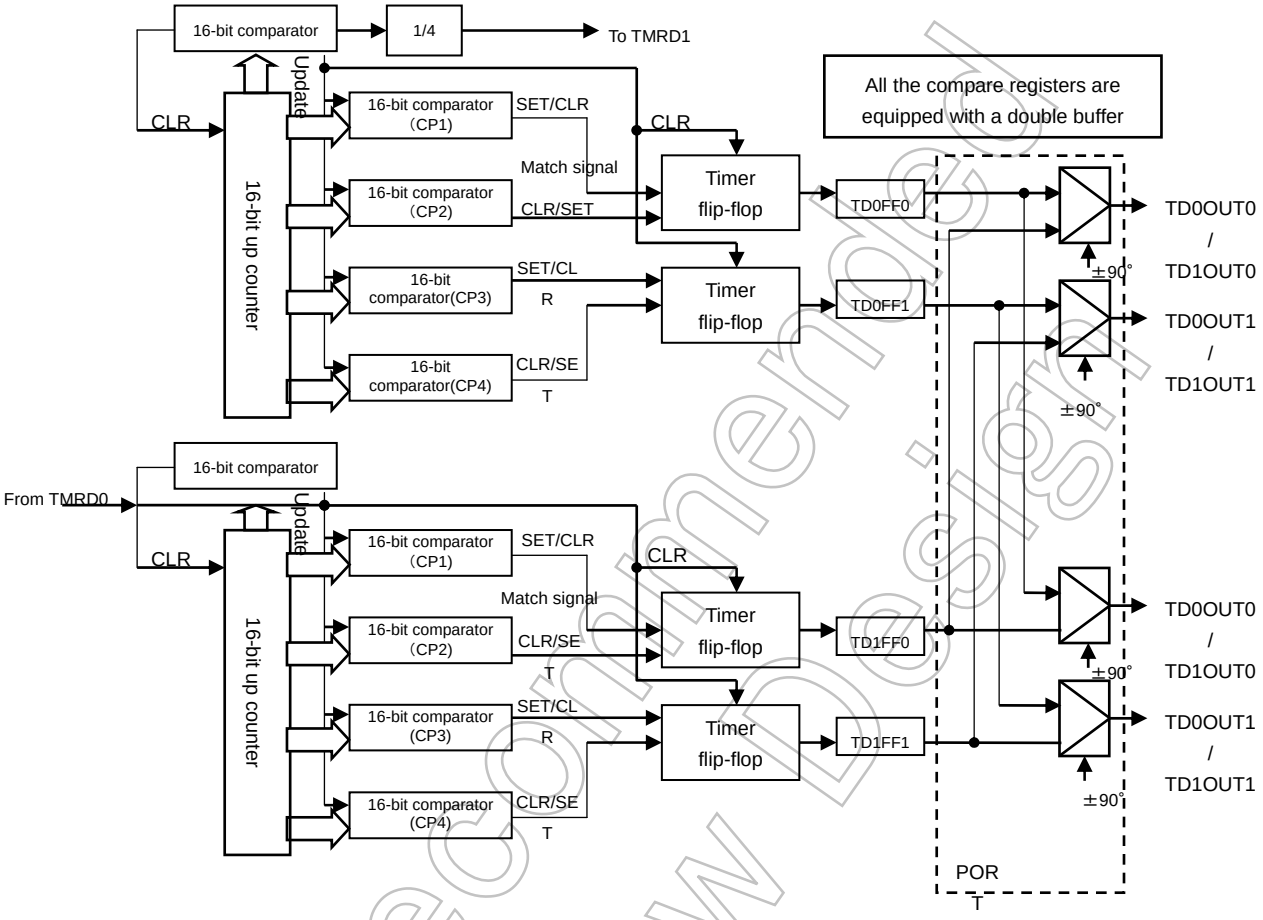


Fig. 12-6 TMRD Block Diagram

16-bit up-counter is used for measuring the timings.

When it reaches to the value set in TD0CP0, it represents one cycle of a waveform.

When it reaches to the values set in TD0CP1 and TD0CP2, the rising and falling edges of the TD0OUT0 signal are generated respectively. The same applies to TD0CP3 and TD0CP4 that for generating TD0OUT1 signal.

If the TMRD0 and TMRD1 that have the same constructions and functions are connected, TMRD1 conforms to the cycle specified by TD0CP0; therefore 4 signals are counted by the same cycle.

12.1.1 Up-counter (UC0) and Up-counter Capture Registers (TD0UC)

This is the 16-bit binary counter that counts up in response to the input clock specified by TD0MOD<TD0CLK1: 0>.

UC0 input clock can be selected from either five types – 1/1, 1/2, 1/4, 1/8 and 1/16- of counting speed by using a prescaler. For UC0, start, stop and clear are specified by TD0RUN<TD0RUN> and if UC0 matches the TD0RG0 timer register, it is cleared to "0" provided the setting is "clear enable." Clear enable/disable is specified by TD0MOD<TD0CLE>.

If the setting is "clear disable," the counter operates as a free-running counter.

The current count value of the UC0 can be captured by reading the TD0UC register.

12.1.2 Timer Registers (TD0CMP0, TD0CMP1, TD0CMP2, TD0CMP3, TD0CMP4)

These are 16-bit registers for specifying counter values and five registers are built into each channel. If a value set on this timer register matches that on a UC0 up-counter, the match detection signal of the comparator becomes active.

TD0RG0 of this timer register is paired with register buffer 0 - a double-buffered configuration. TD0RG0 uses TD0RUN<TD0RDE> to control the enabling/disabling of double buffering. If <TD0RDE> = "0", double buffering is disabled and if <TD0RDE> = "1", it is enabled. If double buffering is enabled, data is transferred from register buffer 0 to the TD0RG0 timer register when there is a match between UC0 and TD0RG1.

The values of TD0RG0 through TD0RG4 become undefined after a reset; therefore it is necessary to write data to them beforehand in case of using a 16-bit timer. A reset initializes TD0RUN <TD0RDE> to "0" and sets double buffering to "disable." To use double buffering, write data to the timer register, set <TD0RDE> to "1" and then write the following data to the register buffers.

TD0RG0 and the register buffers are assigned to the same address: 0xFFFF_E914. If <TD0RDE> = "0", the same value is written to TD0RG0 and each register buffer; if <TD0RDE> = "1," the value is only written to each register buffer. To write an initial value to the timer register, therefore, the register buffers must be set to "disable."

12.1.3 Comparators (CP0, CP1, CP2, CP3, CP4)

These are 16-bit comparators for detecting a match by comparing set values of the UC0 up-counter with set values of the TD0RG0 through TD0RG4 timer registers. If a match is detected, INT0TD0 through INT0TD4 is generated.

12.2 Register Description

TMRDn RUN register (n=0, 1)

TDnRUN (0xFFFF_E900)

	7	6	5	4	3	2	1	0
Bit symbol					I2TDn			TDnRUN
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R	R/W
After reset	0	0	0	0	0	0	0	0
Function								Timer Run/Stop Control 0: Stop & clear 1: Count * This can be read as "0."
	15	14	13	12	11	10	9	8
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

<TDnRUN>: Controls the TMRD0 count operation.

Timer mode: Count-up is started and the contents in the double buffer are updated by writing "1" to this bit. Clearing the bit makes the counter be stopped and be initialized.

PPG single mode: By writing "1" to this bit, the double buffer is updated and the count-up is started. The count-up stops immediately if this bit is cleared when TDSFT is "1".

PPG master mode: The operation of the master module is started by setting "1" to this bit when TDSFT is "0". The operations of this mode and PPG single mode are the same except for the slave module control.

PPG slave mode: Write "0" to this bit at the slave mode. This mode is disabled by writing "1" to this bit.

Fig. 12-7 TMRDn RUN Register

TMRDn control register (n=0, 1)

TDnCR (0xFFFF_E904)

	7	6	5	4	3	2	1	0
Bit symbol	TDnEN	TDnI2TD	TDnMOD1	TDnMOD0	TDnRDE		TDnISO1	TDnISO0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	0	0	0
Function	Specify operation 0 :TMRD stop 1 :TRMD operation	Specify operation /stop during IDLE 0 : Stop 1 : Operation	Select operation mode 00 : Timer mode 01 : PPG single more 10 : PPG master mode 11 : PPG slave mode		Double buffer enable control 0 : Not used 1 : Used *** The master module needs to be set identical to the slave module.**		Select interrupt factor of CMPnINT0 00:No factor 01 : CP0 match 10 : CP0 90° match (only at the master mode) 11 : Counter overflow (only at the slave mode)	
	15	14	13	12	11	10	9	8
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Fig. 12-8 TMRDn Control Register

<TDnISO 1:0>: For selecting CMPnINT0 interrupt factor.
00: No interrupt factor (not to use interrupt)
01: An interrupt occurs when it conforms to TDxCP0 cycle.
10: An interrupt occurs when it conforms to TD0CP0 with 1/4 delay (only at master mode).
11: An interrupt occurs when 16-bit counter overflows.

<TDnRDE>: For selecting use/nonuse of double-buffer.
0: Non-using double-buffering.
1: Using double-buffering
*Master and Slave modes must have the same setup.

<TDnMOD1:0>: For setting the operation mode.
00: Timer mode
01: PPG single mode
10: PPG master mode
11: PPG slave mode

<TDnI2TD>: For setting the operation during IDLE mode.
0: Stop
1: Operating

<TDnEN> : For specify the TMRD operation. Reduced power consumption can be achieved under the TMRD non-operating mode. It is because the clock is only provided to the register in SIO module. Reading/ writing to the other registers are disabled.
Set TMRD to “1” to enable TMRD operation before setting each register of TMRD module. The setup of the each register remains intact when TMRD operation is disabled after its operation.

TMRDn mode register (n=0, 1)

TDnMOD (0xFFFF_E9x8)

	7	6	5	4	3	2	1	0
Bit symbol		TDnIV0	TDnSYNC	TDnCLE	TDnPEN		TDnCLK1	TDnCLK0
Read/Write	R			R/W				
After reset	0	0	0	0	0	0	0	0
Function	Specify PPG1 default value 0 : ↑with CP3 match,↓ with CP4 match 1 : ↓with CP3 match, ↑with CP4 match	Specify PPG0 default value 0:↑with CP1 match,↓ with CP2 match 1 : ↓with CP1 match, ↑with CP2 match	Specify phase 0:0°cycle 1:90°delayed cycle	Counter operation at CP0 match 0:Not initialized 1:Initialized	Permission to use prescaler 0: Not used 1: Used		Select prescaler clock 00: 1/2 01: 1/4 10: 1/8 11: 1/16	

	15	14	13	12	11	10	9	8
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Fig. 12-9 TMRDn Mode Register

<TDnCLK1:0>: For selecting TMRD input clock.
00:1/2
01:1/4
10:1/8
11:1/16 *Master and Slave modes must have the same setup.
<TDnPEN> : Enabling signal for prescaler.
0: Non-using prescaler.
1: Using prescaler.
* Master and Slave modes must have the same setup.
<TDnCLE>: For initializing the TDnCP0 counter (only at timer mode)
0: Not initialized
1: Initialized
<TDnSYNC>: For setting the phase at the synchronous mode (only at timer mode).
0: The same cycle
1: 1/4 delayed cycle
<TDnIV0> : TDxOUT0 output setting (It is reflected to TDnOUT0 once it is set)
0: Default output is low. Rises when it is identical with TDnCP1. Falls when it is identical with TDnCP2.
1: Default output is high. Falls when it is identical with TDnCP1. Rises when it is identical with TDnCP2.
<TDnIV1> : TDxOUT1 output setting (It is reflected to TDnOUT1 once it is set)
0: Default output is low. Rises when it is identical with TDnCP3. Falls when it is identical with TDnCP4.
1: Default output is high. Falls when it is identical with TDnCP3. Rises when it is identical with TDnCP4.

TMRDn update timing control register (n=0, 1)

TDnBCR (0xFFFF_E90C)

	7	6	5	4	3	2	1	0
Bit symbol								TDSFT
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
								Specify update timing.
	15	14	13	12	11	10	9	8
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Fig. 12-10 TMRDn update timing control register

<TDnSFT> Updating the data written in the buffer is enabled by writing to this register.

Timer mode: Double buffer can be updated with the following timing by setting <TDnMOD>CLE bit.

CLE = 0: Updated at overflow.

CLE = 1: Updated when it conforms to TDnCP0.

PPG single mode: Double buffering data is updated every time a new cycle starts after “1” is written. Once the update is started, the bit is cleared.

PPG master mode: It also controls the double buffer in the slave module during the master buffer operation. Double buffering data is updated every time a new cycle starts after “1” is written.

(This bit is cleared after the slave buffer is updated following the master buffer update.

The slave buffer is updated if “0” is manually set after master buffer update is completed.)

PPG slave mode: Writing “1” during slave mode disables double buffer.

When double buffer is enabled, TDnBCR is set to “0” by setting or clearing RUN.

(Note) Do not write “0” or “1” to TDSFT when “1” is already written on it.

TMRDn status register (n=0, 1)

TDnST (0xFFFF_E910)

	7	6	5	4	3	2	1	0
Bit symbol								
Read/Write	R							
After reset	0					0	0	0
Function	The present value in the counter is read.							
	15	14	13	12	11	10	9	8
Bit symbol								
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	The present value in the counter is read.							
	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

<TDnST>: The present value in the counter is read.

Fig. 12-11 TMRDn Status Register

TDnRG0, TDnRG1, TDnRG2, TDnRG3, TDnRG4 timer register

TDnRG0 timer register (n=0, 1)

TDnRG0 (0xFFFF_E914)

	7	6	5	4	3	2	1	0
Bit symbol	TDnRG0L 7	TDnRG0L 6	TDnRG0L 5	TDnRG0L 4	TDnRG0L 3	TDnRG0L 2	TDnRG0L 1	TDnRG0L 0
Read/Write	W							
After reset	Undefined							
Function	To specify the PPG cycle. Min.:0x0000, Max.:0xFFFF (specified value +1clk cycle)							
	15	14	13	12	11	10	9	8
Bit symbol	TDnRG0H 7	TDnRG0H 6	TDnRG0H 5	TDnRG0H 4	TDnRG0H 3	TDnRG0H 2	TDnRG0H 1	TDnRG0H 0
Read/Write	W							
After reset	Undefined							
Function	To specify the PPG cycle. Min.:0x0000, Max.:0xFFFF (specified value +1clk cycle)							
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Fig. 12-12 TDnRG0 Timer Register

TDnRG1 timer register (n=0, 1)

TDnRG1 (0xFFFF_E918)

	7	6	5	4	3	2	1	0
Bit symbol	TDnRG1L 7	TDnRG1L 6	TDnRG1L 5	TDnRG1L 4	TDnRG1L 3	TDnRG1L 2	TDnRG1L 1	TDnRG1L 0
Read/Write	W							
After reset	Undefined							
Function	To specify the rising timing of PPG0.							
	15	14	13	12	11	10	9	8
Bit symbol	TDnRG1H 7	TDnRG1H 6	TDnRG1H 5	TDnRG1H 4	TDnRG1H 3	TDnRG1H 2	TDnRG1H 1	TDnRG1H 0
Read/Write	W							
After reset	Undefined							
Function	To specify the rising timing of PPG0.							
	15	14	13	12	11	10	9	8
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Fig. 12-13 TDnRG1 Timer Register

TDnRG2 timer register (n=0, 1)

TDnRG2 (0xFFFF_E91C)

	7	6	5	4	3	2	1	0
Bit symbol	TDnRG0L 7	TDnRG0L 6	TDnRG0L 5	TDnRG0L 4	TDnRG0L 3	TDnRG0L 2	TDnRG0L 1	TDnRG0L 0
Read/Write	W							
After reset	Undefined							
Function	To specify the falling timing of PPG0.							
	15	14	13	12	11	10	9	8
Bit symbol	TDnRG0H 7	TDnRG0H 6	TDnRG0H 5	TDnRG0H 4	TDnRG0H 3	TDnRG0H 2	TDnRG0H 1	TDnRG0H 0
Read/Write	W							
After reset	Undefined							
Function	To specify the falling timing of PPG0.							
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Fig. 12-14 TDnRG2 Timer Register

TDnRG3 timer register (n=0, 1)

TDnRG3 (0xFFFF_E920)

	7	6	5	4	3	2	1	0
Bit symbol	TDnRG1L 7	TDnRG1L 6	TDnRG1L 5	TDnRG1L 4	TDnRG1L 3	TDnRG1L 2	TDnRG1L 1	TDnRG1L 0
Read/Write	W							
After reset	Undefined							
Function	To specify the rising timing of PPG0.							
	15	14	13	12	11	10	9	8
Bit symbol	TDnRG1H 7	TDnRG1H 6	TDnRG1H 5	TDnRG1H 4	TDnRG1H 3	TDnRG1H 2	TDnRG1H 1	TDnRG1H 0
Read/Write	W							
After reset	Undefined							
Function	To specify the rising timing of PPG0.							
	15	14	13	12	11	10	9	8
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Fig. 12-15 TDnRG3 Timer Register

TDnRG4 timer register (n=0, 1)

TDnRG4 (0xFFFF_E924)

	7	6	5	4	3	2	1	0
Bit symbol	TDnRG0L 7	TDnRG0L 6	TDnRG0L 5	TDnRG0L 4	TDnRG0L 3	TDnRG0L 2	TDnRG0L 1	TDnRG0L 0
Read/Write	R/W							
After reset	0							
Function	To specify the falling timing of PPG0.							
	15	14	13	12	11	10	9	8
Bit symbol	TDnRG0H 7	TDnRG0H 6	TDnRG0H 5	TDnRG0H 4	TDnRG0H 3	TDnRG0H 2	TDnRG0H 1	TDnRG0H 0
Read/Write	R/W							
After reset	0							
Function	To specify the falling timing of PPG0.							
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	23	22	21	20	19	18	17	16
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
	31	30	29	28	27	26	25	24
Bit symbol								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Fig. 12-16 TDnRG4 Timer Register

12.3 PPG mode

12.3.1 16-bit programmable square-wave output mode (PPG)

This mode allows outputting programmable square waves of any frequency and duty. Both low and high active pulse can be output by setting TDnMOD as <TDNIV1:0>.

The falling edge of UC0 synchronizes to the rising edge of TD0REG0. TD0REG0 is used as a reference clock. The rising edges of TD0REG1 and TD0REG2 are the rising/ falling triggers of the waveform to be generated. The wave to be generated is output from TD0OUT0. The same applies to TD0CP3 and TD0CP4 for generating a TD0OUT1 signal.

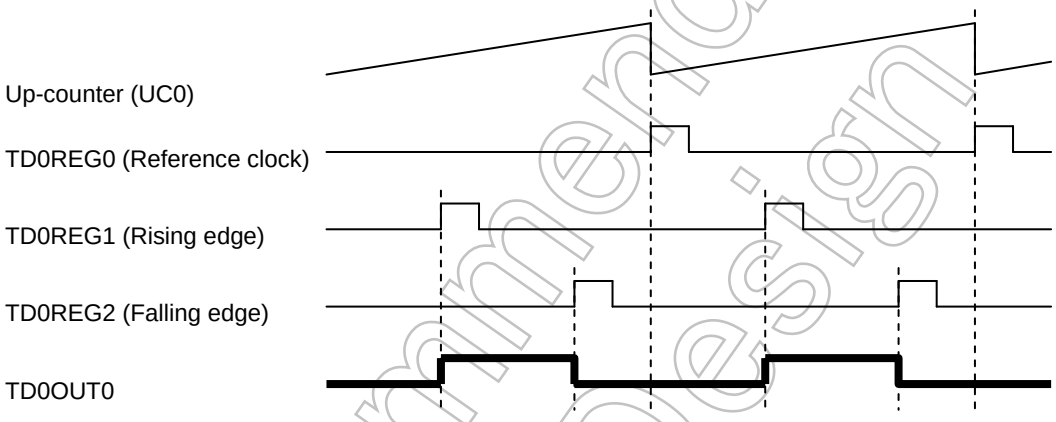


Fig.12-14 Example of programmable square-wave output (PPG)

12.3.2 PPG Mode timing

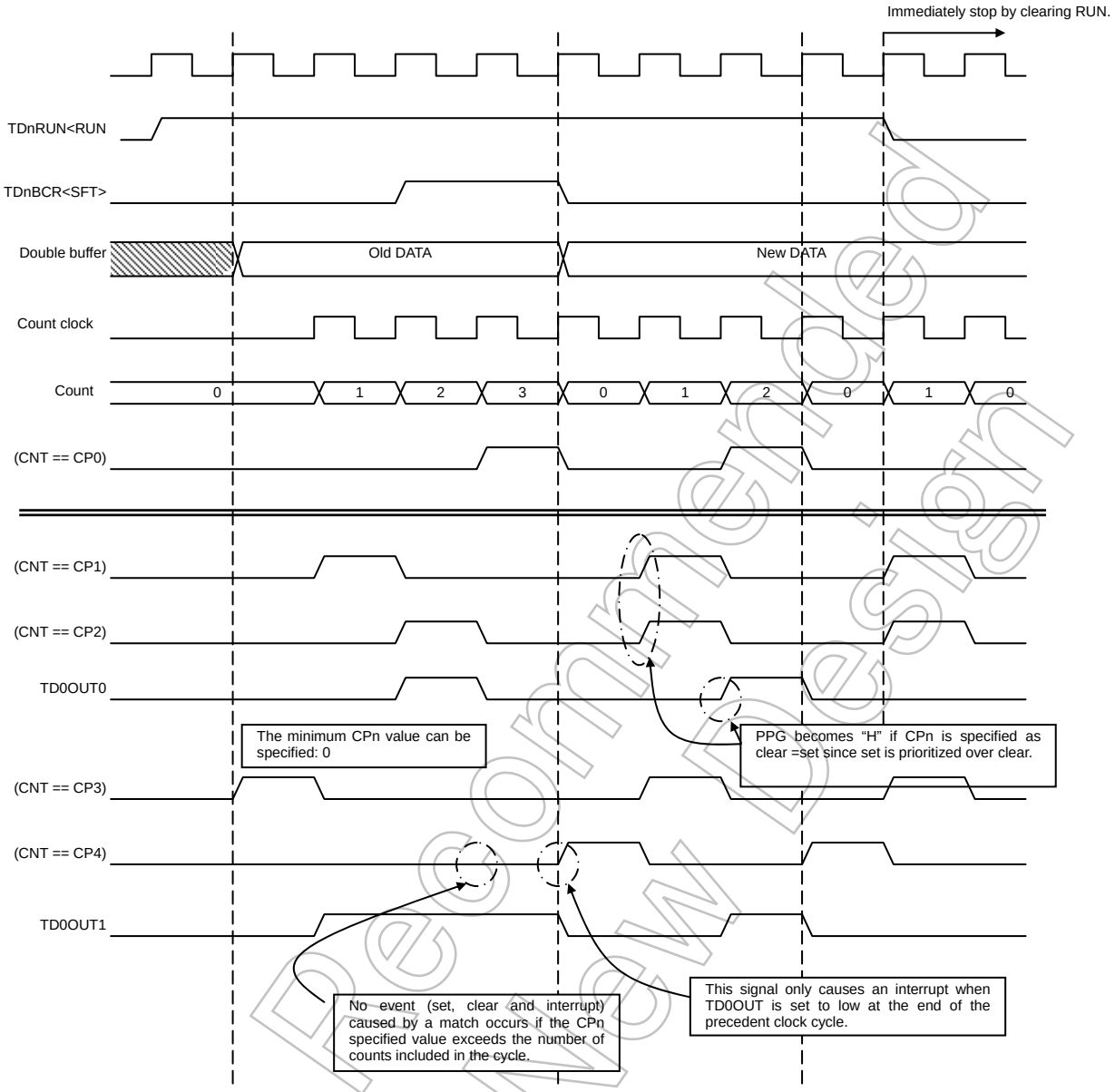


Fig. 12-17 PPG Mode Timing

When it reaches to the value set in TD0CP0, it represents one cycle of a waveform.

・When it reaches to the values set in TD0CP1 and TD0CP2, the rising and falling edges of the TD0OUT0 signal are generated respectively.

・The same applies to TD0CP3 and TD0CP4 that for generating TD0OUT1 signal.

1) The minimum value that can be set to TD0CPn is "0".
2) TD0CP1 has a priority over TD0CP2 if these two have the same values.
3) The value set in TD0CPn becomes invalid when it exceeds the value used as TD0CP0 cycle.
4) If the TD0CP2 and TD0CP4 have the smaller values than TD0CP1 and TD0CP3 respectively, signal shows no fluctuations (interrupts may occur).

Double Buffer Update of PPG master/ slave (with 90 degrees delay)

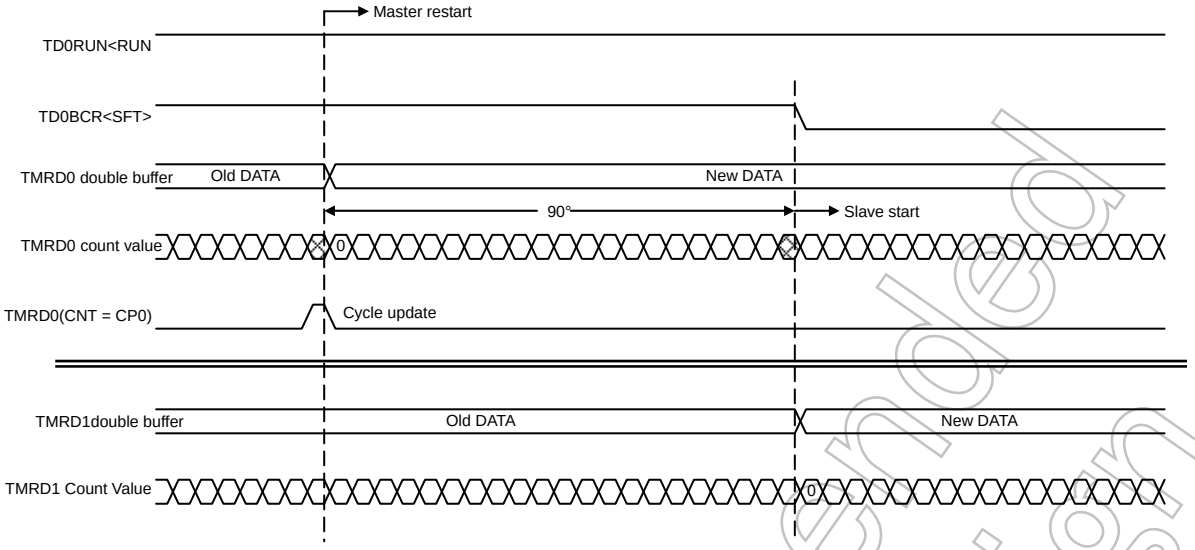


Fig. 12-18 PPG Master/ Slave

If the slave count starts with 1/4 delay from that of the master when <TD0MOD>TD0SYNC is set to “1”, the cycle delayed by 1/4 from the start timing of TD0CP0 of TMRD0 is used as TMRD1 cycle. This setting enables to generate PPG waveform delayed by 1/4 cycle from TMRD0.

13 Serial Channel (SIO)

13.1 Features

This device has three serial I/O channels: SIO0 to SIO2. Each channel operates in either the UART mode (asynchronous communication) or the I/O interface mode (synchronous communication) which is selected by the user.

I/O interface mode
data
— Mode 0: This is the mode to transmit and receive I/O and associated synchronization signals (SCLK) to extend I/O.

Asynchronous (UART) mode:
- Mode 1: TX/RX Data Length: 7 bits
- Mode 2: TX/RX Data Length: 8 bits
- Mode 3: TX/RX Data Length: 9 bits

In the above modes 1 and 2, parity bits can be added. The mode 3 has a wakeup function in which the master controller can start up slave controllers via the serial link (multi-controller system). Fig. 13-1 shows the block diagram of SIO0.

Each channel consists of a prescaler, a serial clock generation circuit, a receive buffer, its control circuit, a transmit buffer and its control circuit. Each channel functions independently.

As the SIOs 0 to 2 operate in the same way, only SIO0 is described here.

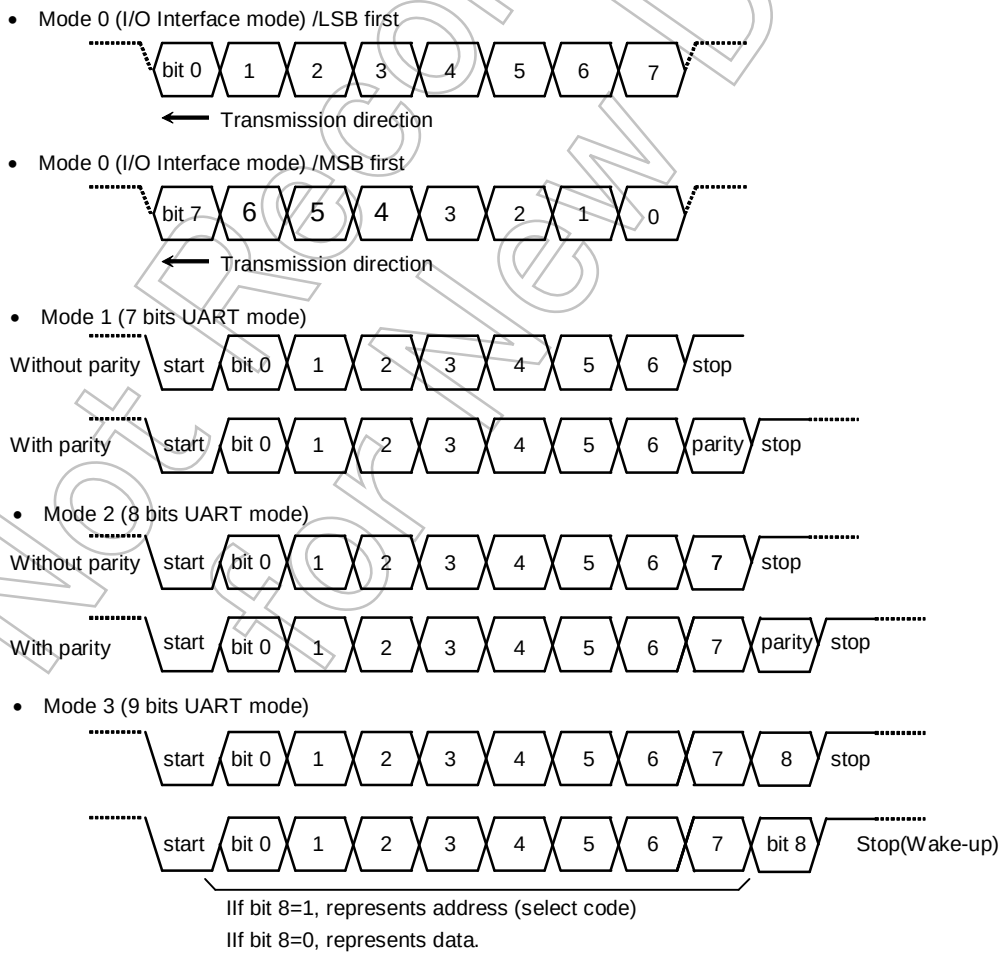


Fig. 13-1 Data Format

13.2 Block Diagram (Channel 0)

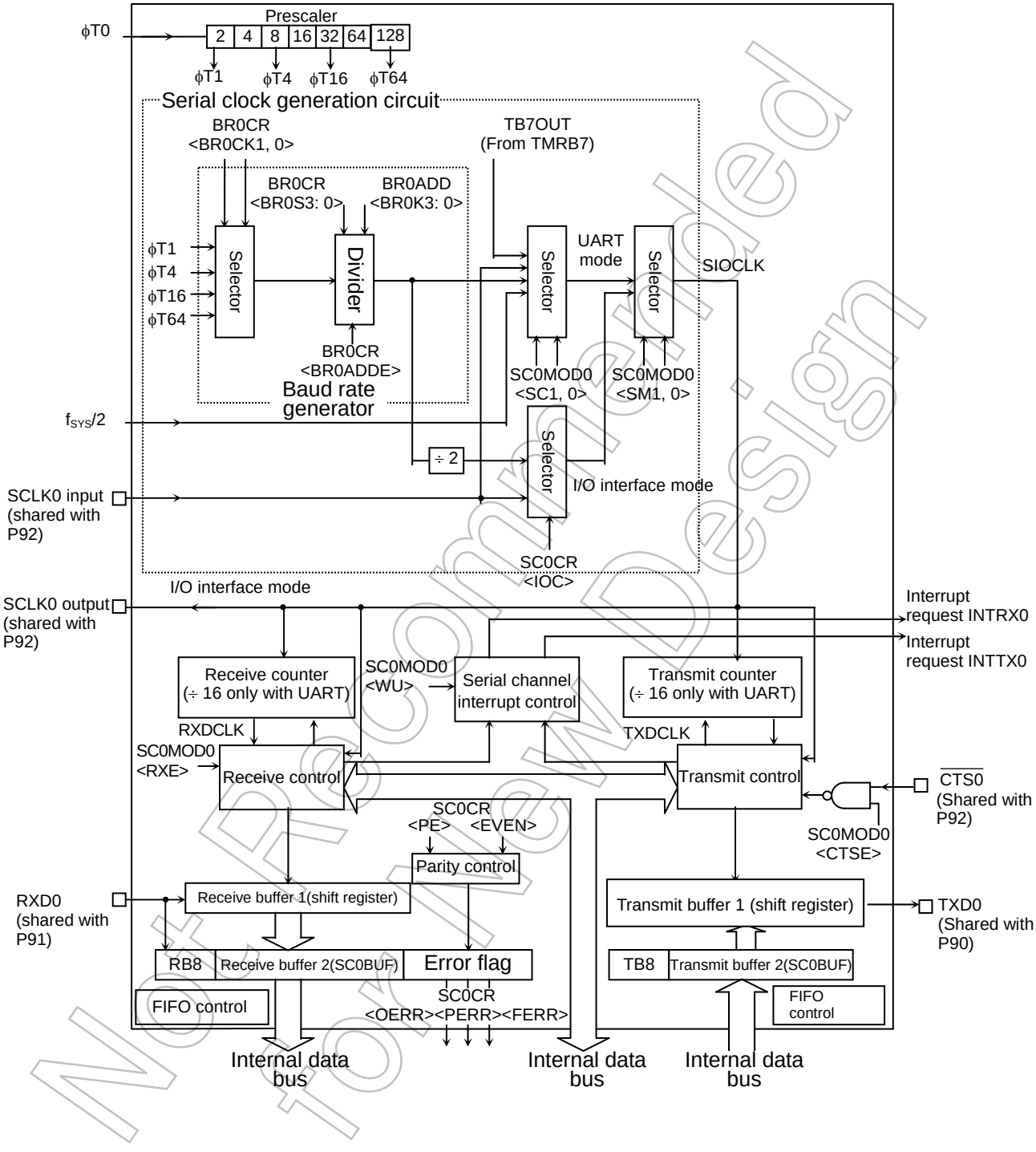


Fig. 13-2 SIO0 Block Diagram

13.3 Operation of Each Circuit (Channel 0)

13.3.1 Prescaler

The device includes a 7-bit prescaler to generate necessary clocks to drive SIO0. The input clock ϕT0 to the prescaler is selected by SYSCR of CG <PRCK1:0> to provide the frequency of either fperiph/2, fperiph/4, fperiph/8, or fperiph/16.

The clock frequency fperiph is either the clock “fgear,” to be selected by SYSCR1<FPSEL> of CG, or the clock “fc” before it is divided by the clock gear.

The prescaler becomes active only when the baud rate generator is selected for generating the serial transfer clock. Table 13-1 lists the prescaler output clock resolution.

Table 13-1 Clock Resolution to the Baud Rate Generator @ = 54MHz

Clear peripheral clock <FPSEL>	Clock gear value <GEAR2 : 0>	Prescaler clock selection <PRCK2 : 0>	Prescaler output clock resolution			
			ϕT1	ϕT4	ϕT16	ϕT64
0 (fgear)	000 (fc)	00(fperiph/16)	fc/2^5(0.6μs)	fc/2^7(2.4μs)	fc/2^9(9.5μs)	fc/2^{11}(37.9μs)
		01(fperiph/8)	fc/2^4(0.3us)	fc/2^6(1.2μs)	fc/2^8(4.7μs)	fc/2^{10}(19.0μs)
		10(fperiph/4)	fc/2^3(0.15us)	fc/2^5(0.6us)	fc/2^7(2.4us)	fc/2^9(9.5μs)
		11(fperiph/2)	fc/2^2(0.07us)	fc/2^4(0.3us)	fc/2^6(1.2μs)	fc/2^8(4.7μs)
	100(fc/2)	00(fperiph/16)	fc/2^6(1.2μs)	fc/2^8(4.7μs)	fc/2^{10}(19.0μs)	fc/2^{12}(75.9μs)
		01(fperiph/8)	fc/2^5(0.6us)	fc/2^7(2.4us)	fc/2^9 (9.5μs)	fc/2^{11}(37.9μs)
		10(fperiph/4)	fc/2^4(0.3us)	fc/2^6(1.2us)	fc/2^8(4.7μs)	fc/2^{10}(19.0μs)
		11(fperiph/2)	fc/2^3(0.15us)	fc/2^5(0.6us)	fc/2^7(2.4us)	fc/2^9(9.5μs)
	110(fc/4)	00(fperiph/16)	fc/2^7(2.4us)	fc/2^9(9.5μs)	fc/2^{11}(37.9μs)	fc/2^{13}(152μs)
		01(fperiph/8)	fc/2^6(1.2μs)	fc/2^8(4.7μs)	Fc/2^{10}(19.0μs)	fc/2^{12}(75.9μs)
		10(fperiph/4)	fc/2^5(0.6us)	fc/2^7(2.4us)	Fc/2^9(9.5μs)	fc/2^{11}(37.9μs)
		11(fperiph/2)	fc/2^4(0.3us)	fc/2^6(1.2μs)	Fc/2^8(4.7μs)	fc/2^{10}(19.0μs)
	111(fc/8)	00(fperiph/16)	fc/2^8(4.7μs)	fc/2^{10}(19.0μs)	fc/2^{12}(75.9μs)	fc/2^{14}(303us)
		01(fperiph/8)	fc/2^7(2.4us)	fc/2^9(9.5μs)	fc/2^{11}(37.9μs)	fc/2^{13}(152μs)
		10(fperiph/4)	fc/2^6(1.2μs)	fc/2^8(4.7μs)	fc/2^{10}(19.0μs)	fc/2^{12}(75.9μs)
		11(fperiph/2)	fc/2^5(0.6us)	fc/2^7(2.4us)	fc/2^9(9.5μs)	fc/2^{11}(37.9μs)
1 (fc)	000 (fc)	00(fperiph/16)	fc/2^5(0.6us)	fc/2^7(2.4us)	fc/2^9(9.5μs)	fc/2^{11}(37.9μs)
		01(fperiph/8)	fc/2^4(0.3us)	fc/2^6(1.2μs)	fc/2^8(4.7μs)	fc/2^{10}(19.0μs)
		10(fperiph/4)	fc/2^3(0.15us)	fc/2^5(0.6us)	fc/2^7(2.4us)	fc/2^9(9.5μs)
		11(fperiph/2)	fc/2^2(0.07us)	fc/2^4(0.3us)	fc/2^6(1.2us)	fc/2^8(4.7μs)
	100(fc/2)	00(fperiph/16)	fc/2^5(0.6us)	fc/2^7(2.4us)	fc/2^9(9.5μs)	fc/2^{11}(37.9μs)
		01(fperiph/8)	fc/2^4(0.3us)	fc/2^6(1.2μs)	fc/2^8(4.7μs)	fc/2^{10}(19.0μs)
		10(fperiph/4)	fc/2^3(0.15us)	fc/2^5(0.6us)	fc/2^7(2.4us)	fc/2^9(9.5μs)
		11(fperiph/2)	—	fc/2^4(0.3us)	fc/2^6(1.2us)	fc/2^8(4.7μs)
	110(fc/4)	00(fperiph/16)	fc/2^5(0.6us)	fc/2^7(2.4us)	fc/2^9(9.5μs)	fc/2^{11}(37.9μs)
		01(fperiph/8)	fc/2^4(0.3us)	fc/2^6(1.2μs)	fc/2^8(4.7μs)	fc/2^{10}(19.0μs)
		10(fperiph/4)	—	fc/2^5(0.6us)	fc/2^7(2.4us)	fc/2^9(9.5μs)
		11(fperiph/2)	—	fc/2^4(0.3us)	fc/2^6(1.2us)	fc/2^8(4.7μs)
	111(fc/8)	00(fperiph/16)	fc/2^5(0.6us)	fc/2^7(2.4us)	fc/2^9(9.5μs)	fc/2^{11}(37.9μs)
		01(fperiph/8)	—	fc/2^6(1.2μs)	fc/2^8(4.7μs)	fc/2^{10}(19.0μs)
		10(fperiph/4)	—	fc/2^5(0.6us)	fc/2^7(2.4us)	fc/2^9(9.5μs)
		11(fperiph/2)	—	—	fc/2^6(1.2us)	fc/2^8(4.7μs)

(Note 1) The prescaler output clock ϕTn must be selected so that the relationship "ϕTn < fsys/2" is satisfied (so that ϕTn is slower than fsys/2).

(Note 2) Do not change the clock gear while SIO is operating.

(Note 3) The horizontal lines in the above table indicate that the setting is prohibited.

The serial interface baud rate generator uses four different clocks, i.e., ϕT1, ϕT4, ϕT16 and ϕT64, supplied from the prescaler output clock.

Table 13-2 Clock Resolution to the Baud Rate Generator **@ = 40MHz**

Clear peripheral clock <FPSEL>	Clock gear value <GEAR2 : 0>	Prescaler clock selection <PRCK2 : 0>	Prescaler output clock resolution			
			ϕT1	ϕT4	ϕT16	ϕT64
0 (fgear)	000 (fc)	00(fperiph/16)	fc/2^5(0.8 μs)	fc/2^7(3.2 μs)	fc/2^9(12.8 μs)	fc/2^{11}(51.2μs)
		01(fperiph/8)	fc/2^4(0.4 us)	fc/2^6(1.6 μs)	fc/2^8(6.4 μs)	fc/2^{10}(25.6 μs)
		10(fperiph/4)	fc/2^3(0.2us)	fc/2^5(0.8us)	fc/2^7(3.2us)	fc/2^9(12.8μs)
		11(fperiph/2)	fc/2^2(0.1us)	fc/2^4(0.4us)	fc/2^6(1.6 μs)	fc/2^8(6.4 μs)
	100(fc/2)	00(fperiph/16)	fc/2^6(1.6 μs)	fc/2^8(6.4 μs)	fc/2^{10}(25.6 μs)	fc/2^{12} (102 μs)
		01(fperiph/8)	fc/2^5(0.8us)	fc/2^7(3.2us)	fc/2^9 (12.8μs)	fc/2^{11}(51.2μs)
		10(fperiph/4)	fc/2^4(0.4us)	fc/2^6(1.6 ms)	fc/2^8(6.4 μs)	fc/2^{10}(25.6 μs)
		11(fperiph/2)	fc/2^3(0.2 us)	fc/2^5(0.8 us)	fc/2^7(3.2 us)	fc/2^9(12.8μs)
	110(fc/4)	00(fperiph/16)	fc/2^7(3.2 us)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)	fc/2^{13}(204μs)
		01(fperiph/8)	fc/2^6(1.6 μs)	fc/2^8(6.4 μs)	fc/2^{10}(25.6 μs)	fc/2^{12} (102μs)
		10(fperiph/4)	fc/2^5(0.8 us)	fc/2^7(3.2 us)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
		11(fperiph/2)	fc/2^4(0.4 us)	fc/2^6(1.6 μs)	fc/2^8(6.4 μs)	fc/2^{10}(25.6 μs)
	111(fc/8)	00(fperiph/16)	fc/2^8(6.4 μs)	fc/2^{10}(25.6 μs)	fc/2^{12} (102 μs)	fc/2^{14}(410us)
		01(fperiph/8)	fc/2^7(3.2 us)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)	fc/2^{13}(204μs)
		10(fperiph/4)	fc/2^6(1.6 μs)	fc/2^8(6.4 μs)	fc/2^{10}(25.6 μs)	fc/2^{12} (102μs)
		11(fperiph/2)	fc/2^5(0.8 us)	fc/2^7(3.2 us)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
1 (fc)	000 (fc)	00(fperiph/16)	fc/2^5(0.8 us)	fc/2^7(3.2 us)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
		01(fperiph/8)	fc/2^4(0.4 us)	fc/2^6(1.6 μs)	fc/2^8(6.4 μs)	fc/2^{10}(25.6 μs)
		10(fperiph/4)	fc/2^3(0.2 us)	fc/2^5(0.8 us)	fc/2^7(3.2 us)	fc/2^9(12.8μs)
		11(fperiph/2)	fc/2^2(0.1 us)	fc/2^4(0.4 us)	fc/2^6(1.6 us)	fc/2^8(6.4 μs)
	100(fc/2)	00(fperiph/16)	fc/2^5(0.8 us)	fc/2^7(3.2 us)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
		01(fperiph/8)	fc/2^4(0.4 us)	fc/2^6(1.6 μs)	fc/2^8(6.4 μs)	fc/2^{10}(25.6 μs)
		10(fperiph/4)	fc/2^3(0.2 us)	fc/2^5(0.8 us)	fc/2^7(3.2 us)	fc/2^9(12.8μs)
		11(fperiph/2)	—	fc/2^4(0.4 us)	fc/2^6(1.6 us)	fc/2^8(6.4 μs)
	110(fc/4)	00(fperiph/16)	fc/2^5(0.8 us)	fc/2^7(3.2 us)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
		01(fperiph/8)	fc/2^4(0.4 us)	fc/2^6(1.6 μs)	fc/2^8(6.4 μs)	fc/2^{10}(25.6 μs)
		10(fperiph/4)	—	fc/2^5(0.8 us)	fc/2^7(3.2 us)	fc/2^9(12.8μs)
		11(fperiph/2)	—	fc/2^4(0.4 us)	fc/2^6(1.6 us)	fc/2^8(6.4 μs)
	111(fc/8)	00(fperiph/16)	fc/2^5(0.8 us)	fc/2^7(3.2 us)	fc/2^9(12.8μs)	fc/2^{11}(51.2μs)
		01(fperiph/8)	—	fc/2^6(1.6 μs)	fc/2^8(6.4 μs)	fc/2^{10}(25.6 μs)
		10(fperiph/4)	—	fc/2^5(0.8 us)	fc/2^7(3.2 us)	fc/2^9(12.8μs)
		11(fperiph/2)	—	—	fc/2^6(1.6 us)	fc/2^8(6.4 μs)

(Note 1) The prescaler output clock ϕTn must be selected so that the relationship "ϕTn < fsys/2" is satisfied (so that ϕTn is slower than fsys/2).

(Note 2) Do not change the clock gear while SIO is operating.

(Note 3) The horizontal lines in the above table indicate that the setting is prohibited.

The serial interface baud rate generator uses four different clocks, i.e., ϕT1, ϕT4, ϕT16 and ϕT64, supplied from the prescaler output clock.

13.3.2 Baud Rate Generator

The baud rate generator generates transmit and receive clocks to determine the serial channel transfer rate.

The baud rate generator uses either the ϕT1, ϕT4, ϕT16 or ϕT64 clock supplied from the 7-bit prescaler. This input clock selection is made by setting the baud rate generator control register, BR0CR <BR0CK1:0>.

The baud rate generator contains built-in dividers for divide by 1, N + m/16 (N=2~15, m=0~15), and 16. The division is performed according to the settings of the baud rate generator control registers BR0CR <BR0ADDE> <BR0S3:0> and BR0ADD <BR0K3:0> to determine the resulting transfer rate.

- UART mode

  1) If BR0CR <BR0ADDE> = 0,

     The setting of BR0ADD <BR0K3:0> is ignored and the counter is divided by N where N is the value set to BR0CR <BR0S3:0>. (N = 1 to 16).

  2) If BR0CR <BR0ADDE> = 1,

     The N + (16 - K)/16 division function is enabled and the division is made by using the values N (set in BR0CR <BR0S3:0>) and K (set in BR0ADD<BR0K3:0>). (N = 2 to 15, K = 1 to 15)

(Note) For the N values of 1 and 16, the above N+(16-K)/16 division function is inhibited. So, be sure to set BR0CR<BR0ADDE> to "0."

- I/O interface mode

  The N + (16 - K)/16 division function cannot be used in the I/O interface mode. Be sure to divide by N, by setting BR0CR <BR0ADDE> to "0".

- Baud rate calculation to use the baud rate generator:

  1) UART mode

$$\text{Baud rate} = \frac{\text{Baud rated generator input clock}}{\text{Frequency divided by the divide ratio}} /16$$

The highest baud rate out of the baud rate generator is 843 kbps when ϕT1 is 13.5 MHz.

The fsys/2 frequency, which is independent of the baud rate generator, can be used as the serial clock. In this case, the highest baud rate will be 1.69 Mbps when fsys is 54 MHz.

  2) I/O interface mode

$$\text{Baud rate} = \frac{\text{Baud rated generator input clock}}{\text{Frequency divided by the divide ratio}} /2$$

The highest baud rate will be generated when ϕT1 is 13.5 MHz. The divide ratio can be set to 1 if double buffer is used and the resulting output baud rate will be 6.75 Mbps. (If double buffering is not used, the highest baud rate will be 3.38 Mbps applying the divide ratio of "2").

- Example baud rate setting:

1) Division by an integer (divide by N):

Selecting fc = 39.321 MHz for fperiph, setting ϕT0 to fperiph/16, using the baud rate generator input clock ϕT1, setting the divide ratio N (BR0CR<BR0S3:0>) = 4, and setting BR0CR<BR0ADDE> = "0," the resulting baud rate in the UART mode is calculated as follows:

* Clocking conditions
  - System clock : High-speed (fc)
  - High speed clock gear : x 1 (fc)
  - Prescaler clock : f_{periph}/16 ($f_{periph} = f_{sys}$)

Baud rate = $\frac{fc/32}{4}$ /16

$= 39.321 \times 10^6 \div 32 \div 4 \div 16 \fallingdotseq 19200$ (bps)

(Note) The divide by (N + (16-K)/16) function is inhibited and thus BR0ADD <BR0K3:0> is ignored.

2) For divide by N + (16-K)/16 (only for UART mode):

Selecting fc = 19.2 MHz for fperiph, setting ϕT0 to fperiph/16, using the baud rate generator input clock ϕT1, setting the divide ratio N (BR0CR<BR0S3:0>) = 7, setting K (BR0ADD<BR0K3:0>) = 3, and selecting BR0CR<BR0ADDE> = 1, the resulting baud rate is calculated as follows:

* Clocking conditions
  - System clock : High-speed (fc)
  - High-speed clock gear : x 1 (fc)
  - Prescaler clock : f_{periph}/4 ($f_{periph} = f_{sys}$)

Baud rate = $\frac{fc/32}{7 + \frac{(16-3)}{16}}$ /16

$= 19.2 \times 10^6 \div 32 \div (7 + \frac{13}{16}) \div 16 = 4800$ (bps)

Also, an external clock input may be used as the serial clock. The resulting baud rate calculation is shown below:

- Baud rate calculation for an external clock input:

  1) UART mode

  Baud Rate = external clock input / 16

  In this, the period of the external clock input must be equal to or greater than 4/fsys.

  If fsys = 54 MHz, the highest baud rate will be 54÷4÷16=844 (kbps).

  2) I/O interface mode

  Baud Rate = external clock input

  When double buffering is used, it is necessary to satisfy the following relationship:

  External clock input period > 12/fsys

  Therefore, when fsys = 54 MHz, the baud rate must be set to a rate lower than 54÷12=4.5 (Mbps).

  When double buffering is not used, it is necessary to satisfy the following relationship:

  External clock input period > 16/fsys

  Therefore, when fsys = 54 MHz, the baud rate must be set to a rate lower than 54÷16=3.38 (Mbps).

  The baud rate examples for the UART mode are shown in Table 13-3 and Table 13-4.

Table 13-3 Selection of UART Baud Rate

(Using the baud rate generator with BR0CR <BR0ADDE> = 0) Unit: (kbps)

fc [MHz]	Input clock / Divide ratio N (Set to BR0CR <BR0S3 : 0>)	ϕT1 (fc/4)	ϕT4 (fc/16)	ϕT16 (fc/64)	ϕT64 (fc/256)
19.6608	1	307.200	76.800	19.200	4.800
↑	2	153.600	38.400	9.600	2.400
↑	4	76.800	19.200	4.800	1.200
↑	8	38.400	9.600	2.400	0.600
↑	0	19.200	4.800	1.200	0.300
24.576	5	76.800	19.200	4.800	1.200
↑	A	38.400	9.600	2.400	0.600
29.4912	1	460.800	115.200	28.800	7.200
↑	2	230.400	57.600	14.400	3.600
↑	3	153.600	38.400	9.600	2.400
↑	4	115.200	28.800	7.200	1.800
↑	6	76.800	19.200	4.800	1.200
↑	C	38.400	9.600	2.400	0.600

(Note) This table shows the case where the system clock is set to fc, the clock gear is set to fc/1, and the prescaler clock is set to f_{periph}/2.

Table 13-4 Selection of UART Baud Rate

(The TMRB7 timer output (internal TB7OUT) is used with the timer input clock set to ϕT0.)

TB7RG \ fc	29.4912 MHz	24.576 MHz	24 MHz	19.6608 MHz	16 MHz	12.288 MHz
1H	230.4	192	187.5	153.6	125	96
2H	115.2	96	93.75	76.8	62.5	48
3H	76.8	64	62.5	51.2	41.67	32
4H	57.6	48	46.88	38.4	31.25	24
5H	46.08	38.4	37.5	30.72	25	19.2
6H	38.4	32	31.25	25.6	20.83	16
8H	28.8	24	23.44	19.2	15.63	12
AH	23.04	19.2	18.75	15.36	12.5	9.6
10H	14.4	12	11.72	9.6	7.81	6
14H	11.52	9.6	9.38	7.68	6.25	4.8

Baud rate calculation to use the TMRB7 timer:

$$\text{Transfer rate} = \frac{\text{Clock frequency selected by SYSCR0<PRCK1:0>}}{\text{TB7RG} \times \underline{2} \times 16}$$

(When input clock to the timer TMRB7 is ϕT0)

(Note 1) In the I/O interface mode, the TMRB0 timer output signal cannot be used internally as the transfer clock.

(Note 2) This table shows the case where the system clock is set to fc, the clock gear is set to fc/1, and the prescaler clock is set to f_{periph}/4.

13.3.3 Serial Clock Generation Circuit

This circuit generates basic transmit and receive clocks.

- I/O interface mode

  In the SCLK output mode with the SC0CR <IOC> serial control register set to “0,” the output of the previously mentioned baud rate generator is divided by 2 to generate the basic clock.

  In the SCLK input mode with SC0CR <IOC> set to “1,” rising and falling edges are detected according to the SC0CR <SCLKS> setting to generate the basic clock.

- Asynchronous (UART) mode :

  According to the settings of the serial control mode register SC0MOD0 <SC1:0>, either the clock from the baud rate register, the system clock (f_{SYS}/2), the internal output signal of the TMRB7 timer, or the external clock (SCLKO pin) is selected to generate the basic clock, SIOCLK.

13.3.4 Receive Counter

The receive counter is a 4-bit binary counter used in the asynchronous (UART) mode and is up-counted by SIOCLK. Sixteen SIOCLK clock pulses are used in receiving a single data bit while the data symbol is sampled at the seventh, eighth, and ninth pulses. From these three samples, majority logic is applied to decide the received data.

13.3.5 Receive Control Unit

- I/O interface mode:

  In the SCLK output mode with SC0CR <IOC> set to “0,” the RXD0 pin is sampled on the rising edge of the shift clock output to the SCLK0 pin.

  In the SCLK input mode with SC0CR <IOC> set to “1,” the serial receive data RXD0 pin is sampled on the rising or falling edge of SCLK input depending on the SC0CR <SCLKS> setting.

- Asynchronous (UART) mode :

  The receive control unit has a start bit detection circuit, which is used to initiate receive operation when a normal start bit is detected.

13.3.6 Receive Buffer

The receive buffer is of a dual structure to prevent overrun errors. The first receive buffer (a shift register) stores the received data bit-by-bit. When a complete set of bits have been stored, they are moved to the second receive buffer (SC0BUF). At the same time, the receive buffer full flag (SC0MOD2 “RBFLL”) is set to “1” to indicate that valid data is stored in the second receive buffer. However, if the receive FIFO is set enabled, the receive data is moved to the receive FIFO and this flag is immediately cleared.

If the receive FIFO has been disabled (SCOFCNF <CNFG> = 0 and SC0MOD1<FDPX1:0> =01), the INTRX0 interrupt is generated at the same time. If the receive FIFO has been enabled (SCNFCNF <CNFG> = 1 and SC0MOD1<FDPX1:0> = 01), an interrupt will be generated according to the SC0RFC <RIL1:0> setting.

The CPU will read the data from either the second receive buffer (SC0BUF) or from the receive FIFO (the address is the same as that of the receive buffer). If the receive FIFO has not been enabled, the receive buffer full flag <RBFLL> is cleared to “0” by the read operation. The next data received can be stored in the first receive buffer even if the CPU has not read the previous data from the second receive buffer (SC0BUF) or the receive FIFO.

If SCLK is set to generate clock output in the I/O interface mode, the double buffer control bit SC0MOD2 <WBUF> can be programmed to enable or disable the operation of the second receive buffer (SCOBUF).

By disabling the second receive buffer (i.e., the double buffer function) and also disabling the receive FIFO (SCOFCNF <CNFG> = 0 and <FDPX1:0> = 01), handshaking with the other side of communication can be enabled and the SCLK output stops each time one frame of data is transferred. In this setting, the CPU reads data from the first receive buffer. By the read operation of CPU, the SCLK output resumes.

If the second receive buffer (i.e., double buffering) is enabled but the receive FIFO is not enabled, the SCLK output is stopped when the first receive data is moved from the first receive buffer to the second receive buffer and the next data is stored in the first buffer filling both buffers with valid data. When the second receive buffer is read, the data of the first receive buffer is moved to the second receive buffer and the SCLK output is resumed upon generation of the receive interrupt INTRX. Therefore, no buffer overrun error will be caused in the I/O interface SCLK output mode regardless of the setting of the double buffer control bit SC0MOD2 <WBUF>.

If the second receive buffer (double buffering) is enabled and the receive FIFO is also enabled (SCNFCNF <CNFG> = 1 and <FDPX1:0> = 01/11), the SCLK output will be stopped when the receive FIFO is full (according to the setting of SCOFNCF <RFST>) and both the first and second receive buffers contain valid data. Also in this case, if SCOFCNF <RXTXCNT> has been set to “1,” the receive control bit RXE will be automatically cleared upon suspension of the SCLK output. If it is set to “0,” automatic clearing will not be performed.

(Note) In this mode, the SC0CR <OEER> flag is insignificant and the operation is undefined. Therefore, before switching from the SCLK output mode to another mode, the SC0CR register must be read to initialize this flag.

In other operating modes, the operation of the second receive buffer is always valid, thus improving the performance of continuous data transfer. If the receive FIFO is not enabled, an overrun error occurs when the data in the second receive buffer (SC0BUF) has not been read before the first receive buffer is full with the next receive data. If an overrun error occurs, data in the first receive buffer will be lost while data in the second receive buffer and the contents of SC0CR <RB8> remain intact. If the receive FIFO is enabled, the FIFO must be read before the FIFO is full and the second receive buffer is written by the next data through the first buffer. Otherwise, an overrun error will be generated and the receive FIFO overrun error flag will be set. Even in this case, the data already in the receive FIFO remains intact.

The parity bit to be added in the 8-bit UART mode as well as the most significant bit in the 9-bit UART mode will be stored in SC0CR <RB8>.

In the 9-bit UART mode, the slave controller can be operated in the wake-up mode by setting the wake-up function SC0MOD0 <WU> to “1.” In this case, the interrupt INTRX0 will be generated only when SC0CR <RB8> is set to “1.”

13.3.7 Receive FIFO Buffer

In addition to the double buffer function already described, data may be stored using the receive FIFO buffer. By setting <CNFG> of the SC0FCNF register and <FDPX1:0> of the SC0MOD1 register, the 4-byte receive buffer can be enabled. Also, in the UART mode or I/O interface mode, data may be stored up to a predefined fill level. When the receive FIFO buffer is to be used, be sure to enable the double buffer function.

If data with parity bit is to be received in the UART mode, parity check must be performed each time a data frame is received.

13.3.8 Receive FIFO Operation

① I/O interface mode with SCLK output:

The following example describes the case a 4-byte data stream is received in the half duplex mode:

SC0RFC<7:6>=01: Clears receive FIFO and sets the condition of interrupt generation.

SC0RFC<1:0>=00: Sets the interrupt to be generated at fill level 4.

SC0FCNF <1:0>=10111: Automatically inhibits continued reception after reaching the fill level. The number of bytes to be used in the receive FIFO is the same as the interrupt generation fill level.

In this condition, 4-byte data reception may be initiated by setting the half duplex transmission mode and writing “1” to the RXE bit. After receiving 4 bytes, the RXE bit is automatically cleared and the receive operation is stopped (SCLK is stopped).

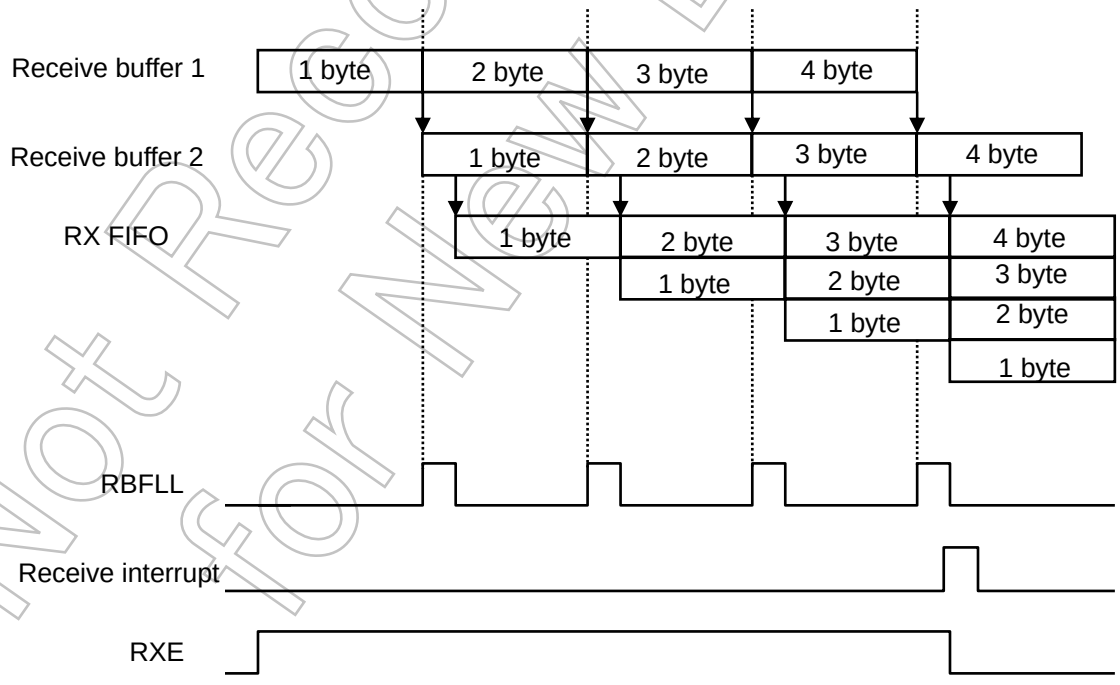


Fig. 13-3 Receive FIFO Operation

② I/O interface mode with SCLK input:

The following example describes the case a 4-byte data stream is received:

SC0RFC <7:6> = 10: Clears receive FIFO and sets the condition of interrupt generation

SC0RFC <1:0> = 00: Sets the interrupt to be generated at fill level 4.

SC0FCNF <1:0> = 10101: Automatically allows continued reception after reaching the fill level. The number of bytes to be used in the receive FIFO is the maximum allowable number.

In this condition, 4-byte data reception may be initiated by setting the half duplex transmission mode and writing “1” to the RXE bit. After receiving 4 bytes, receive FIFO interrupt is generated. This setting enables the next data reception as well. The next 4 bytes can be received before all the data is read from FIFO.

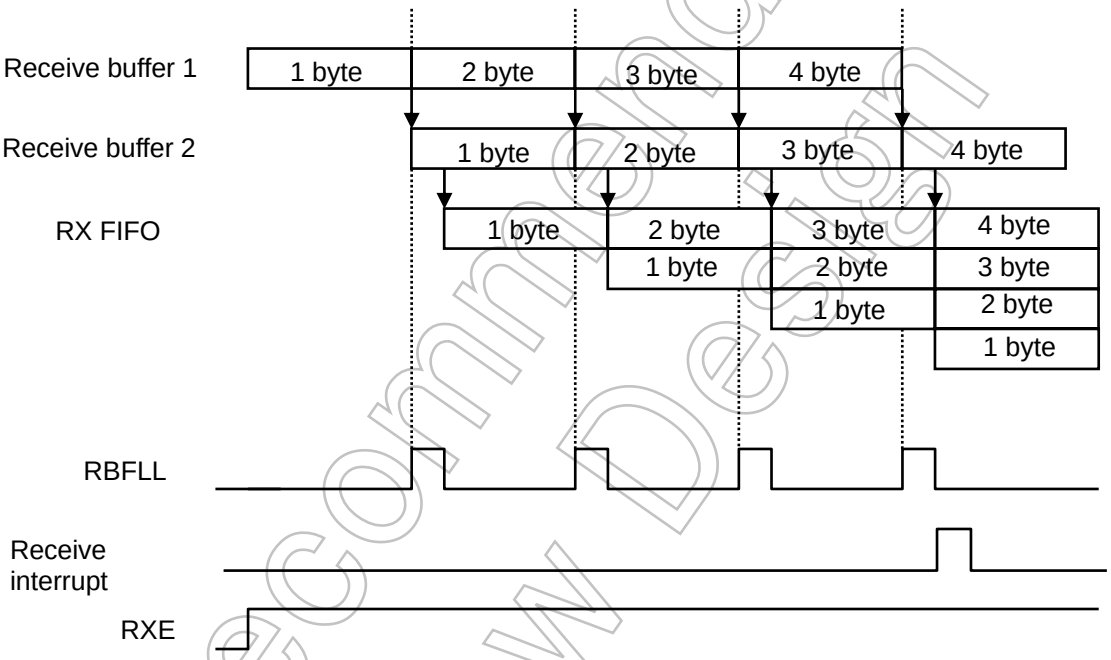


Fig. 13-4 Receive FIFO Operation

13.3.9 Transmit Counter

The transmit counter is a 4-bit binary counter used in the asynchronous communication (UART) mode. It is counted by SIOCLK as in the case of the receive counter and generates a transmit clock (TXDCLK) on every 16th clock pulse.

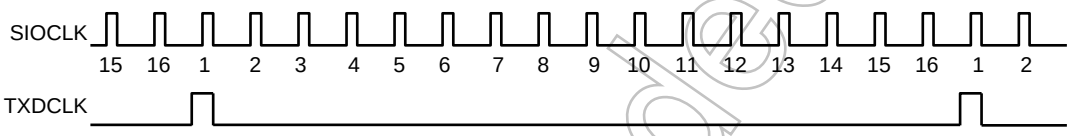


Fig. 13-5 Transmit Clock Generation

13.3.10 Transmit Control Unit

- I/O interface mode:

  In the SCLK output mode with SC0CR <IOC> set to “0,” each bit of data in the transmit buffer is output to the TXD0 pin on the rising edge of the shift clock output from the SCLK0 pin.

  In the SCLK input mode with SC0CR <IOC> set to “1,” each bit of data in the transmit buffer is output to the TXD0 pin on the rising or falling edge of the input SCLK signal according to the SC0CR <SCLKS> setting.

- Asynchronous (UART) mode:

  When the CPU writes data to the transmit buffer, data transmission is initiated on the rising edge of the next TXDCLK and the transmit shift clock (TXDSFT) is also generated.

- Handshake function

The $\overline{CTS}$ pin enables frame by frame data transmission so that overrun errors can be prevented. This function can be enabled or disabled by SC0MOD0 <CTSE>.

When the $\overline{CTS0}$ pin is set to the “H” level, the current data transmission can be completed but the next data transmission is suspended until the $\overline{CTS0}$ pin returns to the “L” level. However in this case, the INTTX0 interrupt is generated, the next transmit data is requested to the CPU, data is written to the transmit buffer, and it waits until it is ready to transmit data.

Although no $\overline{RTS}$ pin is provided, a handshake control function can be easily implemented by assigning a port for the $\overline{RTS}$ function. By setting the port to “H” level upon completion of data reception (in the receive interrupt routine), the transmit side can be requested to suspend data transmission.

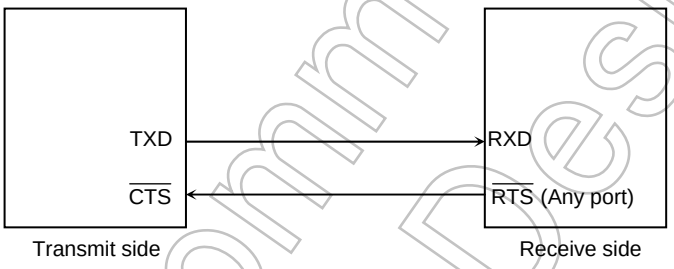


Fig. 13-6 Handshake Function

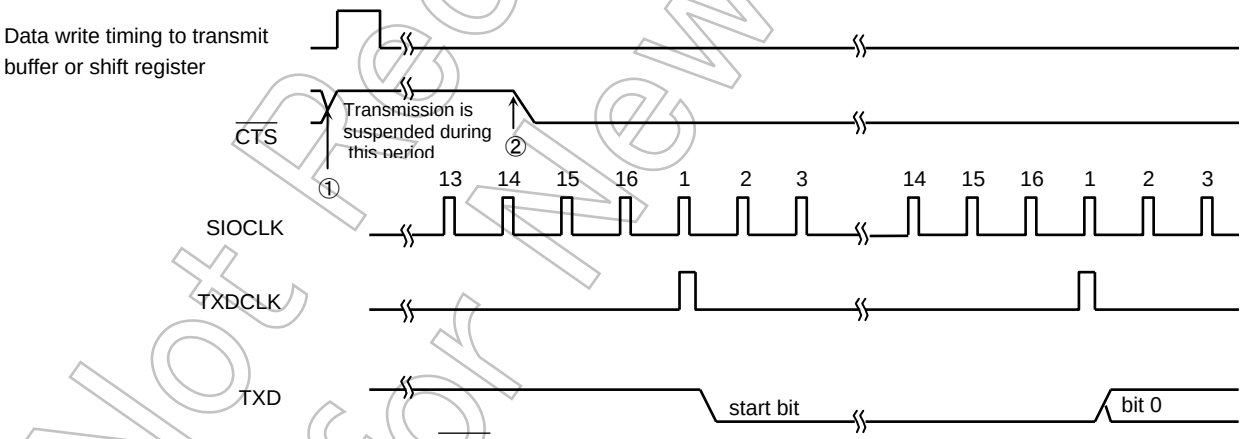


Fig. 13-7 $\overline{CTS}$ (Clear to Transmit) Signal Timing

(Note) **If the $\overline{CTS}$ signal is set to “H” during transmission, the next data transmission is suspended after the current transmission is completed.**

Data transmission starts on the first falling edge of the TXDCLK clock after $\overline{CTS}$ is set to “L.”

13.3.11 Transmit Buffer

The transmit buffer (SC0BUF) is in a dual structure. The double buffering function may be enabled or disabled by setting the double buffer control bit <WBUF> in serial mode control register 2 (SC0MOD2). If double buffering is enabled, data written to Transmit Buffer 2 (SCOBUF) is moved to Transmit Buffer 1 (shift register).

If the transmit FIFO has been disabled (SCOFCNF <CNFG> = 0 or 1 and <FDPX1:0> = 01), the INTTX interrupt is generated at the same time and the transmit buffer empty flag <TBEMP> of SC0MOD2 is set to “1.” This flag indicates that Transmit Buffer 2 is now empty and that the next transmit data can be written. When the next data is written to Transmit Buffer 2, the <TBEMP> flag is cleared to “0.”

If the transmit FIFO has been enabled (SCNFCNF <CNFG> = 1 and <FDPX1:0> = 10/11), any data in the transmit FIFO is moved to the Transmit Buffer 2 and <TBEMP> flag is immediately cleared to “0.” The CPU writes data to Transmit Buffer 2 or to the transmit FIFO.

If the transmit FIFO is disabled in the I/O interface SCLK input mode and if no data is set in Transmit Buffer 2 before the next frame clock input, which occurs upon completion of data transmission from Transmit Buffer 1, an under-run error occurs and a serial control register (SC0CR) <PERR> parity/under-run flag is set.

If the transmit FIFO is enabled in the I/O interface SCLK input mode, when data transmission from Transmit Buffer 1 is completed, the Transmit Buffer 2 data is moved to Transmit Buffer 1 and any data in transmit FIFO is moved to Transmit Buffer 2 at the same time.

If the transmit FIFO is disabled in the I/O interface SCLK output mode, when data in Transmit Buffer 2 is moved to Transmit Buffer 1 and the data transmission is completed, the SCLK output stops. So, no under-run errors can be generated.

If the transmit FIFO is enabled in the I/O interface SCLK output mode, the SCLK output stops upon completion of data transmission from Transmit Buffer 1 if there is no valid data in the transmit FIFO.

Note) In the I/O interface SCLK output mode, the SC0CR <PEER> flag is insignificant. In this case, the operation is undefined. Therefore, to switch from the SCLK output mode to another mode, SC0CR must be read in advance to initialize the flag.

If double buffering is disabled, the CPU writes data only to Transmit Buffer 1 and the transmit interrupt INTTX is generated upon completion of data transmission.

If handshaking with the other side is necessary, set the double buffer control bit <WBUF> to “0” (disable) to disable Transmit Buffer 2; any setting for the transmit FIFO should not be performed.

13.3.12 Transmit FIFO Buffer

In addition to the double buffer function already described, data may be stored using the transmit FIFO buffer. By setting <CNFG> of the SC0FCNF register and <FDPX1:0> of the SC0MOD1 register, the 4-byte transmit buffer can be enabled. In the UART mode or I/O interface mode, up to 4 bytes of data may be stored.
If data is to be transmitted with a parity bit in the UART mode, parity check must be performed on the receive side each time a data frame is received.

13.3.13 Transmit FIFO Operation

① I/O interface mode with SCLK output (normal mode):

The following example describes the case a 4-byte data stream is transmitted:

SC0TFC <7:6> = 01: Clears transmit FIFO and sets the condition of interrupt generation

SC0TFC <1:0> = 00: Sets the interrupt to be generated at fill level 0.

SC0FCNF <1:0> = 01011: Inhibits continued transmission after reaching the fill level.

In this condition, data transmission can be initiated by setting the transfer mode to half duplex, writing 4 bytes of data to the transmit FIFO, and setting the <TXE> bit to “1.” When the last transmit data is moved to the transmit buffer, the transmit FIFO interrupt is generated. When transmission of the last data is completed, the clock is stopped and the transmission sequence is terminated.

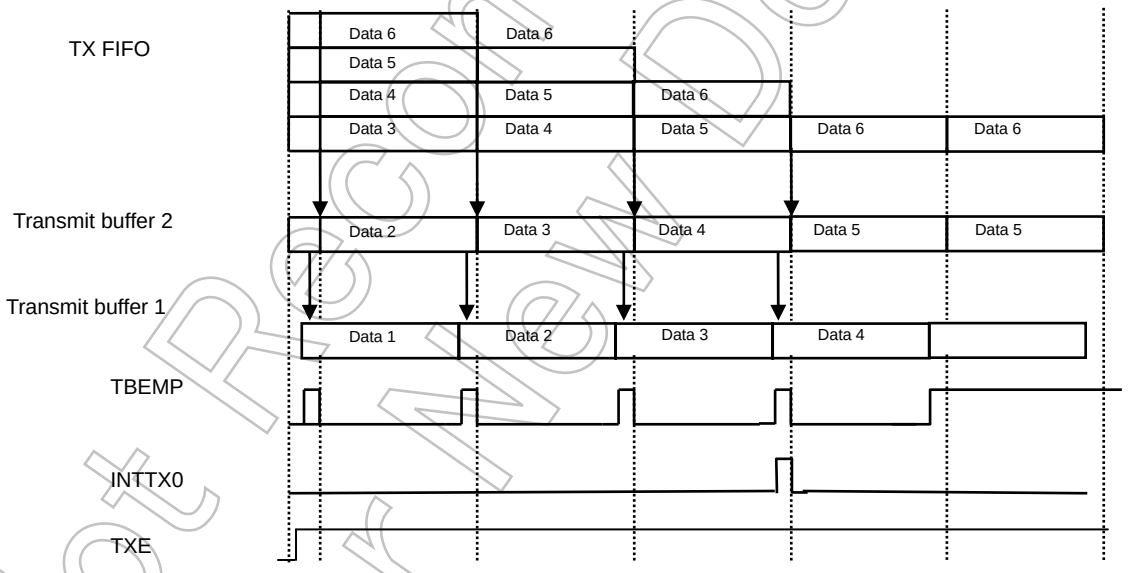


Fig. 13-8 Transmit FIFO Operation

② I/O interface mode with SCLK input (normal mode):

The following example describes the case a 4-byte data stream is transmitted:

SC0TFC <1:0> = 01: Clears the transmit FIFO and sets the condition of interrupt generation.

SC0TFC <7:2> = 000000: Sets the interrupt to be generated at fill level 0.

SC0FCNF <4:0> = 01001: Allows continued transmission after reaching the fill level.

In this condition, data transmission can be initiated along with the input clock by setting the transfer mode to half duplex, writing 4 bytes of data to the transmit FIFO, and setting the <TXE> bit to “1.” When the last transmit data is moved to the transmit buffer, the transmit FIFO interrupt is generated.

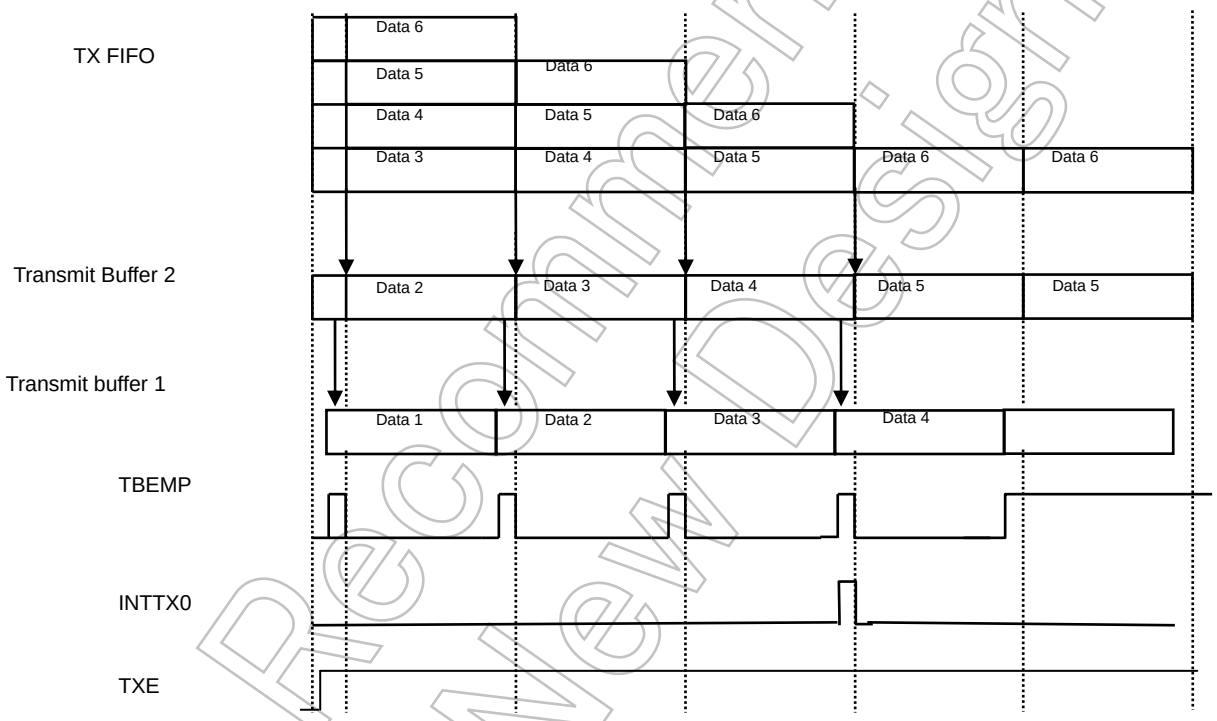


Fig. 13-9 Transmit FIFO Operation

13.3.14 Parity Control Circuit

If the parity addition bit <PE> of the serial control register SC0CR is set to “1,” data is sent with the parity bit. Note that the parity bit may be used only in the 7- or 8-bit UART mode. The <EVEN> bit of SC0CR selects either even or odd parity.

Upon data transmission, the parity control circuit automatically generates the parity with the data written to the transmit buffer (SC0BUF). After data transmission is complete, the parity bit will be stored in SC0BUF bit 7 <TB7> in the 7-bit UART mode and in bit 7 <TB8> in the serial mode control register SC0MOD in the 8-bit UART mode. The <PE> and <EVEN> settings must be completed before data is written to the transmit buffer.

Upon data reception, the parity bit for the received data is automatically generated while the data is shifted to receive buffer 1 and moved to receive buffer 2 (SC0BUF). In the 7-bit UART mode, the parity generated is compared with the parity stored in SC0BUF <RB7>, while in the 8-bit UART mode, it is compared with the bit 7 <RB8> of the SC0CR register. If there is any difference, a parity error occurs and the <PERR> flag of the SC0CR register is set.

In the I/O interface mode, the SC0CR <PERR> flag functions as an under-run error flag, not as a parity flag.

13.3.15 Error Flag

Three error flags are provided to inprove the reliability of received data.

1. Overrun error <OERR>: Bit 4 of the serial control register SC0CR

   In both UART and I/O interface modes, this bit is set to “1” when an error is generated by completing the reception of the next frame receive data before the receive buffer has been read. If the receive FIFO is enabled, the received data is automatically moved to the receive FIFO and no overrun error will be generated until the receive FIFO is full (or until the usable bytes are fully occupied). This flag is set to “0” when it is read. In the I/O interface SCLK output mode, no overrun error is generated and therefore, this flag is inoperative and the operation is undefined.

2. Parity error/under-run error <PERR>: Bit 3 of the SC0CR register

   In the UART mode, this bit is set to “1” when a parity error is generated. A parity error is generated when the parity generated from the received data is different from the parity received. This flag is set to “0” when it is read.

   In the I/O interface mode, this bit indicates an under-run error. When the double buffer control bit <WBUF> of the serial mode control register SC0MOD2 is set to “1” in the SCLK input mode, if no data is set to the transmit double buffer before the next data transfer clock after completing the transmission from the transmit shift register, this error flag is set to “1” indicating an under-run error. If the transmit FIFO is enabled, any data content in the transmit FIFO will be moved to the buffer. When the transmit FIFO and the double buffer are both empty, an under-run error will be generated. Because no under-run errors can be generated in the SCLK output mode, this flag is inoperative and the operation is undefined. If Transmit Buffer 2 is disabled, the under-run flag <PERR> will not be set. This flag is set to “0” when it is read.

3. Framing error <FERR>: Bit 2 of the SC0CR register

In the UART mode, this bit is set to “1” when a framing error is generated. This flag is set to “0” when it is read. A framing error is generated if the corresponding stop bit is determined to be “0” by sampling the bit at around the center. Regardless of the <SBLEN> (stop bit length) setting of the serial mode control register 2, SC0MOD2, the stop bit status is determined by only 1 bit on the receive side.

Operation mode	Error flag	Function
UART	OERR	Overrun error flag
	PERR	Parity error flag
	FERR	Framing error flag
I/O Interface (SCLK input)	OERR	Overrun error flag
	PERR	Underrun error flag (WBUF = 1)
		Fixed to 0 (WBUF = 0)
	FERR	Fixed to 0
I/O Interface (SCLK output)	OERR	Operation undefined
	PERR	Operation undefined
	FERR	Fixed to 0

13.3.16 Direction of Data Transfer

In the I/O interface mode, the direction of data transfer can be switched between “MSB first” and “LSB first” by the data transfer direction setting bit <DRCHG> of the SC0MOD2 serial mode control register 2. Don't switch the direction when data is being transferred.

13.3.17 Stop Bit Length

In the UART transmission mode, the stop bit length can be set to either 1 or 2 bits by bit 4 <SBLEN> of the SC0MOD2 register.

13.3.18 Status Flag

If the double buffer function is enabled (SC0MOD2 <WBUF> = “1”), the bit 6 flag <RBFLL> of the SC0MOD2 register indicates the condition of receive buffer full. When one frame of data has been received and transferred from buffer 1 to buffer 2, this bit is set to “1” to show that buffer 2 is full (data is stored in buffer 2). When the receive buffer is read by CPU/DMAC, it is cleared to “0.” If <WBUF> is set to “0,” this bit is insignificant and must not be used as a status flag. When double buffering is enabled (SC0MOD2 <WBUF> = “1”), the bit 7 flag <TBEMP> of the SC0MOD2 register indicates that Transmit Buffer 2 is empty. When data is moved from Transmit Buffer 2 to Transmit Buffer 1 (shift register), this bit is set to “1” indicating that Transmit Buffer 2 is now empty. When data is set to the transmit buffer by CPU/DMAC, the bit is cleared to “0.” If <WBUF> is set to “0,” this bit is insignificant and must not be used as a status flag.

13.3.19 Configurations of Transmit/Receive Buffer

		<WBUF> = 0	<WBUF> = 1
UART	Transmit buffer	Single	Double
	Receive buffer	Double	Double
I/O Interface (SCLK input)	Transmit buffer	Single	Double
	Receive buffer	Double	Double
I/O Interface (SCLK output)	Transmit buffer	Single	Double
	Receive buffer	Single	Double

13.3.20 Software reset

Software reset is generated by writing the bits 1 and 0 of SC0MOD2 <SWRST1:0> as “10” followed by “01”. As a result, SC0MOD0<RXE>, SC0MOD1<TXE>, SC0MOD2<TBEMP>,<RBFLL>,<TXRUN> of mode registers and SC0CR<OERR>, <PERR>, <FERR> of control registers and internal circuit is initialized. Other states are maintained.

13.3.21 Signal Generation Timing

① UART Mode :

Receive Side

Mode	9-bit	8-bit with parity	8-bit, 7-bit, and 7-bit with parity
Interrupt generation timing	Around the center of the 1st stop bit	Around the center of the 1st stop bit	Around the center of the 1st stop bit
Framing error generation timing	Around the center of the stop bit	Around the center of the stop bit	Around the center of the stop bit
Parity error generation timing	—	Around the center of the last (parity) bit	Around the center of the last (parity) bit
Overrun error generation timing	Around the center of the stop bit	Around the center of the stop bit	Around the center of the stop bit

Transmit Side

Mode	9-bit	8-bit with parity	8-bit, 7-bit, and 7-bit with parity
Interrupt generation timing (<WBUF> = 0)	Just before the stop bit is sent	Just before the stop bit is sent	Just before the stop bit is sent
Interrupt generation timing (<WBUF> = 1)	Immediately after data is moved to transmit buffer 1 (just before start bit transmission)	Immediately after data is moved to transmit buffer 1 (just before start bit transmission).	Immediately after data is moved to transmit buffer 1 (just before start bit transmission)

② I/O interface mode:

Receive Side

Interrupt generation timing (<WBUF> = 0)	SCLK output mode	Immediately after the rising edge of the last SCLK
	SCLK input mode	Immediately after the rising or falling edge of the last SCLK (for rising or falling edge mode, respectively).
Interrupt generation timing (<WBUF> = 1)	SCLK output mode	Immediately after the rising edge of the last SCLK (just after data transfer to receive buffer 2) or just after receive buffer 2 is read.
	SCLK input mode	Immediately after the rising edge or falling edge of the last SCLK (right after data is moved to receive buffer 2).
Overrun error generation timing	SCLK input mode	Immediately after the rising or falling edge of the last SCLK (for rising or falling edge mode, respectively)

Transmit Side

Interrupt generation timing (<WBUF> = 0)	SCLK output mode	Immediately after the rising edge of the last SCLK
	SCLK input mode	Immediately after the rising or falling edge of the last SCLK (for rising or falling edge mode, respectively)
Interrupt generation timing (<WBUF> = 1)	SCLK output mode	Immediately after the rising edge of the last SCLK or just after data is moved to Transmit Buffer 1
	SCLK input mode	Immediately after the rising or falling edge of the last SCLK or just after data is moved to Transmit Buffer 1
Under-run error generation timing	SCLK input mode	Immediately after the falling or rising edge of the next SCLK

Note 1) Do not modify any control register when data is being sent or received (in a state ready to transmit or receive).
Note 2) Do not stop the receive operation (by setting SC0MOD0 <RXE> = "0") when data is being received.
Note 3) Do not stop the transmit operation (by setting SC0MOD1 <TXE> = "0") when data is being transmitted.

13.4 Register Description (Only for Channel 0)

SC0MOD0 (0xFFFF_F702)

	7	6	5	4	3	2	1	0
Bit symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Transmit data bit 8	Handshake function control 0: CTS disable 1: CTS enable	Receive control 0: Disables reception 1: Enables reception	Wake-up function 0: Disable 1: Enable	Serial transfer mode 00: I/O interface mode 01: 7-bit length UARTmode 10: 8-bit length UARTmode 11: 9-bit length UARTmode		Serial transfer clock (for UART) 00: Timer TB7OUT 01: Baud rate generator 10:Internal clock $f_{SYS}/2$ 11: Externalclock (SCLK0 input)	

Note) In the I/O interface mode, the serial control register (SC0CR) is used for clock selection.

Wake-up function

	9-bit UART	Other modes
0	Interrupt when received	don't care
1	Interrupt only when RB8 = 1	

Handshake ($\overline{CTS}$ pin) enables

0	Disable(transmission is always allowed)
1	Enable

(Note) With <RXE> set to "0," set each mode register (SC0MOD0, SC0MOD1 and SC0MOD2). Then set <RXE> to "1."

Fig. 13-10 Serial Mode Control Register 0 (for SIO0, SC0MOD0)

SC0MOD1 (0xFFFF_F705)

	7	6	5	4	3	2	1	0
Bit symbol	I2S0	FDPX1	FDPX0	TXE	SINT2	SINT1	SINT0	−
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	0	0	0
Function	IDLE 0: Stop 1: Start	Transfer mode setting 00: Transfer prohibited 01:Half duplex(RX) 10:Half duplex(TX) 11:Full duplex		Transmit control 0:Disable 1:Enable	Interval time of continuous transmission 000:None 100:8SCLK 001:1SCLK 101:16SCLK 010:2SCLK 110:32SCLK 011:4SCLK 111:64SCLK			Write “0”.

<SINT2:0>: Specifies the interval time of continuous transmission when double buffering or FIFO is enabled in the I/O interface mode. This parameter is invalid for the UART mode or when an external clock is input.

<TXE>: This bit enables transmission and is valid for all the transfer modes. If disabled while transmission is in progress, transmission is inhibited only after the current frame of data is completed for transmission.

<FDPX1:0>: Configures the transfer mode in the I/O interface mode. Also configures the FIFO if it is enabled. In the UART mode, it is used only to specify the FIFO configuration.

<I2S0>: Specifies the Idle mode operation.

(Note) Please specify the mode first and then specify the <TXE> bit.

Fig. 13-11 Serial Mode Control Register 1 (for SIO0, SC0MOD1)

SC0MOD2 (0xFFFF_F706)

	7	6	5	4	3	2	1	0
Bit symbol	TBEMP	RBFLL	TXRUN	SBLEN	DRCHG	WBUF	SWRST1	SWRST0
Read/Write	R			R/W				
After reset	1	0	0	0	0	0	0	0
Function	Transmit buffer empty flag 0: full 1: Empty	Receive buffer full FLAG 0: Empty 1: full	In transmission flag 0: Stop 1: Start	STOP bit 0: 1-bit 1: 2-bit	Setting transfer direction 0: LSB first 1: MSB first	W-buffer 0: Disable 1: Enable	SOFT RESET Overwrite “01” on “10” to reset	

<SWRST1:0>: Overwriting “01” in place of “10” generates a software reset. When this software reset is executed, the mode register parameters SC0MOD0 <RXE>, SC0MOD1<TXE>, SC0MOD2 <TBEMP>, <RBFLL>, and <TXRUN>, control register parameters SC0CR <OERR>, <PERR>, and <FERR>, and their internal circuits are initialized.

<WBUF>: This parameter enables or disables the transmit/receive buffers to transmit (in both SCLK output/input modes) and receive (in SCLK output mode) data in the I/O interface mode and to transmit data in the UART. In all other modes, double buffering is enabled in both cases that 0 or 1 is set to <WBUF> bit.

<DRCHG>: Specifies the direction of data transfer in the I/O interface mode. In the UART mode, it is fixed to LSB first.

<TXRUN>: This is a status flag to show that data transmission is in progress.
When this bit is set to “1,” it indicates that data transmission operation is in progress. If it is “0,” the bit 7 <TBEMP> is set to “1” to indicate that the transmission has been fully completed and the same <TBEMP> is set to “0” to indicate that the transmit buffer contains some data waiting for the next transmission.

<RBFLL>: This is a flag to show that the receive double buffers are full. When a receive operation is completed and received data is moved from the receive shift register to the receive double buffers, this bit changes to “1” while reading this bit changes it to “0.”
If double buffering is disabled, this flag is insignificant.

<TBEMP>: This flag shows that the transmit double buffers are empty. When data in the transmit double buffers is moved to the transmit shift register and the double buffers are empty, this bit is set to “1.” Writing data again to the double buffers sets this bit to “0.”
If double buffering is disabled, this flag is insignificant.

<SBLEN>: This specifies the length of stop bit transmission in the UART mode. On the receive side, the decision is made using only a single bit regardless of the <SBLEN> setting.

(Note 1) While data transmission is in progress, any software reset operation must be executed twice in succession.

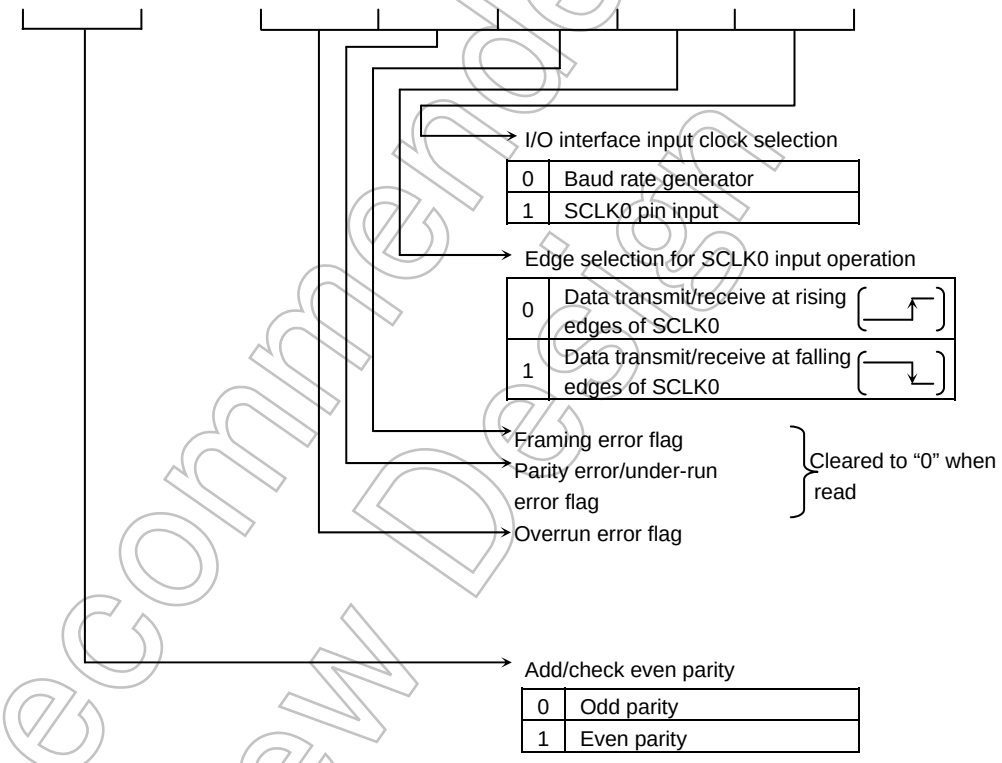
(Note 2) A software reset requires 2 clocks-duration at the time between the end of recognition and the start of execution of software reset instruction. Insert the SYNC and NOP instructions after the software reset instruction to secure the required time.

(Note 3) A software reset initializes other bits. Resetting a mode register and a control register are needed.

Fig. 13-12 Serial Mode Control Register

SC0CR (0xFFFF_F701)

	7	6	5	4	3	2	1	0
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	0	0	0	0	0	0	0	0
Function	Receive data bit 8	Parity 0: Odd 1: Even	Add parity 0: Disable 1: Enable	0: Normal operation 1: Error			0: SCLK0 (↑) 1: SCLK0 (↓)	0: Baud rate generator 1: SCLK0 pin input
				Over run	Parity/ under-run	Framing		



(Note) Any error flag is cleared when read.

Fig. 13-13 Serial Control Register (for SIO0, SC0CR)

BR0CR (0xFFFF_F703)

	7	6	5	4	3	2	1	0
Bit symbol	−	BR0ADDE	BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Write "0".	N + (16 − K)/16 divider function 0: Disable 1: Enable	00: ϕT1 01: ϕT4 10: ϕT16 11: ϕT64		Divide ratio "N"			

Select input clock to the baud rate generator

00	Internal clock ϕT1
01	Internal clock ϕT4
10	Internal clock ϕT16
11	Internal clock ϕT64

BR0ADD (0xFFFF_F704)

	7	6	5	4	3	2	1	0
Bit symbol					BR0K3	BR0K2	BR0K1	BR0K0
Read/Write	R				R/W			
After reset	0				0	0	0	0
Function	Always reads "0".				Specify K for the "N + (16 - K)/16" division			

Setting divide ratio of the baud rate generator

	BR0CR<BR0ADDE> = 1		BR0CR<BR0ADDE> = 0
BR0CR <BR0S3:0> / BR0ADD <BR0K3:0>	0000(N = 16) 〃 0001(N = 1)	0010(N = 2) 〃 1111(N = 15)	0001(N = 1) (ONLY UART) 1111 (N = 15) 0000 (N = 16)
0000	Disable	Disable	
0001(K = 1) 1111(K = 15)	Disable	$N + \frac{(16-K)}{16}$ Division	Divide by N

Fig. 13-14 Baud Rate Generator Control (for SIO0, BR0CR, BR0ADD)

(Note 1) In the UART mode, the division ratio "1" of the baud rate generator can be specified only when the "N + (16 - K)/16" division function is not used. In the I/O interface mode, the division ratio "1" of the baud rate generator can be specified only when double buffering is used.

(Note 2) To use the "N + (16 - K)/16" division function, be sure to set BR0CR <BR0ADDE> to "1" after setting the K value (K = 1 to 15) to BR0ADD <BR0K3:0>. However, don't use the "N + (16 - K)/16" division function when BR0CR <BR0S3:0> is set to either "0000" or "0001" (N = 16 or 1).

(Note 3) The "N + (16 - K)/16" division function can only be used in the UART mode. In the I/O interface mode, the "N + (16 - K)/16" division function is prohibited by setting BR0CR <BR0ADDE> to "0."

SC0BUF (0xFFFF_F700)

	7	6	5	4	3	2	1	0
Bit symbol	TB7/RB7	TB6/RB6	TB5/RB5	TB4/RB4	TB3/RB3	TB2/RB2	TB1/RB1	TB0/RB0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	TB7~0 : Transmit buffer + FIFO RB7~0 : Receive buffer + FIFO							

Fig. 13-15 Serial Transmit/Receive Buffer Register

(Note) SC0BUF works as a transmit buffer for WR operation and as a receive buffer for RD operation.

SC0FCNF (0xFFFF_F70C)

	7	6	5	4	3	2	1	0
Bit symbol	Reserved	Reserved	Reserved	RFST	TFIE	RFIE	RXTXCNT	CNFG
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	**Be sure to write "000".**			Bytes used in RX FIFO 0:Maximum 1: Same as Fill level of RX FIFO	TX interrupt for TX FIFO 0:Disable 1:Enable	RX interrupt for RX FIFO 0:Disable 1:Enable	Automatic disable of RXE/TXE 0:None 1:Auto disable	FIFO enable 0:Disable 1:Enable

<CNFG>: If enabled, the SCOMOD1 <FDPX1:0> setting automatically configures FIFO as follows:
<FDPX1:0>=01 (Half duplex RX) ---- 4-byte RX FIFO
<FDPX1:0>=10 (Half duplex TX) ---- 4-byte TX FIFO
<FDPX1:0>=11 (Full duplex) ---- 2-Byte RX FIFO + 2-Byte TX FIFO

<RXTXCNT>:0 The function to automatically disable RXE/TXE bits is disabled
: 1 If automatic disable is in operation, the SCOMOD1 <FDPX1:0> is used to set as follows:

<FDPX1:0>=01 (Half duplex RX) ------ When the RX FIFO is filled up to the specified number of valid bytes, RXE is automatically set to "0" to inhibit further reception.

<FDPX1:0>=10 (Half duplex TX) ------ When the TX FIFO is empty, TXE is automatically set to "0" to inhibit further transmission.

<FDPX1:0>=11 (Full duplex) ---------- When either of the above two conditions is satisfied, TXE/RXE are automatically set to "0" to inhibit further transmission and reception.

<RFIE>: When RX FIFO is enabled, receive interrupts are enabled or disabled by this parameter.
<TFIE>: When TX FIFO is enabled, transmit interrupts are enabled or disabled by this parameter.
<RFST>: When RX FIFO is enabled, the number of RX FIFO bytes to be used is selected.
0: The maximum number of bytes of the FIFO configured 4 bytes when <FDPX1:0> = 01 (Half duplex RX) and 2 bytes for <FDPX1:0> = 11 (Full duplex)
1: Same as the fill level for receive interrupt generation specified by SC0RFC <RIL1:0>.

(Note) Regarding TX FIFO, the maximum number of bytes being configured is always available. The available number of bytes is the bytes already written to the TX FIFO.

Fig. 13-16 FIFO Configuration Register

SC0RFC (0xFFFF_F708)

	7	6	5	4	3	2	1	0
Bit symbol	RFCS	RFIS					RIL1	RIL0
Read/Write	W	R/W	R				R/W	
After reset	0	0	0				0	0
Function	Clear RX FIFO 1:Clear Always reads “0.”	Select interrupt generation condition	Always reads “0.”				FIFO fill level to generate RX interrupts 00:4 bytes (2 bytes if full duplex) 01:1byte 10:2bytes 11:3bytes Note: RIL1 is ignored when FDPX1:0 = 11 (full duplex)	

0: An interrupt is generated when the data reaches to the specified fill level.
1: An interrupt is generated when the data reaches to the specified fill level or the data exceeds the specified fill level at the time data is read.

Fig. 13-17 RX FIFO Configuration Register

SC0TFC (0xFFFF_F709)

	7	6	5	4	3	2	1	0
Bit symbol	TFCS	TFIS					TIL1	TIL0
Read/Write	w	R/W	R				R/W	
After reset	0	0	0				0	0
Function	TX FIFO clear 1:Clear Always reads “0.”	Select interrupt generation condition	Always reads “0.”				FIFO fill level to generate TX interrupts 00:Empty 01:1byte 10:2bytes 11:3bytes Note: TIL1 is ignored when FDPX1:0 = 11 (full duplex).	

0: An interrupt is generated when the data reaches to the specified fill level.
1: An interrupt is generated when the data reaches to the specified fill level or the data cannot reach the specified fill level at the time new data is read.

Fig. 13-18 TX FIFO Configuration Register

SC0RST (0xFFFF_F70A)

	7	6	5	4	3	2	1	0
Bit symbol	ROR					RLVL2	RLVL1	RLVL0
Read/Write	R	R				R		
After reset	0	0				0	0	0
Function	RX FIFO Overrun 1:Generated	Always reads “0.”				Status of RX FIFO fill level 000:Empty 001:1Byte 010:2Byte 011:3Byte 100:4Byte		

(Note) The <ROR> bit is cleared to “0” when receive data is read from the SC0BUF register.

Fig. 13-19 RX FIFO Status Register

SC0TST (0xFFFF_F70B)

	7	6	5	4	3	2	1	0
Bit symbol	TUR					TLVL2	TLVL1	TLVL0
Read/Write	R	R				R		
After reset	1	0				0	0	0
Function	TX FIFO Under run 1: Generated Cleared by writing to FIFO	Always reads "0."				Status of TX FIFO fill level 000:Empty 001:1Byte 010:2Byte 011:3Byte 100:4Byte		

(Note) The <TUR> bit is cleared to "0" when transmit data is written to the SC0BUF register.

Fig. 13-20 TX FIFO Status Register

SC0EN (0xFFFF_F707)

	7	6	5	4	3	2	1	0
Bit symbol								SIOE
Read/Write	R							R/W
After reset	0							0
Function	Always reads "0."							SIO operation 0:Disable 1:Enable

<SIOE>: It specifies SIO operation. When SIO operation is disabled, the clock will not be supplied to the SIO module except for the register part and thus power consumption can be reduced (other registers cannot be accessed for read/write operation). When SIO is to be used, be sure to enable SIO by setting "1" to this register before setting any other registers of the SIO module. When SIO is disabled after it was enabled, each register setting is maintained.

Fig. 13-21 SIO Enable Register

13.5 Operation in Each Mode

13.5.1 Mode 0 (I/O interface mode)

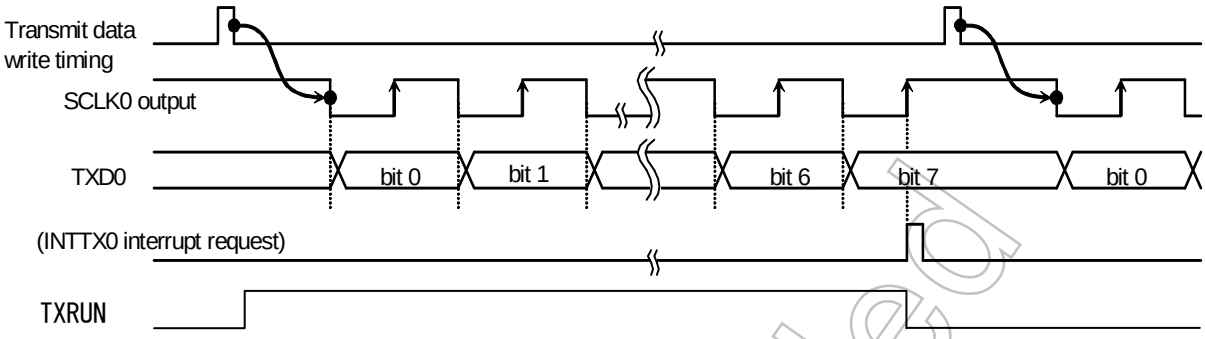
Mode 0 consists of two modes, the “SCLK output” mode to output synchronous clock and the “SCLK input” mode to accept synchronous clock from an external source. The following operational descriptions are for the case use of FIFO is disabled. For details of FIFO operation, refer to the previous sections describing receive/transmit FIFO functions.

① Transmitting data

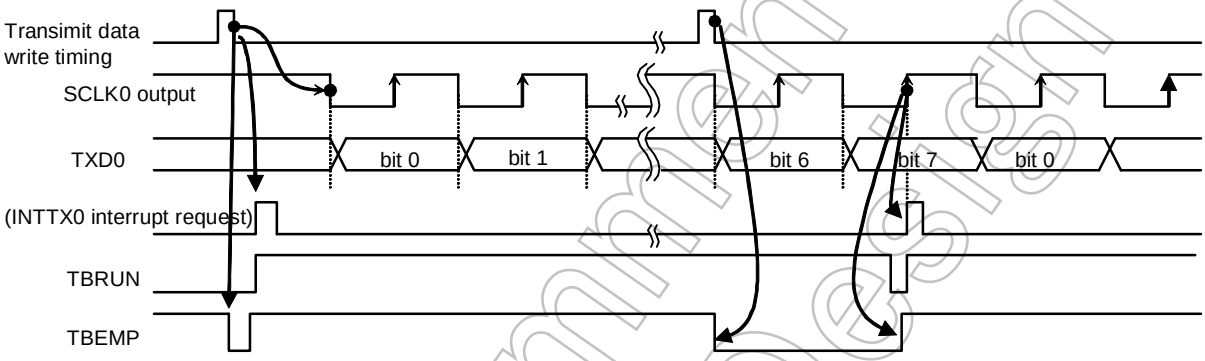
SCLK output mode

In the SCLK output mode, if SC0MOD2<WBUF> is set to “0” and the transmit double buffers are disabled, 8 bits of data are output from the TXD0 pin and the synchronous clock is output from the SCLK0 pin each time the CPU writes data to the transmit buffer. When all data is output, the INTTX0 interrupt is generated.

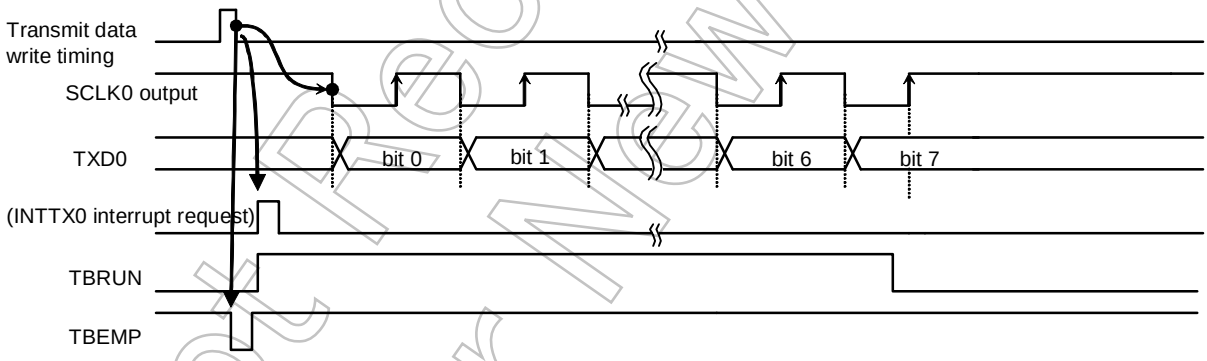
If SC0MOD2 <WBUF> is set to “1” and the transmit double buffers are enabled, data is moved from Transmit Buffer 2 to Transmit Buffer 1 when the CPU writes data to Transmit Buffer 2 while data transmission is halted or when data transmission from Transmit Buffer 1 (shift register) is completed. When data is moved from Transmit Buffer 2 to Transmit Buffer 1, the transmit buffer empty flag SC0MOD2 <TBEMP> is set to “1,” and the INTTX0 interrupt is generated. If Transmit Buffer 2 has no data to be moved to Transmit Buffer 1, the INTTX0 interrupt is not generated and the SCLK0 output stops.



<WBUF>="0" (if double buffering is disabled)



<WBUF>="1" (if double buffering is enabled and there is data in buffer 2)



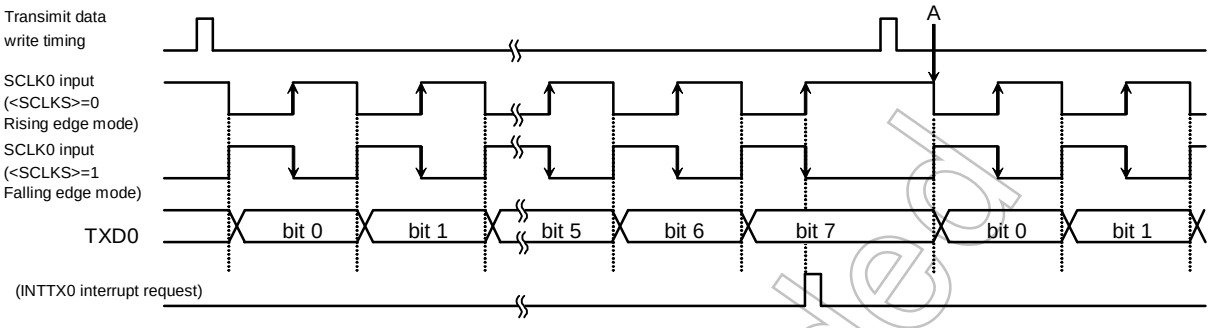
<WBUF>="1" (if double buffering is enabled and there is no data in buffer 2)

Fig. 13-22 Transmit Operation in the I/O Interface Mode (SCLK0 Output Mode)

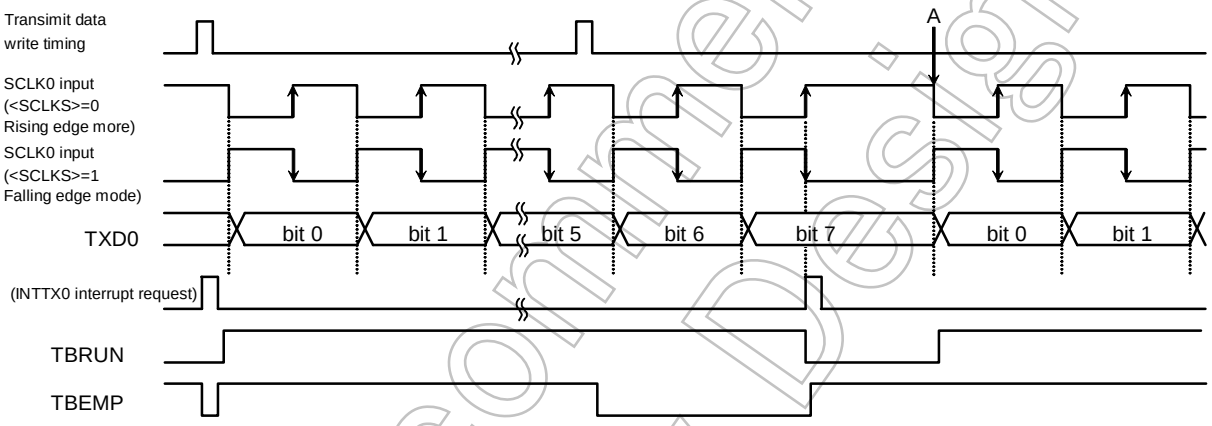
SCLK input mode

In the SCLK input mode, if SC0MOD2 <WBUF> is set to “0” and the transmit double buffers are disabled, 8-bit data that has been written in the transmit buffer is output from the TXD0 pin when the SCLK0 input becomes active. When all 8 bits are sent, the INTTX0 interrupt is generated. The next transmit data must be written before the timing point “A” as shown in Fig. 13-27.

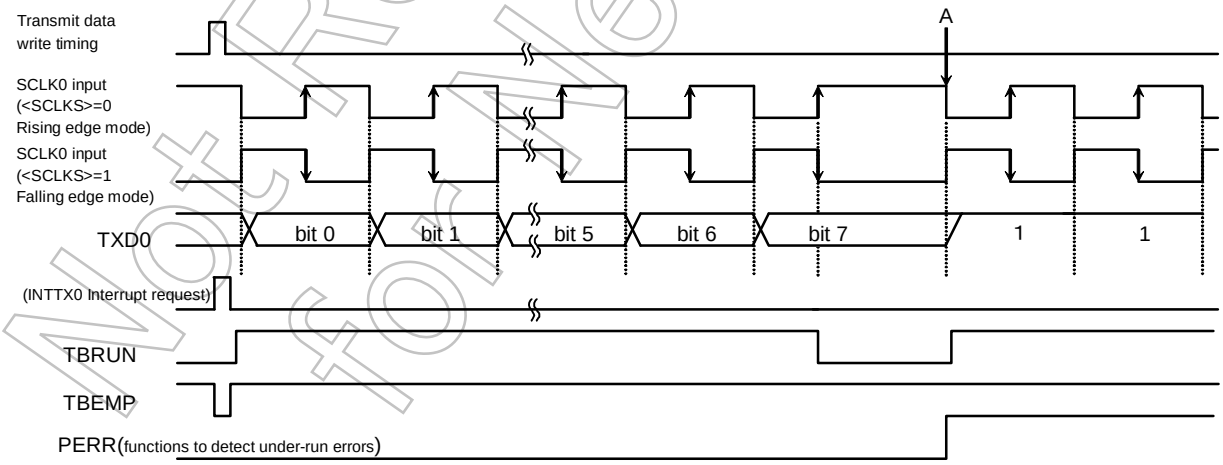
If SC0MOD2 <WBUF> is set to “1” and the transmit double buffers are enabled, data is moved from Transmit Buffer 2 to Transmit Buffer 1 when the CPU writes data to Transmit Buffer 2 before the SCLK0 becomes active or when data transmission from Transmit Buffer 1 (shift register) is completed. As data is moved from Transmit Buffer 2 to Transmit Buffer 1, the transmit buffer empty flag SC0MOD2 <TBEMP> is set to “1” and the INTTX0 interrupt is generated. If the SCLK0 input becomes active while no data is in Transmit Buffer 2, although the internal bit counter is started, an under-run error occurs and 8-bit dummy data (FFh) is sent.



<WBUF>="0" (if double buffering is disabled



<WBUF>="1" (if double buffering is enabled and there is data in buffer 2)



<WBUF>="1" (if double buffering is enabled and there is no data in buffer 2)

Fig. 13-23 Transmit Operation in the I/O Interface Mode (SCLK0 Input Mode)

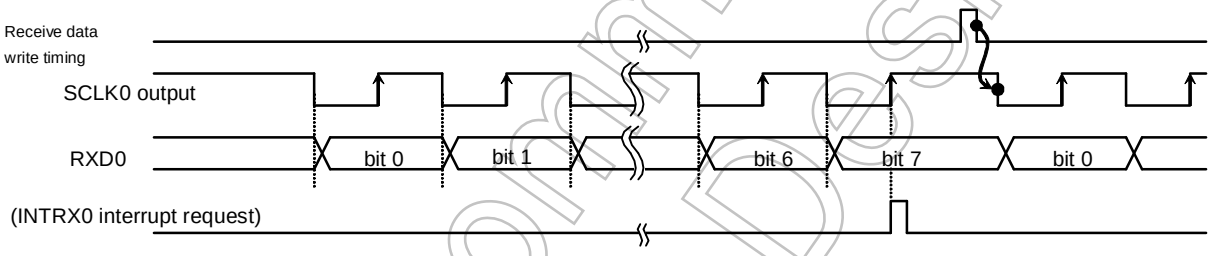
② Receiving data

SCLK output mode

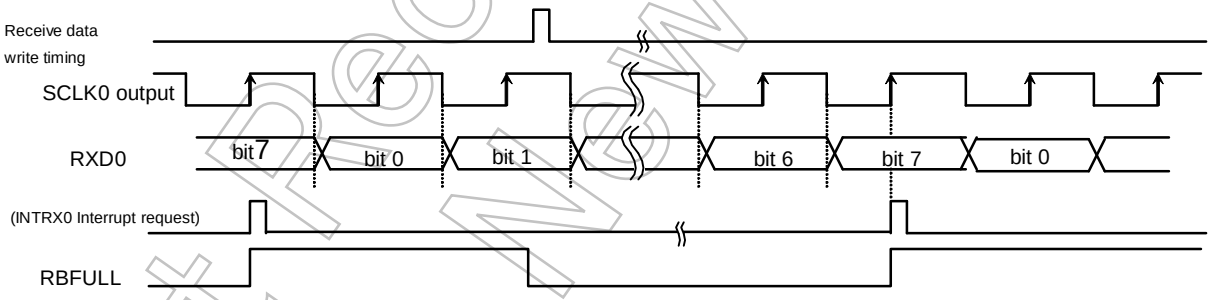
In the SCLK output mode, if SC0MOD2 <WBUF> = “0” and receive double buffering is disabled, a synchronous clock pulse is output from the SCLK0 pin and the next data is shifted into receive buffer 1 each time the CPU reads received data. When all the 8 bits are received, the INTRX0 interrupt is generated.

The first SCLK output can be started by setting the receive enable bit SC0MOD0 <RXE> to “1.” If the receive double buffering is enabled with SC0MOD2 <WBUF> set to “1,” the first frame received is moved to receive buffer 2 and receive buffer 1 can receive the next frame successively. As data is moved from receive buffer 1 to receive buffer 2, the receive buffer full flag SC0MOD2 <RBFULL> is set to “1” and the INTRX0 interrupt is generated.

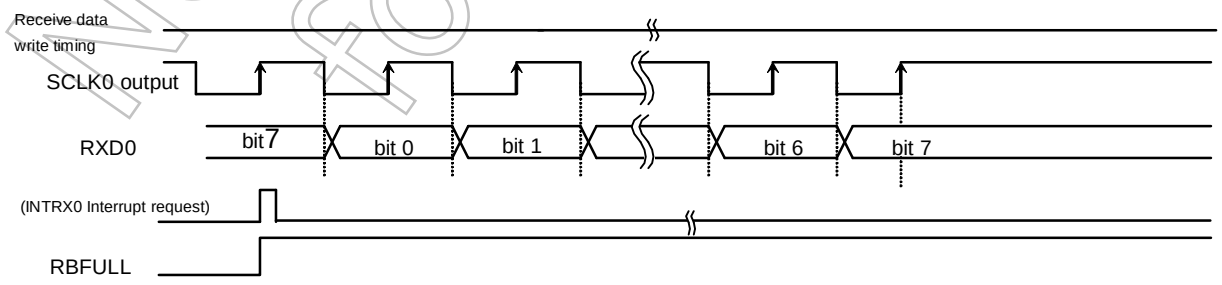
While data is in receive buffer 2, if CPU/DMAC cannot read data from receive buffer 2 before completing reception of the next 8 bits, the INTRX0 interrupt is not generated and the SCLK0 clock stops. In this state, reading data from receive buffer 2 allows data in receive buffer 1 to move to receive buffer 2 and thus the INTRX0 interrupt is generated and data reception resumes.



<WBUF>=“0” (if double buffering is disabled)



<WBUF>=“1” (if double buffering is enabled and data is read from buffer 2)



<WBUF>=“1” (if double buffering is enabled and data cannot be read from buffer 2)

Fig. 13-24 Receive Operation in the I/O Interface Mode (SCLK0 Output Mode)

SCLK input mode

In the SCLK input mode, receiving double buffering is always enabled, the received frame can be moved to receive buffer 2 and receive buffer 1 can receive the next frame successively.

The INTRX receive interrupt is generated each time received data is moved to received buffer 2.

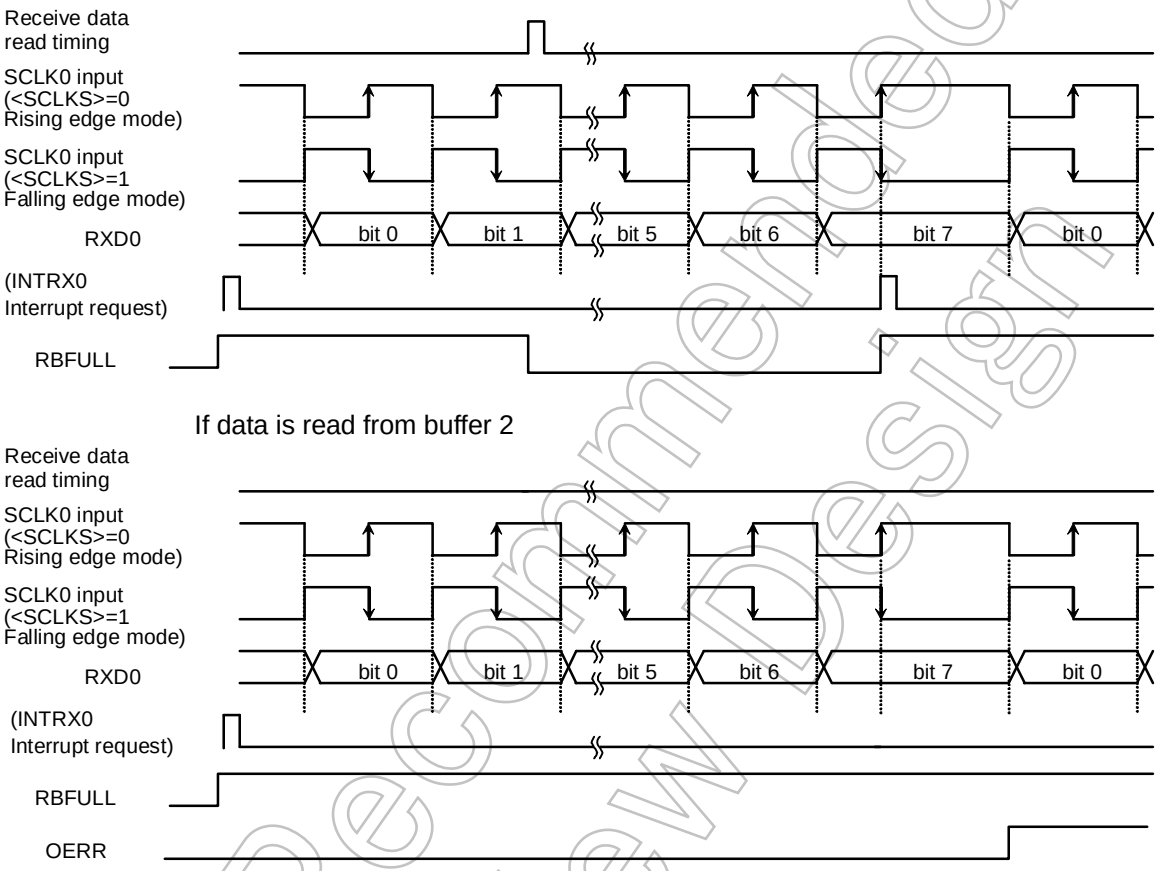


Fig. 13-25 Receive Operation in the I/O Interface Mode (SCLK0 Input Mode)

(Note) To receive data, SC0MOD <RXE> must always be set to "1" (receive enable) in the SCLK output / SCLK input mode.

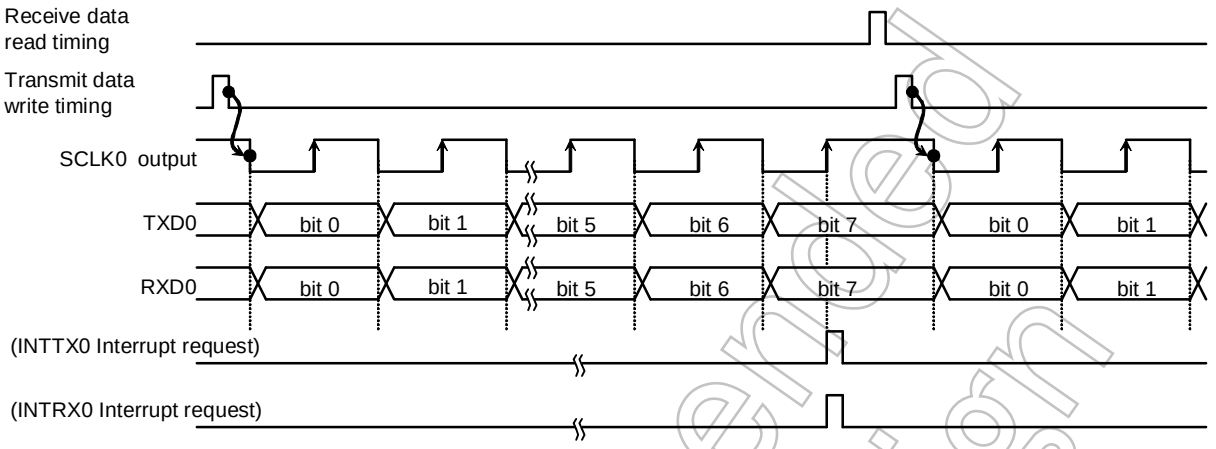
③ Transmit and receive (full-duplex)

The full-duplex mode is enabled by setting bit 6 <FDPX0> of the serial mode control register 1 (SC0MOD1) to “1”.

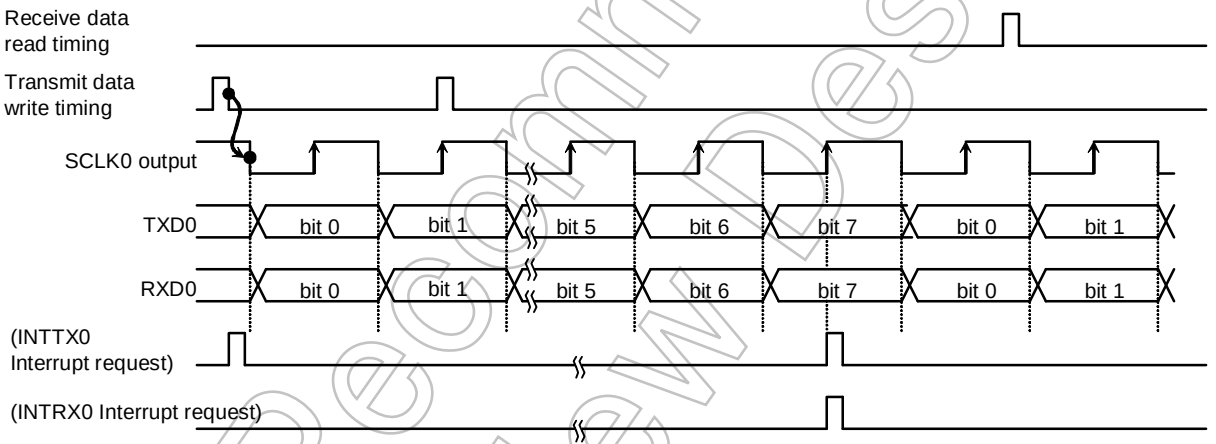
SCLK output mode

In the SCLK output mode, if SC0MOD2 <WBUF> is set to “0” and both the transmit and receive double buffers are disabled, SCLK is output when the CPU writes data to the transmit buffer. Subsequently, 8 bits of data are shifted into receive buffer 1 and the INTRX0 receive interrupt is generated. Concurrently, 8 bits of data written to the transmit buffer are output from the TXD0 pin, the INTTX0 transmit interrupt is generated when transmission of all data bits has been completed. Then, the SCLK output stops. In this, the next round of data transmission and reception starts when the data is read from the receive buffer and the next transmit data is written to the transmit buffer by the CPU. The order of reading the receive buffer and writing to the transmit buffer can be freely determined. Data transmission is resumed only when both conditions are satisfied.

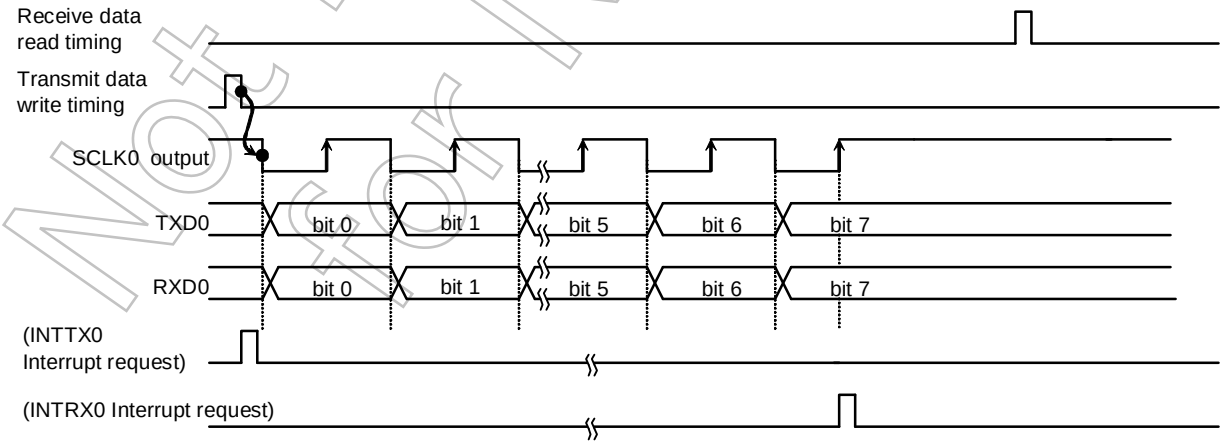
If SC0MOD2 <WBUF> = “1” and double buffering is enabled for both transmission and reception, SCLK is output when the CPU writes data to the transmit buffer. Subsequently, 8 bits of data are shifted into receive buffer 1, moved to receive buffer 2, and the INTRX0 interrupt is generated. While 8 bits of data is received, 8 bits of transmit data is output from the TXD0 pin. When all data bits are sent out, the INTTX0 interrupt is generated and the next data is moved from the Transmit Buffer 2 to Transmit Buffer 1. If Transmit Buffer 2 has no data to be moved to Transmit Buffer 1 (SC0MOD2 <TBEMP> = 1) or when receive buffer 2 is full (SC0MOD2 <RBFULL> = 1), the SCLK clock is stopped. When both conditions, receive data is read and transmit data is written, are satisfied, the SCLK output is resumed and the next round of data transmission is started.



<WBUF>="0" (if double buffering is disabled)



<WBUF>="1" (if double buffering is enabled)



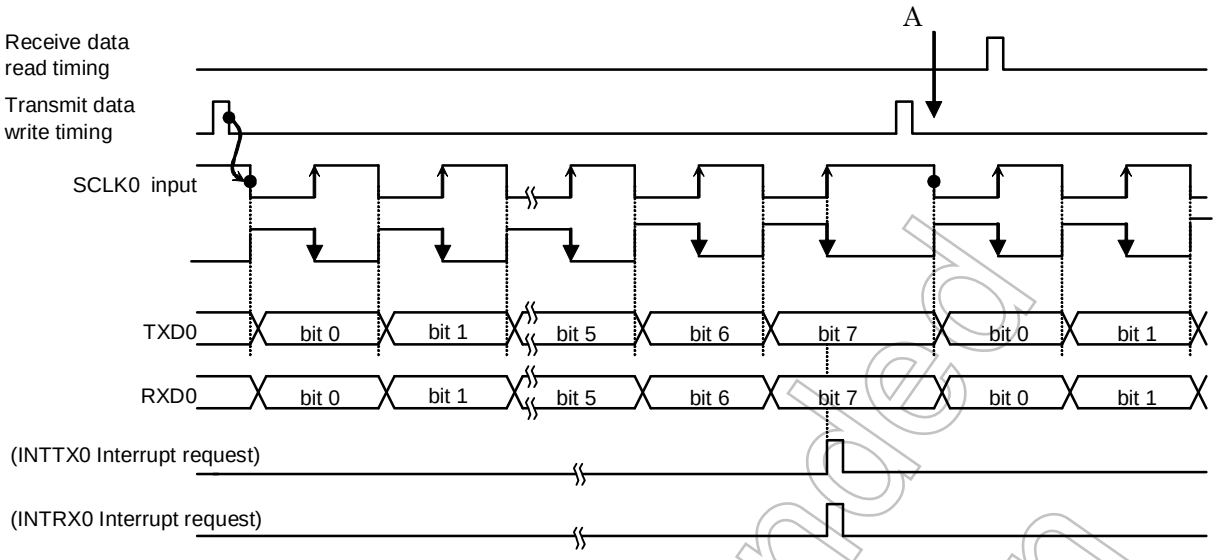
<WBUF>="1" (if double buffering is enabled)

Fig. 13-26 Transmit/Receive Operation in the I/O Interface Mode (SCLK0 Output Mode)

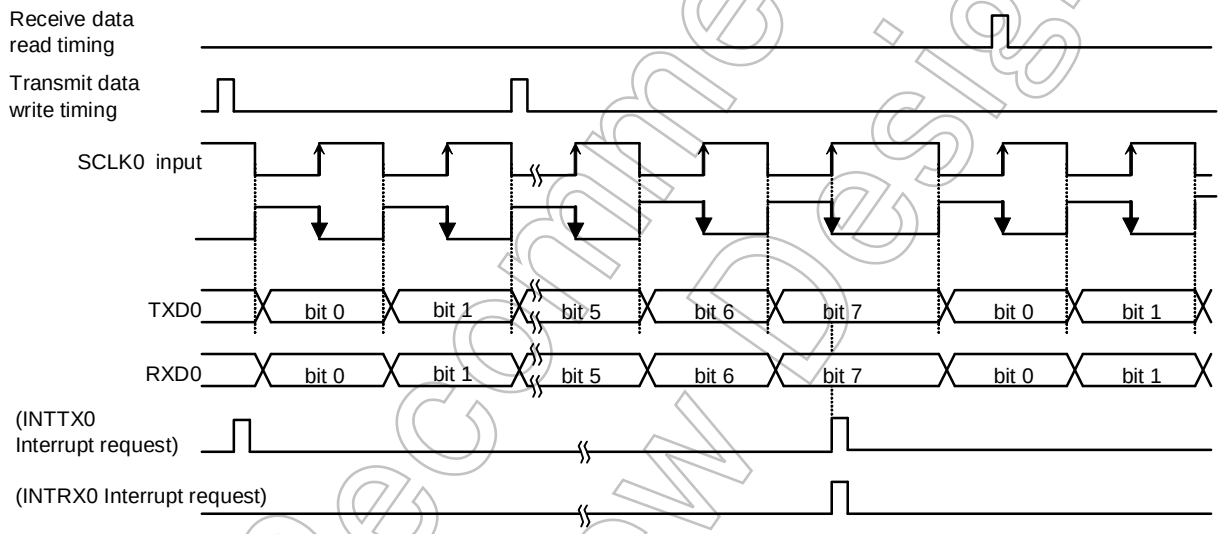
SCLK input mode

In the SCLK input mode with SC0MOD2 <WBUF> set to “0” and the transmit double buffers are disabled (double buffering is always enabled for the receive side), 8-bit data written in the transmit buffer is output from the TXD0 pin and 8 bits of data is shifted into the receive buffer when the SCLK input becomes active. The INTTX0 interrupt is generated upon completion of data transmission and the INTRX0 interrupt is generated at the instant the received data is moved from receive buffer 1 to receive buffer 2. Note that transmit data must be written into the transmit buffer before the SCLK input for the next frame (data must be written before the point A in Fig. 13- 31). As double buffering is enabled for data reception, data must be read before completing reception of the next frame data.

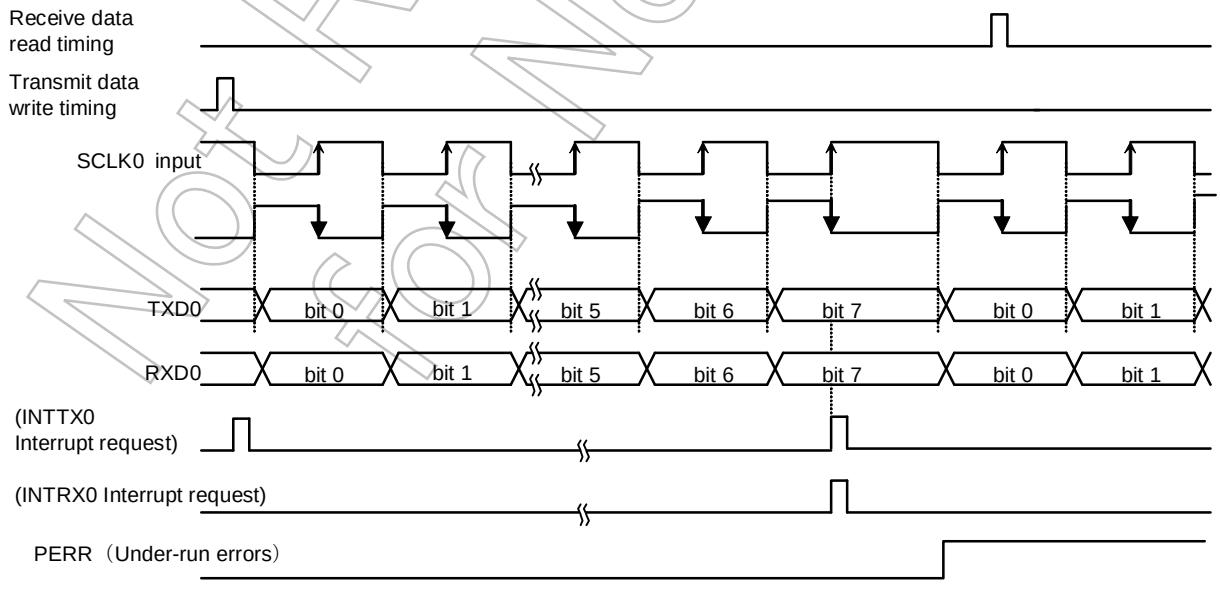
If SC0MOD2 <WBUF> = “1” and double buffering is enabled for both transmission and reception, the interrupt INTRX0 is generated at the timing Transmit Buffer 2 data is moved to Transmit Buffer 1 after completing data transmission from Transmit Buffer 1. At the same time, the 8 bits of data received is shifted to buffer 1, it is moved to receive buffer 2, and the INTRX0 interrupt is generated. Upon the SCLK input for the next frame, transmission from Transmit Buffer 1 (in which data has been moved from Transmit Buffer 2) is started while receive data is shifted into receive buffer 1 simultaneously. If data in receive buffer 2 has not been read when the last bit of the frame is received, an overrun error occurs. Similarly, if there is no data written to Transmit Buffer 2 when SCLK for the next frame is input, an under-run error occurs.



<WBUF>="0" (if double buffering is disabled)



<WBUF>="1" (if double buffering is enabled with no errors)



<WBUF>="1" (if double buffering is enabled with error generation)

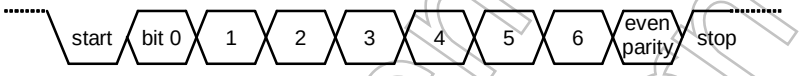
Fig. 13-27 Transmit/Receive Operation in the I/O Interface Mode (SCLK0 Input Mode)

13.5.2 Mode 1 (7-bit UART Mode)

The 7-bit UART mode can be selected by setting the serial mode control register (SC0MOD <SM1, 0>) to “01”.

In this mode, parity bits can be added to the transmit data stream; the serial mode control register (SC0CR <PE>) controls the parity enable/disable setting. When <PE> is set to “1” (enable), either even or odd parity may be selected using the SC0CR <EVEN> bit. The length of the stop bit can be specified using SC0MOD2<SBLEN>.

The following table shows the control register settings for transmitting in the following data format.



← Transmission direction (Transmission rate of 2400 bps @ fc = 24.576 MHz)

* Clocking conditions

System clock	:	high- speed (fc)
High-speed clock gear	:	x1 (fc)
Prescaler clock	:	$f_{periph/4}$ ($f_{periph} = fsys$)

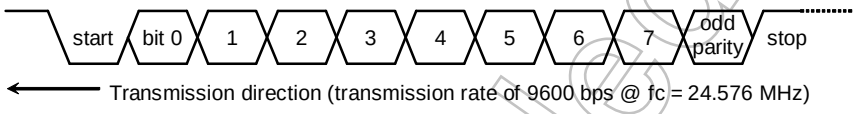
	7	6	5	4	3	2	1	0	
P9CR	← –	–	–	–	–	–	–	1	Designates P90 as the XD0 pin.
P9FC1	← –	–	–	–	–	–	–	1	
SC0MOD	← X	0	–	X	0	1	0	1	Sets the 7-bit UART mode.
SC0CR	← X	1	1	X	X	X	0	0	Adds even parity.
BR0CR	← 0	0	1	0	1	0	1	0	Sets the data rate to 2400 bps.
IMC4	← –	1	1	X	0	1	0	0	Enables the INTTX0 interrupt and sets to level 4 by the <23:16> bits of the 32 bit register.
SC0BUF	← *	*	*	*	*	*	*	*	Sets the data to be sent.

(Note) X: don’t care –: no change

13.5.3 Mode 2 (8-bit UART Mode)

The 8-bit UART mode can be selected by setting SC0MOD0 <SM1:0> to “10.” In this mode, parity bits can be added and parity enable/disable is controlled using SC0CR <PE>. If <PE> = “1” (enabled), either even or odd parity can be selected using SC0CR <EVEN>.

The control register settings for receiving data in the following format are as follows:



* Clocking conditions
- System clock : High-speed (fc)
- High-speed clock gear : x1 (fc)
- Prescaler clock : $f_{periph/4}$ ($f_{periph} = f_{sys}$)

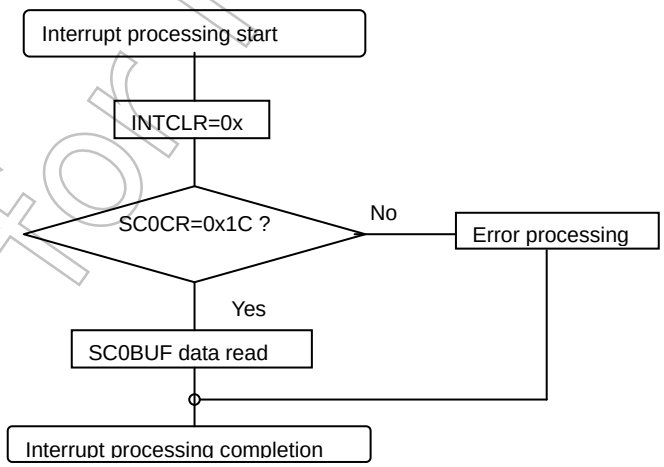
Main routine settings

		7	6	5	4	3	2	1	0	
P9CR	←	–	–	–	–	–	–	0	–	Designate P91 as the RXD0 pin.
P9FC1	←	–	–	–	–	–	–	1	–	
P9IE	←	–	–	–	–	–	–	1	–	
SC0MOD	←	–	0	0	X	1	0	0	1	Selects the 8-bit UART mode.
SC0CR	←	X	0	1	X	X	X	0	0	Sets odd parity.
BR0CR	←	0	0	0	1	0	1	0	1	Sets the data rate to 9600 bps...
IMC4	←	–	1	1	X	0	1	0	0	Enables the INTRX0 interrupt and sets to level 4 by the <15:8> bits of the 32 bit register.
SC0MOD	←	–	–	1	X	–	–	–	–	Enables reception of the data.

An example of interrupt routine process

INTCLR ← X 1 0 0 0 1 0 0	Clears the INTRX0 interrupt request. 0x0000_0044
Reg. ← SC0CR AND 0x1C	Performs error check.
If Reg. ≠ 0, ERROR processing	
Reg. ← SC0BUF	Reads received data.
Interrupt processing is completed	

(Note) X: don’t care –: no change



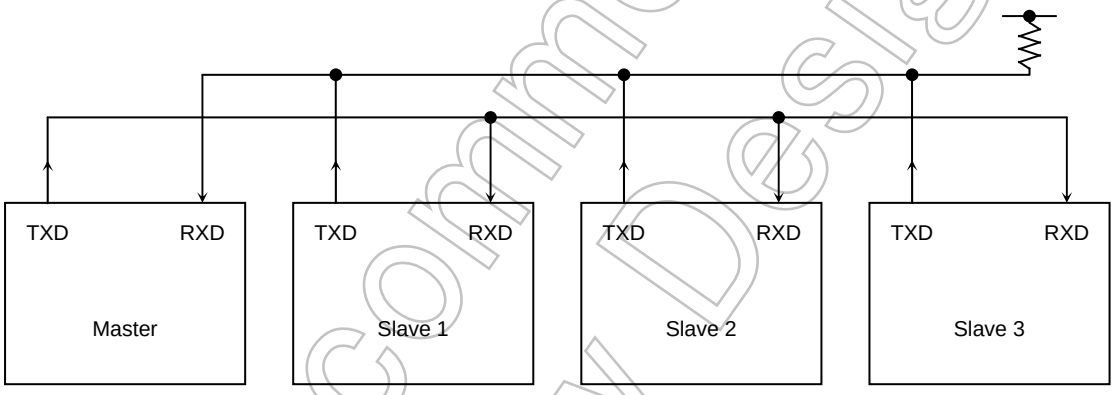
13.5.4 Mode 3 (9-bit UART)

The 9-bit UART mode can be selected by setting SC0MOD0 <SM1:0> to “11.” In this mode, parity bits must be disabled (SC0CR <PE> = “0”).

The most significant bit (9th bit) is written to bit 7 <TB8> of the serial mode control register 0 (SC0MOD0) for transmitting data. The data is stored in bit 7 <RB8> of the serial control register SC0CR. When writing or reading data to/from the buffers, the most significant bit must be written or read first before writing or reading to/from SC0BUF. The stop bit length can be specified using SC0MOD2 <SBLEN>.

Wakeup function

In the 9-bit UART mode, slave controllers can be operated in the wake-up mode by setting the wake-up function control bit SC0MOD0 <WU> to “1.” In this case, the interrupt INTRX0 will be generated only when SC0CR <RB8> is set to “1”.



(Note)The TXD pin of the slave controller must be set to the open drain output mode using the ODE register.

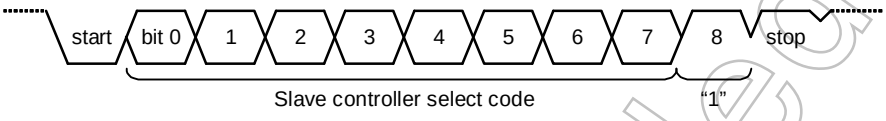
Fig. 13-28 Serial Links to Use Wake-up Function

Protocol

① Select the 9-bit UART mode for the master and slave controllers.

② Set SC0MOD <WU> to “1” for the slave controllers to make them ready to receive data.

③ The master controller is to transmit a single frame of data that includes the slave controller select code (8 bits). In this, the most significant bit (bit 8) <TB8> must be set to “1”.



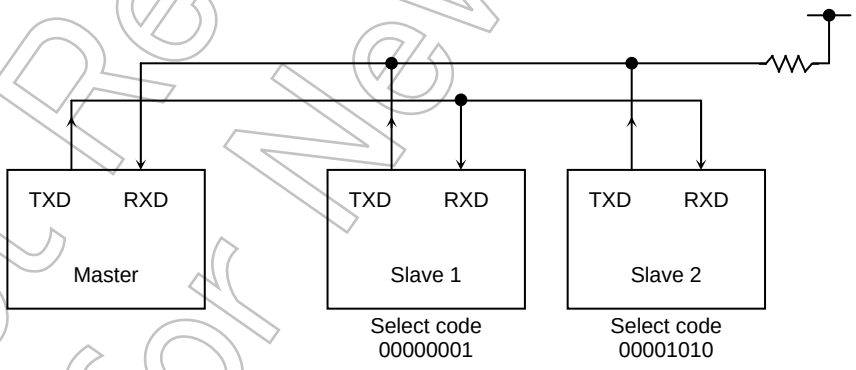
④ Each slave controller receives the above data frame; if the code received matches with the controller's own select code, it clears the WU bit to “0”.

⑤ The master controller transmits data to the designated slave controller (the controller of which SC0MOD <WU> bit is cleared to “0”). In this, the most significant bit (bit 8) <TB8> must be set to “0”.

⑥ The slave controllers with the <WU> bit set to “1” ignore the receive data because the most significant bit (bit 8) <RB8> is set to “0” and thus no interrupt (INTRX0) is generated.

Also, the slave controller with the <WU> bit set to “0” can transmit data to the master controller to inform that the data has been successfully received.

An example: Using the internal clock $f_{SYS}/2$ as the transfer clock, two slave controllers are serially linked as follows.



3) Master controller setting

Main routine

P9CR	← – – – – – – 0 1	Designate P90 and P91 as the TXD0 and RXD0 pins respectively.
P9FC1	← – – – – – – 1 1	
P9IE	← – – – – – – 1 –	
IMC4	← – 1 1 – 0 1 0 1	Enables the INTRX0 and sets the interrupt level to 5 by the <15:8> bits of the 32 bit register.
IMC4	← – 1 1 – 0 1 0 0	Enables the INTTX0 and sets the interrupt level to 4 by the <23:16> bits of the 32 bit register.
SC0MOD0	← 1 0 1 0 1 1 1 0	Sets the 9-bit UART mode and f_{SYS}/2 transfer clock.
SC0BUF	← 0 0 0 0 0 0 0 1	Sets the select code of Slave 1.

Interrupt routine (INTTX0)

INTCLR	← X 1 0 0 1 0 0 0	Clears the interrupt request. (INTTX0)
SC0MOD0	← 0 – – – – – – –	Sets TB8 to “0”.
SC0BUF	← * * * * * * * *	Sets the data to be sent.

Interrupt processing is completed

4) Slave controller setting

Main routine

P9CR	← – – – – – – 0 1	Designates P90 as TXD (open drain output) and P91 as RXD.
P9FC1	← – – – – – – 1 1	
P9ODE	← – – – – – – 1 –	
P9IE	← – – – – – – – 1	
IMC4	← – – 1 1 0 1 1 0	Enables INTTX0 and INTRX0.
IMC4	← – – 1 1 0 1 0 1	
SC0MOD0	← 0 0 1 1 1 1 1 0	Sets the 9-bit UART mode and f_{SYS}/2 transfer clock and then sets <WU> to “1.

Interrupt routine (INTRX0)

INTCLR	← 0 1 0 0 0 1 0 0	Clears the interrupt request.
Reg.	← SC0BUF	
if Reg. = Select code		
Then		
SC0MOD0	← – – – 0 – – – –	Clear <WU> to “0”.

14 Serial Channel (HSIO)

This device has a high-speed serial I/O channel, HSIO0. The UART mode (asynchronous communication) or the I/O interface mode (synchronous communication) are selectable.

I/O interface mode — Mode 0: This is the mode to transmit and receive I/O data and associated synchronization signals (HSCLK) to extend I/O.

Asynchronous (UART) mode
- Mode 1: TX/RX Data Length: 7 bits
- Mode 2: TX/RX Data Length: 8 bits
- Mode 3: TX/RX Data Length: 9 bits

In the above modes 1 and 2, parity bits can be added. The mode 3 has a wakeup function in which the master controller can start up slave controllers via the serial link (multi-controller system). Fig. 14-2 shows the block diagram of HSIO0.

The channel consists of a prescaler, a serial clock generation circuit, a receive buffer and its control circuit, and a transmit buffer and its control circuit.

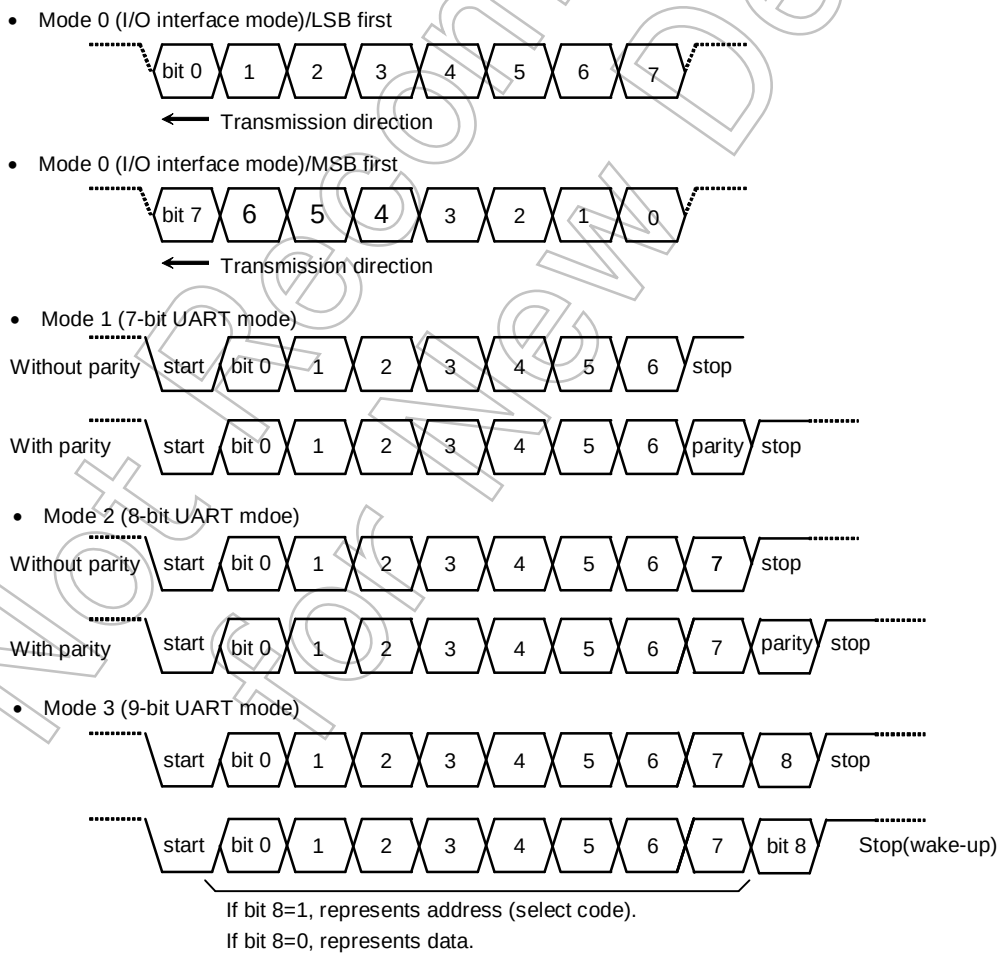


Fig. 14-1 Data Format

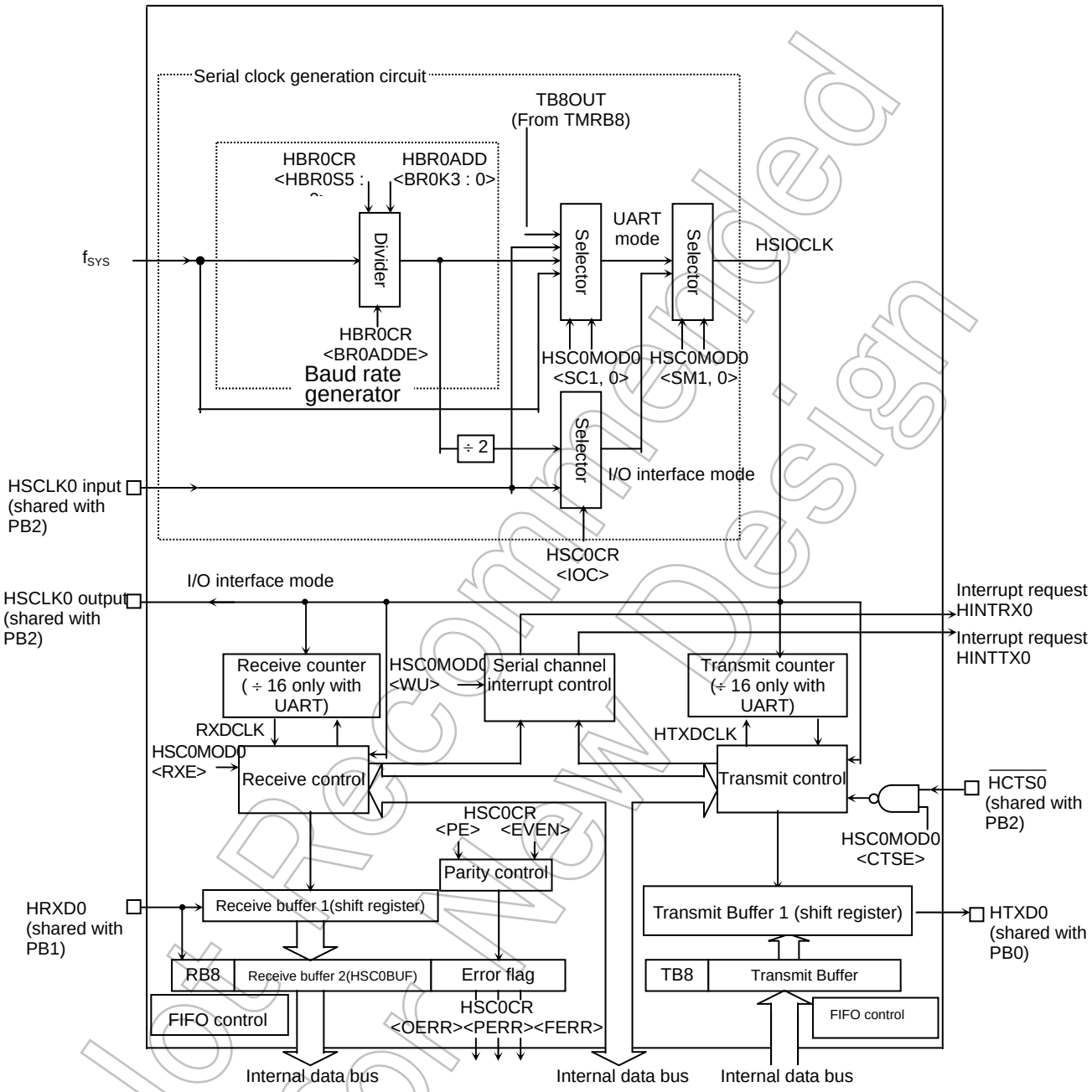


Fig. 14-2 HSIO0 Block Diagram

(Note) The baud rate generator cannot be set for "divide by 1".

14.1 Operation of Each Circuit (HSIO Channel 0)

14.1.1 Baud Rate Generator

The baud rate generator generates transmit and receive clocks to determine the serial channel transfer rate.

The baud rate generator uses the sys/2 clock.

The baud rate generator contains built-in dividers for divide by 1, (N + m/16), and 64 where N is a number from 2 to 63 and m is a number from 0 to 15. The division is performed according to the settings of the baud rate control registers HBR0CR <BR0ADDE> <BR0S5:0> and HBR0ADD <BR0K3:0> to determine the resulting transfer rate.

- UART mode

1) If HBR0CR <BR0ADDE> = 0,

   The setting of HBR0ADD <BR0K3:0> is ignored and the counter is divided by N where N is the value set to HBR0CR <BR0S5:0>. (N = 1 to 64).

2) If HBR0CR<BR0ADDE>=1,

   The N + (16 - K)/16 division function is enabled and the division is made by using the values N (set in HBR0CR <BR0S5:0>) and K (set in HBR0ADD<BR0K3:0>). (N = 2 to 63, K = 1 to 15).

(Note) For the N values of 1 and 64, the above N+(16-K)/16 division function is inhibited. So, be sure to set HBR0CR<BR0ADDE> to "0."

- I/O interface mode

The N + (16 - K)/16 division function cannot be used in the I/O interface mode. Be sure to divide by N, by setting HBR0CR <BR0ADDE> to "0".

- Baud rate calculation to use the baud rate generator:

1) UART mode

$$\text{Baud rate} = \frac{fsys}{\text{Frequency divided by the divide ratio}} \div 16$$

The highest baud rate out of the baud rate generator is 3.38 Mbps when fsys is 54 MHz (2.5 Mbps if fsys when 40MHz).

2) I/O interface mode

$$\text{Baud rate} = \frac{fsys}{\text{Frequency divided by the divide ratio}} \div 2$$

The highest baud rate will be generated when fsys is 54 MHz. If double buffering is used, the divide ratio can be set to "2" and the resulting output baud rate will be 13.5 Mbps. If double buffering is not used, the highest baud rate will be 6.75 Mbps applying the divide ratio of "4".

When fsys is 40MHz, the baud rate will be 10Mbps with double buffering and 5Mbps without double buffering.

- Examples of baud rate setting:

1) Division by an integer (divide by N)

Using the baud rate generator input clock fsys, setting the divide ratio N (HBR0CR<BR0S5:0>) = 4, and setting HBR0CR<BR0ADDE> = "0," the resulting baud rate in the UART mode is calculated as follows:

* Clocking conditions
  - System clock :High-speed (fc)
  - High-speed clock gear:x1 (fc)

$$\text{Baud rate} = \frac{\text{fsys}}{4} \div 16$$

$$= 54 \times 10^6 \div 4 \div 16 = 843.8\text{K (bps)}$$

(625K bps when fsys is 40 MHz)

(Note) The divide by (N + (16-K)/16) function is inhibited and thus HBR0ADD <BR0K3:0> is ignored.

1) For divide by N + (16-K)/16 (only for UART mode)

Using the baud rate generator fsys, setting the divide ratio N (HBR0CR<BR3S5:0>) = 4, setting K (HBR0ADD<BR3K3:0>) = 14, and selecting HBR0CR<BR3ADDE> = 1, the resulting baud rate is calculated as follows:

* Clocking conditions
  - System clock: high-speed (fc)
  - High-speed clock gear: x1 (fc)

$$\text{Baud rate} = \frac{\text{fsys}}{4 + \frac{(16 - 14)}{16}} \div 16$$

$$= 54 \times 10^6 \div \left(4 + \frac{2}{16}\right) \div 16 = 818.2\text{K (bps)}$$

(606.1kbps when fsys is 40MHz)

Also, an external clock input may be used as the serial clock. The resulting baud rate calculation is shown below:

- Baud rate calculation for an external clock input:

  1) UART mode

     Baud Rate = external clock input ÷ 16

     In this, the period of the external clock input must be equal to or greater than 2/fsys.

     If fsys = 54 MHz, the highest baud rate will be 54 / 4 / 16 = 843.8 (kbps).

     (625kbps when fsys is 40MHz)

  2) I/O interface mode

     Baud rate = External clock input

     When double buffering is used, it is necessary to satisfy the following relationship:

     External clock input period > 6/fsys

     Therefore, when fsys = 54 MHz, the baud rate must be set to a rate lower than 54 / 6 = 9 (Mbps). (6.67 Mbps when fsys is 40 Mhz)

     When double buffering is not used, it is necessary to satisfy the following relationship:

     External clock input period > 8/fsys

     Therefore, when fsys = 54 MHz, the baud rate must be set to a rate lower than 54 / 16 = 3.38 (Mbps).(5Mbps when fsys is 40MHz)

14.1.2 High-speed Serial Clock Generation Circuit

This circuit generates basic transmit and receive clocks.

- I/O interface mode

In the HSCLK output mode with the HSC0CR <IOC> serial control register set to "0," the output of the previously mentioned baud rate generator is divided by 2 to generate the basic clock.

In the HSCLK input mode with HSC0CR <IOC> set to "1," rising and falling edges are detected according to the HSC0CR <SCLKS> setting to generate the basic clock.

- Asynchronous (UART) mode

According to the settings of the serial control mode register HSC0MOD0 <SC1:0>, either the clock from the baud rate generator, the system clock (f_{SYS}), the internal output signal of the TMRB8 timer, or the external clock (HSCLKO pin) is selected to generate the basic clock, HSIOCL.

14.1.3 Receive Counter

The receive counter is a 4-bit binary counter used in the asynchronous (UART) mode and is up-counted by HSIOCLK. Sixteen HSIOCLK clock pulses are used in receiving a single data bit while the data symbol is sampled at the seventh, eighth, and ninth pulses. From these three samples, majority logic is applied to decide the received data.

14.1.4 Receive Control Unit

- I/O interface mode

In the HSCLK output mode with HSC0CR <IOC> set to "0," the HRXD0 pin is sampled on the rising edge of the shift clock output to the HSCLK0 pin.

In the HSCLK input mode with HSC0CR <IOC> set to "1," the serial receive data HRXD0 pin is sampled on the rising or falling edge of HSCLK input depending on the HSC0CR <SCLKS> setting.

- Asynchronous (UART) mode :

The receive control unit has a start bit detection circuit, which is used to initiate receive operation when a normal start bit is detected.

14.1.5 Receive Buffer

The receive buffer is of a dual structure to prevent overrun errors. Receive Buffer 1 (a shift register) stores the received data bit-by-bit. When a complete set of bits have been stored, they are moved to Receive Buffer 2 (HSC0BUF). At the same time, the receive buffer full flag (HSC0MOD2 "RBFLL") is set to "1" to indicate that valid data is stored in Receive Buffer 2. However, if the receive FIFO is set enabled, the receive data is moved to the receive FIFO and this flag is immediately cleared.

If the receive FIFO has been disabled (HSCOFCNF <CNFG> = 0 and HSC0MOD1 <FDPX1:0> = 01), the HINTRX0 interrupt is generated at the same time. If the receive FIFO has been enabled (HSCNFCNF <CNFG> = 1 and HSC0MOD1<FDPX1:0>=01/11), an interrupt will be generated according to the HSC0RFC <RIL1:0> setting.

The CPU will read the data from either Receive Buffer 2 (HSC0BUF) or from the receive FIFO (the address is the same as that of the receive buffer). If the receive FIFO has not been enabled, the receive buffer full flag <RBFLL> is cleared to "0" by the read operation. The next data received can be stored in Receive Buffer 1 even if the CPU has not read the previous data from Receive Buffer 2 (HSC0BUF) or the receive FIFO.

If HSCLK is set to generate clock output in the I/O interface mode, the double buffer control bit HSC0MOD2 <WBUF> can be programmed to enable or disable the operation of Receive Buffer 2 (HSCOBUF).

By disabling Receive Buffer 2 and also disabling the receive FIFO (HSCOFCNF <CNFG> = 0 and <FDPX1:0> = 01), handshaking with the other side of communication can be enabled and the HSCLK output stops each time one frame of data is transferred. In this setting, the CPU reads data from Receive Buffer 1. By the read operation of CPU, the HSCLK output resumes.

If the Receive Buffer 2 (i.e., double buffering) is enabled but the receive FIFO is not enabled, the HSCLK output is stopped when the first receive data is moved from Receive Buffer 1 to Receive Buffer 2 and the next data is stored in the first buffer filling both buffers with valid data. When Receive Buffer 2 is read, the data of Receive Buffer 1 is moved to Receive Buffer 2 and the HSCLK output is resumed upon generation of the receive interrupt HINTRX. Therefore, no buffer overrun error will be caused in the I/O interface HSCLK output mode regardless of the setting of the double buffer control bit HSC0MOD2 <WBUF>.

If Receive Buffer 2 (double buffering) is enabled and the receive FIFO is also enabled (HSCNFCNF<CNFG>=1 and HSC0MOD1<FDPX1:0>＝01/11), the HSCLK output will be stopped when the receive FIFO is full (according to the setting of HSCOFNCF<RFST>) and both Receive Buffers 1 and 2 contain valid data. Also in this case, if HSC0FCNF<RXTXCNT> has been set to "1," the receive control bit RXE will be automatically cleared upon suspension of the HSCLK output. If it is set to "0," automatic clearing will not be performed.

(Note) In this mode, the HSC0CR <OERR> flag is insignificant and the operation is undefined. Therefore, before switching from the HSCLK output mode to another mode, the HSC0CR register must be read to initialize this flag.

In other operating modes, the operation of Receive Buffer 2 is always valid, and it enables to improve the performance of continuous data transfer. If the receive FIFO is not enabled, an overrun error occurs when the data in Receive Buffer 2 (HSC0BUF) has not been read before Receive Buffer 1 is full with the next receive data. If an overrun error occurs, data in Receive Buffer 1 will be lost while data in Receive Buffer 2 and the contents of HSC0CR <RB8> remain intact. If the receive FIFO is enabled, the FIFO must be read before the FIFO is full and Receive Buffer 2 is written by the next data through the first buffer. Otherwise, an overrun error will be generated and the receive FIFO overrun error flag will be set. Even in this case, the data already in the receive FIFO remains intact.

The parity bit to be added in the 8-bit UART mode as well as the most significant bit in the 9-bit UART mode will be stored in HSC0CR <RB8>.

In the 9-bit UART mode, the slave controller can be operated in the wake-up mode by setting the wake-up function HSC0MOD0 <WU> to "1." In this case, the interrupt HINTRX0 will be generated only when HSC0CR <RB8> is set to "1."

14.1.6 Receive FIFO Buffer

In addition to the double buffer function already described, data may be stored using the receive FIFO buffer. By setting <CNFG> of the HSC0FCNF register and <FDPX1:0> of the HSC0MOD1 register, the 4-byte receive buffer can be enabled. Also, in the UART mode or I/O interface mode, data may be stored up to a predefined fill level. When the receive FIFO buffer is to be used, be sure to enable the double buffer function.

14.1.7 Receive FIFO Operation

① I/O interface mode with HSCLK output

The following example describes the case a 4-byte data stream is received in the half duplex mode:

HSC0RFC<7:6>=01: Clears receive FIFO and sets the condition of interrupt generation.

HSC0RFC<1:0>=00: Sets the interrupt to be generated at fill level 4.

HSC0FCNF <4:0>=10111: Automatically inhibits continued reception after reaching the fill level.

The number of bytes to be used in the receive FIFO is the same as the interrupt generation fill level.

In this condition, 4-byte data reception may be initiated by setting the half duplex transmission mode and writing "1" to the RXE bit. After receiving 4 bytes, the RXE bit is automatically cleared and the receive operation is stopped (HSCLK is stopped)

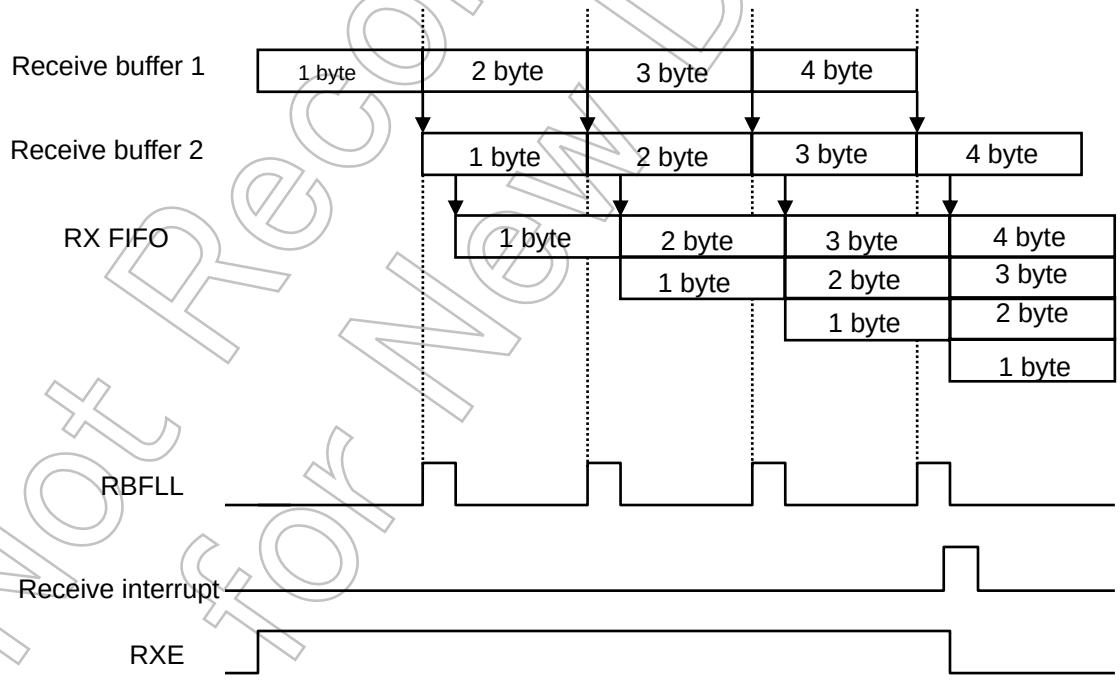


Fig. 14-3 Receive FIFO Operation

② I/O interface mode with HSCLK input

The following example describes the case a 4-byte data stream is received:

HSC0RFC <7:6> = 10: Clears receive FIFO and sets the condition of interrupt generation

HSC0RFC <1:0> = 00: Sets the interrupt to be generated at fill level 4.

HSC0FCNF <1:0> = 10101: Automatically allows continued reception after reaching the fill level.

The number of bytes to be used in the receive FIFO is the maximum allowable number.

In this condition, 4-byte data reception can be initiated along with the input clock by setting the half duplex transmission mode and writing "1" to the RXE bit. When the 4-byte data reception is completed, the receive FIFO interrupt will be generated.

Note that preparation for the next data reception can be managed in this setting, i.e., the next 4-byte data can be received before data is fully read from the FIFO.

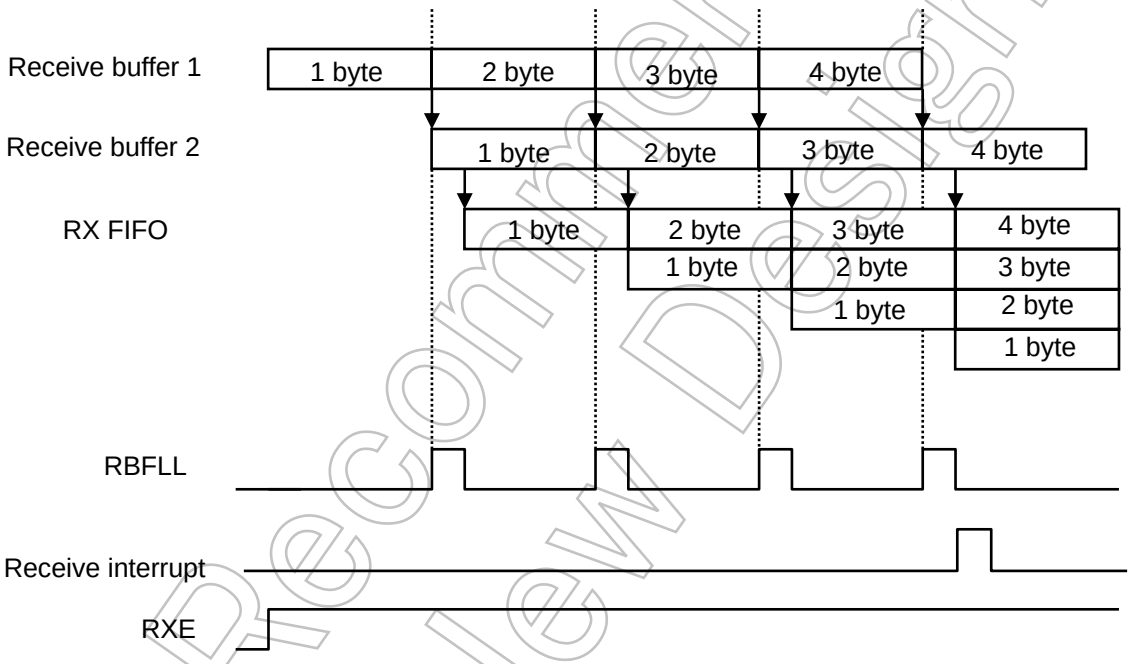


Fig. 14-4 Receive FIFO Operation

14.1.8 Transmit Counter

The transmit counter is a 4-bit binary counter used in the asynchronous communication (UART) mode. It is counted by HSIOCLK as in the case of the receive counter and generates a transmit clock (TXDCLK) on every 16th clock pulse.

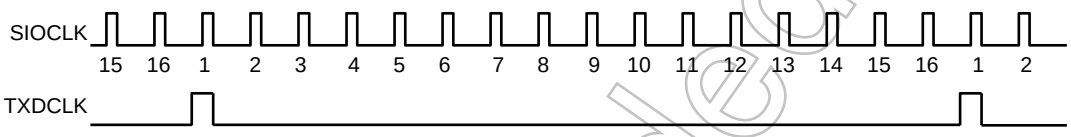


Fig. 14-5 Transmit Clock Generation

14.1.9 Transmit Control Unit

- I/O interface mode:

In the HSCLK output mode with HSC0CR <IOC> set to "0," each bit of data in the transmit buffer is output to the HTXD0 pin on the rising edge of the shift clock output from the HSCLK0 pin.

In the HSCLK input mode with HSC0CR <IOC> set to "1," each bit of data in the transmit buffer is output to the HTXD0 pin on the rising or falling edge of the input HSCLK signal according to the HSC0CR <SCLKS> setting.

- Asynchronous (UART) mode :

When the CPU writes data to the transmit buffer, the transmitting of data begins on the rising edge of the next HTXDCLK and a transmit shift clock (HTXDSFT) is generated.

Handshake function

The $\overline{\text{HCTS}}$ pin enables frame by frame data transmission so that overrun errors can be prevented. This function can be enabled or disabled by HSC0MOD0 <CTSE>.

When the $\overline{\text{HCTS}}$ pin is set to the "H" level, the current data transmission can be completed but the next data transmission is suspended until the $\overline{\text{HCTS}}$ pin returns to the "L" level. However in this case, the HINTTX0 interrupt is generated, the next transmit data is requested to the CPU, data is written to the transmit buffer, and it waits until it is ready to transmit data.

Although no $\overline{\text{HCTS}}$ pin is provided, a handshake control function can be easily implemented by assigning a port for the $\overline{\text{HRTS}}$ function. By setting the port to "H" level upon completion of data reception (in the receive interrupt routine), the transmit side can be requested to suspend data transmission.

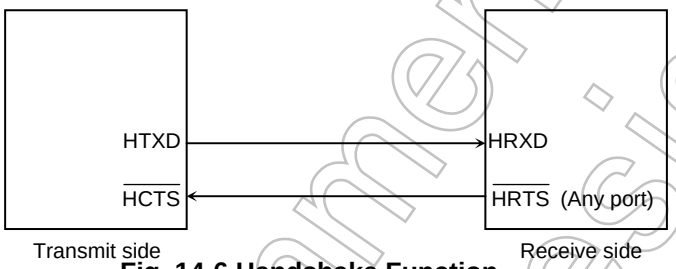


Fig. 14-6 Handshake Function

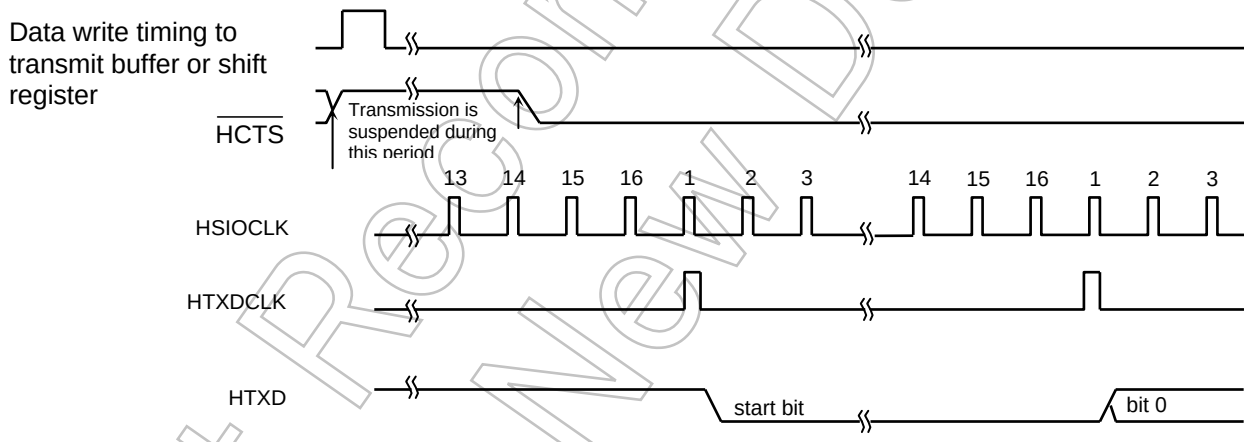


Fig. 14-7 $\overline{\text{HCTS}}$ (Clear to Transmit) Signal Timing

(Note) If the $\overline{\text{HCTS}}$ signal is set to "H" during transmission, the next data transmission is suspended after the current transmission is completed.
Data transmission starts on the first falling edge of the HTXDCLK clock after $\overline{\text{HCTS}}$ is set to "L".

14.1.10 Transmit Buffer

The transmit buffer (HSC0BUF) is in a dual structure. The double buffering function may be enabled or disabled by setting the double buffer control bit <WBUF> in serial mode control register 2 (HSC0MOD2). If double buffering is enabled, data written to Transmit Buffer 2 (HSCOBUF) is moved to Transmit Buffer 1 (shift register).

If the transmit FIFO has been disabled (HSCOFCNF <CNFG> = 0 or 1 and <FDPX1:0> = 01), the HINTTX0 transmit interrupt is generated at the same time and the transmit buffer empty flag <TBEMP> of HSC0MOD2 is set to "1." This flag indicates that Transmit Buffer 2 is now empty and that the next transmit data can be written. When the next data is written to Transmit Buffer 2, the <TBEMP> flag is cleared to "0."

If the transmit FIFO has been enabled (HSCNFCNF <CNFG> = 1 and <FDPX1:0> = 10/11), any data in the transmit FIFO is moved to the Transmit Buffer 2 and <TBEMP> flag is immediately cleared to "0." The CPU writes data to Transmit Buffer 2 or to the transmit FIFO.

If the transmit FIFO is disabled in the I/O interface HSCLK input mode and if no data is set in Transmit Buffer 2 before the next frame clock input, which occurs upon completion of data transmission from Transmit Buffer 1, an under-run error occurs. Then a serial control register (HSC0CR) <PERR> parity/under-run flag is set.

If the transmit FIFO is enabled in the I/O interface HSCLK input mode, when data transmission from Transmit Buffer 1 is completed, the Transmit Buffer 2 data is moved to Transmit Buffer 1 and any data in transmit FIFO is moved to Transmit Buffer 2 at the same time.

If the transmit FIFO is disabled in the I/O interface HSCLK output mode, when data in Transmit Buffer 2 is moved to Transmit Buffer 1. When the data transmission is completed, the HSCLK output stops. So, no under-run errors can be generated.

If the transmit FIFO is enabled in the I/O interface HSCLK output mode, the HSCLK output stops upon completion of data transmission from Transmit Buffer 1 if there is no valid data in the transmit FIFO.

(Note) In the I/O interface HSCLK output mode, the HSC0CR <PERR> flag is insignificant. In this case, the operation is undefined. Therefore, to switch from the HSCLK output mode to another mode, HSC0CR must be read in advance to initialize the flag.

If double buffering is disabled, the CPU writes data only to Transmit Buffer 1 and the transmit interrupt HINTTX0 is generated upon completion of data transmission.

If handshaking with the other side is necessary, set the double buffer control bit <WBUF> to "0" (disable) to disable Transmit Buffer 2 and do not use the transmit FIFO function.

14.1.11 Transmit FIFO Buffer

In addition to the double buffer function already described, data may be stored using the transmit FIFO buffer. By setting <HCNFG> of the HSC0FCNF register and <FDPX1:0> of the HSC0MOD1 register, the 4-byte transmit buffer can be enabled. In the UART mode or I/O interface mode, up to 4 bytes of data may be stored.

14.1.12 Transmit FIFO Operation

① I/O interface mode with HSCLK output (normal mode):

The following example describes the case a 4-byte data stream is transmitted:

HSC0TFC <7:6> = 01: Clears transmit FIFO and sets the condition of interrupt generation

HSC0TFC <1:0> = 00: Sets the interrupt to be generated at fill level 0.

HSC0FCNF <4:0> = 01011: Inhibits continued transmission after reaching the fill level.

In this condition, data transmission can be initiated by setting the transfer mode to half duplex, writing 4 bytes of data to the transmit FIFO, and setting the <TXE> bit to "1." When the last transmit data is moved to the transmit buffer, the transmit FIFO interrupt is generated. When transmission of the last data is completed, the clock is stopped and the transmission sequence is terminated.

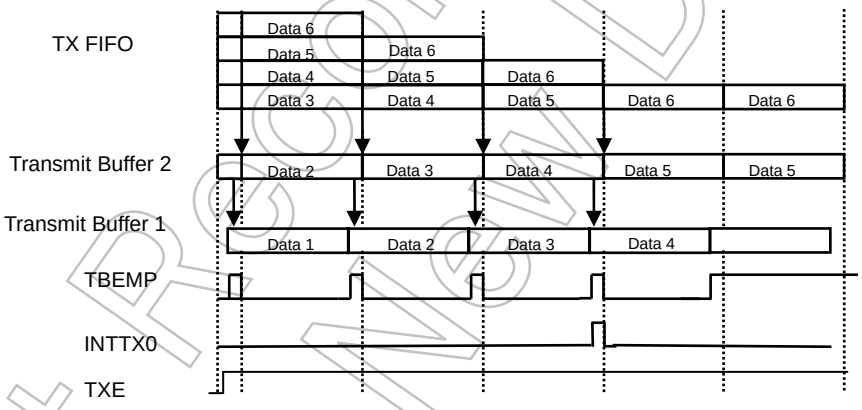


Fig. 14-8 Transmit FIFO Operation

② I/O interface mode with HSCLK input (normal mode):

The following example describes the case a 4-byte data stream is transmitted:

HSC0TFC <1:0> = 01: Clears the transmit FIFO and sets the condition of interrupt generation.

HSC0TFC <7:2> = 000000: Sets the interrupt to be generated at fill level 0.

HSC0FCNF <4:0> = 01001: Allows continued transmission after reaching the fill level.

In this condition, data transmission can be initiated depend on the input clock by setting the transfer mode to half duplex, writing 4 bytes of data to the transmit FIFO, and setting the <TXE> bit to "1." When the last transmit data is moved to the transmit buffer, the transmit FIFO interrupt is generated.

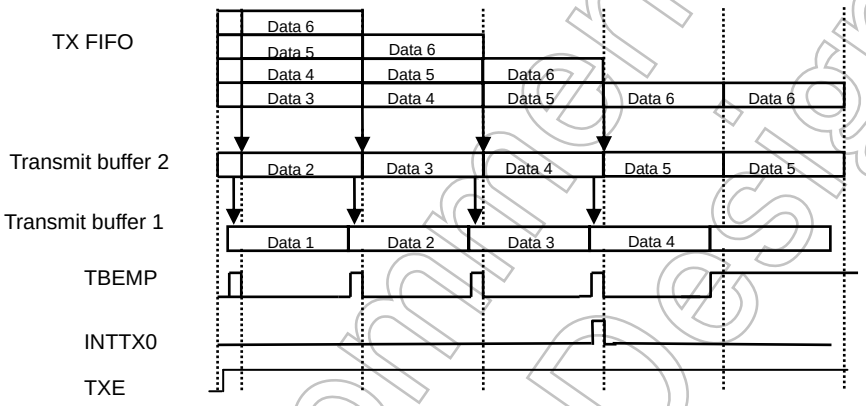


Fig. 14-9 Transmit FIFO Operation

14.1.13 Parity Control Circuit

If the parity addition bit <PE> of the serial control register HSC0CR is set to "1," data is sent with the parity bit. Note that the parity bit may be used only in the 7- or 8-bit UART mode. The <EVEN> bit of HSC0CR selects either even or odd parity.

Upon data transmission, the parity control circuit automatically generates the parity with the data written to the transmit buffer (HSC0BUF). After data transmission is complete, the parity bit will be stored in HSC0BUF bit 7 <TB7> in the 7-bit UART mode and in bit 7 <TB8> in the serial mode control register HSC0MOD in the 8-bit UART mode. The <PE> and <EVEN> settings must be completed before data is written to the transmit buffer.

Upon data reception, the parity bit for the received data is automatically generated while the data is shifted to Receive Buffer 1 and moved to Receive Buffer 2 (HSC0BUF). In the 7-bit UART mode, the parity generated is compared with the parity stored in HSC0BUF <RB7>, while in the 8-bit UART mode, it is compared with the bit 7 <RB8> of the HSC0CR register. If there is any difference, a parity error occurs and the <PERR> flag of the HSC0CR register is set.

In the I/O interface mode, the HSC0CR <PERR> flag functions as an under-run error flag, not as a parity flag.

14.1.14 Error Flag

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

1. Overrun error <OERR>: Bit 4 of the serial control register HSC0CR

   In both UART and I/O interface modes, this bit is set to "1" when an error is generated by completing the reception of the next frame receive data before the receive buffer has been read. If the receive FIFO is enabled, the received data is automatically moved to the receive FIFO and no overrun error will be generated until the receive FIFO is full (or until the usable bytes are fully occupied). This flag is set to "0" when it is read. In the I/O interface HSCLK output mode, no overrun error is generated and therefore, this flag is inoperative and the operation is undefined.

2. Parity error/under-run error <PERR>: Bit 3 of the HSC0CR register

   In the UART mode, this bit is set to "1" when a parity error is generated. A parity error is generated when the parity generated from the received data is different from the parity received. This flag is set to "0" when it is read.

   In the I/O interface mode, this bit indicates an under-run error. When the double buffer control bit <WBUF> of the serial mode control register HSC0MOD2 is set to "1" in the HSCLK input mode, if no data is set to the transmit double buffer before the next data transfer clock after completing the transmission from the transmit shift register, this error flag is set to "1" indicating an under-run error. If the transmit FIFO is enabled, any data content in the transmit FIFO will be moved to the buffer. When the transmit FIFO and the double buffer are both empty, an under-run error will be generated. Because no under-run errors can be generated in the HSCLK output mode, this flag is inoperative and the operation is undefined. If Transmit Buffer 2 is disabled, the under-run flag <PERR> will not be set. This flag is set to "0" when it is read.

3. Framing error <FERR>: Bit 2 of the HSC0CR register:

   In the UART mode, this bit is set to "1" when a framing error is generated. This flag is set to "0" when it is read. A framing error is generated if the corresponding stop bit is determined to be "0" by sampling the bit at around the center. Regardless of the <SBLEN> (stop bit length) setting of the serial mode control register 2, HSC0MOD2, the stop bit status is determined by only 1 bit on the receive side.

Operation mode	Error flag	Function
UART	OERR	Overrun error flag
	PERR	Parity error flag
	FERR	Framing error flag
I/O interface (HSCLK input)	OERR	Overrun error flag
	PERR	Underrun error flag(WBUF＝1)
		Fixed to 0(WBUF＝0)
	FERR	Fixed to 0
I/O interface (HSCLK output)	OERR	Operation undefined
	PERR	Operation undefined
	FERR	Fixed to 0

14.1.15 Direction of Data Transfer

In the I/O interface mode, the direction of data transfer can be switched between "MSB first" and "LSB first" by the data transfer direction setting bit <DRCHG> of the HSC0MOD2 serial mode control register 2. Don't switch the direction when data is being transferred.

14.1.16 Stop Bit Length

In the UART mode transmission, the stop bit length can be set to either 1 or 2 bits by bit 4 <SBLEN> of the HSC0MOD2 register.

14.1.17 Status Flag

If the double buffer function is enabled (HSC0MOD2 <WBUF> = "1"), the bit 6 flag <RBFLL> of the HSC0MOD2 register indicates the condition of receive buffer full. When one frame of data has been received and transferred from buffer 1 to buffer 2, this bit is set to "1" to show that buffer 2 is full (data is stored in buffer 2). When the receive buffer is read by CPU/DMAC, it is cleared to "0." If <WBUF> is set to "0," this bit is insignificant and must not be used as a status flag. When double buffering is enabled (HSC0MOD2 <WBUF> = "1"), the bit 7 flag <TBEMP> of the HSC0MOD2 register indicates that transmit buffer 2 is empty. When data is moved from Transmit Buffer 2 to Transmit Buffer 1 (shift register), this bit is set to "1" indicating that Transmit Buffer 2 is now empty. When data is set to the transmit buffer by CPU/DMAC, the bit is cleared to "0." If <WBUF> is set to "0," this bit is insignificant and must not be used as a status flag.

14.1.18 Configurations of Transmit/Receive Buffers

		WBUF = 0	WBUF = 1
UART	Transmit buffer	Single	Double
	Receive buffer	Double	Double
I/O interface (HSCLK input)	Transmit buffer	Single	Double
	Receive buffer	Double	Double
I/O interface (HSCLK output)	Transmit buffer	Single	Double
	Receive buffer	Single	Double

14.1.19 Software Reset

Software reset is generated by writing bit 1, 0 <SWRST1:0> of HSC0MOD2 register as “10” followed by “01”. As a result, mode registers HSC0MOD0<RXE>, HSC0MOD1<TXE> , HSC0MOD2<TBEMP>,<RBFLL>,<TXRUN>, control registers HSC0CR<OERR>, <PERR>, <FERR> and their internal circuits are initialized. Other conditions are intact.

14.1.20 Signal Generation Timing

① UART Mode:

Receive Side

Mode	9-bit	8-bit with parity	8-bit, 7-bit, and 7-bit with parity
Interrupt generation timing	Around the center of the 1st stop bit	Around the center of the 1st stop bit	Around the center of the 1st stop bit
Framing error generation timing	Around the center of the stop bit	Around the center of the stop bit	Around the center of the stop bit
Parity error generation timing	—	Around the center of the last (parity) bit	Around the center of the last (parity) bit
Overrun error generation timing	Around the center of the stop bit	Around the center of the stop bit	Around the center of the stop bit

Transmit Side

Mode	9-bit	8-bit with parity	8-bit, 7-bit, and 7-bit with parity
Interrupt generation timing (WBUF = 0)	Just before the stop bit is sent	Just before the stop bit is sent	Just before the stop bit is sent
Interrupt generation timing (WBUF = 1)	Immediately after data is moved to Transmit Buffer 1 (just before start bit transmission)	Immediately after data is moved to Transmit Buffer 1 (just before start bit transmission)	Immediately after data is moved to Transmit Buffer 1 (just before start bit transmission)

② I/O interface mode:

Receive Side

Interrupt generation timing (WBUF = 0)	HSCLK output mode	Immediately after the rising edge of the last HSCLK.
	HSCLK input mode	Immediately after the rising or falling edge of the last HSCLK (for rising or falling edge mode, respectively)
Interrupt generation timing (WBUF = 1)	HSCLK output mode	Immediately after the rising edge of the last HSCLK (just after data transfer to Receive Buffer 2) or just after data is read from Receive Buffer 2 .
	HSCLK input mode	Immediately after the rising edge or falling edge of the last HSCLK depending on the rising or falling edge triggering mode, respectively (right after data is moved to Receive Buffer 2)
Overrun error generation timing	HSCLK input mode	Immediately after the rising or falling edge of the last HSCLK (for rising or falling edge mode, respectively)

Transmit Side

Interrupt generation timing (WBUF = 0)	HSCLK output mode	Immediately after the rising edge of the last HSCLK
	HSCLK input mode	Immediately after the rising or falling edge of the last HSCLK (for rising or falling edge mode, respectively)
Interrupt generation timing (WBUF = 1)	HSCLK output mode	Immediately after the rising edge of the last HSCLK or just after data is moved to Transmit Buffer 1
	HSCLK input mode	Immediately after the rising or falling edge of the last HSCLK (for the rising or falling edge mode, respectively) or just after data is moved to Transmit Buffer 1
Undrrun error generation timing	HSCLK input mode	Immediately after the falling or rising edge of the next HSCLK (for the rising or falling edge triggering mode, respectively)

(Note 1) Do not modify any control register when data is being transmitted or received (in a state ready to transmit or receive).
(Note 2) Do not stop the receive operation (by setting HSC0MOD0<RXE>="0") when data is being received.
(Note 3) Do not stop the transmit operation (by setting HSC0MOD1<TXE>="0") when data is being transmitted.

14.2 Register Description (Only for Channel 0)

HSC0MOD0 (0xFFFF_E80E) LITTLE
(0xFFFF_E80D) BIG

	7	6	5	4	3	2	1	0
Bit symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Transmit data bit-8	Handshake function control 0: Disables HCTS 1: Enables HCTS	Receive control 0: Disables reception 1: Enables reception	Wake-up function 0: Disable 1: Enable	Serial transfer mode 00: I/O interface mode 01: 7-bit length UART mode 10: 8-bit length UART mode 11: 9-bit length UART mode		Serial transfer clock (for UART) 00: Timer TB0OUT 01: Baud rate generator 10: Internal f_{SYS} clock 11: External clock (HSCLK0 input)	

Note) In the I/O interface mode, the serial control register (HSC0CR) is used for clock selection

→ Wakeup function

	9-bit UART	Other modes
0	Interrupt when received	don't care
1	Interrupt at RB8 = 1	

→ Handshake function ($\overline{CTS}$ pin) enable

0	Disable(transmission is always allowed)
1	Enable

(Note 1) With <RXE> set to "0," set each mode register (HSC0MOD0, HSC0MOD1 and HSC0MOD2). Then set <RXE> to "1."
(Note 2) The registers must be byte accessed in setting them.
(Note 3) Please specify the mode first and then specify <RXE> bit.

Fig. 14-10 Serial Mode Control Register 0 (for HSIO0, HSC0MOD0)

LITTLE / BIG

HSC0MOD1 (0xFFFF_E805) (0xFFFF_E806)

	7	6	5	4	3	2	1	0
Bit symbol	I2S0	FDPX1	FDPX0	TXE	SINT2	SINT1	SINT0	-
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	0	0	0	0	0	0
Function	IDLE 0: Stop 1: Start	Transfer mode setting 00: Transfer prohibited 01: Half duplex (RX) 10: Half duplex (TX) 11: Full duplex		Transmit control 0: Disable 1: Enable	Interval time of continuous transmission 000: None 100: 8SCLK 001: 1SCLK 101:16SCLK 010: 2SCLK 110: 32SCLK 011: 4SCLK 111: 64SCLK			Write "0."

Fig. 14-11 Serial Mode Control Register 1 (for HSIO0, HSC0MOD1)

<SINT2:0>: Specifies the interval time of continuous transmission when double buffering or/and FIFO is enabled in the I/O interface mode. This parameter is invalid for the UART mode.

<TXE>: This bit enables transmission and is valid for all the transfer modes. If disabled while transmission is in progress, transmission is inhibited only after the current frame of data is completed for transmission.

<FDPX1:0>: Configures the transfer mode in the I/O interface mode. Also configures the FIFO if it is enabled. In the UART mode, it is used only to specify the FIFO configuration.

<I2S0>: Specifies the Idle mode operation.

(Note 1) The registers must be byte accessed in setting them.
(Note 2) Please specify the mode first and then specify <TXE> bit.

HSC0MOD2
LITTLE (0xFFFF_E806)
BIG (0xFFFF_E805)

	7	6	5	4	3	2	1	0
Bit symbol	TBEMP	RBFLL	TXRUN	SBLEN	DRCHG	WBUF	SWRST1	SWRST0
Read/Write	R/W						W	W
After reset	1	0	0	0	0	0	0	0
Function	Transmit buffer empty flag 0: full 1: Empty	Receive buffer full FLAG 0: Empty 1: full	In transmission flag 0: Stop 1: Start	STOP bit 0: 1-bit 1: 2-bit	Setting transfer direction 0: LSB first 1: MSB first	W-buffer 0: Disable 1: Enable	Soft reset Overwrite "01" on "10" to reset	

Fig. 14-12 Serial Mode Control Register

<SWRST1:0>: Overwriting "01" in place of "10" generates a software reset. When this software reset is executed, the mode register parameters HSC0MOD0<RXE>, HSCMOD1<TXE>, HSC0MOD2<TBEMP>,<RBFLL>,<TXRUN>, control register parameters HSC0CR<OERR>, <PERR>, <FERR>, and their internal circuits are initialized.

<WBUF>: This parameter enables or disables the transmit/receive buffers to transmit (in both HSCLK output/input modes) and receive (in HSCLK output mode) data in the I/O interface mode and to transmit data in the UART. In all other modes, double buffering is enabled regardless of the <WBUF> setting.

<DRCHG>: Specifies the direction of data transfer in the I/O interface mode. In the UART mode, it is fixed to LSB first.

<TXRUN>: This is a status flag to show that data transmission is in progress.
When this bit is set to "1," it indicates that data transmission operation is in progress. If it is "0," the bit 7 <TBEMP> is set to "1" to indicate that the transmission has been fully completed and the same <TBEMP> is set to "0" to indicate that the transmit buffer contains some data waiting for the next transmission.

<RBFLL>: This is a flag to show that the receive double buffers are full. When a receive operation is completed and received data is moved from the receive shift register to the receive double buffers, this bit changes to "1" while reading this bit changes it to "0."
If double buffering is disabled, this flag is insignificant.

<TBEMP>: This flag shows that the transmit double buffers are empty. When data in the transmit double buffers is moved to the transmit shift register and the double buffers are empty, this bit is set to "1." Writing data again to the double buffers sets this bit to "0."
If double buffering is disabled, this flag is insignificant.

<SBLEN>: This specifies the length of stop bit transmission in the UART mode. On the receive side, the decision is made using only a single bit regardless of the <SBLEN> setting.

(Note 1) While data transmission is in progress, any software reset operation must be executed twice in succession.

(Note 2) The registers must be byte accessed in setting them.

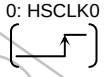
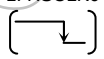
(Note 3) A software reset requires 1 clock at the time between the end of recognition and the start of execution of software reset instruction. Insert the SYNC and NOP instructions after the software reset instruction to secure the required time.

(Note 4) A software reset initializes other bits. Resetting a mode register and a control register are needed.

HSC0CR

LITTLE (0xFFFF_E80D)

BIG (0xFFFF_E80E)

	7	6	5	4	3	2	1	0
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	0	0	0	0	0	0	0	0
Function	Receive data bit-8	Parity 0: Odd 1: Even	Add parity 0: Disable 1: Enable	1: Error			0: HSCLK0  1: HSCLK0 	0: Baud rate generator 1: HSCLK0 pin input
				Overrun	Parity/ Under run	Framing		

I/O interface input clock selection

0	Baud rate generator
1	HSCLK0 pin input

HSCLK0 for HSCLK0 input operation

0	Data transmit/receive at rising edges of HSCLK0 
1	Data transmit/receive at falling edges of HSCLK0 

Framing error flag

Parity error/under-run error flag

Overrun error flag

Cleared to "0" when read

Add/check even parity

0	Odd parity
1	Even parity

Fig. 14-13 Serial Control Register (for HSIO0, HSC0CR)

(Note 1) All the error flags are cleared when read.

(Note 2) The registers must be byte accessed in setting them.

HBR0CR
LITTLE (0xFFFF_E80F)
BIG (0xFFFF_E80C)

	7	6	5	4	3	2	1	0
Bit symbol	-	HBR0ADDE	HBR0S5	HBR0S4	HBR0S3	HBR0S2	HBR0S1	HBR0S0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Write "0."	N+(16-K)/16 divider function 0: Disable 1: Enable	Divide ratio "N"					

(Note) The registers must be byte accessed in setting them.

HBR0ADD
LITTLE (0xFFFF_E804)
BIG (0xFFFF_E807)

	7	6	5	4	3	2	1	0
Bit symbol					HBR0K3	HBR0K2	HBR0K1	HBR0K0
Read/Write	R				R/W			
After reset	0				0	0	0	0
Function	Always reads "0."				Specify K for the "N + (16 - K)/16" division			

Setting divide ratio of the baud rate generator

	HBR0CR<HBR0ADDE> = 1		HBR0CR<HBR0ADDE> = 0
HBR0CR <HBR0S5:0> / HBR0DD <HR03:0>	000000(N = 64) ~ 000001(N = 1)	000010(N = 2) ~ 111111(N = 63)	000001(N = 1) (ONLY UART) ~ 111111 (N = 63) 000000 (N = 64)
0000	Disable	Disable	
0001(K = 1) ~ 1111(K = 15)	Disable	$N + \frac{(16 - K)}{16}$ Division	Divide by N

Fig. 14-14 Baud Rate Generator Control (for HSIO0, HBR0CR, HBR0ADD)

(Note 1) In the UART mode, the division ratio "1" of the baud rate generator can be specified only when the "N + (16 - K)/16" division function is not used. In the I/O interface mode, "divide by 1" must not be specified as a divisor for the baud rate generator.

(Note 2) To use the "N + (16 - K)/16" division function, be sure to set HBR0CR <HBR0ADDE> to "1" after setting the K value (K = 1 to 15) to HBR0ADD <BR3K3:0>. However, don't use the "N + (16 - K)/16" division function when HBR0CR <BR0S5:0> is set to either "000000" or "000001" (N = 64 or 1).

(Note 3) The "N + (16 - K)/16" division function can only be used in the UART mode. In the I/O interface mode, the "N + (16 - K)/16" division function must be disabled (prohibited) by setting HBR0CR <HBR0ADDE> to "0."

HSC0BUF

LITTLE (0xFFFF_E800)

BIG (0xFFFF_E803)

7	6	5	4	3	2	1	0	
TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(Transmit BUFFER +FIFO)
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(Transmit BUFFER +FIFO)

Note: HSC0BUF works as a transmit buffer for WR operation and as a receive buffer for RD operation.

HSC0FCNF

LITTLE (0xFFFF_E80C)

BIG (0xFFFF_E80F)

	7	6	5	4	3	2	1	0
Bit symbol	-	-	-	RFST	TFIE	RFIE	RXTXCNT	CNFG
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Be sure to write "000."			Bytes used in RX FIFO 0: Maximum 1: Same as Fill level of RX FIFO	TX interrupt for TX FIFO 0: Disable 1: Enable	RX interrupt for RX FIFO 0: Disable 1: Enable	Automatic disable of RXE/TXE 0: None 1: Auto disable	FIFO Enable 0: Disable 1: Enable

<CNFG>: If enabled, the HSCOMOD1 <FDPX1:0> setting automatically configures FIFO as follows:

<FDPX1:0>=01 (Half duplex RX) ---- 4-byte RX FIFO

<FDPX1:0>=10 (Half duplex TX) ---- 4-byte TX FIFO

<FDPX1:0>=11 (Full duplex) ---- 2-Byte RX FIFO + 2-Byte TX FIFO

<RXTXCNT>:0 The function to automatically disable RXE/TXE bits is disabled.

1: If enabled, the HSCOMOD1 <FDPX1:0> is used to set as follows:

<FDPX1:0> = 01 (Half duplex RX) ------When the RX FIFO is filled up with the specified number of valid bytes, RXE is automatically set to "0" to inhibit further reception.

<FDPX1:0> = 10 (Half duplex TX) ------When the TX FIFO is empty, TXE is automatically set to "0" to inhibit further transmission.

<FDPX1:0> = 11 (Full duplex) ---------- When either of the above two conditions is satisfied, TXE/RXE are automatically set to "0" to inhibit further transmission and reception.

<RFIE>: When RX FIFO is enabled, receive interrupts are enabled or disabled by this parameter.

<TFIE>: When TX FIFO is enabled, transmit interrupts are enabled or disabled by this parameter.

<RFST>: When RX FIFO is enabled, the number of RX FIFO bytes to be used is selected.

0: The maximum number of bytes of the FIFO configured 4 bytes when <FDPX1:0> = 01 (Half duplex RX) and 2 bytes for <FDPX1:0> = 11 (Full duplex

1: Same as the fill level for receive interrupt generation specified by HSC0RFC <RIL1:0>

(Note 1) Regarding TX FIFO, the maximum number of bytes being configured is always available. The available number of bytes is the bytes already written to the TX FIFO.

(Note 2) The registers must be byte accessed in setting them.

Fig. 14-15 FIFO Configuration Register

HSC0RFC
LITTLE (0xFFFF_E808)
BIG (0xFFFF_E80B)

	7	6	5	4	3	2	1	0
Bit symbol	RFCS	RFIS	-	-	-	-	RIL1	RIL0
Read/Write	w	R					W/R	
After reset	0	0	0	0	0	0	0	0
Function	Clear RX FIFO 1: Clear Always reads "0."	Select interrupt generation condition	Always reads "0."				FIFO fill level to generate RX interrupts 00: 4 bytes (2 bytes at full duplex mode) 01: 1byte 10: 2bytes 11: 3bytes Note: RIL1 is ignored when FDPX1:0 = 11 (full duplex)	

0: An interrupt is generated when it reaches to the specified fill level.
1: An interrupt is generated when it is reaches to the specified fill level or if it exceeds the specified fill level at the time data is read.

Fig. 14-16 Transmit FIFO Configuration Register

HSC0TFC
LITTLE (0xFFFF_E809)
BIG (0xFFFF_E80A)

	7	6	5	4	3	2	1	0
Bit symbol	TFCS	TFIS	-	-	-	-	TIL1	TIL0
Read/Write	w	R					W/R	
After reset	0	0	0	0	0	0	0	0
Function	Clear TX FIFO 1: Clear Always reads "0."	Select interrupt generation condition	Always reads "0."				FIFO fill level to generate TX interrupts 00: Empty 01: 1byte 10: 2byte 11: 3byte Note: TIL1 is ignored when FDPX1:0 = 11 (full duplex).	

0: An interrupt is generated when it reaches to the specified fill level.
1: An interrupt is generated when it reaches to the specified fill level or if it is lower than the specified fill level at the time new data is written.

Fig. 14-17 Transmit FIFO Control Register

(Note) The registers must be byte accessed in setting them.

HSC0RST
LITTLE (0xFFFF_E80A)
BIG (0xFFFF_E809)

	7	6	5	4	3	2	1	0
Bit symbol	ROR	-	-	-	-	RLVL2	RLVL1	RLVL0
Read/Write	R	R						
After reset	0	0	0	0	0	0	0	0
Function	RX FIFO Overrun 1: Generated Cleared when read	Always reads "0."				Status of RX FIFO fill level 000: Empty 001: 1Byte 010: 2Bytes 011: 3Bytes 100: 4Bytes		

Fig. 14-18 Receive FIFO Status Register

HSC0TST
LITTLE (0xFFFF_E80B)
BIG (0xFFFF_E808)

	7	6	5	4	3	2	1	0
Bit symbol	TUR	-	-	-	-	TLVL2	TLVL1	TLVL0
Read/Write	R	R						
After reset	1	0	0	0	0	0	0	0
Function	TX FIFO Under run 1: Generated Cleared by writing to FIFO	Always reads "0."				Status of TX FIFO fill level 000: Empty 001: 1Byte 010: 2Bytes 011: 3Byte s 100: 4Bytes		

Fig. 14-19 Transmit FIFO Status Register

HSC0EN
LITTLE (0xFFFF_E807)
BIG (0xFFFF_E804)

	7	6	5	4	3	2	1	0
Bit symbol	-	-	-	-	-	-	-	SIOE
Read/Write	R							R/W
After reset	0	0	0	0	0	0	0	0
Function	Always reads "0."							HSIO operation 0: Disable 1: Enable

<SIOE>: It specifies HSIO operation. When HSIO operation is disabled, the clock will not be supplied to the HSIO module except for the register part and thus power dissipation can be reduced (other registers cannot be accessed for read/write operation). When HSIO is to be used, be sure to enable HSIO by setting "1" to this register before setting any other registers of the HSIO module. If HSIO is enabled once and then disabled, all the register settings are maintained.

(Note) The registers must be byte accessed in setting them.

Fig. 14-20 HSIO Enable Register

14.3 Operation in Each Mode

14.3.1 Mode 0 (I/O Interface Mode)

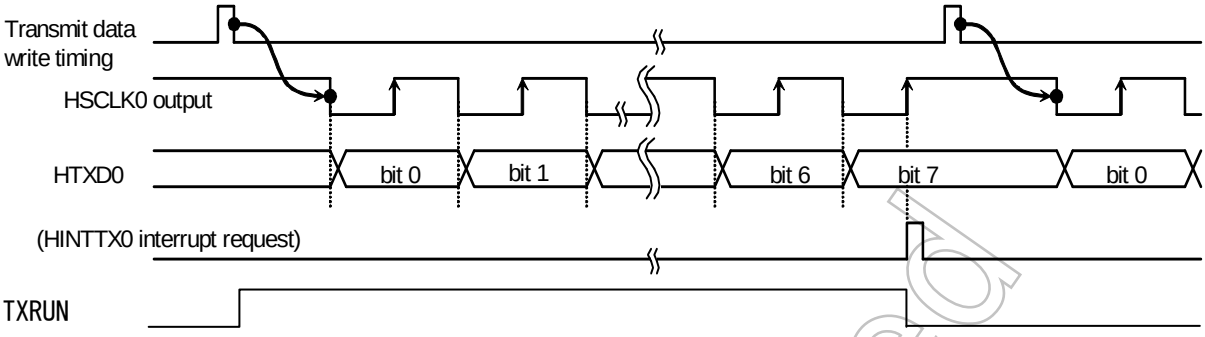
Mode 0 consists of two modes, i.e., the "HSCLK output" mode to output synchronous clock and the "HSCLK input" mode to accept synchronous clock from an external source. The following operational descriptions are for the case use of FIFO is disabled. For details of FIFO operation, refer to the previous sections describing receive/transmit FIFO functions.

① Transmitting data

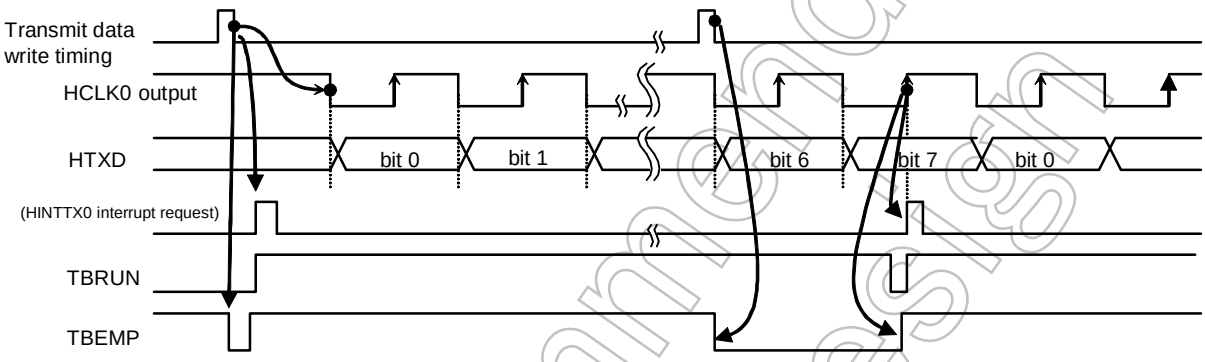
HSCLK output mode

In the HSCLK output mode, if HSC0MOD2<WBUF> is set to "0" and the transmit double buffers are disabled, 8 bits of data are output from the HXD0 pin and the synchronous clock is output from the HSCLK0 pin each time the CPU writes data to the transmit buffer. When all data is output, the HINTTX0 interrupt is generated.

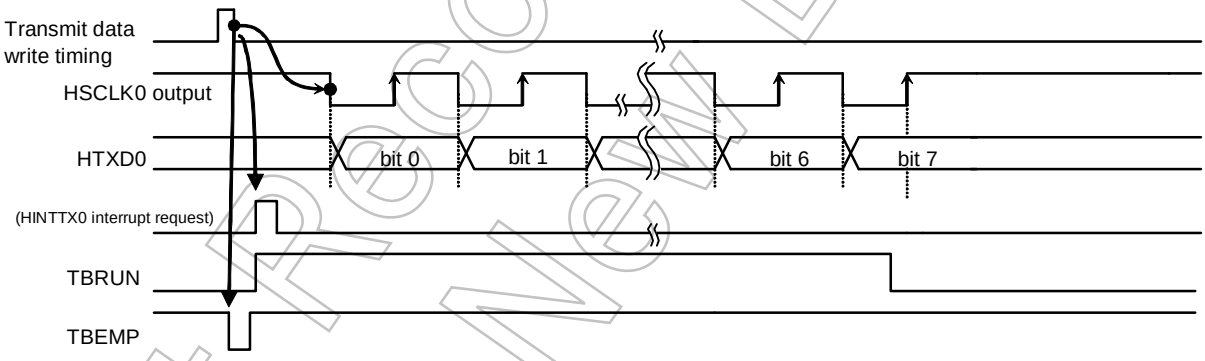
If HSC0MOD2 <WBUF> is set to "1" and the transmit double buffers are enabled, data is moved from Transmit Buffer 2 to Transmit Buffer 1 when the CPU writes data to Transmit Buffer 2 while data transmission is halted or when data transmission from Transmit Buffer 1 (shift register) is completed. When data is moved from Transmit Buffer 2 to Transmit Buffer 1, the transmit buffer empty flag HSC0MOD2 <TBEMP> is set to "1," and the HINTTX0 interrupt is generated. If Transmit Buffer 2 has no data to be moved to Transmit Buffer 1, the HINTTX0 interrupt is not generated and the HSCLK0 output stops.



<WBUF>="0"(if double buffering is disabled)



<WBUF>="1"(if double buffering is enabled and there is data in buffer 2)



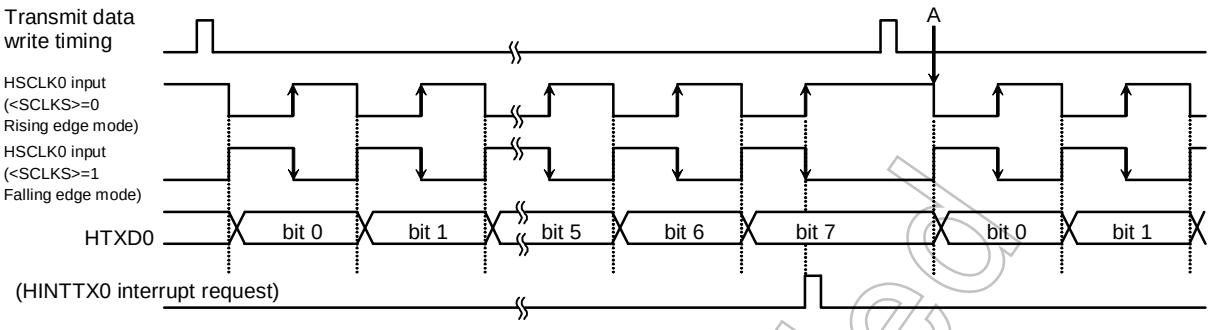
<WBUF>="1"(if double buffering is enabled and there is no data in buffer 2)

Fig. 14-21 Transmit Operation in the I/O Interface Mode (HSCLK0 Output Mode)

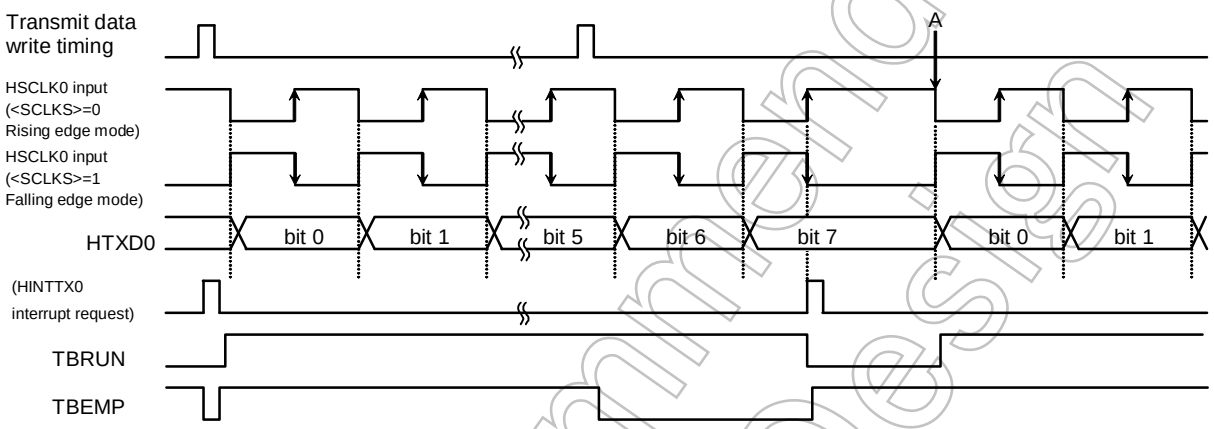
HSCLK input mode

In the HSCLK input mode, if HSC0MOD2 <WBUF> is set to "0" and the transmit double buffers are disabled, 8-bit data that has been written in the transmit buffer is output from the HTXD0 pin when the HSCLK0 input becomes active. When all 8 bits are sent, the HINTTX0 interrupt is generated. The next transmit data must be written before the timing point "A".

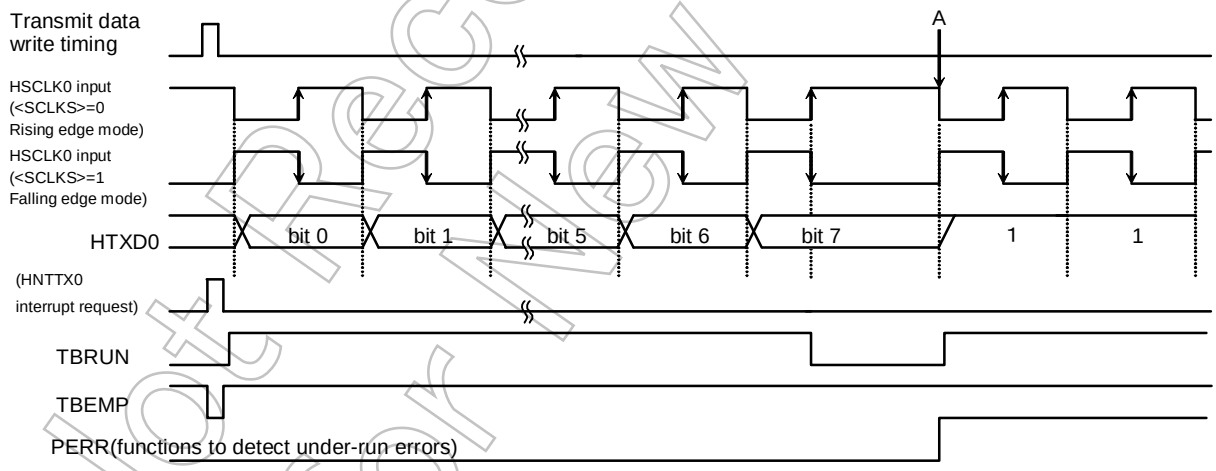
If HSC0MOD2 <WBUF> is set to "1" and the transmit double buffers are enabled, data is moved from Transmit Buffer 2 to Transmit Buffer 1 when the CPU writes data to Transmit Buffer 2 before the HSCLK0 becomes active or when data transmission from Transmit Buffer 1 (shift register) is completed. As data is moved from Transmit Buffer 2 to Transmit Buffer 1, the Transmit Buffer empty flag HSC0MOD2 <TBEMP> is set to "1" and the HINTTX0 interrupt is generated. If the HSCLK0 input becomes active while no data is in Transmit Buffer 2, the internal bit counter is started; however, an under-run error occurs and 8-bit dummy data (FFh) is sent.



<WBUF>="0"(if double buffering is disabled)



<WBUF>="1"(if double buffering is enabled and there is data in buffer 2)



<WBUF>="1"(if double buffering is enabled and there is no data in buffer 2)

Fig. 14-22 Transmit Operation in the I/O Interface Mode (HSCLK0 Input Mode)

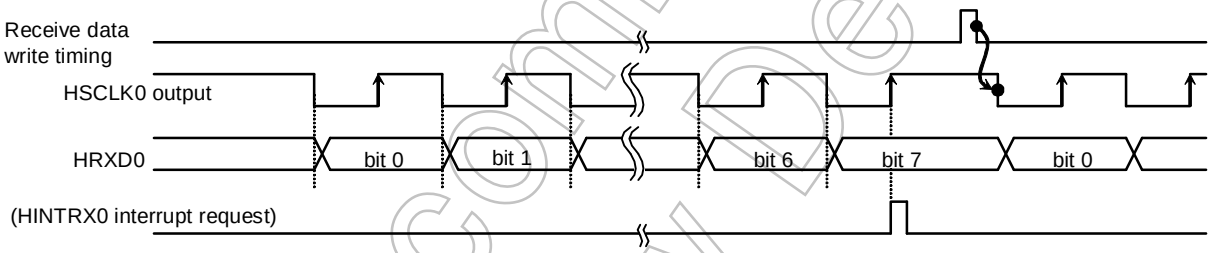
② Receiving data

HSCLK output mode

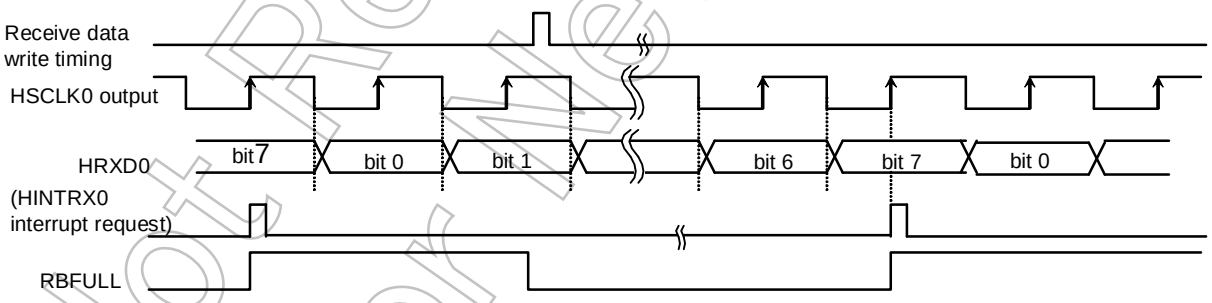
In the HSCLK output mode, if HSC0MOD2 <WBUF> = "0" and receive double buffering is disabled, a synchronous clock pulse is output from the HSCLK0 pin and the next data is shifted into Receive Buffer 1 each time the CPU reads received data. When all the 8 bits are received, the HINTRX0 interrupt is generated.

The first HSCLK output can be started by setting the receive enable bit HSC0MOD0 <RXE> to "1." If the receive double buffering is enabled with HSC0MOD2 <WBUF> set to "1," the first frame received is moved to Receive Buffer 2 and Receive Buffer 1 can receive the next frame successively. As data is moved from Receive Buffer 1 to Receive Buffer 2, the receive buffer full flag HSC0MOD2 <RBFULL> is set to "1" and the HINTRX0 interrupt is generated.

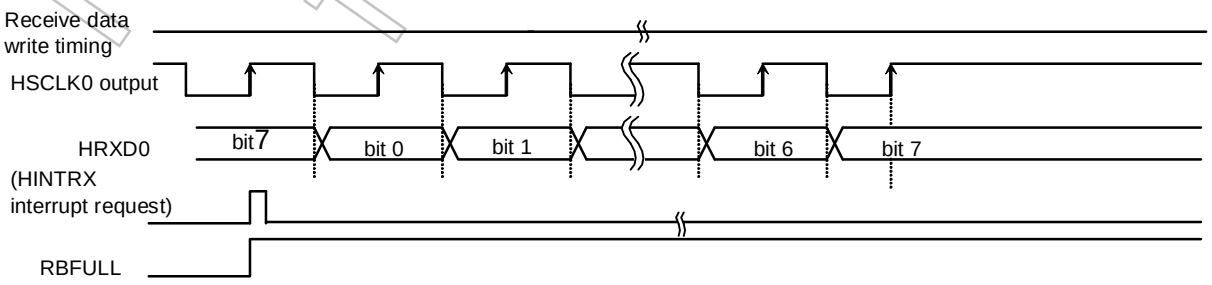
While data is in Receive Buffer 2, if CPU/DMAC cannot read data from Receive Buffer 2 in time before completing reception of the next 8 bits, the HINTRX0 interrupt is not generated and the HSCLK0 clock stops. In this state, reading data from Receive Buffer 2 allows data in Receive Buffer 1 to move to Receive Buffer 2 and thus the HINTRX0 interrupt is generated and data reception resumes.



<WBUF>="0"(if double buffering is disabled)



<WBUF>="1"(if double buffering is enabled and data is read from buffer 2)



<WBUF>="1"(if double buffering is enabled and data cannot be read from buffer 2)

Fig. 14-23 Receive Operation in the I/O Interface Mode (HSCLK0 Output Mode)

HSCLK input mode

In the HSCLK input mode, since receive double buffering is always enabled, the received frame can be moved to Receive Buffer 2 and Receive Buffer 1 can receive the next frame successively.

The HINTRX0 receive interrupt is generated each time received data is moved to Received Buffer 2.

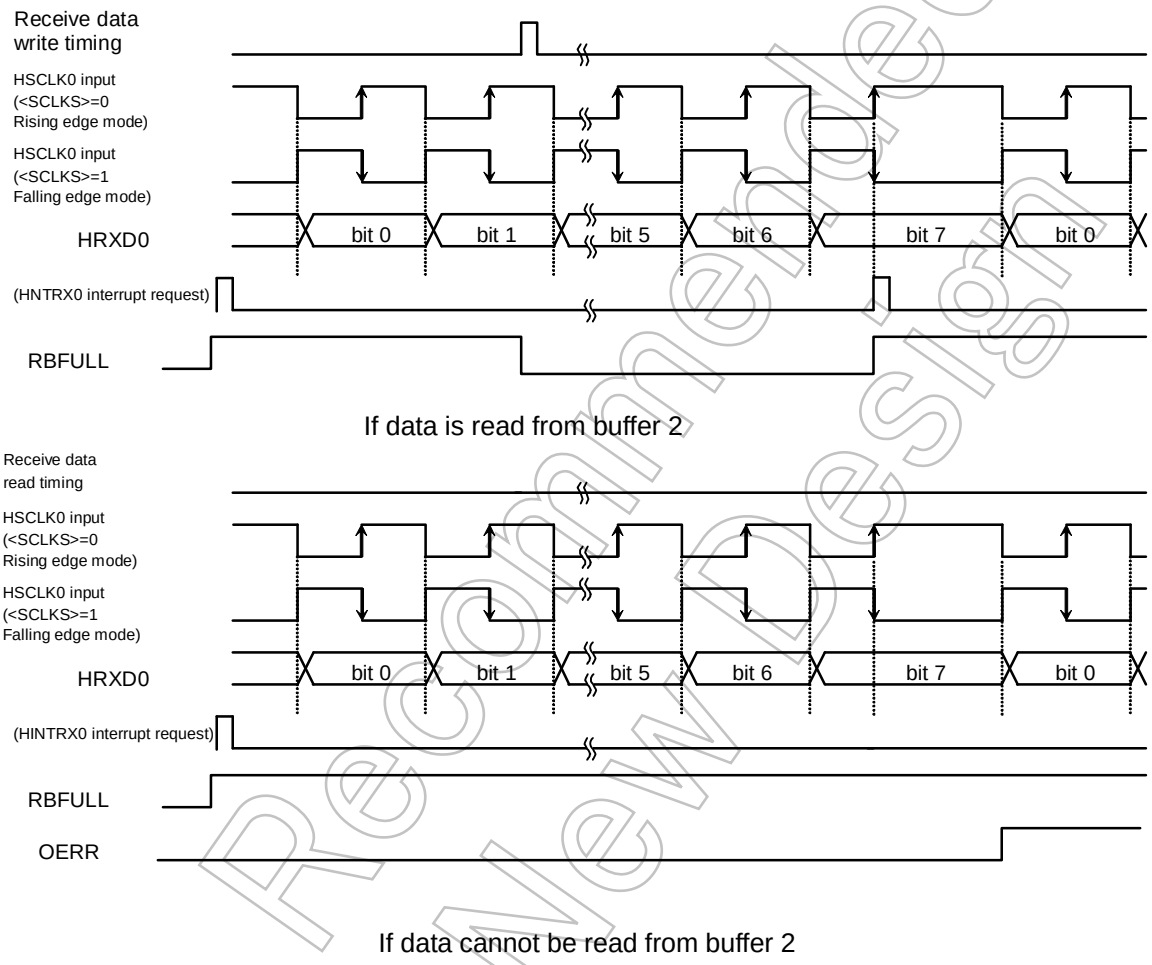


Fig. 14-24 Receive Operation in the I/O Interface Mode (HSCLK0 Input Mode)

(Note) To receive data, HSC0MOD <RXE> must always be set to "1" (receive enable) regardless of the HSCLK input or output mode.

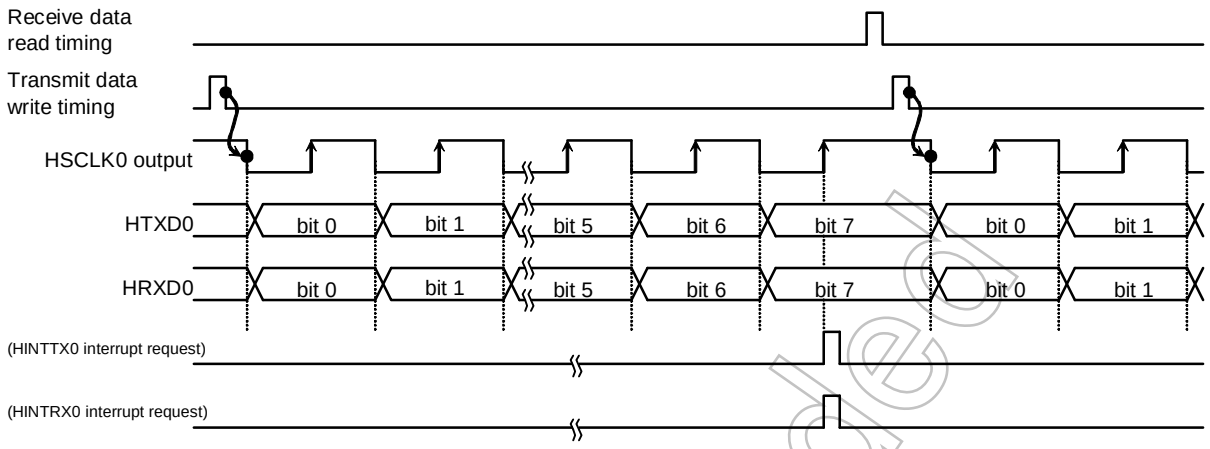
③ Transmit and receive (full-duplex)

The full-duplex mode is enabled by setting bit 6 <FDPX0> of the serial mode control register 1 (HSC0MOD1) to "1".

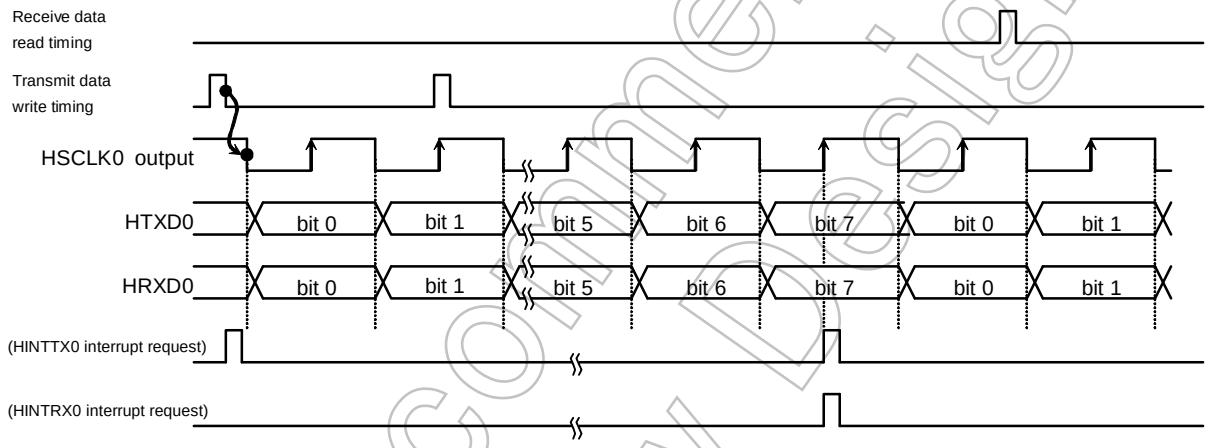
HSCLK output mode

In the HSCLK output mode, if HSC0MOD2 <WBUF> is set to "0" and both the transmit and receive double buffers are disabled, HSCLK is output when the CPU writes data to the transmit buffer. Subsequently, 8 bits of data are shifted into Receive Buffer 1 and the HINTRX0 receive interrupt is generated. Concurrently, 8 bits of data written to the transmit buffer are output from the HTXD0 pin, the HINTTX0 transmit interrupt is generated when transmission of all data bits has been completed. Then, the HSCLK output stops. In this, the next round of data transmission and reception starts when the data is read from the receive buffer and the next transmit data is written to the transmit buffer by the CPU. The order of reading the receive buffer and writing to the transmit buffer can be freely determined. Data transmission is resumed only when both conditions are satisfied.

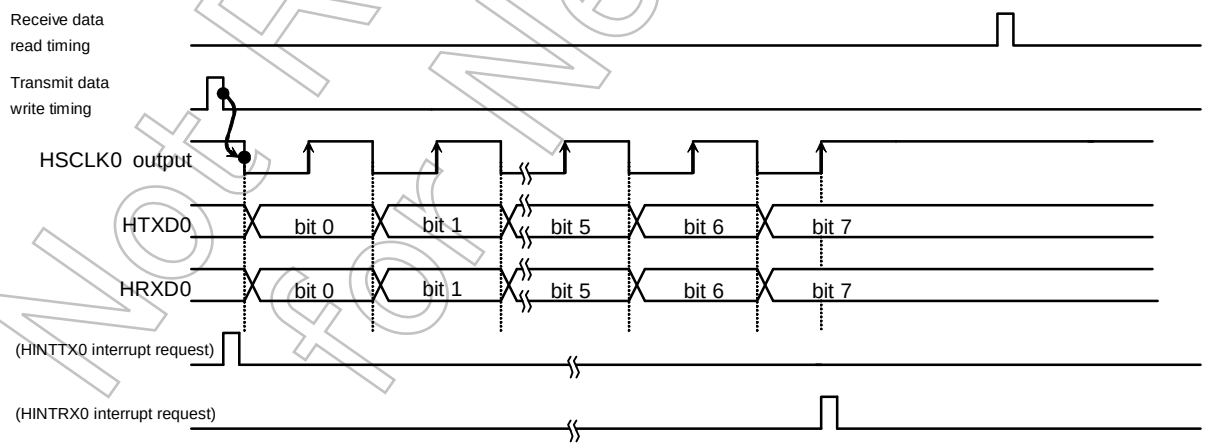
If HSC0MOD2 <WBUF> = "1" and double buffering is enabled for both transmission and reception, HSCLK is output when the CPU writes data to the transmit buffer. Subsequently, 8 bits of data are shifted into Receive Buffer 1, moved to Receive Buffer 2, and the HINTRX0 interrupt is generated. While 8 bits of data is received, 8 bits of transmit data is output from the HTXD0 pin. When all data bits are sent out, the HINTTX0 interrupt is generated and the next data is moved from the Transmit Buffer 2 to Transmit Buffer 1. If Transmit Buffer 2 has no data to be moved to Transmit Buffer 1 (HSC0MOD2 <TBEMP> = 1) or when Receive Buffer 2 is full (HSC0MOD2 <RBFULL> = 1), the HSCLK output is stopped. When both conditions are satisfied, i.e., receive data is read and transmit data is written, the HSCLK output is resumed and the next round of data transmission is started.



<WBUF>="0"(if double buffering is disabled)



<WBUF>="1"(if double buffering is enabled)



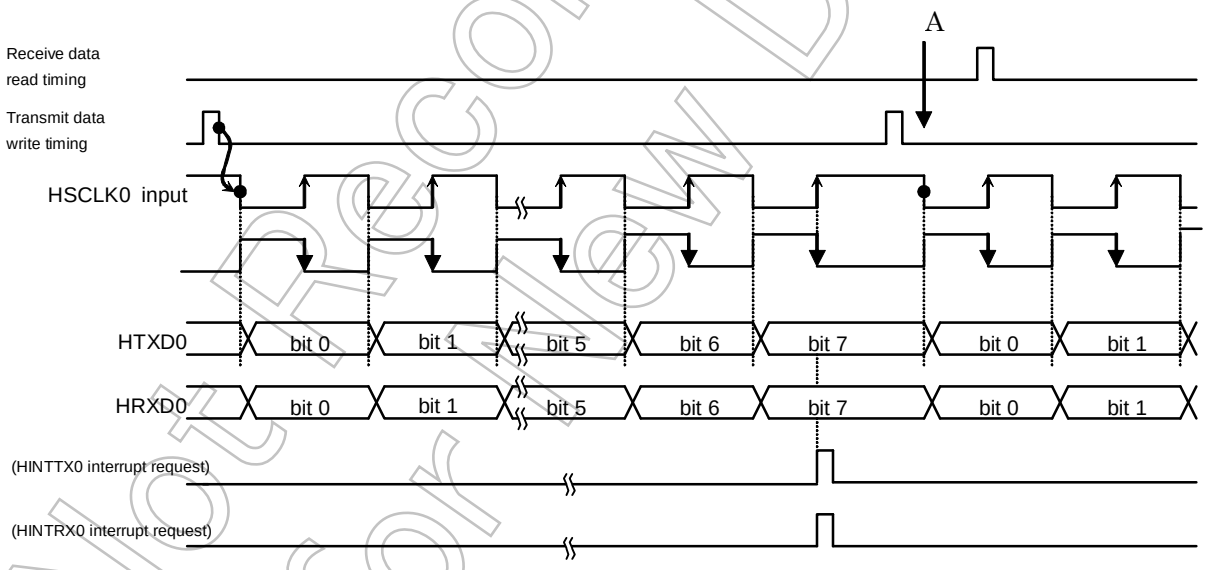
<WBUF>="1"(if double buffering is enabled)

Fig. 14-25 Transmit/Receive Operation in the I/O Interface Mode (HSCLK0 Output Mode)

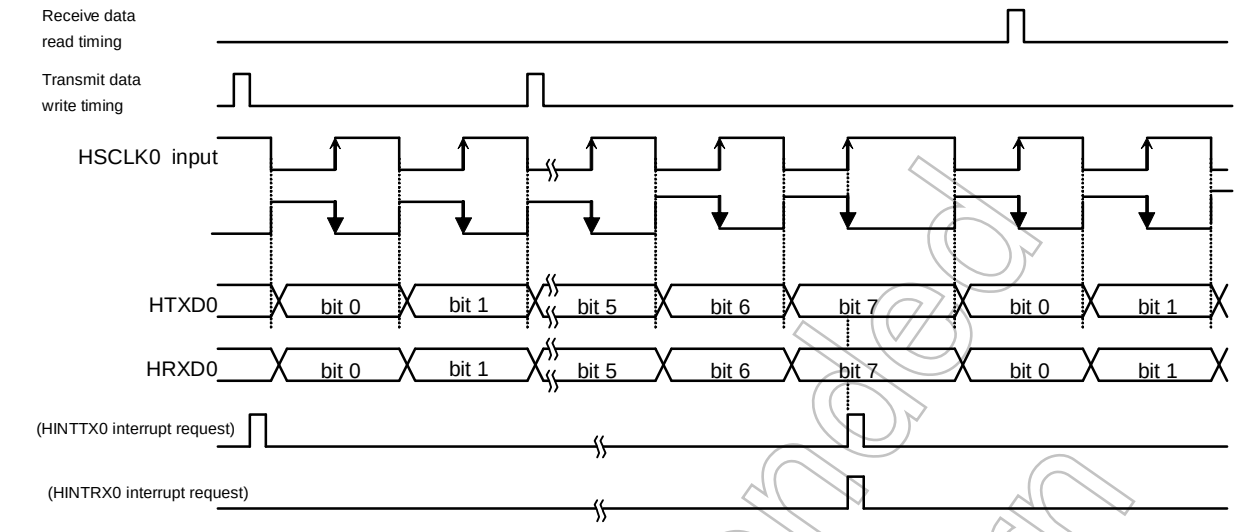
HSCLK input mode

In the HSCLK input mode with HSC0MOD2 <WBUF> set to "0" and the transmit double buffers are disabled (double buffering is always enabled for the receive side), 8-bit data written in the transmit buffer is output from the HTXD0 pin and 8 bits of data is shifted into the receive buffer when the HSCLK input becomes active. The HINTTX0 interrupt is generated upon completion of data transmission and the HINTRX0 interrupt is generated at the instant the received data is moved from Receive Buffer 1 to Receive Buffer 2. Note that transmit data must be written into the transmit buffer before the HSCLK input for the next frame (data must be written before the point A). As double buffering is enabled for data reception, data must be read before the completion of the next frame data reception.

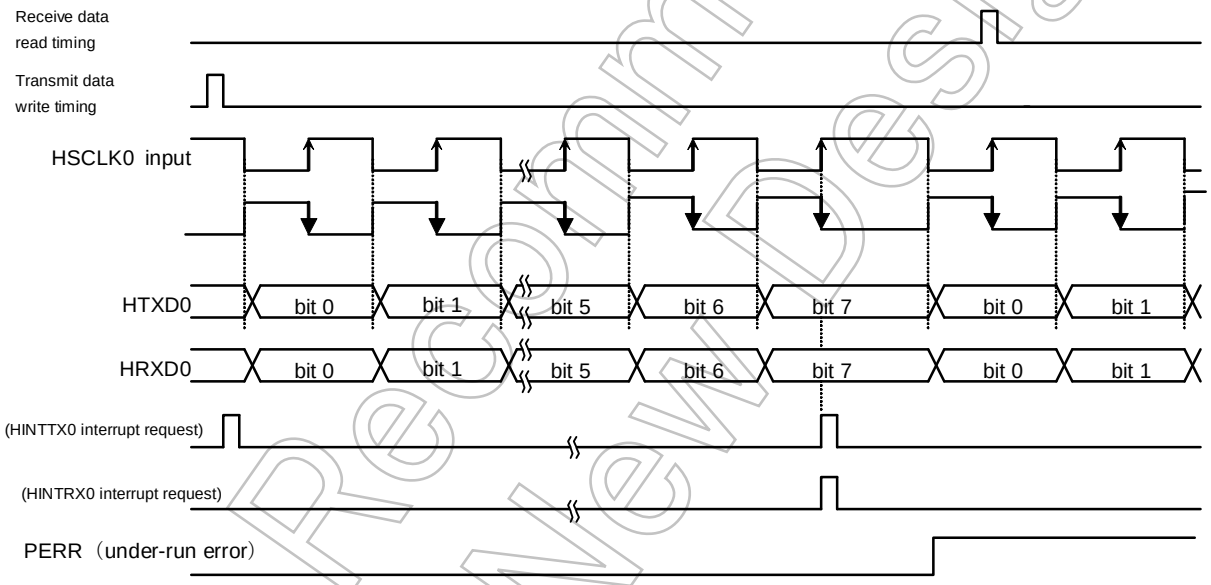
If HSC0MOD2 <WBUF> = "1" and double buffering is enabled for both transmission and reception, the interrupt HINTRX0 is generated at the timing Transmit Buffer 2 data is moved to Transmit Buffer 1 after completing data transmission from Transmit Buffer 1. At the same time, the 8 bits of data received is shifted to Receive Buffer 1, moved to Receive Buffer 2, and the HINTRX0 interrupt is generated. Upon the HSCLK input for the next frame, transmission from Transmit Buffer 1 (in which data has been moved from Transmit Buffer 2) is started while receive data is shifted into Receive Buffer 1 simultaneously. If data in Receive Buffer 2 has not been read when the last bit of the frame is received, an overrun error occurs. Similarly, if there is no data written to Transmit Buffer 2 when HSCLK for the next frame is input, an under-run error occurs.



<WBUF>="0"(if double buffering is disabled)



<WBUF>＝”1”(if double buffering is enabled with no errors)



<WBUF>＝”1”(if double buffering is enabled with error generation)

Fig. 14-26 Transmit/Receive Operation in the I/O Interface Mode (HSCLK0 Input Mode)

14.3.2 Mode 1 (7-bit UART Mode)

The 7-bit UART mode can be selected by setting the serial mode control register (HSC0MOD <SM1, 0>) to "01."

In this mode, parity bits can be added to the transmit data stream; the serial mode control register (HSC0CR <PE>) controls the parity enable/disable setting. When <PE> is set to "1" (enable), either even or odd parity may be selected using the HSC0CR <EVEN> bit. The length of the stop bit can be specified using HSC0MOD2<SBLEN>.

14.3.3 Mode 2 (8-bit UART Mode)

The 8-bit UART mode can be selected by setting HSC0MOD0 <SM1:0> to "10." In this mode, parity bits can be added and parity enable/disable is controlled using HSC0CR <PE>. If <PE> = "1" (enabled), either even or odd parity can be selected using HSC0CR <EVEN>.

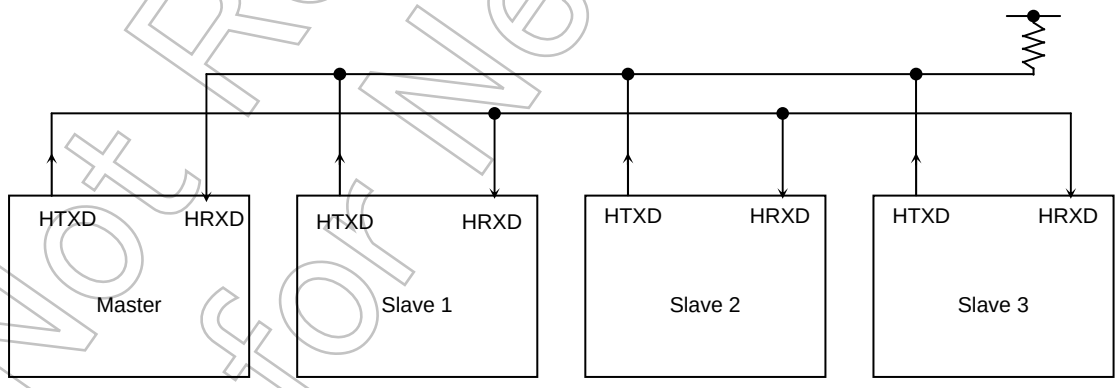
14.3.4 Mode 3 (9-bit UART)

The 9-bit UART mode can be selected by setting HSC0MOD0 <SM1:0> to "11." In this mode, parity bits must be disabled (HSC0CR <PE> = "0").

The most significant bit (9th bit) is written to bit 7 <TB8> of the serial mode control register 0 (HSC0MOD0) to transmit data and it is stored in bit 7 <RB8> of the serial control register HSC0CR upon receiving data. When writing or reading data to/from the buffers, the most significant bit must be written or read first before writing or reading to/from HSC0BUF. The stop bit length can be specified using HSC0MOD2 <SBLEN>.

Wakeup function

In the 9-bit UART mode, slave controllers can be operated in the wake-up mode by setting the wake-up function control bit HSC0MOD0 <WU> to "1." In this case, the interrupt HINTRX0 will be generated only when HSC0CR <RB8> is set to "1.



(Note) The HTXD pin of the slave controller must be set to the open drain output mode using the ODE register.

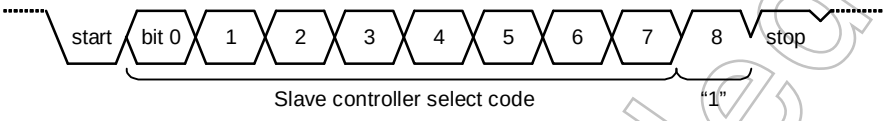
Fig. 14-27 Serial Links to Use Wake-up Function

Protocol

① Select the 9-bit UART mode for the master and slave controllers.

② Set HSC0MOD <WU> to "1" for the slave controllers to make them ready to receive data.

③ The master controller transmits a single frame of data that includes the slave controller select code (8 bits). In this, the most significant bit (bit 8) <TB8> must be set to "1".



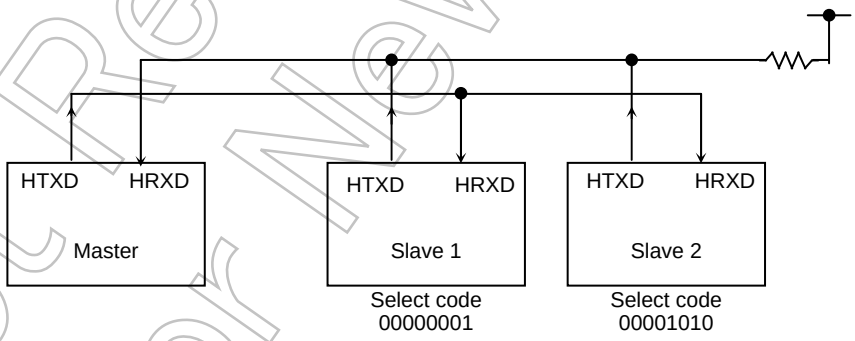
④ Every slave controller receives the above data frame; if the code received matches with the controller's own select code, it clears the WU bit to "0.

⑤ The master controller transmits data to the designated slave controller (the controller of which HSC0MOD <WU> bit is cleared to "0"). In this, the most significant bit (bit 8) <TB8> must be set to "0".

⑥ The slave controllers with the <WU> bit set to "1" ignore the receive data because the most significant bit (bit 8) <RB8> is set to "0" and thus no interrupt (HINTRX0) is generated.

Also, the slave controller with the <WU> bit set to "0" can transmit data to the master controller to inform that the data has been successfully received.

Example setting: Using the internal clock f_{SYS} as the transfer clock, two slave controllers are serially linked as follows:



15. Serial Bus Interface (SBI)

The TMP19A23 contains two Serial Bus Interface (SBI) channels, in which the following two operating modes are included:

- I^2C bus mode (with multi-master capability)
- Clock-synchronous 8-bit SIO mode

In the I^2C bus mode, the SBI is connected to external devices via PA0 (SDA0) and PA1 (SCL0) on channel 0 and via PA4 (SDA1) and PA5 (SCL1) on channel 1. In the clock-synchronous 8-bit SIO mode, the SBI is connected to external devices via PA2 (SCK0), PA0 (SO0), PA1 (SI0) on channel 0 and PA6 (SCLK1), PA4 (SO1), PA5 (SI1) on channel 1.

The following table shows the programming required to put the SBI in each operating mode.

	PCODE <PCODE1:0>	PCCR <PC7C, PC6C, PC5C>	PCFC <PC7F, PC6F, PC5F>
I^2C bus mode	11	X11	011
clock-synchronous 8-bit SIO mode	XX	101(clock output) 001(clock input)	111

X: Don't care

15.1 Configuration

The configuration is shown in Fig. 15.1.

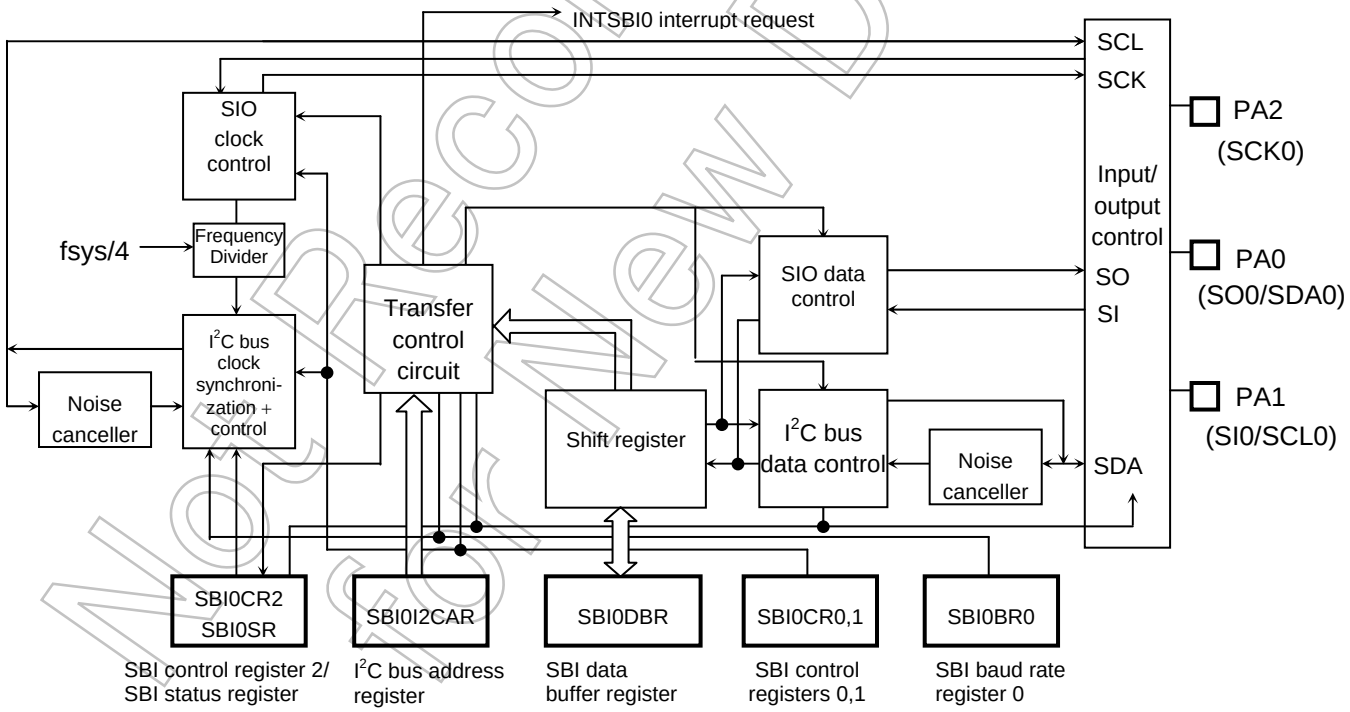


Fig. 15-1 SBI0 Block Diagram (SBI1has the same construction except for the pin numbers)

15.2 Control

The following registers control the serial bus interface and provide its status information for monitoring.

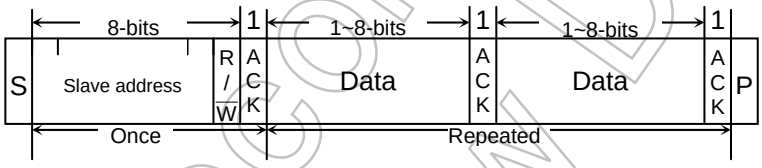
- Serial bus interface control registers 0 (SBI0CR0,SBI1CR0)
- Serial bus interface control registers 1 (SBI0CR1,SBI1CR1)
- Serial bus interface control registers 2 (SBI0CR2,SBI1CR2)
- Serial bus interface buffer registers (SBI0DBR,SBI1DBR)
- I^2C bus address register (I2CAR)
- Serial bus interface status registers (SBI0SR,SBI1SR)
- Serial bus interface baud rate registers 0 (SBI0BR0,SBI1BR0)

The functions of these registers vary, depending on the mode in which the SBI is operating. For a detailed description of the registers, refer to "15.5 Control in the I^2C Bus Mode" and "15.7 Control in the Clock-synchronous 8-bit SIO Mode".

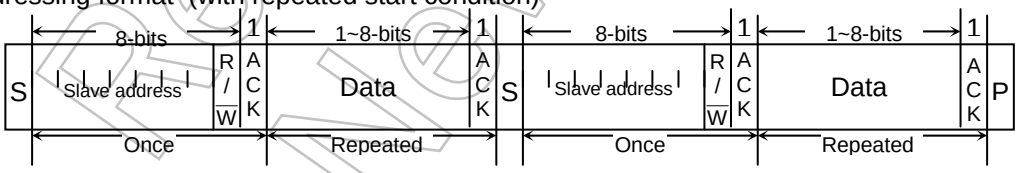
15.3 I^2C Bus Mode Data Formats

Fig. 15.2 shows the data formats used in the I^2C bus mode.

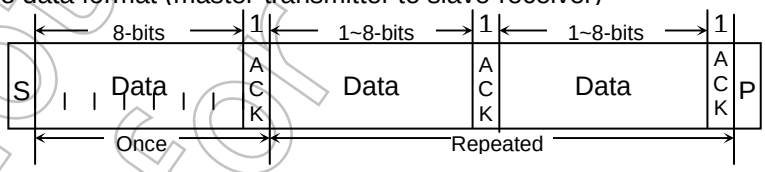
(a) Addressing format



(b) Addressing format (with repeated start condition)



(c) Free data format (master-transmitter to slave-receiver)



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

Fig. 15-2 I^2C Bus Mode Data Formats

15.4 Control Registers in the I^2C Bus Mode

The following registers control the serial bus interface (SBI) in the I^2C bus mode and provide its status information for monitoring.

Serial bus interface control register 0

SBIxCR0
(0xFFFF_F607)
(0xFFFF_F617)

	7	6	5	4	3	2	1	0
Bit symbol	SBIEN							
Read/Write	R/W	R						
After reset	0	0						
Function	SBI operation 0: Disable 1: Enable	This can be read as “0.”						

<SBIEN>: To use the SBI, enable the SBI operation (“1”) before setting each register in the SBI module.

Fig. 15-3 Serial bus interface control register 0

Serial bus interface control register 1

SBIxCR1
(0xFFFF_F600)
(0xFFFF_F610)

	7	6	5	4	3	2	1	0
Bit symbol	BC2	BC1	BC0	ACK		SCK2	SCK1	SCK0/ SWRMON
Read/Write	R/W			R/W	R	R/W		R/W
After reset	0	0	0	0	1	0	0	1
Function	Select the number of bits per transfer (Note 1)			Acknowledgment clock 0: Not generate 1: Generate	This can be read as "1."	Select internal SCL output clock frequency (Note 2) and reset monitor.		

On writing <SCK2:0>: Select internal SCL output clock frequency

000	n=5	265 kHz
001	n=6	201 kHz
010	n=7	136 kHz
011	n=8	83 kHz
100	n=9	46 kHz
101	n=10	25 kHz
110	n=11	13 kHz
111		reserved

System clock : fsys(=54 MHz)
Clock gear : fc/1

$$\text{Frequency} = \frac{\text{fsys}/2}{2^n + 70} \quad [\text{Hz}]$$

On writing <SCK2:0>: Select internal SCL output clock frequency

000	n=5	196 kHz
001	n=6	149 kHz
010	n=7	z101kHz
011	n=8	61 kHz
100	n=9	34 kHz
101	n=10	18 kHz
110	n=11	9 kHz
111		reserved

System clock : fsys(=40 MHz)
Clock gear : fc/1

$$\text{Frequency} = \frac{\text{fsys}/2}{2^n + 70} \quad [\text{Hz}]$$

On reading <SWRMON>: Software reset status monitor

0	Software reset operation is in progress.
1	Software reset operation is not in progress.

Select the number of bits per transfer

<BC2:0>	When <ACK> = 0		When <ACK> = 1	
	Number of clock cycles	Data length	Number of clock cycles	Data
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) **Clear <BC2:0> to "000" before switching the operation mode to the clock-synchronous 8-bit SIO mode.**

(Note 2) **For details on the SCL line clock frequency, refer to "15.7.1 (3) Serial Clock."**

(Note 3) **After a reset, the <SCK0/SWRMON> bit is read as "1." However, if the SIO mode is selected at the SBIxCR2 register, the initial value of the <SCK0> bit is "0."**

Fig. 15-4 Serial bus interface control register 1

Serial bus interface control register 2

SBIxCR2
(0xFFFF_F603)
(0xFFFF_F613)

	7	6	5	4	3	2	1	0
Bit symbol	MST	TRX	BB	PIN	SBIM1	SBIM0	SWRST1	SWRST0
Read/Write	W				W (Note 1)		W (Note 1)	
After reset	0	0	0	1	0	0	0	0
Function	Select master/slave 0: Slave 1: Master	Select transmit/ receive 0: Receive 1: Transmit	Start/stop condition generation 0: Stop condition generated 1: Start condition generated	Clear INTS0 interrupt request 0: – v1: Clear interrupt request	Select serial bus interface operating mode (Note 2) 00: Port mode 01: SIO mode 10: I²C bus mode 11: (Reserved))		Software reset generation Write “10” followed by “01” to generate a reset.	

→ Select serial bus interface operating mode (Note 2)

00	Port mode (Serial bus interface output disabled)
01	Clock-synchronous 8-bit SIO mode
10	I^2C bus mode
11	(Reserved)

(Note 1) **Reading this register causes it to function as the SBISR register.**

(Note 2) **Ensure that the bus is free before switching the operating mode to the port mode. Ensure that the port is at the “H” level before switching the operating mode from the port mode to the I^2C bus or clock-synchronous 8-bit SIO mode.**

(Note 3) **Ensure that serial transfer is completed before switching the mode.**

Fig. 15-5 Serial bus interface control register 2

Table-15-6 Base Clock Resolution

Clock gear value <GEAR1:0>	Base clock resolution (fsys = 54 MHz)	Base clock resolution (fsys = 40 MHz)
00 (fc)	fsys/2^2 (0.07 s)	fsys/2^2 (0.1 s)
01 (fc/2)	fsys/2^3 (0.15 s)	fsys/2^3 (0.2 s)
10 (fc/4)	fsys/2^4 (0.30 s)	fsys/2^4 (0.4 s)
11 (fc/8)	fsys/2^5 (0.59 s)	fsys/2^5 (0.8 s)

Serial bus interface status register

SBIxSR
(0xFFFF_F603)
(0xFFFF_F613)

	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 selection monitor 0: Slave 1: Master	Transmit/ receive selection monitor 0: Receive 1: Transmit	I^2C bus state monitor 0: Free 1: Busy	INTS0 interrupt request monitor 0: Interrupt request generated 1: Interrupt request cleared	Arbitration lost detection 0: – 1: Detected	Slave address match detection 0: – 1: Detected	General call detection 0: – 1: Detected	Last received bit monitor 0: “0” 1: “1”

Last received bit monitor

0	The last bit received was “0.”
1	The last bit received was “1.”

Addressed as slave

0	-
1	Addressed as slave or general call detected.

Arbitration lost

0	-
1	Arbitration was lost to another master

(Note) Writing to this register causes it to function as SBICR2.

Fig. 15-7 Serial bus interface status register

Serial bus interface baud rate register 0

SBIxBR0 (0xFFFF_F604) (0xFFFF_F614)

	7	6	5	4	3	2	1	0
Bit symbol		I2SBI0						
Read/Write	R	R/W	R					R/W
After reset	1	0	1					0
Function	This can be read as "1".	IDLE 0: Stop 1: Operate	This can be read as "1".					Make sure that you write "0." (Note)

Operation at the IDLE mode

0	Stop
1	Operate

(Note) This is read as "1" at the SIO mode.

Serial bus interface data buffer register

SBIxDBR (0xFFFF_F601) (0xFFFF_F611)

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

(Note) Transmit data must be written to this register, with bit 7 being the most-significant bit (MSB).

I^2C bus address register

SBIxI2CAR (0xFFFF_F602) (0xFFFF_F612)

	7	6	5	4	3	2	1	0
Bit symbol	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ALS
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Set the slave address when the SBI acts as a slave device.							Specify address recognition mode

Specify address recognition mode

0	Recognizes the slave address.
1	Does not recognize slave address.

(Note) Please set the bit 0 <ALS> of I2C bus address register I2CAR to "0", except when you use the free data format. It operates as a free data format when setting it to "1", it fixes to the transmission at the master, and the transmission direction is fixed to the reception at the slave.

Fig. 15-8 I^2C Bus Mode Register

15.5 Control in the I²C Bus Mode

15.5.1 Setting the Acknowledgement Mode

Setting SBICR1<ACK> to "1" selects the acknowledge mode. When operating as a master, the SBI adds one clock for acknowledgment signals. As a transmitter, the SBI releases the SDA pin during this clock cycle to receive acknowledgment signals from the receiver. As a receiver, the SBI pulls the SDA pin to the "L" level during this clock cycle and generates acknowledgment signals.

By setting <ACK> to "0", the non-acknowledgment mode is activated. When operating as a master, the SBI does not generate clock for acknowledgement signals.

15.5.2 Setting the Number of Bits per Transfer

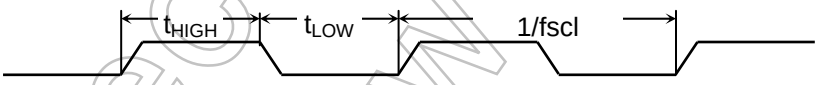
SBICR1 <BC2:0> specifies the number of bits of the next data to be transmitted or received.

Under the start condition, <BC2:0> is set to "000," causing a slave address and the direction bit to be transferred in a packet of eight bits. At other times, <BC2:0> keeps a previously programmed value.

15.5.3 Serial Clock

① Clock source

SBICR1 <SCK2:0> specifies the maximum frequency of the serial clock to be output from the SCL pin in the master mode.



$t_{LOW} = 2^{n-1}/(fsys/2) + 58/(fsys/2)$

$t_{HIGH} = 2^{n-1}/(fsys/2) + 12/(fsys/2)$

$fscl = 1/(t_{Low} + t_{HIGH}) = \frac{fsys/2}{2^n + 70}$

SBIxCR1 <SCK2:0>	n
000	5
001	6
010	7
011	8
100	9
101	10
110	11

Fig. 15-9 Clock Source

(Note) The highest speeds in the standard and high-speed modes are specified to 100KHz and 400KHz respectively following the communications standards. Note that the internal SCL clock frequency is determined by the fsys used and the calculation formula shown above.

② Clock Synchronization

The I^2C bus is driven by using the wired-AND connection due to its pin structure. The first master that pulls its clock line to the “L” level overrides other masters producing the “H” level on their clock lines. This must be detected and responded by the masters producing the “H” level.

Clock synchronization assures correct data transfer on a bus that has two or more master.

For example, the clock synchronization procedure for a bus with two masters is shown below.

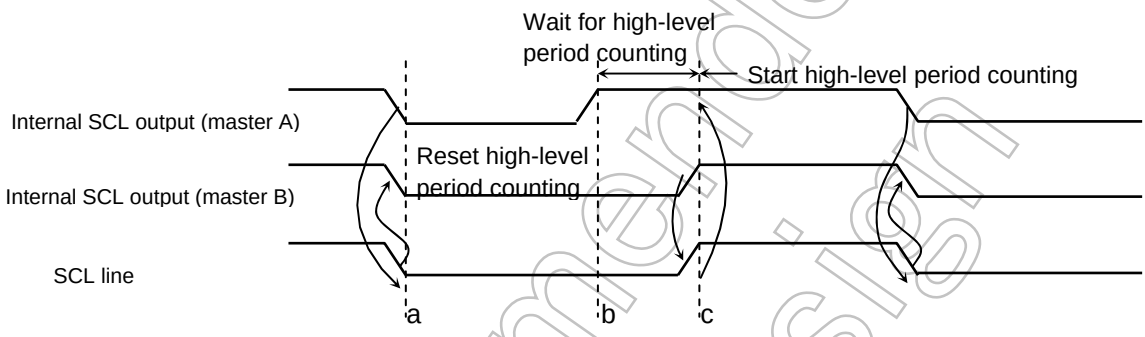


Fig. 15-10 Example of Clock Synchronization

At the point a, Master A pulls its internal SCL output to the “L” level, bringing the SCL bus line to the “L” level. Master B detects this transition, resets its “H” level period counter, and pulls its internal SCL output level to the “L” level.

Master A completes counting of its “L” level period at the point b, and brings its internal SCL output to the “H” level. However, Master B still keeps the SCL bus line at the “L” level, and Master A stops counting of its “H” level period counting. After Master A detects that Master B brings its internal SCL output to the “H” level and brings the SCL bus line to the “H” level at the point c, it starts counting of its “H” level period.

This way, the clock on the bus is determined by the master with the shortest “H” level period and the master with the longest “L” level period among those connected to the bus.

15.5.4 Slave Addressing and Address Recognition Mode

When the SBI is configured to operate as a slave device, the slave address <SA6:0> and <ALS> must be set at SBIxI2CAR. Setting <ALS> to “0” selects the address recognition mode.

15.5.5 Configuring the SBI as a Master or a Slave

Setting SBICR2<MST> to “1” configures the SBI to operate as a master device.

Setting <MST> to “0” configures the SBI as a slave device. <MST> is cleared to “0” by the hardware when it detects the stop condition on the bus or the arbitration lost.

15.5.6 Configuring the SBI as a Transmitter or a Receiver

Setting SBICR2 <TRX> to “1” configures the SBI as a transmitter. Setting <TRX> to “0” configures the SBI as a receiver.

At the slave mode, the SBI receives the direction bit ($R/\overline{W}$) from the master device on the following occasions:

- when data is transmitted in the addressing format
- when the received slave address matches the value specified at SBIxI2CCR
- when a general-call address is received; i.e., the eight bits following the start condition are all zeros

If the value of the direction bit ($R/\overline{W}$) is “1,” <TRX> is set to “1” by the hardware. If the bit is “0,” <TRX> is set to “0”.

As a master device, the SBI receives acknowledgement from a slave device. If the direction bit of “1” is transmitted, <TRX> is set to “0” by the hardware. If the direction bit is “0,” <TRX> changes to “1.” If the SBI does not receive acknowledgement, <TRX> retains the previous value.

<TRX> is cleared to “0” by the hardware when it detects the stop condition on the bus or the arbitration lost.

15.5.7 Generating Start and Stop Conditions

When SBIxSR<BB> is “0,” writing “1” to SBIxCR2 <MST, TRX, BB, PIN> causes the SBI to generate the start condition on the bus and output 8-bit data. <ACK> must be set to “1” in advance.

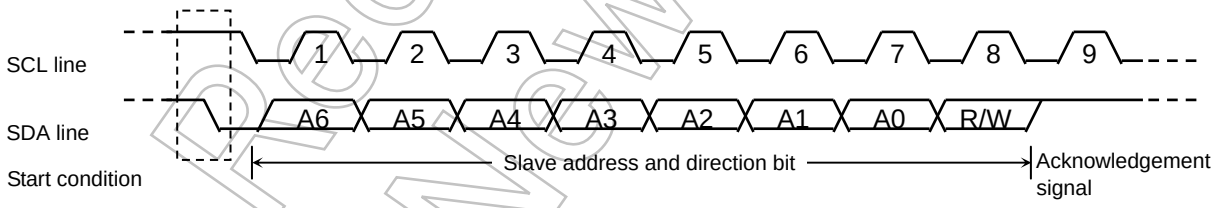


Fig. 15-11 Generating the Start Condition and a Slave Address

When <BB> is “1,” writing “1” to <MST, TRX, PIN> and “0” to <BB> causes the SBI to start a sequence for generating the stop condition on the bus. The contents of <MST, TRX, BB, PIN> should not be altered until the stop condition appears on the bus.

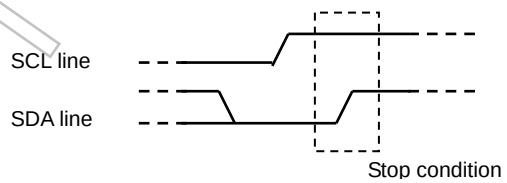


Fig. 15-12 Generating the Stop Condition

SBIxSR<BB> can be read to check the bus state. <BB> is set to “1” when the start condition is detected on the bus (the bus is busy), and set to “0” when the stop condition is detected (the bus is free).

15.5.8 Interrupt Service Request and Release

When a serial bus interface interrupt request (INTSBI0, INTSBI1) is generated, SBIxCR2 <PIN> is cleared to "0." While <PIN> is "0," the SBI pulls the SCL line to the "L" level.

After transmission or reception of one data word, <PIN> is cleared to "0." It is set to "1" when data is written to or read from SBIxDBR. It takes a period of t_{LOW} for the SCL line to be released after <PIN> is set to "1."

In the address recognition mode (<ALS> = "0"), <PIN> is cleared to "0" when the received slave address matches the value specified at I2CAR or when a general-call address is received; i.e., the eight bits following the start condition are all zeros. When the program writes "1" to SBIxCR2<PIN>, it is set to "1." However, writing "0" does clear this bit to "0".

15.5.9 Serial Bus Interface Operating Modes

SBIxCR2 <SBIM1:0> selects an operating mode of the serial bus interface. <SBIM1:0> must be set to "10" to configure the SBI for the I^2C bus mode. Make sure that the bus is free before switching the operating mode to the port mode.

15.5.10 Lost-arbitration Detection Monitor

The I^2C bus has the multi-master capability (there are two or more masters on a bus), and requires the bus arbitration procedure to ensure correct data transfer.

A master that attempts to generate the start condition while the bus is busy loses bus arbitration, with no start condition occurring on the SDA and SCL lines. The I^2C-bus arbitration takes place on the SDA line.

The arbitration procedure for two masters on a bus is shown below. Up until the point a, Master A and Master B output the same data. At the point a, Master A outputs the "L" level and Master B outputs the "H" level. Then Master A pulls the SDA bus line to the "L" level because the line has the wired-AND connection. When the SCL line goes high at the point b, the slave device reads the SDA line data, i.e., data transmitted by Master A. At this time, data transmitted by Master B becomes invalid. This condition of Master B is called "Lost Arbitration". Master B releases its SDA pin, so that it does not affect the data transfer initiated by another master. If two or more masters have transmitted exactly the same first data word, the arbitration procedure continues with the second data word.

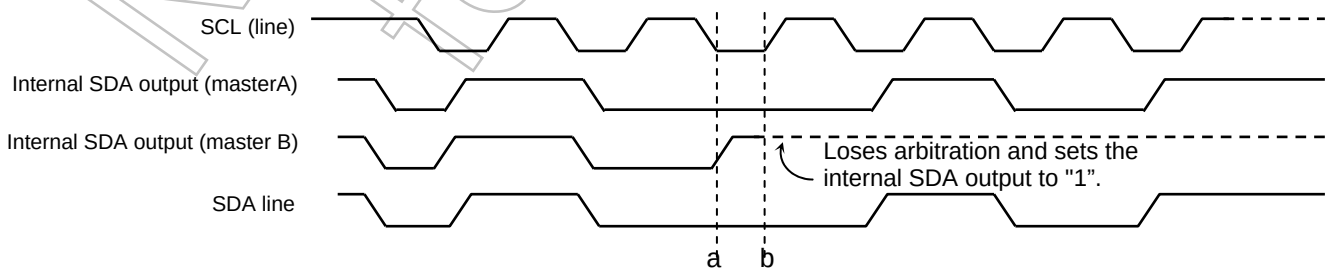


Fig. 15-13 Lost Arbitration

A master compares the SDA bus line level and the internal SDA output level at the rising of the SCL line. If there is a difference between these two values, Arbitration Lost occurs and SBI0SR <AL> is set to “1”.

When <AL> is set to “1,” SBISR <MST, TRX> are cleared to “0,” causing the SBI to operate as a slave receiver. <AL> is cleared to “0” when data is written to or read from SBIxDBR or data is written to SBIxCR2.

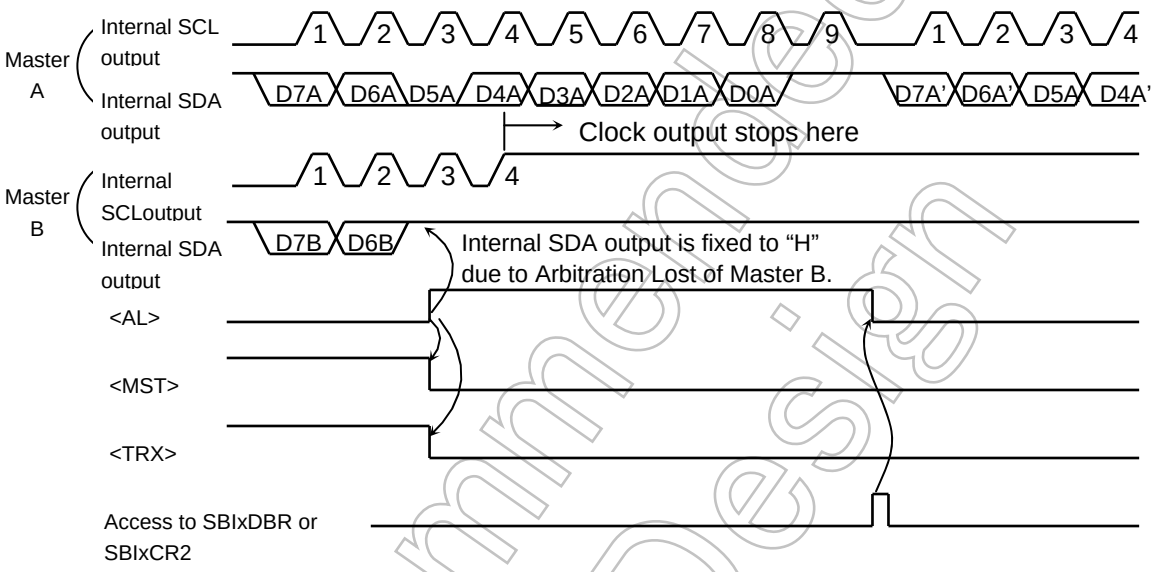


Fig. 15-14 Example of Master B Lost Arbitration (D7A = D7B, D6A = D6B)

15.5.11 Slave Address Match Detection Monitor

When the SBI operates as a slave device in the address recognition mode (SBIxI2CAR <ALS> = ”0”), SBIxSR <AAS> is set to “1” on receiving the general-call address or the slave address that matches the value specified at SBIxI2CAR. When <ALS> is “1,” <AAS> is set to “1” when the first data word has been received. <AAS> is cleared to “0” when data is written to or read from SBIxDBR.

15.5.12 General-call Detection Monitor

When the SBI operates as a slave device, SBIxSR <AD0> is set to “1” when it receives the general-call address; i.e., the eight bits following the start condition are all zeros. <AD0> is cleared to “0” when the start or stop condition is detected on the bus.

15.5.13 Last Received Bit Monitor

SBIxSR <LRB> is set to the SDA line value that was read at the rising of the SCL line. In the acknowledgment mode, reading SBIxSR <LRB> immediately after generation of the INTSBIx interrupt request causes ACK signal to be read.

15.5.14 Software Reset

If the serial bus interface circuit locks up due to external noise, it can be initialized by using a software reset.

Writing “10” followed by “01” to SBIxCR2 <SWRST1:0> generates a reset signal that initializes the serial bus interface circuit. After a reset, all control registers and status flags are initialized to their reset values. When the serial bus interface is initialized, <SWRST> is automatically cleared to “0”.

(Note)	**A software reset causes the SBI operating mode to switch from the I^2C mode to the port mode.**

15.5.15 Serial Bus Interface Data Buffer Register (SBIxDBR)

Reading or writing SBIxDBR initiates reading received data or writing transmitted data. When the SBI is acting as a master, setting a slave address and a direction bit to this register generates the start condition.

15.5.16 I^2C Bus Address Register (SBIxI2CAR)

When the SBI is configured as a slave device, the SBIxI2CAR<SA6:0> bit is used to specify a slave address. If SBIxI2CAR <ALS> is set to “0,” the SBI recognizes a slave address transmitted by the master device and receives data in the addressing format. If <ALS> is set to “1,” the SBI does not recognize a slave address and receives data in the free data format.

15.5.17 IDLE Setting Register (SBIxBR0)

The SBIxBR0<I2SBI> register determines if the SBI operates or not when it enters the IDLE mode. This register must be programmed before executing an instruction to switch to the standby mode.

15.6 Data Transfer Procedure in the I^2C Bus Mode

15.6.1 Device Initialization

First, program SBIxCR1<ACK, SCK2:0> by writing “0” to bits 7 to 5 and bit 3 in SBIxCR1.

Next, program SBIxI2CAR by specifying a slave address at <SA6:0> and an address recognition mode at <ALS>. (<ALS> must be set to”0” when using the addressing format).

Next, program SBIxCR2 to initially configure the SBI in the slave receiver mode by writing “0” to <MST, TRX, BB> , “1” to <PIN> , “10” to <SBIM1:0> and “0” to bits 1 and 0.

	7 6 5 4 3 2 1 0	
SBIxCR1	← 0 0 0 X 0 X X X	Specifies ACK and SCL clock.
SBIxI2CAR	← X X X X X X X X	Specifies a slave address and an address recognition mode.
SBIxCR2	← 0 0 0 1 1 0 0 0	Configures the SBI as a slave receiver.

(Note) X: Don’t care

15.6.2 Generating the Start Condition and a Slave Address

① Master mode

In the master mode, the following steps are required to generate the start condition and a slave address.

First, ensure that the bus is free (<BB> = “0”). Then, write “1” to SBIxCR1 <ACK> to select the acknowledgment mode. Write to SBIxDBR a slave address and a direction bit to be transmitted.

When <BB> = “0,” writing “1111” to SBIxCR2 <MST, TRX, BB, PIN> generates the start condition on the bus. Following the start condition, the SBI generates nine clocks from the SCL pin. The SBI outputs the slave address and the direction bit specified at SBIxDBR with the first eight clocks, and releases the SDA line in the ninth clock to receive an acknowledgment signal from the slave device.

The INTSBIx interrupt request is generated on the falling of the ninth clock, and <PIN> is cleared to ”0.” In the master mode, the SBI holds the SCL line at the “L” level while <PIN> is “0.” <TRX> changes its value according to the transmitted direction bit at generation of the INTSBIx interrupt request, provided that an acknowledgment signal has been returned from the slave device.

Settings in main routine

	7 6 5 4 3 2 1 0	
→ Reg.	← SBISR	
Reg.	← Reg. e 0x20	
if Reg.	≠ 0x00	Ensures that the bus is free.
Then		
SBIxCR1	← X X X 1 0 X X X	Selects the acknowledgement mode.
SBIxDR1	← X X X X X X X X	Specifies the desired slave address and direction.
SBIxCR2	← 1 1 1 1 1 0 0 0	Generates the start condition.

Example of INTSBIx interrupt routine

INTCLR ← 0x78	Clears the interrupt request.
Processing	
End of interrupt	

② Slave mode

In the slave mode, the SBI receives the start condition and a slave address.

After receiving the start condition from the master device, the SBI receives a slave address and a direction bit from the master device during the first eight clocks on the SCL line. If the received address matches its slave address specified at SBIxI2CAR or is equal to the general-call address, the SBI pulls the SDA line to the "L" level during the ninth clock and outputs an acknowledgment signal.

The INTSBIx interrupt request is generated on the falling of the ninth clock, and <PIN> is cleared to "0." In the slave mode, the SBI holds the SCL line at the "L" level while <PIN> is "0".

(Note) The user can only use a DMA transfer:
- **when there is only one master and only one slave and**
- **continuous transmission or reception is possible.**

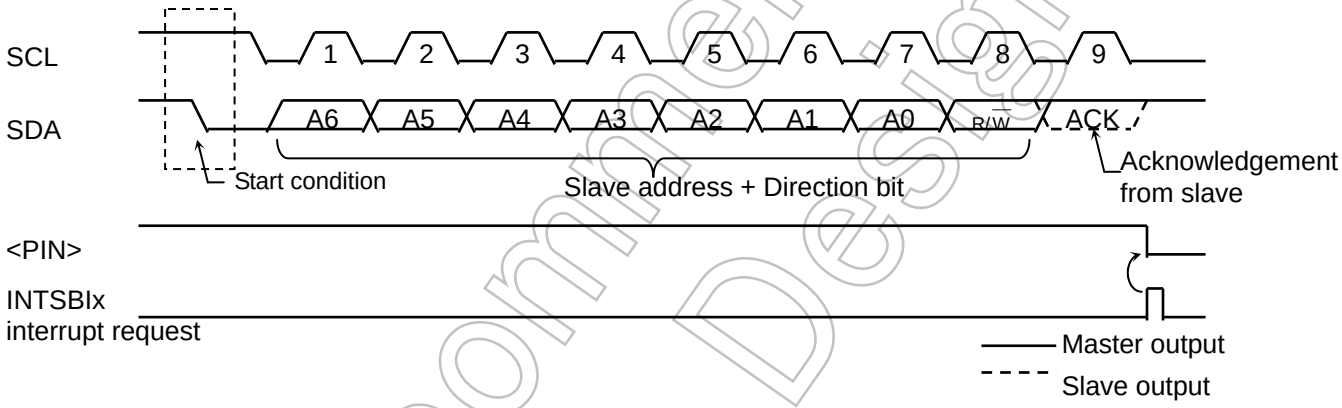


Fig. 15-15 Generation of the Start Condition and a Slave Address

15.6.3 Transferring a Data Word

At the end of a data word transfer, the INTSBIx interrupt is generated to test <MST> to determine whether the SBI is in the master or slave mode.

① Master mode (<MST> = "1")

Test <TRX> to determine whether the SBI is configured as a transmitter or a receiver.

Transmitter mode (<TRX> = "1")

Test <LRB>. If <LRB> is "1," that means the receiver requires no further data. The master then generates the stop condition as described later to stop transmission.

If <LRB> is "0," that means the receiver requires further data. If the next data to be transmitted has eight bits, the data is written into SBIxDBR. If the data has different length, <BC2:0> and <ACK> are programmed and the transmit data is written into SBIxDBR. Writing the data makes <PIN> to"1," causing the SCL pin to generate a serial clock for transferring a next data word, and the SDA pin to transfer the data word. After the transfer is completed, the INTSBIx interrupt request is generated, <PIN> is set to "0," and the SCL pin is pulled to the "L" level. To transmit more data words, test <LRB> again and repeat the above procedure.

INTSBIx interrupt

if MST = 0
Then go to the slave-mode processing
if TRX = 0
Then go to the receiver-mode processing
if LRB = 0
Then go to processing for generating the stop condition

SBIxCR1 ← X X X X 0 X X X Specifies the number of bits to be transmitted and specify whether ACK is required.

SBIxDBR ← X X X X X X X X Writes the transmit data.

End of interrupt processing

(Note) X: Don't care

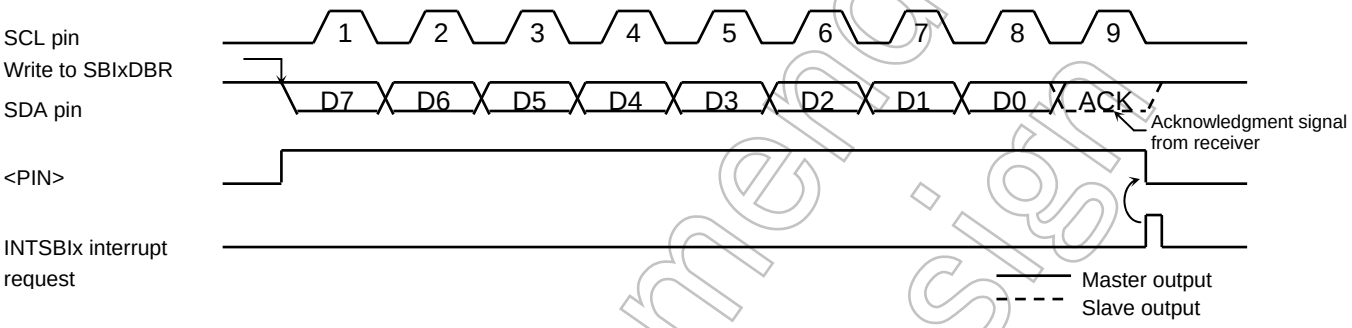


Fig. 15-16 <BC2:0> = "000" and <ACK> = "1" (Transmitter Mode)

Receiver mode (<TRX> = "0")

If the next data to be transmitted has eight bits, the transmit data is written into SBIxDBR. If the data has different length, <BC2:0> and <ACK> are programmed and the received data is read from SBIxDBR to release the SCL line. (The data read immediately after transmission of a slave address is undefined.) On reading the data, <PIN> is set to "1," and the serial clock is output to the SCL pin to transfer the next data word. In the last bit, when the acknowledgment signal becomes the "L" level, "0" is output to the SDA pin.

After that, the INTSBIx interrupt request is generated, and <PIN> is cleared to "0," pulling the SCL pin to the "L" level. Each time the received data is read from SBIDBR, one-word transfer clock and an acknowledgement signal are output.

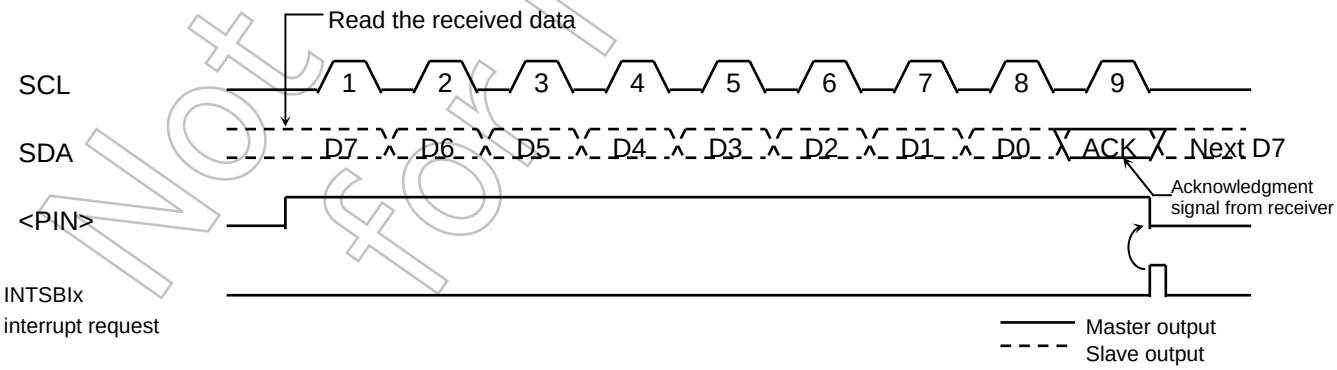


Fig. 15-17 <BC2:0> = "000" and <ACK> = "1" (Receiver Mode)

To terminate the data transmission from the transmitter, <ACK> must be set to “0” immediately before reading the data word second to last. This disables generation of an acknowledgment clock for the last data word. When the transfer is completed, an interrupt request is generated. After the interrupt processing, <BC2:0> must be set to “001” and the data must be read so that a clock is generated for 1-bit transfer. At this time, the master receiver holds the SDA bus line at the “H” level, which signals the end of transfer to the transmitter as an acknowledgment signal.

In the interrupt processing for terminating the reception of 1-bit data, the stop condition is generated to terminate the data transfer.

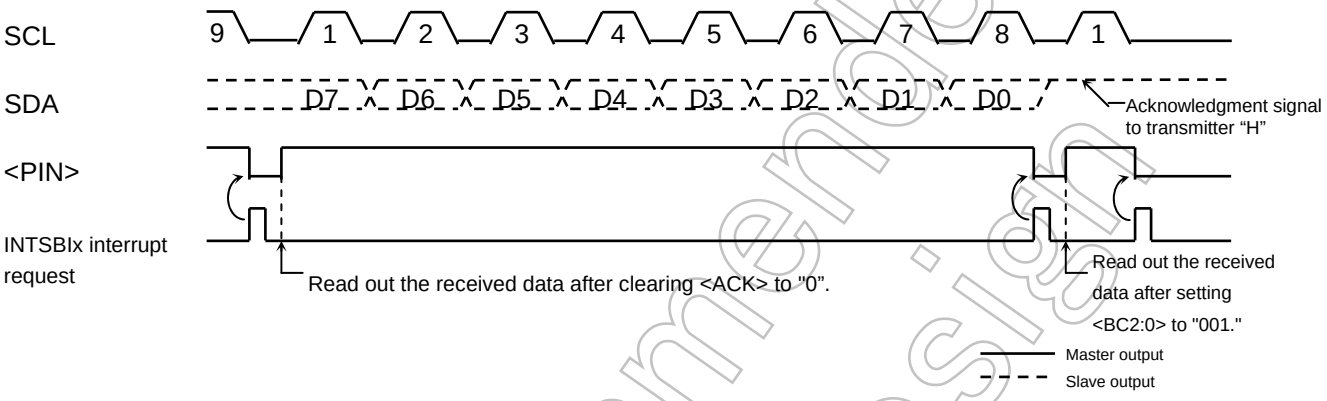


Fig. 15-18 Terminating Data Transmission in the Master Receiver Mode

Example: When receiving N data word

INTSBIx interrupt (after data transmission)

```
                  7 6 5 4 3 2 1 0
SBIxCR1  ← X X X X 0 X X X      Sets the number of bits of data to be received and
                                specify whether ACK is required.
Reg.     ← SBIxDBR              Reads dummy data.
End of interrupt
```

INTSBIx interrupt (first to (N-2)th data reception)

```
                  7 6 5 4 3 2 1 0
Reg.     ← SBIxDBR              Reads the first to (N-2)th data words.
End of interrupt
```

INTSBIx interrupt ((N-1)th data reception)

```
                  7 6 5 4 3 2 1 0
SBIxCR1  ← X X X 0 0 X X X      Disables generation of acknowledgement clock.
Reg.     ← SBIxDBR              Reads the (N-1)th data word.
End of interrupt
```

INTSBIx interrupt (Nth data reception)

```
                  7 6 5 4 3 2 1 0
SBIxCR1  ← 0 0 1 0 0 X X X      Disables generation of acknowledgement clock.
Reg.     ← SBIxDBR              Reads the (N-1)th data word.
End of interrupt
```

INTSBIx interrupt (after completing data reception)

```
Processing to generate the stop condition.   Terminates the data transmission.
End of interrupt
```

(Note) X: Don’t care

② Slave mode (<MST> = “0”)

In the slave mode, the SBI generates the INTSBIx interrupt request on four occasions: 1) when the SBI has received any slave address from the master, 2) when the SBI has received a general-call address, 3) when the received slave address matches its own address, and 4) when a data transfer has been completed in response to a general-call. Also, if the SBI detects Arbitration Lost in the master mode, it switches to the slave mode. Upon the completion of data word transfer in which Arbitration Lost is detected, the INTSBIx interrupt request is generated, <PIN> is cleared to “0,” and the SCL pin is pulled to the “L” level. When data is written to or read from SBIxDBR or when <PIN> is set to “1,” the SCL pin is released after a period of t_{LOW}.

In the slave mode, the normal slave mode processing or the processing as a result of Arbitration Lost is carried out.

SBIxSR <AL>, <TRX>, <AAS> and <AD0> are tested to determine the processing required. Table 15-19 shows the slave mode states and required processing.

Example: When the received slave address matches the SBI's own address and the direction bit is “1” in the slave receiver mode.

INTSBIx interrupt

if TRX = 0
Then go to other processing
if AL = 1
Then go to other processing
if AAS = 0
Then go to other processing

SBIxCR1 ← X X X 1 0 X X X	Sets the number of bits to be transmitted.
SBIxDBR ← X X X X 0 X X X	Sets the transmit data.

(Note) X: Don’t care

Table 15-19 Processing in Slave Mode

<TRX>	<AL>	<AAS>	<AD0>	State	Processing
1	1	1	0	Arbitration Lost is detected while the slave address was being transmitted, and the SBI received a slave address with the direction bit “1” transmitted by another master.	Set the number of bits in a data word to <BC2:0> and write the transmit data into SBIxDBR.
	0	1	0	In the slave receiver mode, the SBI received a slave address with the direction bit “1” transmitted by the master.	
		0	0	In the slave transmitter mode, the SBI has completed a transmission of one data word.	Test LRB. If it has been set to “1,” that means the receiver does not require further data. Set <PIN> to 1 and reset <TRX> to 0 to release the bus. If <LRB> has been reset to “0,” that means the receiver requires further data. Set the number of bits in the data word to <BC2:0> and write the transmit data to the SBIxDBR.
0	1	1	1/0	Arbitration Lost is detected while a slave address is being transmitted, and the SBI receives either a slave address with the direction bit “0” or a general-call address transmitted by another master.	Read the SBIxDBR (a dummy read) to set <PIN> to 1, or write “1” to <PIN>.
		0	0	Arbitration Lost is detected while a slave address or a data word is being transmitted, and the transfer is terminated.	
	0	1	1/0	In the slave receiver mode, the SBI received either a slave address with the direction bit “0” or a general-call address transmitted by the master.	
		0	1/0	In the slave receiver mode, the SBI has completed a reception of a data word.	Set the number of bits in the data word to <BC2:0> and read the received data from SBIxDBR.

15.6.4 Generating the Stop Condition

When SBIxSR <BB> is “1,” writing “1” to SBIxCR2 <MST, TRX, PIN> and “0” to <BB> causes the SBI to start a sequence for generating the stop condition on the bus. Do not alter the contents of <MST, TRX, BB, PIN> until the stop condition appears on the bus.

If another device is holding down the SCL bus line, the SBI waits until the SCL line is released. After that, the SDA pin goes high, causing the stop condition to be generated.

```
                  7 6 5 4 3 2 1 0
SBICR2    ←       1 1 0 1 1 0 0 0      Generates the stop condition.
```

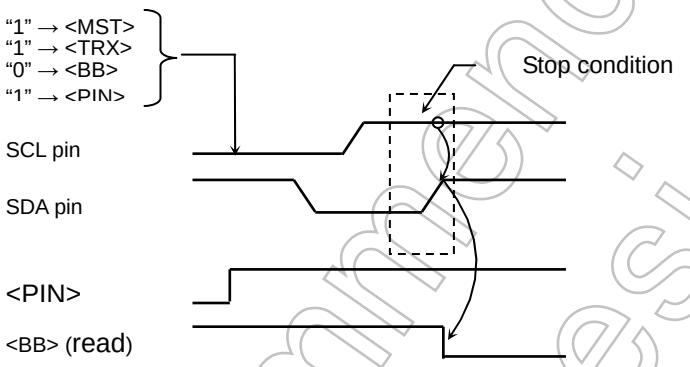


Fig. 15-20 Generating the Stop Condition

15.6.5 Restart Procedure

Restart is used when a master device changes the data transfer direction without terminating the transfer to a slave device. The procedure of generating a restart in the master mode is described below.

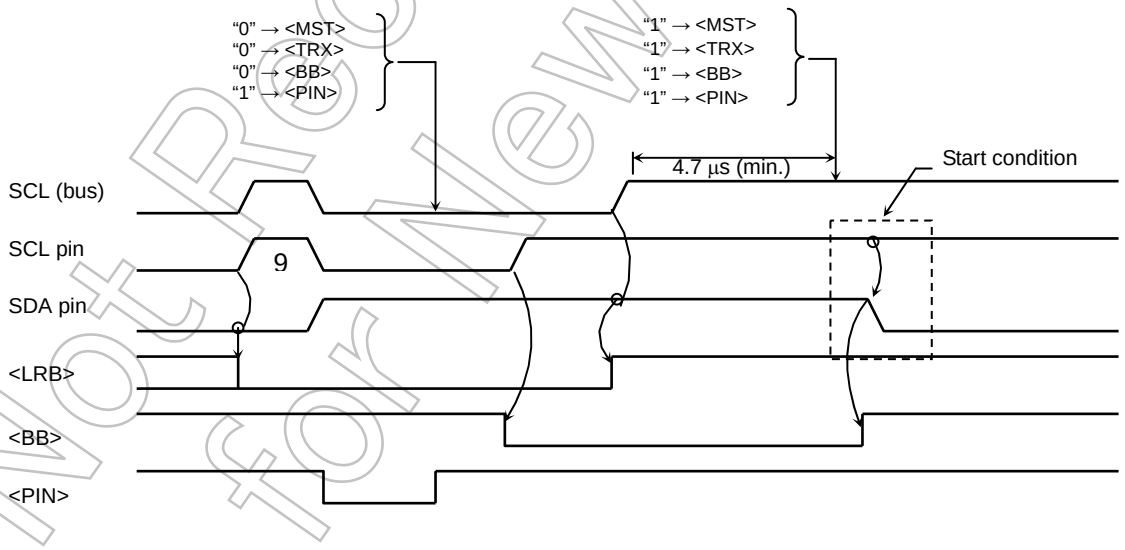
First, set SBIxCR2 <MST, TRX, BB> to "0" and write "1" to <PIN> to release the bus. At this time, the SDA pin is held at the "H" level and the SCL pin is released. Because no stop condition is generated on the bus, other devices recognize that the bus is busy. Then, test SBIxSR <BB> and wait until it becomes "0" to ensure that the SCL pin is released. Next, test <LRB> and wait until it becomes "1" to ensure that no other device is pulling the SCL bus line to the "L" level. Once the bus is determined to be free this way, use the above-mentioned steps ② to generate the start condition.

To satisfy the setup time of restart, at least 4.7-μs wait period (in the standard mode) must be created by the software after the bus is determined to be free.

```
                    7 6 5 4 3 2 1 0
SBIxCR2  ← 0 0 0 1 1 0 0 0   Releases the bus.
if SBIxSR<BB> ≠ 0            Checks that the SCL pin is released.
Then
if SBIxSR<LRB> ≠ 1           Checks that no other device is pulling the SCL pin to the
                             "L" level.

Then
4.7 μs Wait
SBIxCR1  ← X X X 1 0 X X X   Selects the acknowledgment mode.
SBIxDBR  ← X X X X X X X X   Sets the desired slave address and direction.
SBIxCR2  ← 1 1 1 1 1 0 0 0   Generates the start condition.

(Note) X: Don't care
```



(Note) Do not write <MST> to "0" when it is "0." (Restart cannot be initiated.)

Fig. 15-21 Timing Chart of Generating a Restart

15.7 Control in the Clock-synchronous 8-bit SIO Mode

The following registers control the serial bus interface in the clock-synchronous 8-bit SIO mode and provide its status information for monitoring.

Serial bus interface control register 0

SBIxCR0
(0xFFFF_F607)
(0xFFFF_F617)

	7	6	5	4	3	2	1	0
Bit symbol	SBIEN							
Read/Write	R/W	R						
After reset	0	0						
Function	SBI operation 0: Disable 1: Enable	This can be read as “0.”						

<SBIEN>: To use the SBI, enable the SBI operation (“1”) before setting each register of SBI module.

Serial bus interface control register 1

SBIxCR1
(0xFFFF_F600)
(0xFFFF_F610)

	7	6	5	4	3	2	1	0
Bit symbol	SIOS	SIOINH	SIOM1	SIOM0		SCK2	SCK1	SCK0
Read/Write	W				R	W		R/W
After reset	0	0	0	0	1	0	0	1
Function	Start transfer 0: Stop 1: Start	Transfer 0: Continue 1: Forced termination	Select transfer mode 00: Transmit mode 01: (Reserved) 10: Transmit/receive mode 11: Receive mode		This can be read as “1”.	Select serial clock frequency		

On writing <SCK2:0>: Select serial clock frequency

000	n = 3	1.69 MHz
001	n = 4	844 kHz
010	n = 5	422 kHz
011	n = 6	211 kHz
100	n = 7	105 kHz
101	n = 8	53 kHz
110	n = 9	26 kHz
111	—	External clock

System clock : fsys (=54 MHz)
Clock gear : fc/1
Frequency = $\frac{fsys/4}{2^n}$ [Hz]

On writing <SCK2:0>: Select serial clock frequency

000	n = 3	1.25 MHz
001	n = 4	625 kHz
010	n = 5	313 kHz
011	n = 6	156 kHz
100	n = 7	78 kHz
101	n = 8	39 kHz
110	n = 9	20 kHz
111	—	External clock

System clock : fsys (=40 MHz)
Clock gear : fc/1
Frequency = $\frac{fsys/4}{2^n}$ [Hz]

(Note) Set <SIOS> to "0" and <SIOINH> to "1" before programming the transfer mode and the serial clock.

Fig. 15-22 SIO Mode Registers

Serial bus interface data buffer register

SBIxDBR
(0xFFFF_F601)
(0xFFFF_F611)

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

Serial bus interface control register 2

SBIxCR2
(0xFFFF_F603)
(0xFFFF_F613

	7	6	5	4	3	2	1	0
Bit symbol					SBIM1	SBIM0		
Read/Write	R				W		R	
After reset	1				0	0	1	
Function	This can be read as “1”.				Select serial bus interface operating mode 00: Port mode 01: Clock-synchronous 8-bit SIO mode 10: I^2C bus mode 11: (Reserved)		This can be read as “1”.	

Serial bus interface register

SBIxSR
(0xFFFF_F603)
(0xFFFF_F613)

	7	6	5	4	3	2	1	0
Bit symbol					SIOF	SEF		
Read/Write	R				R		R	
After reset	1				0	0	1	
Function	This can be read as “1”.				Serial transfer status monitor 0: Completed 1: In progress	Shift operation status monitor 0: Completed 1: In progress	This can be read as “1”.	

Serial bus interface baud rate register 0

SBIxBR0
(0xFFFF_F604)
(0xFFFF_F614)

	7	6	5	4	3	2	1	0
Bit symbol		I2SBI						
Read/Write	R	R/W	R					W
After reset	1	0	1					0
Function	This can be read as “1”.	IDLE 0: Stop 1: Operate	This can be read as “1”.					Make sure to write “0.”

Fig. 15-23 SIO Mode Registers

15.7.1 Serial Clock

① Clock source

Internal or external clocks can be selected by programming SBIxCR1 <SCK2:0>.

Internal clocks

In the internal clock mode, one of the seven frequencies can be selected as a serial clock, which is output to the outside through the SCK pin. At the beginning of a transfer, the SCK pin output becomes the "H" level.

If the program cannot keep up with this serial clock rate in writing the transmit data or reading the received data, the SBI automatically enters a wait period. During this period, the serial clock is stopped automatically and the next shift operation is suspended until the processing is completed.

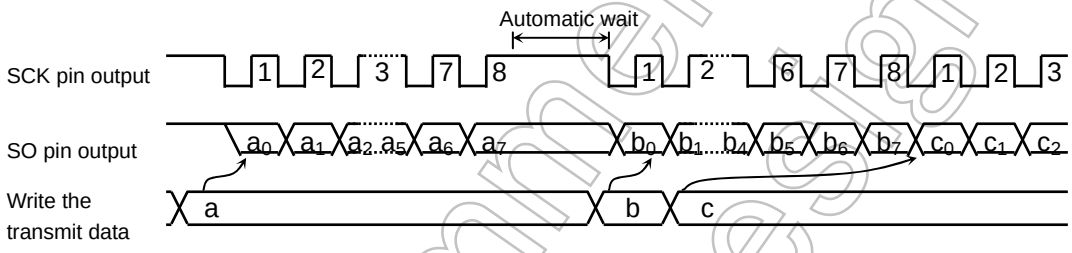


Fig. 15-24 Automatic Wait

External clock (<SCK2:0> = "111")

The SBI uses an external clock supplied from the outside to the SCK pin as a serial clock. For proper shift operations, the serial clock at the "H" and "L" levels must have the pulse widths as shown below.

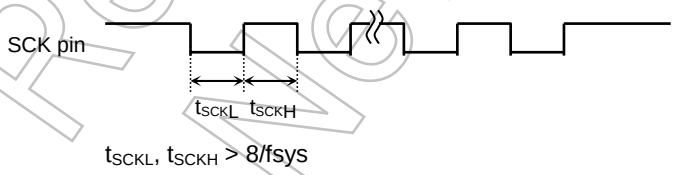


Fig. 15-25 Maximum Transfer Frequency of External Clock Input

② Shift Edge

Leading-edge shift is used in transmission. Trailing-edge shift is used in reception.

Leading-edge shift

Data is shifted at the leading edge of the serial clock (or the falling edge of the SCK pin input/output).

Trailing-edge shift

Data is shifted at the trailing edge of the serial clock (or the rising edge of the SCK pin input/output).

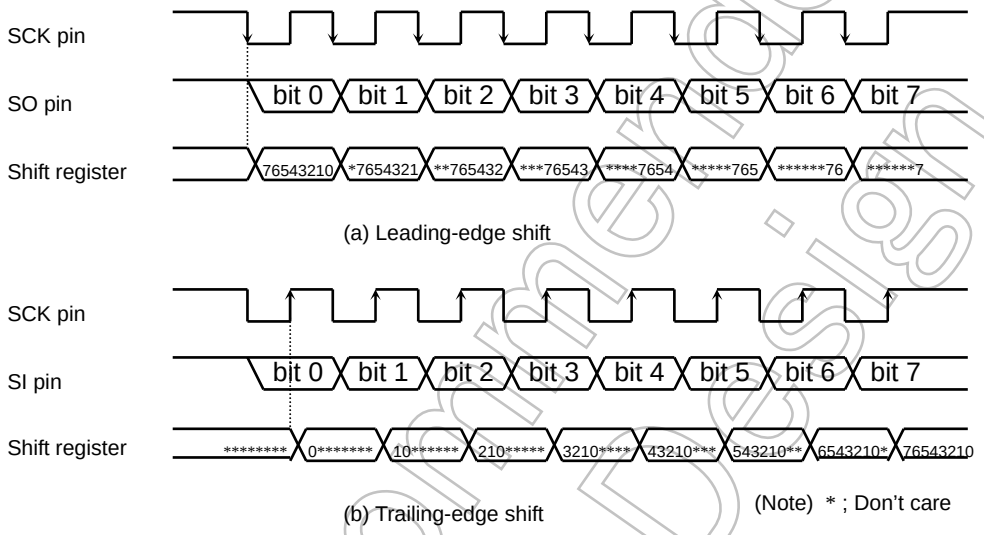


Fig. 15-26 Shift Edge

15.7.2 Transfer Modes

The transmit mode, the receive mode or the transmit/receive mode can be selected by programming SBIxCR1 <SIOM1:0>.

① 8-bit transmit mode

Set the control register to the transmit mode and write the transmit data to SBIxDBR.

After writing the transmit data, writing “1” to SBIxCR1 <SIOS> starts the transmission. The transmit data is moved from SBIxDBR to a shift register and output to the SO pin, with the least-significant bit (LSB) first, in synchronization with the serial clock. Once the transmit data is transferred to the shift register, SBIxDBR becomes empty, and the INTSBIx (buffer-empty) interrupt is generated, requesting the next transmit data.

In the internal clock mode, the serial clock will be stopped and automatically enter the wait state, if next data is not loaded after the 8-bit data has been fully transmitted. The wait state will be cleared when SBIxDBR is loaded with the next transmit data.

In the external clock mode, SBIxDBR must be loaded with data before the next data shift operation is started. Therefore, the data transfer rate varies depending on the maximum latency between when the interrupt request is generated and when SBIxDBR is loaded with data in the interrupt service program.

At the beginning of transmission, the same value as in the last bit of the previously transmitted data is output in a period from setting SBIxSR <SIOF> to “1” to the falling edge of SCK.

Transmission can be terminated by clearing <SIOS> to “0” or setting <SIOINH> to “1” in the INTSBIx interrupt service program. If <SIOS> is cleared, remaining data is output before transmission ends. The program checks SBIxSR <SIOF> to determine whether transmission has come to an end. <SIOF> is cleared to “0” at the end of transmission. If <SIOINH> is set to “1,” the transmission is aborted immediately and <SIOF> is cleared to “0”.

In the external clock mode, <SIOS> must be set to “0” before the next transmit data shift operation is started. Otherwise, operation will stop after dummy data is transmitted.

```
                7 6 5 4 3 2 1 0
SBIxCR1  ← 0 1 0 0 0 X X X      Selects the transmit mode.

SBIxDBR  ← X X X X X X X X      Writes the transmit data.
SBIxCR1  ← 1 0 0 0 0 X X X      Starts transmission.

INTSBIx interrupt
SBIxDBR  ← X X X X X X X X      Writes the transmit data.
```

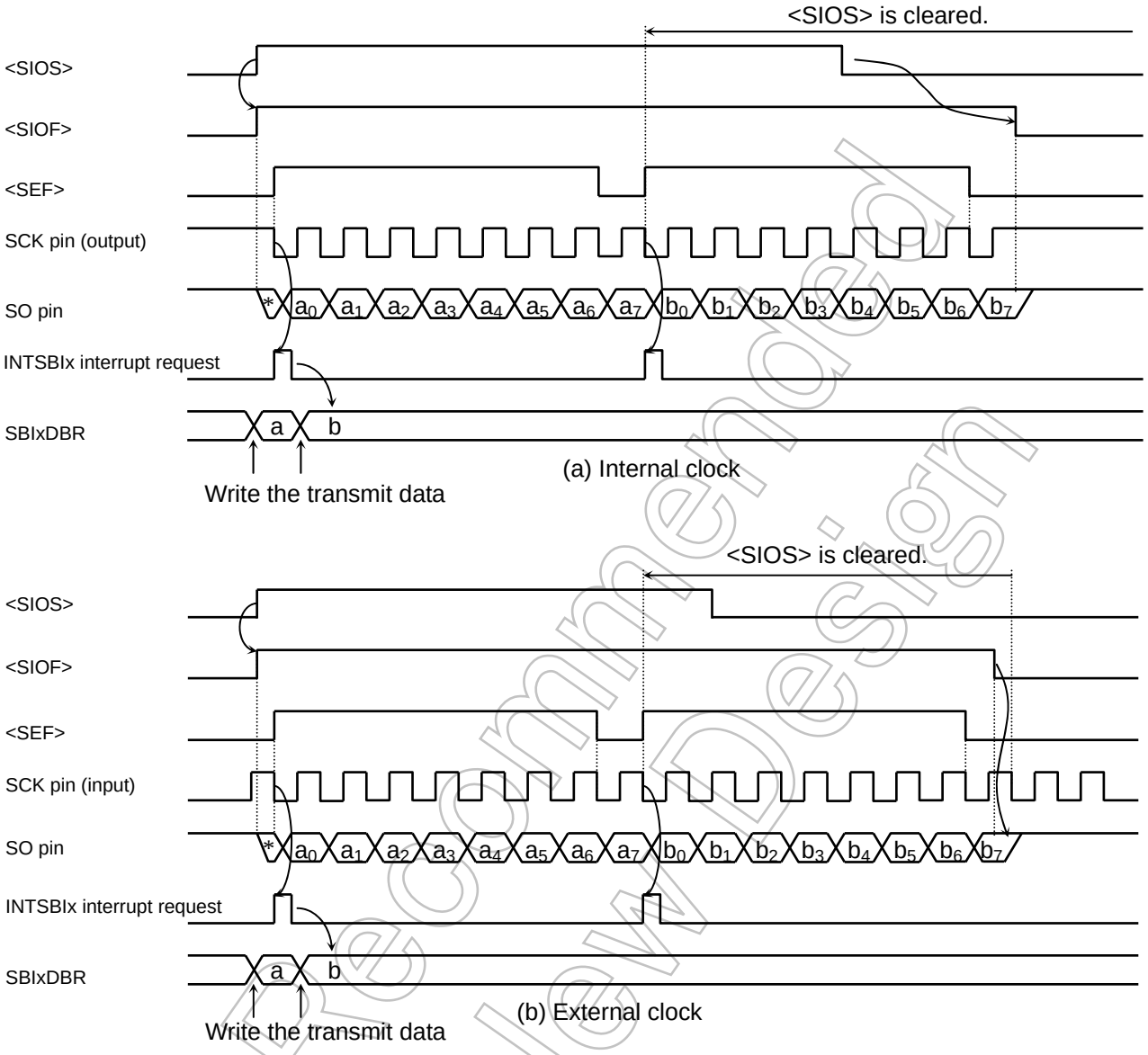


Fig. 15-27 Transmit Mode

Example: Example of programming (MIPS16) to terminate transmission by <SIO>

```
        ADDIU   r3,  r0,  0x04
STEST1  : LB    r2, (SBIxSR)          ; If SBIxSR<SEF> = 1 then loop
        AND     r2,  r3
        BNEZ    r2,  STEST1
        ADDIU   r3,  r0,  0x04
STEST2  : LB    r2, (PA)              ; If SCK = 0 then loop
        AND     r2,  r3
        BEQZ    r2,  STEST2
        ADDIU   r3,  r0,  0y00000111
        STB     r3, (SBIxCR1)         ; <SIOS> ← 0
```

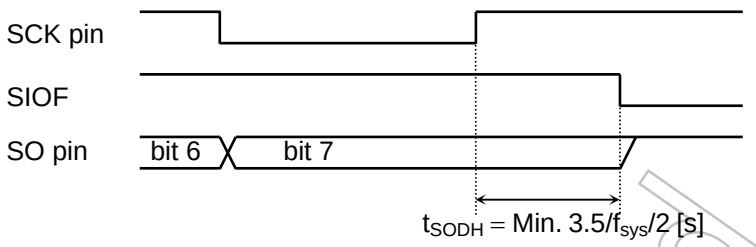


Fig. 15-28 Transmit Data Retention Time at the End of Transmission

② 8-bit receive mode

Set the control register to the receive mode. Then writing "1" to SBIxCR1 <SIOS> enables reception. Data is taken into the shift register from the SI pin, with the least-significant bit (LSB) first, in synchronization with the serial clock. Once the shift register is loaded with the 8-bit data, it transfers the received data to SBIxDBR and the INTSBIx (buffer-full) interrupt request is generated to request reading the received data. The interrupt service program then reads the received data from SBIxDBR.

In the internal clock mode, the serial clock will be stopped and automatically be in the wait state until the received data is read from SBIxDBR.

In the external clock mode, shift operations are executed in synchronization with the external clock. The maximum data transfer rate varies, depending on the maximum latency between generating the interrupt request and reading the received data.

Reception can be terminated by clearing <SIOS> to "0" or setting <SIOINH> to "1" in the INTSBIx interrupt service program. If <SIOS> is cleared, reception continues until all the bits of received data are written to SBIxDBR. The program checks SBISR <SIOF> to determine whether reception has come to an end. <SIOF> is cleared to "0" at the end of reception. After confirming the completion of the reception, last received data is read. If <SIOINH> is set to "1," the reception is aborted immediately and <SIOF> is cleared to "0." (The received data becomes invalid, and there is no need to read it out.)

(Note) The contents of SBIxDBR will not be retained after the transfer mode is changed. The ongoing reception must be completed by clearing <SIOS> to "0" and the last received data must be read before the transfer mode is changed.

```
                    7 6 5 4 3 2 1 0
SBIxCR1  ← 0 1 1 1 0 X X X     Selects the receive mode.

SBIxCR1  ← 1 0 1 1 0 0 0 0     Starts reception.
```

INTSBIx interrupt

```
Reg.     ← SBIxDBR             Reads the received data.
```

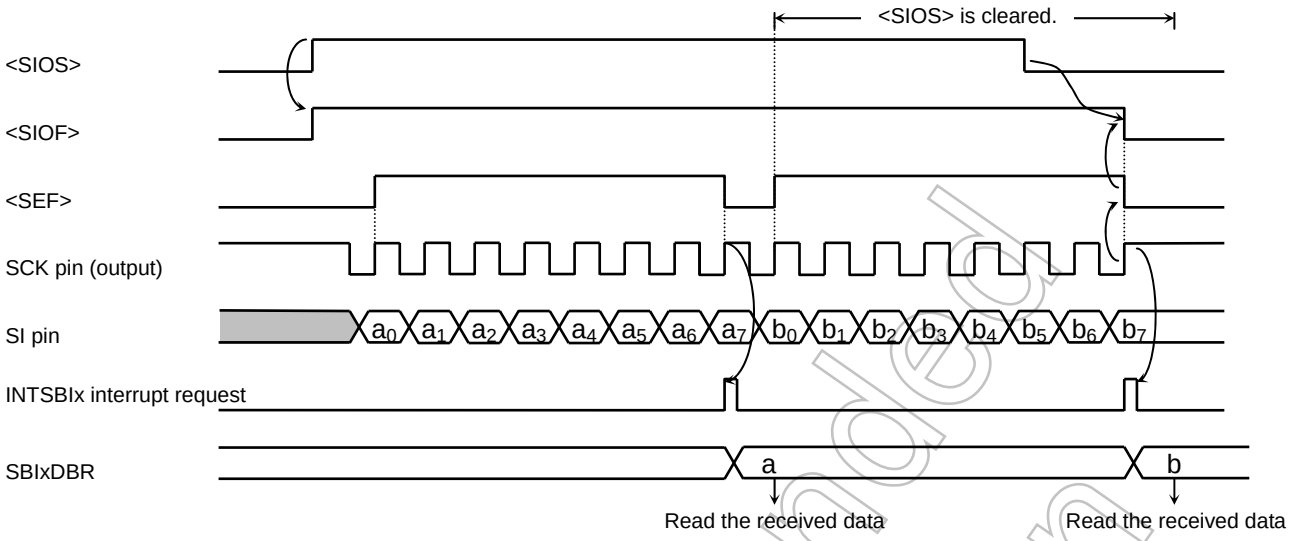


Fig. 15-29 Receive Mode (Example: Internal Clock)

③ 8-bit transmit/receive mode

Set the control register to the transfer/receive mode. Then writing the transmit data to SBIxDBR and setting SBIxCR1 <SIOS> to "1" enables transmission and reception. The transmit data is output through the SO pin at the falling of the serial clock, and the received data is taken in through the SI pin at the rising of the serial clock, with the least-significant bit (LSB) first. Once the shift register is loaded with the 8-bit data, it transfers the received data to SBIxDBR and the INTSBIx interrupt request is generated. The interrupt service program reads the received data from the data buffer register and writes the next transmit data. Because SBIxDBR is shared between transmit and receive operations, the received data must be read before the next transmit data is written.

In the internal clock operation, the serial clock will be automatically in the wait state until the received data is read and the next transmit data is written.

In the external clock mode, shift operations are executed in synchronization with the external serial clock. Therefore, the received data must be read and the next transmit data must be written before the next shift operation is started. The maximum data transfer rate for the external clock operation varies depending on the maximum latency between when the interrupt request is generated and when the transmit data is written.

At the beginning of transmission, the same value as in the last bit of the previously transmitted data is output in a period from setting <SIOF> to "1" to the falling edge of SCK.

Transmission and reception can be terminated by clearing <SIOS> to "0" or setting SBIxCR1 <SIOINH> to "1" in the INTSBIx interrupt service program. If <SIOS> is cleared, transmission and reception continue until the received data is fully transferred to SBIxDBR. The program checks SBISR <SIOF> to determine whether transmission and reception have come to an end. <SIOF> is cleared to "0" at the end of transmission and reception. If <SIOINH> is set, the transmission and reception are aborted immediately and <SIOF> is cleared to "0."

(Note) The contents of SBIxDBR will not be retained after the transfer mode is changed. The ongoing transmission and reception must be completed by clearing <SIOS> to "0" and the last received data must be read before the transfer mode is changed.

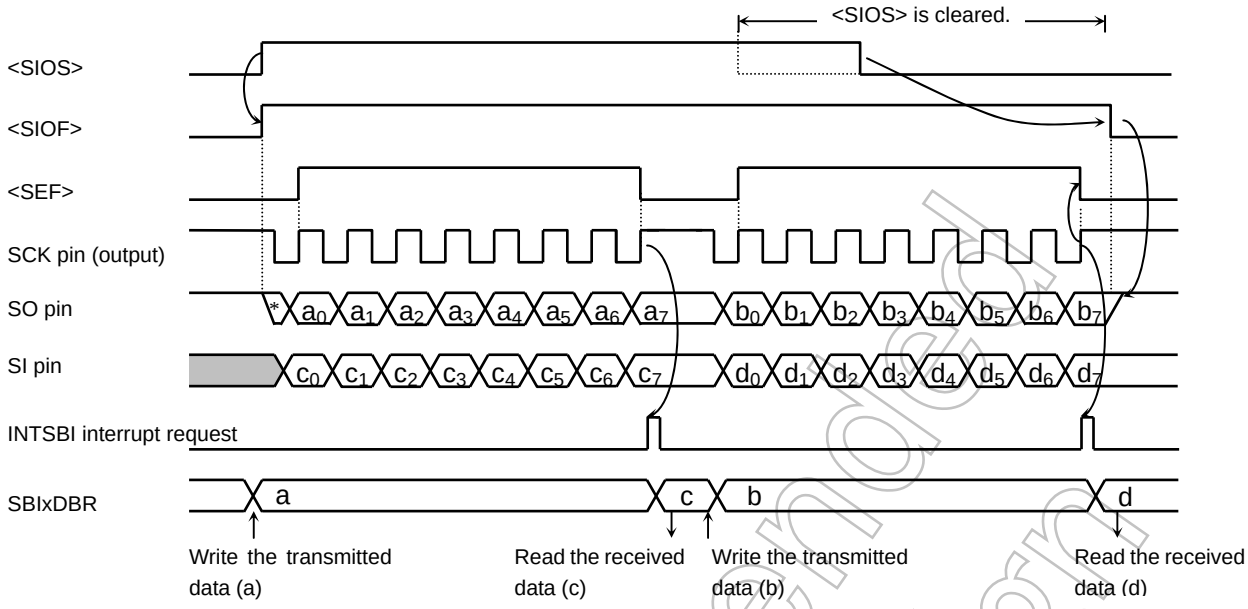


Fig. 15-30 Transmit/Receive Mode (Example: Internal Clock)

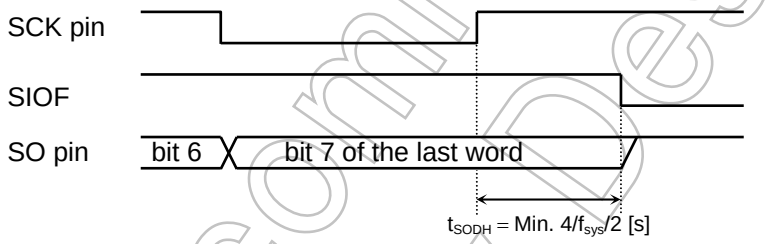


Fig. 15-31 Transmit Data Retention Time at the End of Transmission/Reception (In the Transmit/Receive Mode)

```
                    7 6 5 4 3 2 1 0
SBIxCR1  ← 0 1 1 0 0 X X X    Selects the transmit mode.

SBIxDBR  ← X X X X X X X X    Writes the transmit data.
SBIxCR1  ← 1 0 1 0 0 X X X    Starts reception/transmission.
```

INTSBIx interrupt

```
Reg.     ← SBIODBR            Reads the received data.
SBIxDBR  ← X X X X X X X X    Writes the transmit data.
```

16. Analog/Digital Converter

A 10-bit, sequential-conversion analog/digital converter (A/D converter) is built into the TMP19A23. This A/D converter is equipped with 13 analog input channels.

Fig. 16-1 shows the block diagram of this A/D converter.

These 13 analog input channels (pins AN0 through AN12) are also used as input/ output ports.

(Note) If it is necessary to reduce a power current by operating the TMP19A23 in IDLE or STOP mode and if either case shown below is applicable, you must first stop the A/D converter and then execute the instruction to put the TMP19A23 into standby mode:

1) The TMP19A23 must be put into IDLE mode when ADMOD1<I2AD> is "0."
2) The TMP19A23 must be put into STOP mode.

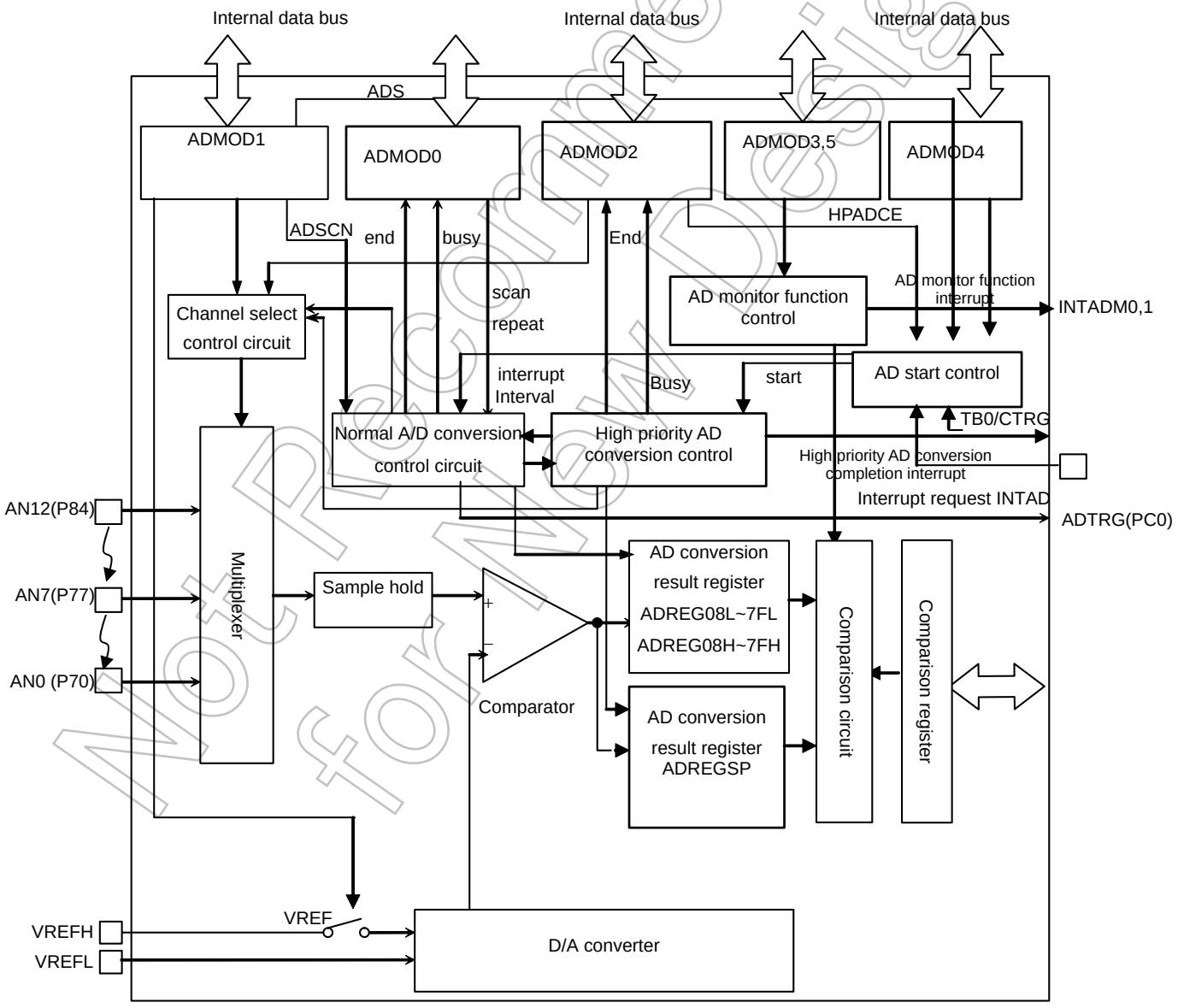


Fig. 16-1 A/D Converter Block Diagram

Note) The following bits must be set before the initiation of conversion from analog to digital to complete the conversion with accuracy.

0xFFFF_F819 = 0x58 (~40MHz)
0xFFFF_F819 = 0x40 (~54MHz)

ADCBAS (0xFFFF_F819)

	7	6	5	4	3	2	1	0
Bit symbol								
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
After reset	0	0	1	1	1	0	0	0
Function	Write "0".	Write "1".	Write "0".	Write "0" or "1".	Write "0" or "1".	Write "0".	Write "0".	"Write "0".

16.1 Control Register

The A/D converter is controlled by A/D mode control registers (ADMOD0, ADMOD1, ADMOD2, ADMOD3, ADMOD4 and ADMOD5). Results of A/D conversion are stored in 16 upper and lower A/D conversion result registers ADREG08H/L through ADREG7FH/L. Results of top-priority conversion are stored in ADREGSPH/L.

Fig. 16-2 shows the registers related to the A/D converter.

A/D Mode Control Register 0

ADMOD0 (0xFFFF_F814)

	7	6	5	4	3	2	1	0
Bit symbol	EOCFN	ADBFN		ITM1	ITM0	REPEAT	SCAN	ADS
Read/Write	R		R	R/W				
After reset	0	0	0	0	0	0	0	0
Function	Normal A/D conversion completion flag 0: Before or during conversion 1: Completion	Normal A/D conversion BUSY flag 0: Conversion stop 1: During conversion	"0" is read.	Specify interrupt in fixed channel repeat conversion mode	Specify interrupt in fixed channel repeat conversion mode.	Specify repeat mode 0: Single conversion mode 1: Repeat conversion mode	Specify scan mode 0: Fixed channel mode 1: Channel scan mode	Start A/D conversion 0: Don't care 1: Start conversion "0" is always read.

Specify A/D conversion interrupt in fixed channel repeat conversion mode

	Fixed channel repeat conversion mode <SCAN> = "0", <REPEAT> = "1"
00	Generate interrupt once every single conversion
01	Generate interrupt once every 4 conversions
10	Generate interrupt once every 8 conversions
11	Setting prohibited

Fig. 16-2 Registers related to the A/D Converter

A/D Mode Control Register 1

ADMOD1 (0xFFFF_ F815)

	7	6	5	4	3	2	1	0
Bit symbol	VREFON	I2AD	ADSCN	−	ADCH3	ADCH2	ADCH1	ADCH0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	VREF application control 0 :OFF 1 :ON	IDLE 0 : Stop 1 : Operation	Specify operation mode for channel scanning 0: 4ch scan 1: 8ch scan	Write "0."	Select analog input channel			

Select analog input channel

<SCAN> / <ADCH3.2, 1, 0>	0 Fixed channel	1 Channel scan (ADSCN=0)	1 Channel scan (ADSCN= 1)
0000	AN0	AN0	AN0
0001	AN1	AN0 to AN1	AN0 to AN1
0010	AN2	AN0 to AN2	AN0 to AN2
0011	AN3	AN0 to AN3	AN0 to AN3
0100	AN4	AN4	AN0 to AN4
0101	AN5	AN4 to AN5	AN0 to AN5
0110	AN6	AN4 to AN6	AN0 to AN6
0111	AN7	AN4 to AN7	AN0 to AN7
1000	AN8	AN 8	AN8
1001	AN9	AN8 to AN9	AN8 to AN9
1010	AN10	AN8 to AN10	AN8 to AN10
1011	AN11	AN8 to AN11	AN8 to AN11
1100	AN12	AN12	AN8 to AN12
1101	Setting prohibited		
1110			
1111			

(Note 1) Before starting AD conversion, write "1" to the <VREFON> bit, wait for 3 μs during which time the internal reference voltage should stabilize, and then write "1" to the ADMOD0<ADS> bit.

(Note 2) To go into standby mode upon completion of AD conversion, set <VREFON> to "0."

Fig. 16-3 A/D Mode Control Register 1

A/D Mode Control Register 2

ADMOD2 (0xFFFF_ F816)

	7	6	5	4	3	2	1	0
Bit symbol	EOCFHP	ADBFHP	HPADCE	−	HPADCH3	HPADCH2	HPADCH1	HPADCH0
Read/Write	R	R	R/W					
After reset	0	0	0	0	0	0	0	0
Function	Top-priority AD conversion completion flag 0: Before or during conversion 1: Upon completion	Top-priority AD conversion BUSY flag 0: During conversion halts 1: During conversion	Activate top-priority conversion 0: Don't care 1: Start conversion. "0" is always read.	Write "0".	Select analog input channel when activating top-priority conversion.			

<HPADCH4,3.2, 1, 0>	Analog input channel when executing top-priority conversion
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	Setting prohibited
1110	
1111	

Fig. 16-4 A/D Mode Control Register 2

A/D Mode Control Register 3

ADMOD3 (0xFFFF _F81E)

	7	6	5	4	3	2	1	0
Bit symbol			ADOBIC	REGS3	REGS2	REGS1	REGS0	ADOBSV
Read/Write	R/W	R	R/W					
After reset	0	0	0	0	0	0	0	0
Function	Write "0".	"0" can be read.	Make AD monitor function interrupt setting 0: Smaller than comparison Reg. 1: Larger than comparison Reg.	BIT for selecting the AD conversion result storage Reg. that is to be compared with the comparison Reg. if the AD monitor function is enabled				AD monitor function 0 : Disable 1 : Enable

<REGS.2, 1, 0>	AD conversion result storage Reg. to be compared
0000	ADREG08
0001	ADREG19
0010	ADREG2A
0011	ADREG3B
0100	ADREG4C
0101	ADREG5D
0110	ADREG6E

Fig. 16-5 A/D Mode Control Register 3

A/D Mode Control Register 5

ADMOD5 (0xFFFF _F81F)

	7	6	5	4	3	2	1	0
Bit symbol			ADOBIC	REGS3	REGS2	REGS1	REGS0	ADOBSV
Read/Write	R/W	R	R/W					
After reset	0	0	0	0	0	0	0	0
Function	rite "0".	" can be read.	Make AD monitor function interrupt setting Smaller than comparison Reg. Larger than comparison Reg.	BIT for selecting the AD conversion result storage Reg. that is to be compared with the comparison Reg. if the AD monitor function is enabled .				AD monitor function 0 : Disable 1 : Enable

<REGS.2, 1, 0>	AD conversion result storage Reg. to be compared
0000	ADREG08
0001	ADREG19
0010	ADREG2A
0011	ADREG3B
0100	ADREG4C
0101	ADREG5D
0110	ADREG6E
0111	ADREG7F

Fig. 16-6 A/D Mode Control Register 5

A/D Mode Control Register 4

ADMOD4 (0xFFFF_F818)

	7	6	5	4	3	2	1	0
Bit symbol	HADHS	HADHTG	ADHS	ADHTG			ADRST1	ADRST0
Read/Write	R/W				R		W	W
After reset	0	0	0	0	0		–	–
Function	HW source for activating top-priority A/D conversion 0: External TRG 1:Match with TB2RG0	HW for activating top-priority A/D conversion 0: Disable 1: Enable	HW source for activating normal A/D conversion 0: External TRG 1:Match with TB1RG0	HW for activating normal A/D conversion 0: Disable 1: Enable	"0" can be read.		Overwriting 10 with 01 allows ADC to be software reset.	

(Note 1) If AD conversion is executed with the match triggers <ADHTG> and <HADHTG> of a 16-bit timer set to "1" by using a source for triggering H/W, A/D conversion can be activated at specified intervals by performing three steps shown below when the timer is idle:

1. **Select a source for triggering HW: <ADHS>, <HADHS>**
2. **Enable H/W activation of AD conversion: <ADHTG>, <HADHTG>**
3. **Start the timer.**

(Note 2) Do not make a top-priority AD conversion setting and a normal AD conversion setting simultaneously.

(Note 3) The external trigger cannot be used for H/W activation of AD conversion when it is used for H/W activation of top priority AD conversion.

Fig. 16-7 A/D Mode Control Register 4

Lower A/D Conversion Result Register 08

ADREG08L (0xFFFF_F800)

	7	6	5	4	3	2	1	0
Bit symbol	ADR01	ADR00					OVR0	ADR0RF
Read/Write	R		R				R	R
After reset	0		1				0	0
Function	Store lower 2 bits of A/D conversion result		"1" can be read.				Over RUN flag 0: Not generate 1: Generate	A/D conversion result storage flag 1:Presence of conversion result

Upper A/D Conversion Result Register 08

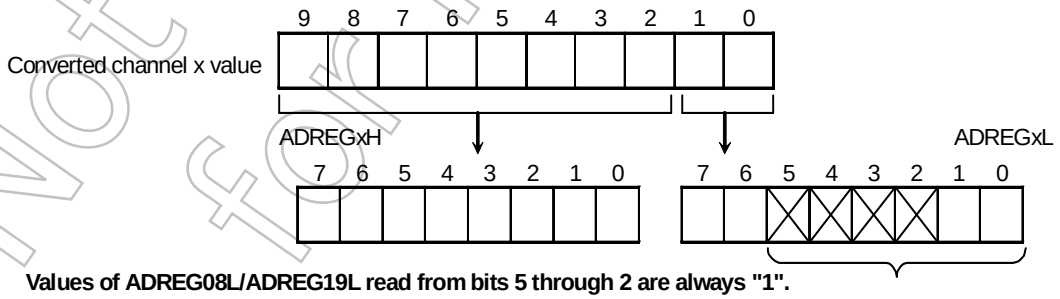
	7	6	5	4	3	2	1	0
Bit symbol	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							

Lower A/D Conversion Result Register 19

	7	6	5	4	3	2	1	0
Bit symbol	ADR11	ADR10					OVR1	ADR1RF
Read/Write	R		R				R	R
After reset	0		1				0	0
Function	Store lower 2 bits of A/D conversion result		"1" can be read.				Over RUNflag 0: Not generate 1: Generate	A/D conversion result storage flag 1:Presence of conversion result

Upper A/D Conversion Result Register 19

	7	6	5	4	3	2	1	0
Bit symbol	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							



- **Values of ADREG08L/ADREG19L read from bits 5 through 2 are always "1".**
- **Bit 0 of ADREG08L/ADREG19L is the A/D conversion result storage flag <ADRxRF>. This bit is set to "1" after an A/D converted value is stored. A read of a lower register (ADREGxL) will set this bit to "0".**
- **Bit 1 of ADREG08L/ADREG19L is the over RUN flag <OVRx>. This bit is set to "1" if a conversion result is overwritten before both conversion result storage registers (ADREGxH and ADREGxL) are read. A read of a flag will clear this bit to "0."**
- **To read from conversion result storage registers, first read upper registers and then lower registers.**

Fig. 16-8 A/D Conversion Result Register

Lower A/D Conversion Result Register 2A

ADREG2AL (0xFFFF_F804)

	7	6	5	4	3	2	1	0
Bit symbol	ADR21	ADR20					OVR2	ADR2RF
Read/Write	R		R				R	R
After reset	0		1				0	0
Function	Store lower 2 bits of A/D conversion result		"1" is read.				Over RUN flag 0:Not generate 1: Generate	A/D conversion result storage flag 1: Presence of conversion result

Upper A/D Conversion Result Register 2A

ADREG2AH (0xFFFF_F805)

	7	6	5	4	3	2	1	0
bit Symbol	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							

Lower A/D Conversion Result Register 3B

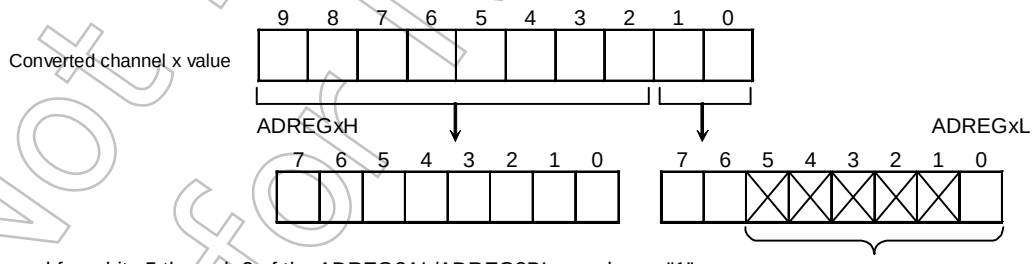
ADREG3BL (0xFFFF_F806)

	7	6	5	4	3	2	1	0
bit Symbol	ADR31	ADR30					OVR3	ADR3RF
Read/Write	R		R				R	R
After reset	0		1				0	0
Function	Store lower 2 bits of A/D conversion result		"1" is read.				Over RUN flag 0:Not generate 1: Generate	A/D conversion result storage flag 1: Presence of conversion result

Upper A/D Conversion Result Register 3B

ADREG3BH (0xFFFF_F807)

	7	6	5	4	3	2	1	0
bit Symbol	ADR39	ADR38	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							



- Values read from bits 5 through 2 of the ADREG2AL/ADREG3BL are always "1".
- Bit 0 of the ADREG2AL/ADREG3BL is the A/D conversion result storage flag <ADRxRF>. It is set to "1" after an A/D converted value is stored. A read of a lower register (ADREGxL) will set this bit to "0".
- Bit 1 of the ADREG2AL/ADREG3BL is the over RUN flag <OVRx>. It is set to "1" if a conversion result is overwritten before both conversion result storage registers (ADREGxH,ADREGxL) are read. A read of a flag will clear this bit to "0.
- When reading conversion result storage registers, first read upper registers and then read lower registers.

Fig. 16-9 A/D Conversion Result Register

Lower A/D Conversion Result Register 4C

ADREG 4 CL (0xFFFF_F808)

	7	6	5	4	3	2	1	0
Bit symbol	ADR41	ADR40					OVR4	ADR4RF
Read/Write	R		R				R	R
After reset	0		1				0	0
Function	Store lower 2 bits of A/D conversion result		"1" is read.				Over RUN flag 0: Not generate 1: Generate	A/D conversion result storage flag 1: Presence of conversion result

Upper A/D Conversion Result Register 4C

ADREG 4 CL (0xFFFF_F808)

	7	6	5	4	3	2	1	0
Bit symbol	ADR49	ADR48	ADR47	ADR46	ADR45	ADR44	ADR43	ADR42
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							

Lower A/D Conversion Result Register 5D

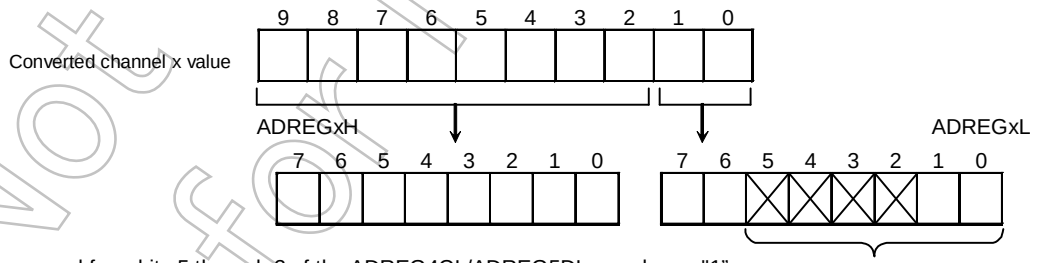
ADREG 4 CL (0xFFFF_F808)

	7	6	5	4	3	2	1	0
Bit symbol	ADR51	ADR50					OVR5	ADR5RF
Read/Write	R		R				R	R
After reset	0		1				0	0
Function	Store lower 2 bits of A/D conversion result		"1" is read.				Over RUN flag 0: Not generate 1: Generate	A/D conversion result storage flag Presence of conversion result

Upper A/D Conversion Result Register 5D

ADREG 4 CL (0xFFFF_F808)

	7	6	5	4	3	2	1	0
Bit symbol	ADR59	ADR58	ADR57	ADR56	ADR55	ADR54	ADR53	ADR52
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							



- Values read from bits 5 through 2 of the ADREG4CL/ADREG5DL are always "1".
- Bit 0 of the ADREG4CL/ADREG5DL is the A/D conversion result storage flag <ADRxRF>. It is set to "1" after an A/D converted value is stored. A read of a lower register (ADREGxL) will set this bit to "0".
- Bit 1 of the ADREG4CL/ADREG5DL is the over Run flag <OVRx>. It is set to "1" if a conversion result is overwritten before both conversion result storage registers (ADREGxH and ADREGxL) are read. A read of a flag will clear this bit to "0".
- When reading conversion result storage registers, first read upper registers and then read lower registers.

Fig. 16-10 A/D Conversion Result Register

Lower A/D Conversion Result Register 6E

ADREG6EL (0xFFFF_F80C)

	7	6	5	4	3	2	1	0
Bit Symbol	ADR61	ADR60					OVR6	ADR6RF
Read/Write	R		R				R	R
After reset	0		1				0	0
Function	Store lower 2 bits of A/D conversion result		"1" is read.				Over RUN flag 0: Not generate 1: Generate	A/D conversion result storage flag 1: Presence of conversion result

Upper A/D Conversion Result Register 6E

ADREG6EL (0xFFFF_F80C)

	7	6	5	4	3	2	1	0
Bit Symbol	ADR69	ADR68	ADR67	ADR66	ADR65	ADR64	ADR63	ADR62
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							

Lower A/D Conversion Result Register 7F

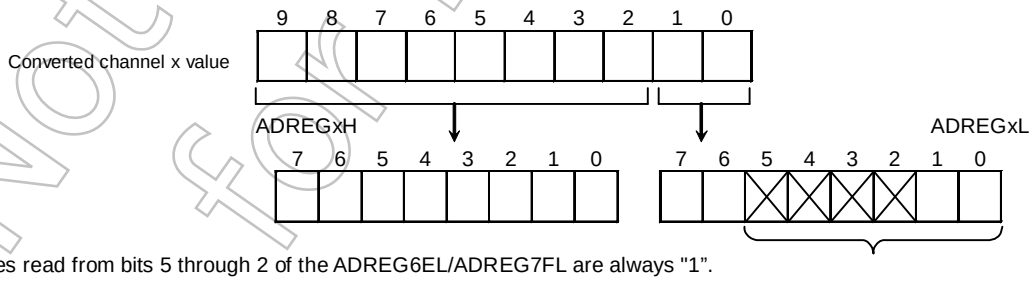
ADREG6EL (0xFFFF_F80C)

	7	6	5	4	3	2	1	0
Bit Symbol	ADR71	ADR70					OVR7	ADR7RF
Read/Write	R		R				R	R
After reset	0		1				0	0
Function	Store lower 2 bits of A/D conversion result		"1" is read.				Over RUN flag 0: Not generate 1: Generate	A/D conversion result storage flag 1: Presence of conversion result

Upper A/D Conversion Result Register 7F

ADREG6EL (0xFFFF_F80C)

	7	6	5	4	3	2	1	0
Bit Symbol	ADR79	ADR78	ADR77	ADR76	ADR75	ADR74	ADR73	ADR72
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							



- Values read from bits 5 through 2 of the ADREG6EL/ADREG7FL are always "1".
- Bit 0 of the ADREG6EL/ADREG7FL is the A/D conversion result storage flag <ADRxRF>. It is set to "1" if an A/D converted value is stored. A read of a lower register (ADREGxL) will set this bit to "0".
- Bit 1 of the ADREG6EL/ADREG7FL is the over Run flag <OVRx>. It is set to "1" if a conversion result is overwritten before both conversion result storage registers (ADREGxH and ADREGxL) are read. A read of a flag will clear this bit to "0".
- When reading conversion result storage registers, first read upper registers and then read lower registers.

Fig. 16-11 A/D Conversion Result Register

Lower A/D Conversion Result Register SP

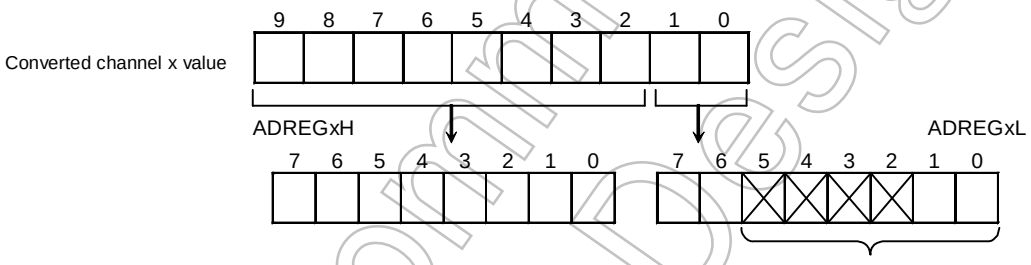
ADREGSPL (0xFFFF_F810)

	7	6	5	4	3	2	1	0
Bit symbol	ADRSP1	ADRSP0					OVRSP	ADRSPRF
Read/Write	R		R				R	R
After reset	0		1				0	0
Function	Store lower 2 bits of A/D conversion result		"1" is read.				Over RUN flag 0: Not generate 1: Generate	A/D conversion result storage flag 1: Presence of conversion result

Upper A/D Conversion Result Register SP

ADREGSPL (0xFFFF_F810)

	7	6	5	4	3	2	1	0
Bit symbol	ADRSP9	ADRSP8	ADRSP7	ADRSP6	ADRSP5	ADRSP4	ADRSP3	ADRSP2
Read/Write	R							
After reset	0							
Function	Store upper 8 bits of A/D conversion result							



- Values read from bits 5 through 2 of the ADREGSP are always "1".
- Bit 0 of the ADREGSP is the A/D conversion result storage flag <ADRxRF>. It is set to "1" after an A/D converted value is stored. A read of a lower register (ADREGxL) will set this bit to "0."
- Bit 1 of the ADREGSP is the over RUN flag <OVRx>. It is set to "1" if a conversion result is overwritten before both conversion result storage registers (ADREGxH and ADREGxL) are read. A read of a flag will clear this bit to "0".
- When reading conversion result storage registers, first read upper registers and then read lower registers.

Fig. 16-12 A/D Conversion Result Register

Lower A/D Conversion Result Comparison Register 0

ADCMP0L (0xFFFF_F820)

	7	6	5	4	3	2	1	0
Bit symbol	ADR021	ADR020						
Read/Write	R/W		R					
After reset	0		0					
Function	Store lower 2 bits of A/D conversion result comparison		"0" is read.					

Upper A/D Conversion Result Comparison Register 0

ADCMP0H (0xFFFF_F821)

	7	6	5	4	3	2	1	0
Bit symbol	ADR029	ADR028	ADR027	ADR026	ADR025	ADR024	ADR023	ADR022
Read/Write	R/W							
After reset	0							
Function	Store upper 8 bits of A/D conversion result comparison							

Lower A/D Conversion Result Comparison Register 1

ADCMP1L (0xFFFF_F822)

	7	6	5	4	3	2	1	0
Bit symbol	ADR121	ADR120						
Read/Write	R/W		R					
After reset	0		0					
Function	Store lower 2 bits of A/D conversion result comparison		"0" is read.					

Upper A/D Conversion Result Comparison Register 1

ADCMP1H (0xFFFF_F823)

	7	6	5	4	3	2	1	0
Bit symbol	ADR129	ADR128	ADR127	ADR126	ADR125	ADR124	ADR123	ADR122
Read/Write	R/W							
After reset	0							
Function	Store upper 8 bits of A/D conversion result comparison							

(Note) To set or change a value in this register, the AD monitor function must be disabled (ADCMPx<ADOBSVx>="0").

Fig. 16-13 A/D Conversion Result Register

16.2 Conversion Clock

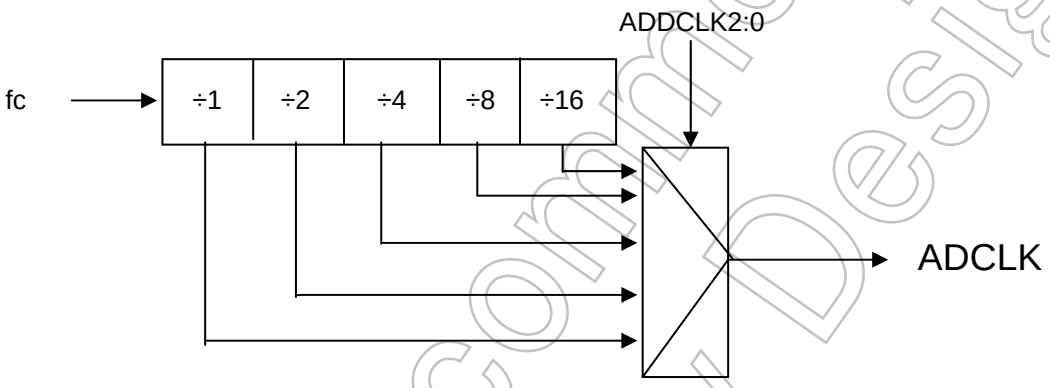
- The conversion time is calculated by the 46 conversion clock.

A/D Conversion Clock Setting Register

ADCLK (0xFFFF_F81C)

	7	6	5	4	3	2	1	0
Bit symbol	TSH3	tSH2	tSH1	tSH0		ADCLK2	ADCLK1	ADCLK0
Read/Write	R/W	R/W	R/W	R/W	R	R/W	R/W	R/W
After reset	1	0	0	0	0	0	0	0
Function	Select the A/D sample hold time 1000: 8 conversion clock 1001: 16 conversion clock 1010: 24 conversion clock 1011: 32 conversion clock 0011: 64 conversion clock 1100: 128 conversion clock 1101: 512 conversion clock The setup other than those above: reserved				"0" is read.	Select the A/D prescaler output 000: fc 001: fc／2 010: fc／4 011: fc／8 100: fc／16 111:reserved		

Fig. 16-16 A/D Conversion Clock Setting Register



Example: If fsys = fc = 40 MHz/ 54MHz

Prescaler [ADDCLK2:0]	tconv.(conversion time) 40MHz	tconv.（conversion time） 54MHz
1	1.15μs	-
1/2	2.3μs	1.22μs (temporary)
1/4	4.6μs	2.44μs（temporary）

Variable S/H time

Conversion clock	S/H time 40MHz	tconv.（conversion time）
40MHz	Conversion clk*8 （0.2 us）	1.15μs
	Conversion clk*16 （0.4 us）	1.35μs
	Conversion clk*24 （0.6 us）	1.55μs
	Conversion clk*32 （0.8 us）	1.75μs
	Conversion clk*64 （1.6 us）	2.55μs
	Conversion clk*128 （3.2 us）	4.15μs
	Conversion clk*512 （12.8 us）	13.75μs

(Note):Please do not change the analog to digital conversion clock setting during the analog to digital translation.

Fig. 16-17 A/D Conversion Time

16.3 Description of Operations

16.3.1 Analog Reference Voltage

The "H" level of the analog reference voltage shall be applied to the VREFH pin, and the "L" level shall be applied to the VREFL pin. By writing "0" to the ADMOD1<VREFON> bit, a switched-on state of VREFH-VREFL can be turned into a switched-off state. To start AD conversion, make sure that you first write "1" to the <VREFON> bit, wait for 3 μs during which time the internal reference voltage should stabilize, and then write "1" to the ADMOD0<ADS> bit.

16.3.2 Selecting the Analog Input Channel

How the analog input channel is selected is different depending on A/D converter operation mode used.

(1) Normal AD conversion mode

- If the analog input channel is used in a fixed state (ADMOD0<SCAN>="0"):

One channel is selected from analog input pins AIN0 through AIN12 by setting ADMOD1<ADCH3 to 0> to an appropriate setting.

- If the analog input channel is used in a scan state (ADMOD0<SCAN>="1"):

One scan mode is selected from 13 scan modes by setting ADMOD1<ADCH3 to 0> and ADSCN to appropriate settings.

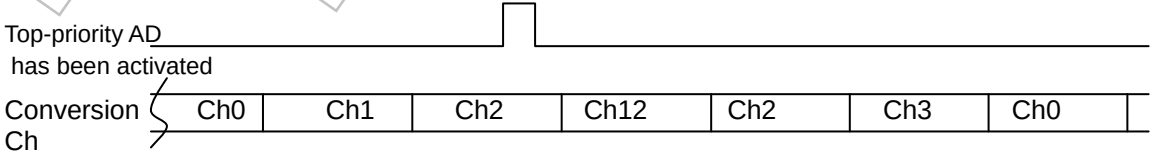
（2）Top-priority AD conversion mode

One channel is selected from analog input pins AIN0 through AIN12 by setting ADMOD2<HPADCH3 to 0> to an appropriate setting.

After a reset, ADMOD0<SCAN> is initialized to "0" and ADMOD1<ADCH3:0> is initialized to "0000." This initialization works as a trigger to select a fixed channel input through the AN0 pin. The pins that are not used as analog input channels can be used as ordinary input ports.

If top-priority AD conversion is activated during normal AD conversion, normal AD conversion is discontinued, top-priority AD conversion is executed and completed, and then normal AD conversion is resumed.

Example: A case in which repeat-scan conversion is ongoing at channels AIN0 through AIN3 with ADMOD0<REPEAT:SCAN> set to "11" and ADMOD1<ADCH3:0> set to 0011, and top-priority AD conversion has been activated at AIN12 with ADMOD2<HPADCH3:0>=1100.



16.3.3 Starting A/D Conversion

Two types of A/D conversion are supported: normal AD conversion and top-priority AD conversion. Normal AD conversion is software activated by setting ADMOD0<ADS> to "1." Top-priority AD conversion is software activated by setting ADMOD2<HPADCE> to "1." 4 operation modes are made available to normal AD conversion. In performing normal AD conversion, one of these operation modes must be selected by setting ADMOD0<2:1> to an appropriate setting. For top-priority AD conversion, only one operation mode can be used: fixed channel single conversion mode. Normal AD conversion can be activated using the HW activation source selected by ADMOD3<ADHS>, and top-priority AD conversion can be activated using the HW activation source selected by ADMOD3<HADHS>. If this bit is "0," normal and top-priority AD conversions are activated in response to the input of a falling edge through the $\overline{\text{ADTRG}}$ pin. If this bit is "1," normal AD conversion is activated in response to TB1RG0 generated by the 16-bit timer 1, and top-priority AD conversion is activated in response to TB2RG0 generated by the 16-bit timer 9. Software activation is still valid even after H/W activation has been authorized.

(Note) When an external trigger is used for the HW activation source of a top priority analog to digital translation, an external trigger cannot usually be set as analog to digital translation HW activation.

When normal A/D conversion starts, the A/D conversion Busy flag (ADMOD0<ADBF>) showing that A/D conversion is under way is set to "1." When top-priority A/D conversion starts, the A/D conversion Busy flag (ADMOD2<ADBFHP>) showing that A/D conversion is under way is set to "1." At that time, the value of the Busy flag for normal A/D conversion before the start of top-priority A/D conversion is retained. The value of the conversion completion flag EOCFN for normal A/D conversion before the start of top-priority A/D conversion can also be retained.

(Note) Normal A/D conversion must not be activated when top-priority A/D conversion is under way. In that case, the top-priority A/D conversion completion flag cannot be set, and the flag for previous normal A/D conversion cannot be cleared.

To reactivate normal A/D conversion, a software reset (ADMOD4<ADRST1:0>) must be performed before starting A/D conversion. The HW activation method must not be used to reactivate normal A/D conversion.

If ADMOD2<HPADCE> is set to "1" during normal A/D conversion, ongoing A/D conversion is discontinued and top-priority A/D conversion starts; specifically, A/D conversion (fixed channel single conversion) is executed for a channel designated by ADMOD2<3:0>. After the result of this top-priority A/D conversion is stored in the storage register ADREGSP, normal A/D conversion is resumed.

If HW activation of top-priority A/D conversion is authorized during normal A/D conversion, ongoing A/D conversion is discontinued when requirements for activation using a resource are met, and top-priority A/D conversion (fixed channel single conversion) starts for a channel designated by ADMOD2<3:0>. After the result of this top-priority A/D conversion is stored in the storage register ADREGSP, normal A/D conversion is resumed.

16.3.4 A/D Conversion Modes and A/D Conversion Completion Interrupts

For A/D conversion, the following four operation modes are supported. For normal A/D conversion, an operation mode can be selected by setting ADMOD0<2:1> to an appropriate setting. For top-priority A/D conversion, the fixed channel single conversion mode is automatically selected, irrespective of the ADMOD0<2:1> setting.

- Fixed channel single conversion mode
- Channel scan single conversion mode
- Fixed channel repeat conversion mode
- Channel scan repeat conversion mode

（1）Normal A/D conversion

An operation mode is selected with ADMOD0<REPEAT, SCAN>. As A/D conversion starts, ADMOD0<ADBFN> is set to "1." When specified A/D conversion is completed, the A/D conversion completion interrupt (INTAD) is generated, and ADMOD0<EOCF> showing the completion of A/D conversion is set to "1." If <REPEAT>="0," <ADBFN> returns to "0" concurrently with the setting of EOCF. If <REPEAT> is set to "1," <ADBFN> remains at "1" and A/D conversion continues.

1. Fixed channel single conversion mode

   If ADMOD0 <REPEAT, SCAN> is set to "00," A/D conversion is performed in the fixed channel single conversion mode.

   In this mode, A/D conversion is performed once for one channel selected. After A/D conversion is completed, ADMOD0<EOCF> is set to "1," ADMOD0<ADBF> is cleared to "0," and the interrupt request INTAD is generated. <EOCF> is cleared to "0" upon read.

2. Channel scan single conversion mode

   If ADMOD0 <REPET,SCAN> is set to "01," A/D conversion is performed in the channel scan single conversion mode.

   In this mode, A/D conversion is performed once for each scan channel selected. After A/D scan conversion is completed, ADMOD0<EOCF> is set to "1," ADMOD0<ADBF> is cleared to "0," and the interrupt request INTAD is generated. <EOCF> is cleared to "0" upon read.

3. Fixed channel repeat conversion mode

   If ADMOD0<REPEAT,SCAN> is set to "10," A/D conversion is performed in fixed channel repeat conversion mode.

   In this mode, A/D conversion is performed repeatedly for one channel selected. After A/D conversion is completed, ADMOD <EOCF> is set to "1." ADMOD0 <ADBF> is not cleared to "0." It remains at "1." The timing with which the interrupt request INTAD is generated can be selected by setting ADMOD0 <ITM1:0> to an appropriate setting. <EOCF> is set with the same timing as this interrupt INTAD is generated. I

   <EOCF> is cleared to "0" upon read.

   With <ITM1:0> set to "00," an interrupt request is generated each time one A/D conversion is completed. In this case, the conversion results are always stored in the

storage register ADREG08. After the conversion result is stored, EOCF changes to "1."

With <ITM1:0> set to "01," an interrupt request is generated each time four A/D conversion are completed. In this case, the conversion results are sequentially stored in storage registers ADREG08 through ADREG3B. After the conversion results are stored in ADREG3B, <EOCF> is set to "1," and the storage of subsequent conversion results starts from ADREG08. <EOCF> is cleared to "0" upon read.

With <ITM1:0> set to "10," an interrupt request is generated each time eight A/D conversions are completed. In this case, the conversion results are sequentially stored in storage registers ADREG08 through ADREG7F. After the conversion results are stored in ADREG7F, <EOCF> is set to "1," and the storage of subsequent conversion results starts from ADREG08.

<EOCF> is cleared to "0" upon read.

4. Channel scan repeat conversion mode

If ADMOD0 <REPEAT, SCAN> is set to "11," A/D conversion is performed in the channel scan repeat conversion mode.

In this mode, A/D conversion is performed repeatedly for a scan channel selected. Each time one A/D scan conversion is completed, ADMOD0 <EOCF> is set to "1," and the interrupt request INTAD is generated. ADMOD0 <ADBF> is not cleared to "0." It remains at "1." <EOCF> is cleared to "0" upon read.

To stop the A/D conversion operation in the repeat conversion mode (modes described in 3. and 4. above), write "0" to ADMOD0 <REPEAT>. When ongoing A/D conversion is completed, the repeat conversion mode terminates, and ADMOD0 <ADBF> is set to "0."

Before switching from one mode to standby mode (such standby modes as IDLE, STOP, etc.), check that A/D conversion is not being executed. If A/D conversion is under way, you must stop it or wait until it is completed.

（2）Top-priority A/D conversion

Top-priority A/D conversion is performed only in fixed channel single conversion mode. The ADMOD0<REPEAT, SCAN> setting has no relevance to the top-priority A/D conversion operations or preparations. As activation requirements are met, A/D conversion is performed only once for a channel designated by ADMOD2<HPADCH3:0>. After the A/D conversion is completed, the top-priority A/D conversion completion interrupt is generated, ADMOD2<EOCFHP> is set to "1," and <ADBFHP> returns to "0." The EOCFHP Flag is cleared upon read.

Table 16-18 Relationships among A/D Conversion Modes, Interrupt Generation Timings and Flag Operations

Conversion mode	Interrupt generation timing	EOCF setting timing (see Note)	ADBF (after the interrupt is generated)	ADMOD0		
				ITM1:0	REPEAT	SCAN
Fixed channel single conversion	After conversion is completed	After conversion is completed	0	——	0	0
Fixed channel repeat conversion	Each time one conversion is completed	After one conversion is completed	1	00	1	0
	Each time four conversions are completed	After four conversions are completed	1	01		
	Each time eight conversions are completed	After eight conversions are completed	1	10		
Channel scan single conversion	After scan conversion is completed	After scan conversion is completed	0	——	0	1
Channel scan repeat conversion	Each time one scan conversion is completed	After one scan conversion is completed	1	——	1	1

(Note) EOCF is cleared upon read.

16.3.5 High-priority Conversion Mode

By interrupting ongoing normal A/D conversion, top-priority A/D conversion can be performed. Top-priority A/D conversion can be software activated by setting ADMOD2<HPADCE> to "1" or it can be activated using the HW resource by setting ADMOD3<7:6> to an appropriate setting. If top-priority A/D conversion has been activated during normal A/D conversion, ongoing normal A/D conversion is interrupted, and single conversion is performed for a channel designated by ADMOD2<3:0>. The result of single conversion is stored in ADREGSP, and the top-priority A/D conversion interrupt is generated. After top-priority A/D conversion is completed, normal A/D conversion is resumed; the status of normal A/D conversion immediately before being interrupted is maintained. Top-priority A/D conversion activated while top-priority A/D conversion is under way is ignored.

For example, if channel repeat conversion is activated for channels AN0 through AN8 and if <HPADCE> is set to "1" during AN3 conversion, AN3 conversion is suspended, and conversion is performed for a channel designated by <HPADC3:0>. After the result of conversion is stored in ADREGSP, channel repeat conversion is resumed, starting from AN3.

16.3.6 A/D Monitor Function

If ADCMPx<ADOBSVx> is set to "1," the A/D monitor function is enabled. If the value of the conversion result storage register specified by REGS<3:0> becomes larger or smaller ("larger" or "smaller" to be designated by ADOBIC) than the value of a comparison register, the A/D monitor function interrupt is generated. This comparison operation is performed each time a result is stored in a corresponding conversion result storage register, and the interrupt is generated if the conditions are met. Because storage registers assigned to perform the A/D monitor function are usually not read by software, overrun flag <OVRn> is always set and the conversion result storage flag <ADRnRF> is also set. To use the A/D monitor function, therefore, a flag of a corresponding conversion result storage register must not be used.

16.3.7 Storing and Reading A/D Conversion Results

A/D conversion results are stored in upper and lower A/D conversion result registers for normal A/D conversion (ADREG08H/L through ADRG7FH/L).

In fixed channel repeat conversion mode, A/D conversion results are sequentially stored in ADREG08H/L through ADREG7FH/L. If <ITM1:0> is so set as to generate the interrupt each time one A/D conversion is completed, conversion results are stored only in ADREG08H/L. If <ITM1:0> is so set as to generate the interrupt each time four A/D conversions are completed, conversion results are sequentially stored in ADREG08H/L through ADREG3BH/L.

Table 16. 19 shows analog input channels and related A/D conversion result registers.

Table 16-19 Analog Input Channels and Related A/D Conversion Result Registers

Analog input channel (port A)	A/D conversion result register			
	Conversion modes other than shown to the right	Fixed channel repeat conversion mode (every one conversion)	Fixed channel repeat conversion mode (every four conversions)	Fixed channel repeat conversion mode (every eight conversions)
AN0	ADREG08H/L	ADREG08H/L fixed	ADREG08H/L → ADREG3BH/L	ADREG08H/L → ADREG4CH/L
AN1	ADREG19H/L			
AN2	ADREG2AH/L			
AN3	ADREG3BH/L			
AN4	ADREG4CH/L			
AN5	ADREG5DH/L			
AN6	ADREG6EH/L			
AN7	ADREG7FH/L			
AN8	ADREG08H/L			
AN9	ADREG19H/L			
AN10	ADREG2AH/L			
AN11	ADREG3BH/L			
AN12	ADREG4CH/L			

16.3.8 Data Polling

To process A/D conversion results without using interrupts, ADMOD0<EOCF> must be polled. If this flag is set, conversion results are stored in a specified A/D conversion result register. After confirming that this flag is set, read that conversion result storage register. In reading the register, make sure that you first read upper bits and then lower bits to detect an overrun. If OVRn is "0" and ADRnRF is "1" in lower bits, a correct conversion result has been obtained.

17. Watchdog Timer (Runaway Detection Timer)

The TMP19A23 has a built-in watchdog timer for detecting runaways.

The watchdog timer (WDT) is for detecting malfunctions (runaways) of the CPU caused by noises or other disturbances and remedying them to return the CPU to normal operation. If the timer detects a runaway, it generates a non-maskable interrupt to notify the CPU.

By connecting the output of the watchdog timer to a reset pin (inside the chip), it is possible to force the watchdog timer to reset itself.

17.1 Configuration

Fig. 17-1 shows the block diagram of the watchdog timer

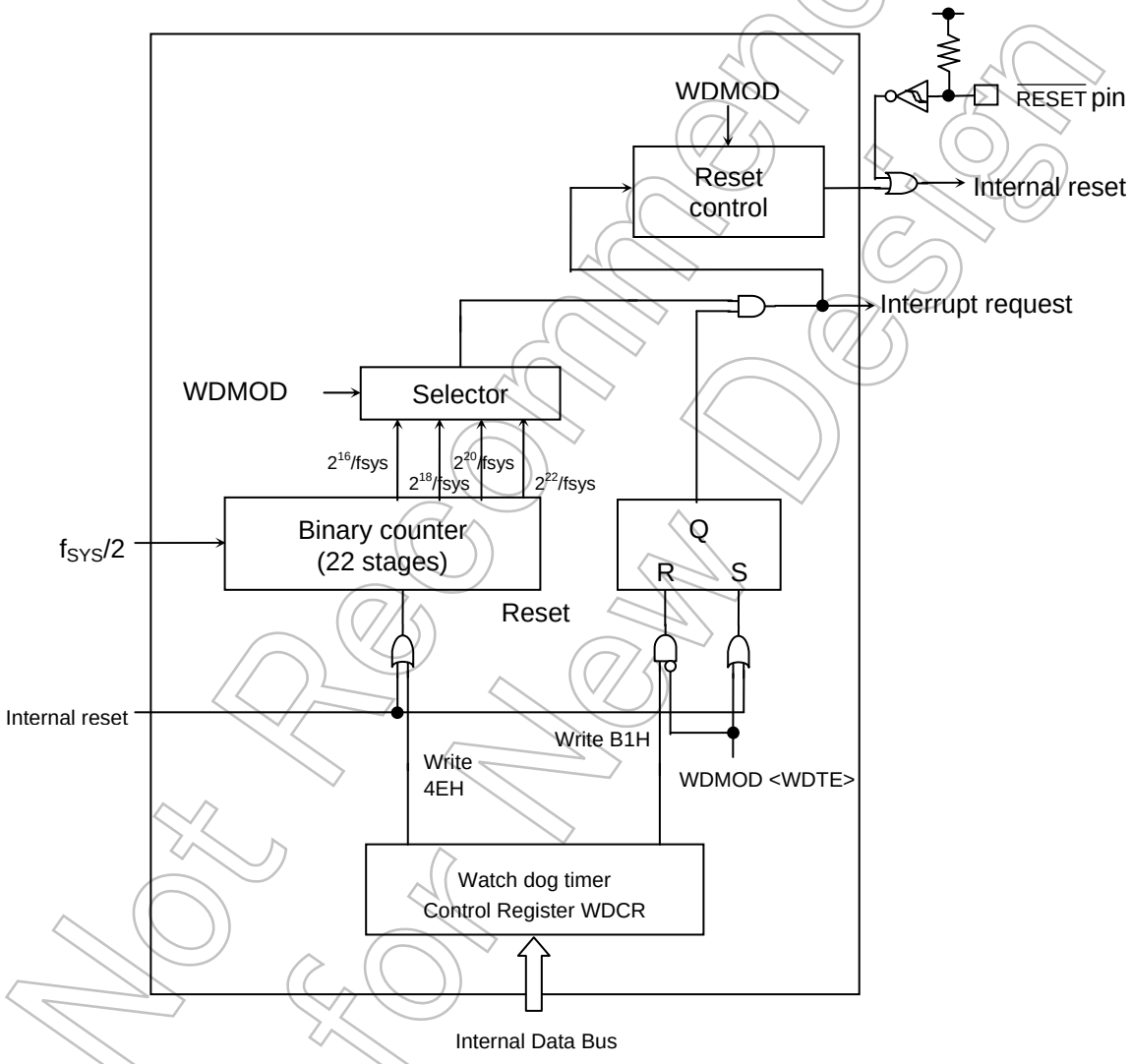


Fig. 17-1 Block Diagram of the Watchdog Timer

17.2 Watchdog Timer Interrupt

The watchdog timer consists of the binary counters that are arranged in 22 stages and work using the $f_{SYS/2}$ system clock as an input clock. The outputs produced by these binary counters are 2^{16}, 2^{18}, 2^{20} and 2^{22}. By selecting one of these outputs with WDMOD <WDTP1:0>, a watchdog timer interrupt can be generated when an overflow occurs, as shown in Fig. 17-2.

Because the watchdog timer interrupt is a non-maskable interrupt factor, NMIFLG <WDT> at the INTC performs a task of identifying it.

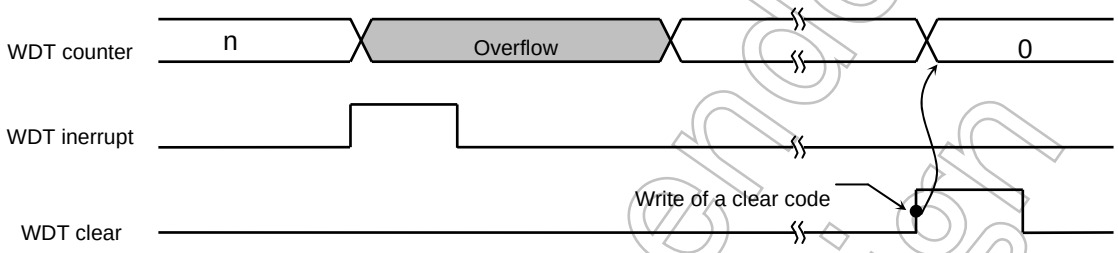
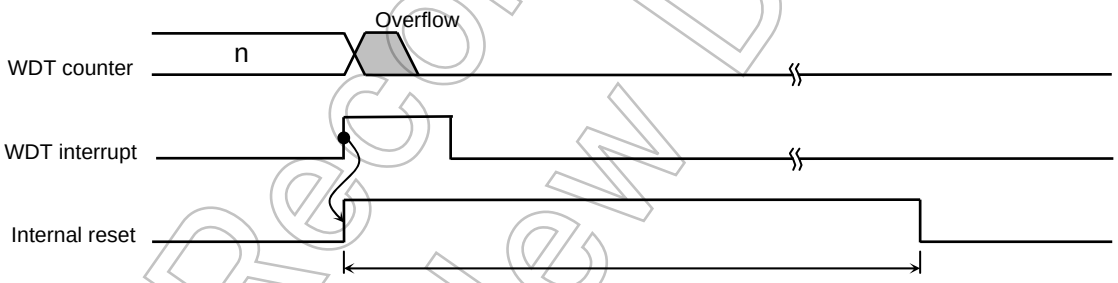


Fig. 17-2 Normal Mode

When an overflow occurs, resetting the chip itself is an option to choose. If the chip is reset, a reset is effected for a 32-state time, as shown in Fig. 17-3. If this reset is effected, the clock f_{SYS} that the clock gear generates by dividing the clock f_C of the high-speed oscillator by 8 is used as an input clock $f_{SYS/2}$.



32-state (9.48 μs @ f_C = 54MHz, f_{sys} = 6.75MHz, $f_{sys}/2$ = 3.375 MHz)
(12.8 μs @ f_C = 40MHz, f_{sys} = 5MHz, $f_{sys}/2$ = 2.5 MHz)

Fig. 17-3 Reset Mode

17.3 Control Registers

The watchdog timer (WDT) is controlled by two control registers WDMOD and WDCR.

17.3.1 Watchdog Timer Mode Register (WDMOD)

1. Specifying the detection time of the watchdog timer <WDTP1: 0>

   This is a 2-bit register for specifying the watchdog timer interrupt time for runaway detection. When a reset is effected, this register is initialized to WDMOD <WDTP1, 0> = "00." Fig. 17-4 shows the detection time of the watchdog timer.

2. Enabling/disabling the watchdog timer <WDTE>

   When reset, WDMOD <WDTE> is initialized to "1" and the watchdog timer is enabled.

   To disable the watchdog timer, this bit must be set to "0" and, at the same time, the disable code (B1H) must be written to the WDCR register. This dual setting is intended to minimize the probability that the watchdog timer may inadvertently be disabled if a runaway occurs.

   To change the status of the watchdog timer from "disable" to "enable," set the <WDTE> bit to "1".

3. Watchdog timer out reset connection <RESCR>

   This is a register for specifying whether or not to reset the watchdog timer itself after a runaway is detected. As a reset initializes this setting to WDMOD <RESCR>="0," a reset initiated the output of the watchdog timer is not performed.

17.3.2 Watchdog Timer Control Register (WDCR)

This is a register for disabling the watchdog timer function and controlling the clearing function of the binary counter.

- Disabling control

By writing the disable code (B1H) to this WDCR register after setting WDMOD <WDTE> to "0," the watchdog timer can be disabled.

```
WDMOD  ← 0 – – – – – – –   Clears WDTE to "0."
WDCR   ← 1 0 1 1 0 0 0 1   Writes the disable code (B1H).
```

- Enabling control

Set WDMOD <WDTE> to "1".

- Watchdog timer clearing control

Writing the clear code (4EH) to the WDCR register clears the binary counter and allows it to resume counting.

```
WDCR   ← 0 1 0 0 1 1 1 0   Writes the clear code (4EH)
```

(Note) Writing the disable code (BIH) clears the binary counter.

WDMOD (0xFFFF_FA00)

	7	6	5	4	3	2	1	0
Bit symbol	WDTE	WDTP1	WDTP0			I2WDT	RESCR	
Read/Write	R/W	R/W		R		R/W		R/W
After reset	1	0	0	0		0	0	0
Function	WDT control 1: Enable	Selects WDT detection time 00: $2^{16}/f_{SYS}$ 01: $2^{18}/f_{SYS}$ 10: $2^{20}/f_{SYS}$ 11: $2^{22}/f_{SYS}$		"0" is read.		IDLE 0: Stop 1: Start	1:Internally connect WDT output to reset pin	Write "0."

Watchdog timer out control

0	Generates NMI interrupt
1	Connects WDT out to reset

Detection time of watchdog timer

SYSCR1 clock gear value <GEAR2 : 0>	Detection Time of Watchdog Timer @ fc = 54 MHz			
	WDMOD<WDTP1, 0>			
	00	01	10	11
000 (fc)	1.2 ms	4.9 ms	19.4 ms	77.7 ms
100 (fc/2)	2.4 ms	9.7 ms	38.8 ms	155.3 ms
110 (fc/4)	4.9 ms	19.4 ms	77.7 ms	310.7 ms
111 (fc/8)	9.7 ms	38.8 ms	155.3 ms	621.4 ms

SYSCR1 clock gear value <GEAR2 : 0>	Detection Time of Watchdog Timer @ fc = 40 MHz			
	WDMOD<WDTP1, 0>			
	00	01	10	11
000 (fc)	1.6 ms	6.5 ms	26.2 ms	105 ms
100 (fc/2)	3.3 ms	13.1 ms	52.4 ms	210 ms
110 (fc/4)	6.5 ms	26.2 ms	105.0 ms	419 ms
111 (fc/8)	13.1 ms	52.4 ms	209.0 ms	839 ms

Enable/disable control of the watchdog timer

0	Disable
1	Enable

Fig. 17-4 Watchdog Timer Mode Register

WDCR (0xFFFF_FA01)

	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 of WDT

B1H	Disable code
4EH	Clear code
Others	—

Fig. 17-5 Watchdog Timer Control Register

17.4 Operation Description

The watchdog timer generates the INTWDT interrupt after a lapse of the detection time specified by the WDMOD <WDTP1, 0> register. Before generating the INTWD interrupt, the binary counter for the watchdog timer must be cleared to "0" using software (instruction). If the CPU malfunctions (runs away) due to noise or other disturbances and cannot execute the instruction to clear the binary counter, the binary counter overflows and the INTWD interrupt is generated. The CPU is able to recognize the occurrence of a malfunction (runaway) by identifying the INTWD interrupt and to restore the faulty condition to normal by using a malfunction (runaway) countermeasure program. Additionally, it is possible to resolve the problem of a malfunction (runaway) of the CPU by connecting the watchdog timer out pin to reset pins of peripheral devices.

The watchdog timer begins operation immediately after a reset is cleared.

In STOP mode, the watchdog timer is reset and in an idle state. When the bus is open ($\overline{\mathrm{BUSAK}}$ = "L"), it continues counting. In IDLE mode, its operation depends on the WDMOD <I2WDT> setting. Before putting it in IDLE mode, WDMOD <I2WDT> must be set to an appropriate setting, as required.

Example:

1. To clear the binary counter

```
                7 6 5 4 3 2 1 0
WDCR    ← 0 1 0 0 1 1 1 0      Writes the clear code (4EH)
```

2. To set the detection time of the watchdog timer to $2^{18}/f_{SYS.}$

```
                7 6 5 4 3 2 1 0
WDMOD   ← 1 0 1 – – – – –
```

3. To disable the watchdog timer.

```
                7 6 5 4 3 2 1 0
WDMOD   ← 0 – – – – – – –      Clears WDTE to "0"
WDCR    ← 1 0 1 1 0 0 0 1      Writes the disable code (B1H)
```

(Note): If the watchdog timer is operated when the high-frequency oscillator is idle, the system reset operation initiated by the watchdog timer becomes erratic due to the unstable oscillation of the high-frequency oscillator. Therefore, do not operate the watchdog timer when the high-frequency oscillator is idle.

18. ROM Correction Function

This chapter describes the ROM correction function built into the TMP19A23.

18.1 Features

- Using this function, twelve pieces of eight-word data can be replaced.
- If an address (lower 5 or 2 bits are "don't care" bits) written to the address register matches an address generated by the PC or DMAC, ROM data is replaced by data generated by the ROM correction data register which is established in a RAM area assigned to the above address register.
- ROM correction is automatically authorized by writing an address to each address register.
- If ROM correction cannot be executed using eight-word data due to a program modification or for other reasons, it is possible to place a "jump-to-RAM" instruction in a data register in a RAM area and to correct ROM data in that RAM area.

18.2 Description of Operations

By setting in the address register ADDREGn a physical address (including a projection area) of the ROM area to be corrected, ROM data can be replaced by data generated by a data register in a RAM area assigned to ADDREGn. The ROM correction function is automatically enabled when an address is set in ADDREGn, and it cannot be disabled. After a reset, the ROM correction function is disabled. Therefore, to execute ROM correction with the initialization after a reset is cleared, it is necessary to set an address in ADDREG. As an address is set in ADDREG, the ROM correction function is enabled for this register. If the CPU has the bus authority, ROM data is replaced when the value generated by the PC matches that of the address register. If the DMAC has the bus authority, ROM data is replaced when a source or destination address generated by the DMAC matches the value of the address register. For example, if an address is set in ADDREG0 and ADDREG3, the ROM correction function is enabled for this area; match detection is performed on these registers, and data replacement is executed if there is a match. Data replacement is not executed for ADDREG1, ADDREG2, and ADDREG4 through ADDREG7. Although the bit <31:5> exists in address registers, match detection is performed on A<17:5> for reasons of circuitry simplification. Internal processing is that data replacement is executed when the calculation of a logical product is completed by multiplying the ROMCS signal showing a ROM area by the result of a match detection operation performed by ROM correction circuitry.

If eight-word data is replaced, an address for ROM correction can be established only on an eight-word boundary, and data is replaced in units of 32 bytes. If only part of 32-byte data must be replaced with different data, the addresses that do not need to be replaced must be overwritten with the same data as the one existing prior to data replacement.

ADDREGn registers and RAM areas assigned to them are as follows:

Table 18-1 ADDREGn and RAM area

Register	Address	RAM area	Number of words
ADDREG0	0xFFFF_E540	0xFFFF_DE80 - 0xFFFF_DE9C	8
ADDREG1	0xFFFF_E544	0xFFFF_DEA0 - 0xFFFF_DEBC	8
ADDREG2	0xFFFF_E548	0xFFFF_DEC0 - 0xFFFF_DEDC	8
ADDREG3	0xFFFF_E54C	0xFFFF_DEE0 - 0xFFFF_DEFC	8
ADDREG4	0xFFFF_E550	0xFFFF_DF00 - 0xFFFF_DF1C	8
ADDREG5	0xFFFF_E554	0xFFFF_DF20 - 0xFFFF_DF3C	8
ADDREG6	0xFFFF_E558	0xFFFF_DF40 - 0xFFFF_DF5C	8
ADDREG7	0xFFFF_E55C	0xFFFF_DF60 - 0xFFFF_DF7C	8
ADDREG8	0xFFFF_E560	0xFFFF_DF80 - 0xFFFF_DF9C	8
ADDREG9	0xFFFF_E564	0xFFFF_DFA0 - 0xFFFF_DFBC	8
ADDREGA	0xFFFF_E568	0xFFFF_DFC0 - 0xFFFF_DFDC	8
ADDREGB	0xFFFF_E56C	0xFFFF_DFE0 - 0xFFFF_DFFC	8

Note: To use the ROM correction function, the ROM must be unprotected.

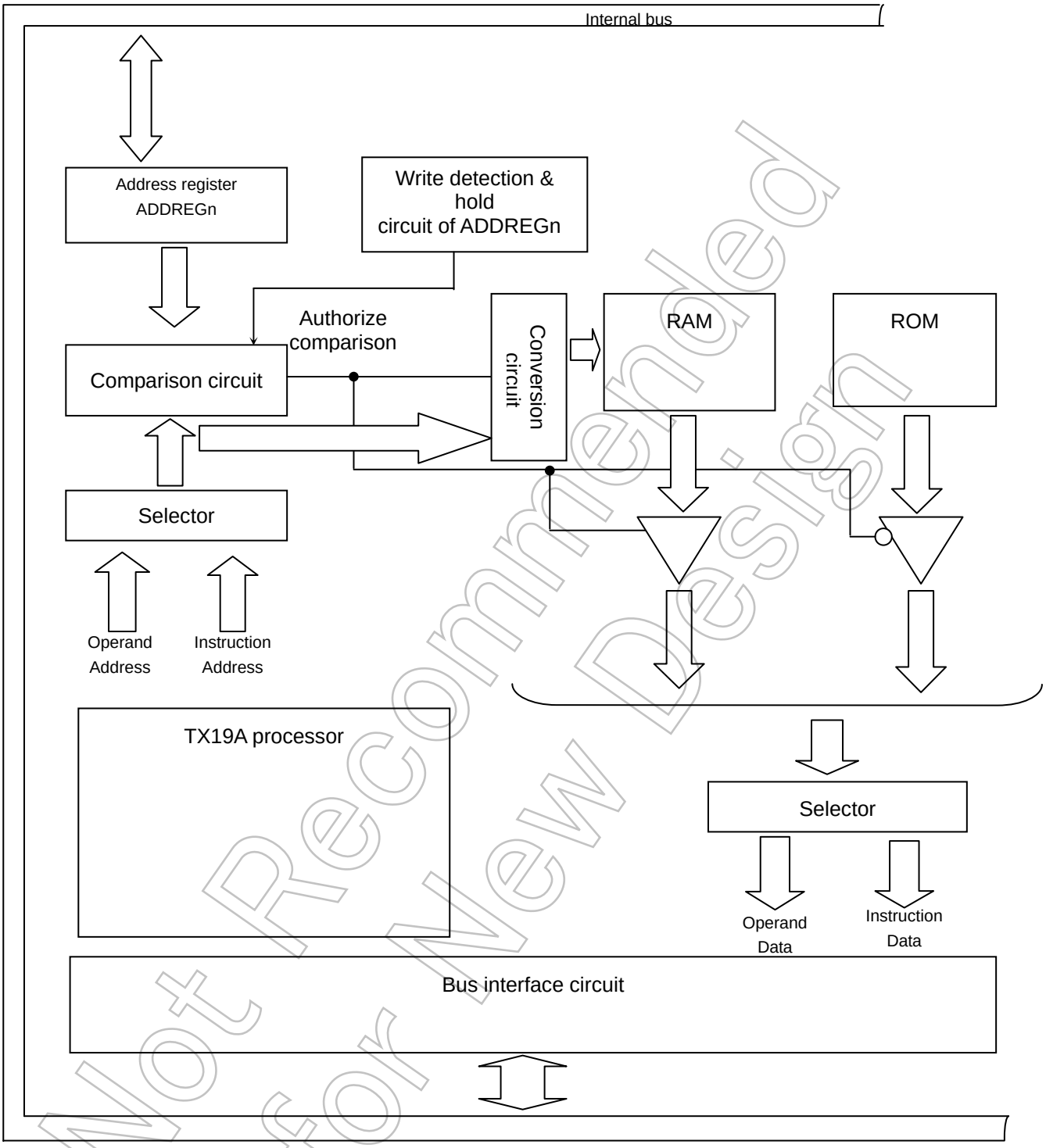


Fig. 18-2 ROM correction system diagram

18.3 Registers

(1) Address registers

ADDREG0 (0xFFFF_E540)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD07	ADD06	ADD05					
Read/Write	R/W			R				
After reset	0			1				
	15	14	13	12	11	10	9	8
Bit Symbol	ADD015	ADD014	ADD013	ADD012	ADD011	ADD010	ADD09	ADD08
Read/Write	R/W							
After reset	0							
	23	22	21	20	19	18	17	16
Bit Symbol							ADD017	ADD016
Read/Write	R						R/W	
After reset	0						0	
	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset	0							

ADDREG1 (0xFFFF_E544)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD17	ADD16	ADD15					
Read/Write	R/W			R				
After reset	0			1				
	15	14	13	12	11	10	9	8
Bit Symbol	ADD115	ADD114	ADD113	ADD112	ADD111	ADD110	ADD19	ADD18
Read/Write	R/W							
After reset	0							
	23	22	21	20	19	18	17	16
Bit Symbol							ADD117	ADD116
Read/Write	R						R/W	
After reset	0						0	
	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset	0							

ADDREG2 (0xFFFF_E548)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD27	ADD26	ADD25					
Read/Write	R/W			R				
After reset	0			1				
	15	14	13	12	11	10	9	8
Bit Symbol	ADD215	ADD214	ADD213	ADD212	ADD211	ADD210	ADD29	ADD28
Read/Write	R/W							
After reset	0							
	23	22	21	20	19	18	17	16
Bit Symbol							ADD217	ADD216
Read/Write	R						R/W	
After reset	0						0	
	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset	0							

Fig. 18-3 ROM correction related registers

ADDREG3 (0xFFFF_E54C)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD37	ADD36	ADD35					
Read/Write	R/W			R				
After reset	0			1				
	15	14	13	12	11	10	9	8
Bit Symbol	ADD315	ADD314	ADD313	ADD312	ADD311	ADD310	ADD39	ADD38
Read/Write	R/W							
After reset	0							
	23	22	21	20	19	18	17	16
Bit Symbol							ADD317	ADD316
Read/Write	R						R/W	
After reset	0						0	
	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset	0							

ADDREG4 (0xFFFF_E550)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD47	ADD46	ADD45					
Read/Write	R/W			R				
After reset	0			1				
	15	14	13	12	11	10	9	8
Bit Symbol	ADD415	ADD414	ADD413	ADD412	ADD411	ADD410	ADD49	ADD48
Read/Write	R/W							
After reset	0							
	23	22	21	20	19	18	17	16
Bit Symbol							ADD417	ADD416
Read/Write	R						R/W	
After reset	0						0	
	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset	0							

ADDREG5 (0xFFFF_E554)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD57	ADD56	ADD55					
Read/Write	R/W			R				
After reset	0			1				
	15	14	13	12	11	10	9	8
Bit Symbol	ADD515	ADD514	ADD513	ADD512	ADD511	ADD510	ADD59	ADD58
Read/Write	R/W							
After reset	0							
	23	22	21	20	19	18	17	16
Bit Symbol							ADD517	ADD516
Read/Write	R						R/W	
After reset	0						0	
	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset	0							

Fig. 18-4 ROM correction related registers

ADDREG6 (0xFFFF_E558)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD67	ADD66	ADD65					
Read/Write	R/W			R				
After reset	0			1				
	15	14	13	12	11	10	9	8
Bit Symbol	ADD615	ADD614	ADD613	ADD612	ADD611	ADD610	ADD69	ADD68
Read/Write	R/W							
After reset	0							
	23	22	21	20	19	18	17	16
Bit Symbol							ADD617	ADD616
Read/Write	R						R/W	
After reset	0						0	
	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset	0							

ADDREG7 (0xFFFF_E55C)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD77	ADD76	ADD75					
Read/Write	R/W			R				
After reset	0			1				
	15	14	13	12	11	10	9	8
Bit Symbol	ADD715	ADD714	ADD713	ADD712	ADD711	ADD710	ADD79	ADD78
Read/Write	R/W							
After reset	0							
	23	22	21	20	19	18	17	16
Bit Symbol							ADD717	ADD716
Read/Write	R						R/W	
After reset	0						0	
	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset	0							

ADDREG8 (0xFFFF_E560)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD87	ADD86	ADD85					
Read/Write	R/W			R				
After reset	0			1				
	15	14	13	12	11	10	9	8
Bit Symbol	ADD815	ADD814	ADD813	ADD812	ADD811	ADD810	ADD89	ADD88
Read/Write	R/W							
After reset	0							
	23	22	21	20	19	18	17	16
Bit Symbol							ADD817	ADD816
Read/Write	R						R/W	
After reset	0						0	
	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset	0							

Fig. 18-5 ROM correction related registers

ADDREG9 (0xFFFF_E564)

	7	6	5	4	3	2	1	0
Bit Symbol	ADD97	ADD96	ADD95					
Read/Write	R/W			R				
After reset	0			1				
	15	14	13	12	11	10	9	8
Bit Symbol	ADD915	ADD914	ADD913	ADD912	ADD911	ADD910	ADD99	ADD98
Read/Write	R/W							
After reset	0							
	23	22	21	20	19	18	17	16
Bit Symbol							ADD917	ADD916
Read/Write	R						R/W	
After reset	0						0	
	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset	0							

ADDREGA (0xFFFF_E568)

	7	6	5	4	3	2	1	0
Bit Symbol	ADDA7	ADDA6	ADDA5					
Read/Write	R/W			R				
After reset	0			1				
	15	14	13	12	11	10	9	8
Bit Symbol	ADDA15	ADDA14	ADDA13	ADDA12	ADDA11	ADDA10	ADDA9	ADDA8
Read/Write	R/W							
After reset	0							
	23	22	21	20	19	18	17	16
Bit Symbol							ADDA17	ADDA16
Read/Write	R						R/W	
After reset	0						0	
	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset	0							

ADDREGB (0xFFFF_E56C)

	7	6	5	4	3	2	1	0
Bit Symbol	ADDB7	ADDB6	ADDB5					
Read/Write	R/W			R				
After reset	0			1				
	15	14	13	12	11	10	9	8
Bit Symbol	ADDB15	ADDB14	ADDB13	ADDB12	ADDB11	ADDB10	ADDB9	ADDB8
Read/Write	R/W							
After reset	0							
	23	22	21	20	19	18	17	16
Bit Symbol							ADDB17	ADDB16
Read/Write	R						R/W	
After reset	0						0	
	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset	0							

(Note 1) Data cannot be transferred by DMA to the address register. However, data can be transferred by DMA to the RAM area where data for replacement is placed. The ROM correction function supports data replacement for both CPU and DMA access.

(Note 2) Writing back the initial value "0x00" allows data at the reset address to be replaced.

Fig. 18-6 ROM correction related registers

19. Table of Special Function Registers

Special function registers are allocated to an 8K-byte address space from FFFFE000H to 0xFFFFFFFF.

[1] Port registers
[2] 16-bit timer
[3] I^2CBUS/serial channel
[4] UART/serial channel
[5] 10-bit A/D converter
[6] Watchdog timer
[7] Interrupt controller
[8] DMA controller
[9] Chip select/ wait controller
[10] FLASH control
[11] ROM correction
[12] UART/high-speed serial channel
[13] High accuracy PPG
[14] Clock generator

(Note 1) The register mapped to an address within 0xFFFF_F000~0xFFFF_FFFF is not affected by a little-endian setup. The address of the register mapped within 0xFFFF_E000~0xFFFF_EFFF varies depending on the little-endian setup.

(Note 2) For continuous 8-bit long registers, 16- or 32-bit access is possible. The use of 16- or 32-bit access requires that an even-number address be accessed and that an even-number address does not contain undefined areas.

1. Big-endian

[1] PORT registers

ADR	Register name
FFFFF000H	P0
1H	P1
2H	
3H	
4H	P0CR
5H	P0FC1
6H	
7H	P0PUP
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFF010H	
1H	
2H	
3H	
4H	P1CR
5H	P1FC1
6H	P1FC2
7H	P1PUP
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFF020H	P2
1H	P2CR
2H	P2FC1
3H	P2FC2
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	P2PUP
CH	
DH	
EH	P2IE
FH	

ADR	Register name
FFFFF030H	P3
1H	P3CR
2H	P3FC1
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	P3PUP
CH	
DH	
EH	P3IE
FH	

ADR	Register name
FFFFF040H	P4
1H	P4CR
2H	P4FC1
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	P4OD
BH	P4PUP
CH	
DH	
EH	P4IE
FH	

ADR	Register name
FFFFF050H	P5
1H	P5CR
2H	P5FC1
3H	P5FC2
4H	P5FC3
5H	
6H	Reserved
7H	
8H	
9H	
AH	P5OD
BH	P5PUP
CH	
DH	
EH	P5IE
FH	

ADR	Register name
FFFFF060H	P6
1H	P6CR
2H	P6FC1
3H	P6FC2
4H	P6FC3
5H	P6FC4
6H	Reserved
7H	
8H	
9H	
AH	P6OD
BH	P6PUP
CH	
DH	
EH	P6IE
FH	

ADR	Register name
FFFFF070H	P7
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	P7PUP
CH	
DH	
EH	P7IE
FH	

ADR	Register name
FFFFF080H	P8
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	P8PUP
CH	
DH	
EH	P8IE
FH	

ADR	Register name
FFFFF090H	P9
1H	P9CR
2H	P9FC1
3H	P9FC2
4H	P9FC3
5H	
6H	
7H	
8H	
9H	
AH	
BH	P9PUP
CH	
DH	P9SEL
EH	P9IE
FH	

ADR	Register name
FFFFF0A0H	PA
1H	PACR
2H	PAFC1
3H	PAFC2
4H	
5H	
6H	Reserved
7H	
8H	
9H	
AH	PAOD
BH	PAPUP
CH	
DH	PASEL
EH	PAIE
FH	

ADR	Register name
FFFFF0B0H	PB
1H	PBCR
2H	PBFC1
3H	PBFC2
4H	PBFC3
5H	
6H	
7H	
8H	
9H	
AH	PBOD
BH	PBPUP
CH	
DH	PBSEL
EH	PBIE
FH	

ADR	Register name
FFFFF0C0H	PC
1H	PCCR
2H	PCFC1
3H	PCFC2
4H	
5H	
6H	
7H	
8H	
9H	
AH	PCOD
BH	PCPUP
CH	
DH	
EH	PCIE
FH	

ADR	Register name
FFFFF0D0H	PD
1H	PDCR
2H	PDFC1
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	PDPUP
CH	
DH	***reserved***
EH	PDIE
FH	

ADR	Register name
FFFFF0E0H	PE
1H	PECR
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	PEPUP
CH	
DH	***reserved***
EH	PEIE
FH	

ADR	Register name
FFFFF0F0H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[2] 16-bit timer

ADR	Register name
FFFFF200H	TB0RUN
1H	TB0CR
2H	TB0MOD
3H	TB0FFCR
4H	TB0ST
5H	TB0OUTCR
6H	TB0UCL
7H	TB0UCH
8H	TB0RG0L
9H	TB0RG0H
AH	TB0RG1L
BH	TB0RG1H
CH	TB0CP0L
DH	TB0CP0H
EH	TB0CP1L
FH	TB0CP1H

ADR	Register name
FFFFF210H	TB1RUN
1H	TB1CR
2H	TB1MOD
3H	TB1FFCR
4H	TB1ST
5H	
6H	TB1UCL
7H	TB1UCH
8H	TB1RG0L
9H	TB1RG0H
AH	TB1RG1L
BH	TB1RG1H
CH	TB1CP0L
DH	TB1CP0H
EH	TB1CP1L
FH	TB1CP1H

ADR	Register name
FFFFF220H	TB2RUN
1H	TB2CR
2H	TB2MOD
3H	TB2FFCR
4H	TB2ST
5H	
6H	TB2UCL
7H	TB2UCH
8H	TB2RG0L
9H	TB2RG0H
AH	TB2RG1L
BH	TB2RG1H
CH	TB2CP0L
DH	TB2CP0H
EH	TB2CP1L
FH	TB2CP1H

ADR	Register name
FFFFF230H	TB3RUN
1H	TB3CR
2H	TB3MOD
3H	TB3FFCR
4H	TB3ST
5H	
6H	TB3UCL
7H	TB3UCH
8H	TB3RG0L
9H	TB3RG0H
AH	TB3RG1L
BH	TB3RG1H
CH	TB3CP0L
DH	TB3CP0H
EH	TB3CP1L
FH	TB3CP1H

ADR	Register name
FFFFF240H	TB4RUN
1H	TB4CR
2H	TB4MOD
3H	TB4FFCR
4H	TB4ST
5H	
6H	TB4UCL
7H	TB4UCH
8H	TB4RG0L
9H	TB4RG0H
AH	TB4RG1L
BH	TB4RG1H
CH	TB4CP0L
DH	TB4CP0H
EH	TB4CP1L
FH	TB4CP1H

ADR	Register name
FFFFF250H	TB5RUN
1H	TB5CR
2H	TB5MOD
3H	TB5FFCR
4H	TB5ST
5H	
6H	TB5UCL
7H	TB5UCH
8H	TB5RG0L
9H	TB5RG0H
AH	TB5RG1L
BH	TB5RG1H
CH	TB5CP0L
DH	TB5CP0H
EH	TB5CP1L
FH	TB5CP1H

ADR	Register name
FFFFF260H	TB6RUN
1H	TB6CR
2H	TB6MOD
3H	TB6FFCR
4H	TB6ST
5H	
6H	TB6UCL
7H	TB6UCH
8H	TB6RG0L
9H	TB6RG0H
AH	TB6RG1L
BH	TB6RG1H
CH	TB6CP0L
DH	TB6CP0H
EH	TB6CP1L
FH	TB6CP1H

ADR	Register name
FFFFF270H	TB7RUN
1H	TB7CR
2H	TB7MOD
3H	TB7FFCR
4H	TB7ST
5H	
6H	TB7UCL
7H	TB7UCH
8H	TB7RG0L
9H	TB7RG0H
AH	TB7RG1L
BH	TB7RG1H
CH	TB7CP0L
DH	TB7CP0H
EH	TB7CP1L
FH	TB7CP1H

ADR	Register name
FFFFF280H	TB8RUN
1H	TB8CR
2H	TB8MOD
3H	TB8FFCR
4H	TB8ST
5H	
6H	TB8UCL
7H	TB8UCH
8H	TB8RG0L
9H	TB8RG0H
AH	TB8RG1L
BH	TB8RG1H
CH	TB8CP0L
DH	TB8CP0H
EH	TB8CP1L
FH	TB8CP1H

ADR	Register name
FFFFF290H	TB9RUN
1H	TB9CR
2H	TB9MOD
3H	TB9FFCR
4H	TB9ST
5H	
6H	TB9UCL
7H	TB9UCH
8H	TB9RG0L
9H	TB9RG0H
AH	TB9RG1L
BH	TB9RG1H
CH	TB9CP0L
DH	TB9CP0H
EH	TB9CP1L
FH	TB9CP1H

[3] SBI

ADR	Register name
FFFFF600H	SBI0CR1
1H	SBI0DBR
2H	SBI0I2CAR
3H	SBI0CR2/SR
4H	SBI0BR0
5H	
6H	
7H	SBI0CR0
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFF610H	SBI1CR1
1H	SBI1DBR
2H	SBI1I2CAR
3H	SBI1CR2/SR
4H	SBI1BR0
5H	
6H	
7H	SBI1CR0
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[4] SIO/UART

ADR	Register name
FFFFF700H	SC0BUF
1H	SC0CR
2H	SC0MOD0
3H	BR0CR
4H	BR0ADD
5H	SC0MOD1
6H	SC0MOD2
7H	SC0EN
8H	SC0RFC
9H	SC0TFC
AH	SC0RST
BH	SC0TST
CH	SC0FCNF
DH	
EH	
FH	

ADR	Register name
FFFFF710H	SC1BUF
1H	SC1CR
2H	SC1MOD0
3H	BR1CR
4H	BR1ADD
5H	SC1MOD1
6H	SC1MOD2
7H	SC1EN
8H	SC1RFC
9H	SC1TFC
AH	SC1RST
BH	SC1TST
CH	SC1FCNF
DH	
EH	
FH	

ADR	Register name
FFFFF720H	SC2BUF
1H	SC2CR
2H	SC2MOD0
3H	BR2CR
4H	BR2ADD
5H	SC2MOD1
6H	SC2MOD2
7H	SC2EN
8H	SC2RFC
9H	SC2TFC
AH	SC2RST
BH	SC2TST
CH	SC2FCNF
DH	
EH	
FH	

[5] ADC

ADR	Register name
FFFFF800H	ADREG08L
1H	ADREG08H
2H	ADREG19L
3H	ADREG19H
4H	ADREG2AL
5H	ADREG2AH
6H	ADREG3BL
7H	ADREG3BH
8H	ADREG4CL
9H	ADREG4CH
AH	ADREG5DL
BH	ADREG5DH
CH	ADREG6EL
DH	ADREG6EH
EH	ADREG7FL
FH	ADREG7FH

ADR	Register name
FFFFF810H	ADREGSPL
1H	ADREGSPH
2H	
3H	
4H	ADMOD0
5H	ADMOD1
6H	ADMOD2
7H	
8H	ADMOD4
9H	ADCBAS
AH	***reserved***
BH	***reserved***
CH	ADCLK
DH	
EH	ADMOD3
FH	ADMOD5

ADR	Register name
FFFFF820H	ADCMP0L
1H	ADCMP0H
2H	ADCMP1L
3H	ADCMP1H
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[6] WDT

ADR	Register name
FFFFFA00H	WDMOD
1H	WDCR
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[7] INTC

ADR	Register name
FFFFE000H	IMC0
1H	"
2H	"
3H	"
4H	IMC1
5H	"
6H	"
7H	"
8H	IMC2
9H	"
AH	"
BH	"
CH	IMC3
DH	"
EH	"
FH	"

ADR	Register name
FFFFE010H	IMC4
1H	"
2H	"
3H	"
4H	IMC5
5H	"
6H	"
7H	"
8H	IMC6
9H	"
AH	"
BH	"
CH	IMC7
DH	"
EH	"
FH	"

ADR	Register name
FFFFE020H	IMC8
1H	"
2H	"
3H	"
4H	IMC9
5H	"
6H	"
7H	"
8H	IMCA
9H	"
AH	"
BH	"
CH	IMCB
DH	"
EH	"
FH	"

ADR	Register name
FFFFE030H	IMCC
1H	"
2H	"
3H	"
4H	IMCD
5H	"
6H	"
7H	"
8H	IMCE
9H	"
AH	"
BH	"
CH	IMCF
DH	"
EH	"
FH	"

ADR	Register name
FFFFE040H	IVR
1H	“
2H	“
3H	“
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE050H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE060H	INTCLR
1H	“
2H	“
3H	“
4H	**DREQFLG**
5H	“
6H	“
7H	“
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE070H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE100H	***reserved***
1H	“
2H	“
3H	“
4H	***reserved***
5H	“
6H	“
7H	“
8H	***reserved***
9H	“
AH	“
BH	“
CH	ILEV
DH	“
EH	“
FH	“

[8] DMAC

ADR	Register name
FFFFE200H	CCR0
1H	“
2H	“
3H	“
4H	CSR0
5H	“
6H	“
7H	“
8H	SAR0
9H	“
AH	“
BH	“
CH	DAR0
DH	“
EH	“
FH	“

ADR	Register name
FFFFE210H	BCR0
1H	“
2H	“
3H	“
4H	
5H	
6H	
7H	
8H	DTCR0
9H	“
AH	“
BH	“
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE220H	CCR1
1H	“
2H	“
3H	“
4H	CSR1
5H	“
6H	“
7H	“
8H	SAR1
9H	“
AH	“
BH	“
CH	DAR1
DH	“
EH	“
FH	“

ADR	Register name
FFFFE230H	BCR1
1H	“
2H	“
3H	“
4H	
5H	
6H	
7H	
8H	DTCR1
9H	“
AH	“
BH	“
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE240H	CCR2
1H	“
2H	“
3H	“
4H	CSR2
5H	“
6H	“
7H	“
8H	SAR2
9H	“
AH	“
BH	“
CH	DAR2
DH	“
EH	“
FH	“

ADR	Register name
FFFFE250H	BCR2
1H	“
2H	“
3H	“
4H	
5H	
6H	
7H	
8H	DTCR2
9H	“
AH	“
BH	“
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE260H	CCR3
1H	“
2H	“
3H	“
4H	CSR3
5H	“
6H	“
7H	“
8H	SAR3
9H	“
AH	“
BH	“
CH	DAR3
DH	“
EH	“
FH	“

ADR	Register name
FFFFE270H	BCR3
1H	“
2H	“
3H	“
4H	
5H	
6H	
7H	
8H	DTCR3
9H	“
AH	“
BH	“
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE300H	DCR
1H	“
2H	“
3H	“
4H	RSR
5H	“
6H	“
7H	“
8H	
9H	
AH	
BH	
CH	DHR
DH	“
EH	“
FH	“

ADR	Register name
FFFFE310H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE320H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE330H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[9] CS/WAIT controller

ADR	Register name 名
FFFFE400H	BMA0
1H	“
2H	“
3H	“
4H	BMA1
5H	“
6H	“
7H	“
8H	BMA2
9H	“
AH	“
BH	“
CH	BMA3
DH	“
EH	“
FH	“

ADR	Register name
FFFFE410H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE480H	B01CS
1H	“
2H	“
3H	“
4H	B23CS
5H	“
6H	“
7H	“
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE490H	
1H	
2H	
3H	
4H	***reserved***
5H	“
6H	“
7H	“
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[10] FLASH control

ADR	Register name
FFFFE510H	SEQMOD
1H	“
2H	“
3H	“
4H	SEQCNT
5H	“
6H	“
7H	“
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE520H	FLCS
1H	“
2H	“
3H	“
4H	FLPGEND
5H	“
6H	“
7H	“
8H	***reserved***
9H	“
AH	“
BH	“
CH	
DH	
EH	
FH	

[11] ROM correction

ADR	Register name
FFFFE540H	ADDREG0
1H	“
2H	“
3H	“
4H	ADDREG1
5H	“
6H	“
7H	“
8H	ADDREG2
9H	“
AH	“
BH	“
CH	ADDREG3
DH	“
EH	“
FH	“

ADR	Register name
FFFFE550H	ADDREG4
1H	“
2H	“
3H	“
4H	ADDREG5
5H	“
6H	“
7H	“
8H	ADDREG6
9H	“
AH	“
BH	“
CH	ADDREG7
DH	“
EH	“
FH	“

ADR	Register name
FFFFE560H	ADDREG8
1H	“
2H	“
3H	“
4H	ADDREG9
5H	“
6H	“
7H	“
8H	ADDREGA
9H	“
AH	“
BH	“
CH	ADDREGB
DH	“
EH	“
FH	“

ADR	Register name
FFFFE570H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE610H	*reserved*
1H	*reserved*
2H	*reserved*
3H	*reserved*
4H	*reserved*
5H	*reserved*
6H	*reserved*
7H	*reserved*
8H	*reserved*
9H	*reserved*
AH	*reserved*
BH	*reserved*
CH	*reserved*
DH	*reserved*
EH	*reserved*
FH	*reserved*

ADR	Register name
FFFFE620H	*reserved*
1H	*reserved*
2H	*reserved*
3H	*reserved*
4H	*reserved*
5H	*reserved*
6H	*reserved*
7H	*reserved*
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[12]HSIO/UART

ADR	Register name
FFFFE800H	
1H	
2H	
3H	HSC0BUF
4H	HSC0EN
5H	HSC0MOD2
6H	HSC0MOD1
7H	HBR0ADD
8H	HSC0TST
9H	HSC0RST
AH	HSC0TFC
BH	HSC0RFC
CH	HBR0CR
DH	HSC0MOD0
EH	HSC0CR
FH	HSC0FCNF

ADR	Register name
FFFFE810H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE820H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE830H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[13] High accuracy PPG (TMRD)

ADR	Register name
FFFFE900H	TD0RUN
1H	“
2H	“
3H	“
4H	TD0CR
5H	“
6H	“
7H	“
8H	TD0MOD
9H	“
AH	“
BH	“
CH	TB0BCR
DH	“
EH	“
FH	“

ADR	Register name
FFFFE910H	TD0ST
1H	“
2H	“
3H	“
4H	TD0REG0
5H	“
6H	“
7H	“
8H	TD0REG1
9H	“
AH	“
BH	“
CH	TD0REG2
DH	“
EH	“
FH	“

ADR	Register name
FFFFE920H	TD0REG3
1H	“
2H	“
3H	“
4H	TD0REG4
5H	“
6H	“
7H	“
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE930H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE940H	TD1RUN
1H	“
2H	“
3H	“
4H	TD1CR
5H	“
6H	“
7H	“
8H	TD1MOD
9H	“
AH	“
BH	“
CH	TD1BCR
DH	“
EH	“
FH	“

ADR	Register name
FFFFE950H	TD1ST
1H	“
2H	“
3H	“
4H	TD1REG0
5H	“
6H	“
7H	“
8H	TD1REG1
9H	“
AH	“
BH	“
CH	TD1REG2
DH	“
EH	“
FH	“

ADR	Register name
FFFFE960H	TD1REG3
1H	“
2H	“
3H	“
4H	TD1REG4
5H	“
6H	“
7H	“
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE970H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[14] CG

ADR	Register name
FFFFEE00H	SYSCR3
1H	SYSCR2
2H	SYSCR1
3H	SYSCR0
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFEE10H	IMCGA
1H	“
2H	“
3H	“
4H	IMCGB
5H	“
6H	“
7H	“
8H	IMCGC
9H	“
AH	“
BH	“
CH	IMCGD
DH	“
EH	“
FH	“

ADR	Register name
FFFFEE20H	EICRCG
1H	“
2H	“
3H	“
4H	NMIFLG
5H	“
6H	“
7H	“
8H	PWMCG
9H	“
AH	“
BH	“
CH	
DH	
EH	
FH	

ADR	Register name
FFFFEE30H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

2. Little-endian

[1] PORT registers

ADR	Register name
FFFFF000H	P0
1H	P1
2H	
3H	
4H	P0CR
5H	P0FC1
6H	
7H	P0PUP
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFF010H	
1H	
2H	
3H	
4H	P1CR
5H	P1FC1
6H	P1FC2
7H	P1PUP
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFF020H	P2
1H	P2CR
2H	P2FC1
3H	P2FC2
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	P2PUP
CH	
DH	
EH	P2IE
FH	

ADR	Register name
FFFFF030H	P3
1H	P3CR
2H	P3FC1
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	P3PUP
CH	
DH	
EH	P3IE
FH	

ADR	Register name
FFFFF040H	P4
1H	P4CR
2H	P4FC1
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	P4OD
BH	P4PUP
CH	
DH	
EH	P4IE
FH	

ADR	Register name
FFFFF050H	P5
1H	P5CR
2H	P5FC1
3H	P5FC2
4H	P5FC3
5H	
6H	Reserved
7H	
8H	
9H	
AH	P5OD
BH	P5PUP
CH	
DH	
EH	P5IE
FH	

ADR	Register name
FFFFF060H	P6
1H	P6CR
2H	P6FC1
3H	P6FC2
4H	P6FC3
5H	P6FC4
6H	Reserved
7H	
8H	
9H	
AH	P6OD
BH	P6PUP
CH	
DH	
EH	P6IE
FH	

ADR	Register name
FFFFF070H	P7
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	P7PUP
CH	
DH	
EH	P7IE
FH	

ADR	Register name
FFFFF080H	P8
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	P8PUP
CH	
DH	
EH	P8IE
FH	

ADR	Register name
FFFFF090H	P9
1H	P9CR
2H	P9FC1
3H	P9FC2
4H	P9FC3
5H	
6H	
7H	
8H	
9H	
AH	
BH	P9PUP
CH	
DH	P9SEL
EH	P9IE
FH	

ADR	Register name
FFFFF0A0H	PA
1H	PACR
2H	PAFC1
3H	PAFC2
4H	
5H	
6H	Reserved
7H	
8H	
9H	
AH	PAOD
BH	PAPUP
CH	
DH	PASEL
EH	PAIE
FH	

ADR	Register name
FFFFF0B0H	PB
1H	PBCR
2H	PBFC1
3H	PBFC2
4H	PBFC3
5H	
6H	
7H	
8H	
9H	
AH	PBOD
BH	PBPUP
CH	
DH	PBSEL
EH	PBIE
FH	

ADR	Register name
FFFFF0C0H	PC
1H	PCCR
2H	PCFC1
3H	PCFC2
4H	
5H	
6H	
7H	
8H	
9H	
AH	PCOD
BH	PCPUP
CH	
DH	
EH	PCIE
FH	

ADR	Register name
FFFFF0D0H	PD
1H	PDCR
2H	PDFC1
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	PDPUP
CH	
DH	***reserved***
EH	PDIE
FH	

ADR	Register name
FFFFF0E0H	PE
1H	PECR
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	PEPUP
CH	
DH	***reserved***
EH	PEIE
FH	

ADR	Register name
FFFFF0F0H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[2] 16-bit timer

ADR	Register name
FFFFF200H	TB0RUN
1H	TB0CR
2H	TB0MOD
3H	TB0FFCR
4H	TB0ST
5H	TB0OUTCR
6H	TB0UCL
7H	TB0UCH
8H	TB0RG0L
9H	TB0RG0H
AH	TB0RG1L
BH	TB0RG1H
CH	TB0CP0L
DH	TB0CP0H
EH	TB0CP1L
FH	TB0CP1H

ADR	Register name
FFFFF210H	TB1RUN
1H	TB1CR
2H	TB1MOD
3H	TB1FFCR
4H	TB1ST
5H	
6H	TB1UCL
7H	TB1UCH
8H	TB1RG0L
9H	TB1RG0H
AH	TB1RG1L
BH	TB1RG1H
CH	TB1CP0L
DH	TB1CP0H
EH	TB1CP1L
FH	TB1CP1H

ADR	Register name
FFFFF220H	TB2RUN
1H	TB2CR
2H	TB2MOD
3H	TB2FFCR
4H	TB2ST
5H	
6H	TB2UCL
7H	TB2UCH
8H	TB2RG0L
9H	TB2RG0H
AH	TB2RG1L
BH	TB2RG1H
CH	TB2CP0L
DH	TB2CP0H
EH	TB2CP1L
FH	TB2CP1H

ADR	Register name
FFFFF230H	TB3RUN
1H	TB3CR
2H	TB3MOD
3H	TB3FFCR
4H	TB3ST
5H	
6H	TB3UCL
7H	TB3UCH
8H	TB3RG0L
9H	TB3RG0H
AH	TB3RG1L
BH	TB3RG1H
CH	TB3CP0L
DH	TB3CP0H
EH	TB3CP1L
FH	TB3CP1H

ADR	Register name
FFFFF240H	TB4RUN
1H	TB4CR
2H	TB4MOD
3H	TB4FFCR
4H	TB4ST
5H	
6H	TB4UCL
7H	TB4UCH
8H	TB4RG0L
9H	TB4RG0H
AH	TB4RG1L
BH	TB4RG1H
CH	TB4CP0L
DH	TB4CP0H
EH	TB4CP1L
FH	TB4CP1H

ADR	Register name
FFFFF250H	TB5RUN
1H	TB5CR
2H	TB5MOD
3H	TB5FFCR
4H	TB5ST
5H	
6H	TB5UCL
7H	TB5UCH
8H	TB5RG0L
9H	TB5RG0H
AH	TB5RG1L
BH	TB5RG1H
CH	TB5CP0L
DH	TB5CP0H
EH	TB5CP1L
FH	TB5CP1H

ADR	Register name
FFFFF260H	TB6RUN
1H	TB6CR
2H	TB6MOD
3H	TB6FFCR
4H	TB6ST
5H	
6H	TB6UCL
7H	TB6UCH
8H	TB6RG0L
9H	TB6RG0H
AH	TB6RG1L
BH	TB6RG1H
CH	TB6CP0L
DH	TB6CP0H
EH	TB6CP1L
FH	TB6CP1H

ADR	Register name
FFFFF270H	TB7RUN
1H	TB7CR
2H	TB7MOD
3H	TB7FFCR
4H	TB7ST
5H	
6H	TB7UCL
7H	TB7UCH
8H	TB7RG0L
9H	TB7RG0H
AH	TB7RG1L
BH	TB7RG1H
CH	TB7CP0L
DH	TB7CP0H
EH	TB7CP1L
FH	TB7CP1H

ADR	Register name
FFFFF280H	TB8RUN
1H	TB8CR
2H	TB8MOD
3H	TB8FFCR
4H	TB8ST
5H	
6H	TB8UCL
7H	TB8UCH
8H	TB8RG0L
9H	TB8RG0H
AH	TB8RG1L
BH	TB8RG1H
CH	TB8CP0L
DH	TB8CP0H
EH	TB8CP1L
FH	TB8CP1H

ADR	Register name
FFFFF290H	TB9RUN
1H	TB9CR
2H	TB9MOD
3H	TB9FFCR
4H	TB9ST
5H	
6H	TB9UCL
7H	TB9UCH
8H	TB9RG0L
9H	TB9RG0H
AH	TB9RG1L
BH	TB9RG1H
CH	TB9CP0L
DH	TB9CP0H
EH	TB9CP1L
FH	TB9CP1H

[3] SBI

ADR	Register name
FFFFF600H	SBI0CR1
1H	SBI0DBR
2H	SBI0I2CAR
3H	SBI0CR2/SR
4H	SBI0BR0
5H	
6H	
7H	SBI0CR0
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFF610H	SBI1CR1
1H	SBI1DBR
2H	SBI1I2CAR
3H	SBI1CR2/SR
4H	SBI1BR0
5H	
6H	
7H	SBI1CR0
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[4] SIO/UART

ADR	Register name
FFFFF700H	SC0BUF
1H	SC0CR
2H	SC0MOD0
3H	BR0CR
4H	BR0ADD
5H	SC0MOD1
6H	SC0MOD2
7H	SC0EN
8H	SC0RFC
9H	SC0TFC
AH	SC0RST
BH	SC0TST
CH	SC0FCNF
DH	
EH	
FH	

ADR	Register name
FFFFF710H	SC1BUF
1H	SC1CR
2H	SC1MOD0
3H	BR1CR
4H	BR1ADD
5H	SC1MOD1
6H	SC1MOD2
7H	SC1EN
8H	SC1RFC
9H	SC1TFC
AH	SC1RST
BH	SC1TST
CH	SC1FCNF
DH	
EH	
FH	

ADR	Register name
FFFFF720H	SC2BUF
1H	SC2CR
2H	SC2MOD0
3H	BR2CR
4H	BR2ADD
5H	SC2MOD1
6H	SC2MOD2
7H	SC2EN
8H	SC2RFC
9H	SC2TFC
AH	SC2RST
BH	SC2TST
CH	SC2FCNF
DH	
EH	
FH	

[5] ADC

ADR	Register name
FFFFF800H	ADREG08L
1H	ADREG08H
2H	ADREG19L
3H	ADREG19H
4H	ADREG2AL
5H	ADREG2AH
6H	ADREG3BL
7H	ADREG3BH
8H	ADREG4CL
9H	ADREG4CH
AH	ADREG5DL
BH	ADREG5DH
CH	ADREG6EL
DH	ADREG6EH
EH	ADREG7FL
FH	ADREG7FH

ADR	Register name
FFFFF810H	ADREGSPL
1H	ADREGSPH
2H	
3H	
4H	ADMOD0
5H	ADMOD1
6H	ADMOD2
7H	
8H	ADMOD4
9H	ADCBAS
AH	*reserved*
BH	*reserved*
CH	ADCLK
DH	
EH	ADMOD3
FH	ADMOD5

ADR	Register name
FFFFF820H	ADCMP0L
1H	ADCMP0H
2H	ADCMP1L
3H	ADCMP1H
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[6] WDT

ADR	Register name
FFFFFA00H	WDMOD
1H	WDCR
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[7] INTC

ADR	Register name
FFFFE000H	IMC0
1H	“
2H	“
3H	“
4H	IMC1
5H	“
6H	“
7H	“
8H	IMC2
9H	“
AH	“
BH	“
CH	IMC3
DH	“
EH	“
FH	“

ADR	Register name
FFFFE010H	IMC4
1H	“
2H	“
3H	“
4H	IMC5
5H	“
6H	“
7H	“
8H	IMC6
9H	“
AH	“
BH	“
CH	IMC7
DH	“
EH	“
FH	“

ADR	Register name
FFFFE020H	IMC8
1H	“
2H	“
3H	“
4H	IMC9
5H	“
6H	“
7H	“
8H	IMCA
9H	“
AH	“
BH	“
CH	IMCB
DH	“
EH	“
FH	“

ADR	Register name
FFFFE030H	IMCC
1H	“
2H	“
3H	“
4H	IMCD
5H	“
6H	“
7H	“
8H	IMCE
9H	“
AH	“
BH	“
CH	IMCF
DH	“
EH	“
FH	“

ADR	Register name
FFFFE040H	IVR
1H	“
2H	“
3H	“
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE050H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE060H	INTCLR
1H	“
2H	“
3H	“
4H	**DREQFLG**
5H	“
6H	“
7H	“
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE070H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE100H	***reserved***
1H	“
2H	“
3H	“
4H	***reserved***
5H	“
6H	“
7H	“
8H	***reserved***
9H	“
AH	“
BH	“
CH	ILEV
DH	“
EH	“
FH	“

[8] DMAC

ADR	Register name
FFFFE200H	CCR0
1H	“
2H	“
3H	“
4H	CSR0
5H	“
6H	“
7H	“
8H	SAR0
9H	“
AH	“
BH	“
CH	DAR0
DH	“
EH	“
FH	“

ADR	Register name
FFFFE210H	BCR0
1H	“
2H	“
3H	“
4H	
5H	
6H	
7H	
8H	DTCR0
9H	“
AH	“
BH	“
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE220H	CCR1
1H	“
2H	“
3H	“
4H	CSR1
5H	“
6H	“
7H	“
8H	SAR1
9H	“
AH	“
BH	“
CH	DAR1
DH	“
EH	“
FH	“

ADR	Register name
FFFFE230H	BCR1
1H	“
2H	“
3H	“
4H	
5H	
6H	
7H	
8H	DTCR1
9H	“
AH	“
BH	“
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE240H	CCR2
1H	“
2H	“
3H	“
4H	CSR2
5H	“
6H	“
7H	“
8H	SAR2
9H	“
AH	“
BH	“
CH	DAR2
DH	“
EH	“
FH	“

ADR	Register name
FFFFE250H	BCR2
1H	“
2H	“
3H	“
4H	
5H	
6H	
7H	
8H	DTCR2
9H	“
AH	“
BH	“
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE260H	CCR3
1H	“
2H	“
3H	“
4H	CSR3
5H	“
6H	“
7H	“
8H	SAR3
9H	“
AH	“
BH	“
CH	DAR3
DH	“
EH	“
FH	“

ADR	Register name
FFFFE270H	BCR3
1H	“
2H	“
3H	“
4H	
5H	
6H	
7H	
8H	DTCR3
9H	“
AH	“
BH	“
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE300H	DCR
1H	“
2H	“
3H	“
4H	RSR
5H	“
6H	“
7H	“
8H	
9H	
AH	
BH	
CH	DHR
DH	“
EH	“
FH	“

ADR	Register name
FFFFE310H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE320H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE330H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[9] CS/WAIT controller

ADR	Register name
FFFFE400H	BMA0
1H	“
2H	“
3H	“
4H	BMA1
5H	“
6H	“
7H	“
8H	BMA2
9H	“
AH	“
BH	“
CH	BMA3
DH	“
EH	“
FH	“

ADR	Register name
FFFFE410H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE480H	B01CS
1H	“
2H	“
3H	“
4H	B23CS
5H	“
6H	“
7H	“
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE490H	
1H	
2H	
3H	
4H	***reserved***
5H	“
6H	“
7H	“
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[10] FLASH control

ADR	Register name
FFFFE510H	SEQMOD
1H	“
2H	“
3H	“
4H	SEQCNT
5H	“
6H	“
7H	“
8H	ROMSEC1
9H	“
AH	“
BH	“
CH	ROMSEC2
DH	“
EH	“
FH	“

ADR	Register name
FFFFE520H	FLCS
1H	“
2H	“
3H	“
4H	FLPGEND
5H	“
6H	“
7H	“
8H	*reserved*
9H	*reserved*
AH	*reserved*
BH	*reserved*
CH	
DH	
EH	
FH	

[11] ROM correction

ADR	Register name
FFFFE540H	ADDREG0
1H	“
2H	“
3H	“
4H	ADDREG1
5H	“
6H	“
7H	“
8H	ADDREG2
9H	“
AH	“
BH	“
CH	ADDREG3
DH	“
EH	“
FH	“

ADR	Register name
FFFFE550H	ADDREG4
1H	“
2H	“
3H	“
4H	ADDREG5
5H	“
6H	“
7H	“
8H	ADDREG6
9H	“
AH	“
BH	“
CH	ADDREG7
DH	“
EH	“
FH	“

ADR	Register name
FFFFE560H	ADDREG8
1H	“
2H	“
3H	“
4H	ADDREG9
5H	“
6H	“
7H	“
8H	ADDREGA
9H	“
AH	“
BH	“
CH	ADDREGB
DH	“
EH	“
FH	“

ADR	Register name
FFFFE570H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE610H	*reserved*
1H	*reserved*
2H	*reserved*
3H	*reserved*
4H	*reserved*
5H	*reserved*
6H	*reserved*
7H	*reserved*
8H	*reserved*
9H	*reserved*
AH	*reserved*
BH	*reserved*
CH	*reserved*
DH	*reserved*
EH	*reserved*
FH	*reserved*

ADR	Register name
FFFFE620H	*reserved*
1H	*reserved*
2H	*reserved*
3H	*reserved*
4H	*reserved*
5H	*reserved*
6H	*reserved*
7H	*reserved*
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[12] UART/HSIO

ADR	Register name
FFFFE800H	HSC0BUF
1H	
2H	
3H	
4H	HBR0ADD
5H	HSC0MOD1
6H	HSC0MOD2
7H	HSC0EN
8H	HSC0RFC
9H	HSC0TFC
AH	HSC0RST
BH	HSC0TST
CH	HSC0FCNF
DH	HSC0CR
EH	HSC0MOD0
FH	HBR0CR

ADR	Register name
FFFFE810H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE820H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE830H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[13] High accuracy PPG (TMRD)

ADR	Register name
FFFFE900H	TD0RUN
1H	"
2H	"
3H	"
4H	TD0CR
5H	"
6H	"
7H	"
8H	TD0MOD
9H	"
AH	"
BH	"
CH	TD0BCR
DH	"
EH	"
FH	"

ADR	Register name
FFFFE910H	TD0ST
1H	"
2H	"
3H	"
4H	TD0REG0
5H	"
6H	"
7H	"
8H	TD0REG1
9H	"
AH	"
BH	"
CH	TD0REG2
DH	"
EH	"
FH	"

ADR	Register name
FFFFE920H	TD0REG3
1H	"
2H	"
3H	"
4H	TD0REG4
5H	"
6H	"
7H	"
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE930H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE940H	TD1RUN
1H	“
2H	“
3H	“
4H	TD1CR
5H	“
6H	“
7H	“
8H	TD1MOD
9H	“
AH	“
BH	“
CH	TD1BCR
DH	“
EH	“
FH	“

ADR	Register name
FFFFE950H	TD1ST
1H	“
2H	“
3H	“
4H	TD1REG0
5H	“
6H	“
7H	“
8H	TD1REG1
9H	“
AH	“
BH	“
CH	TD1REG2
DH	“
EH	“
FH	“

ADR	Register name
FFFFE960H	TD1REG3
1H	“
2H	“
3H	“
4H	TD1REG4
5H	“
6H	“
7H	“
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFE970H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[14] CG

ADR	Register name
FFFFEE00H	SYSCR0
1H	SYSCR1
2H	SYSCR2
3H	SYSCR3
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

ADR	Register name
FFFFEE10H	IMCGA
1H	“
2H	“
3H	“
4H	IMCGB
5H	“
6H	“
7H	“
8H	IMCGC
9H	“
AH	“
BH	“
CH	IMCGD
DH	“
EH	“
FH	“

ADR	Register name
FFFFEE20H	EICRCG
1H	“
2H	“
3H	“
4H	NMIFLG
5H	“
6H	“
7H	“
8H	PWMCG
9H	“
AH	“
BH	“
CH	
DH	
EH	
FH	

ADR	Register name
FFFFEE30H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

20. JTAG Interface

The TMP19A23 is equipped with the boundary scan interface that conforms to the Joint Test Action Group (JTAG) standard. This interface uses the industry-standard JTAG protocol (IEEE Standard 1149.1/D6). This chapter describes this JTAG interface with a mention of boundary scan, interface pins, interface signals, and test access ports (TAP).

20.1 Boundary Scan Overview

IC (Integrated Circuit) density is ever increasing, SMDs (Surface Mount Devices) continue to decrease in size, components are now mounted on both sides of printed circuit boards (PCBs), and there are considerable technical developments related to embedding holes. Conventional internal circuit testing techniques are dependent on the physical contact between internal circuitry and chips and, therefore, their limitations with respect to efficiency and accuracy are manifest. With the ever-increasing IC complexity, tests conducted to perform inspections on all chips integrated into an IC are becoming larger in scale, and it is becoming more difficult to design an efficient, reliable IC testing program.

To overcome this difficulty in performing IC tests, the "boundary scan" circuit was developed. It is a group of shift registers called "boundary scan circuit" established between pins and internal circuitry of IC connected to the pins (see Fig. 20-1). These boundary scan cells are bypassed under normal conditions. When an IC goes into test mode, data is sent from the boundary scan cells through the shift register bus in response to the instruction given by a test program, and various diagnostic tests are executed. In IC tests, five signals TDI, TDO, TMS, TCK and $\overline{\text{TRST}}$ are used. These signals are explained in the next section.

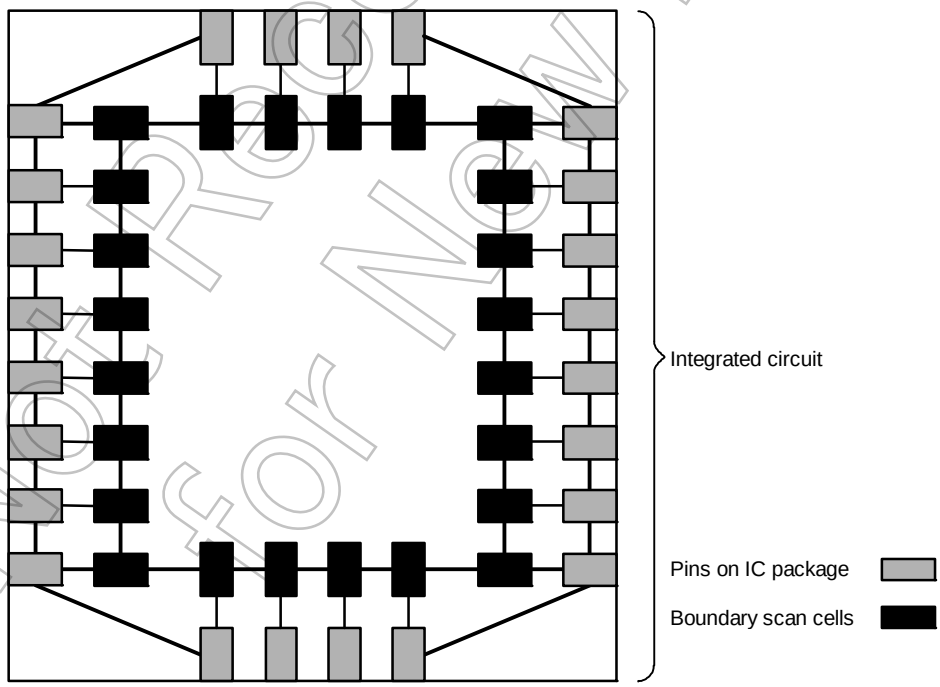


Fig. 20-1 JTAG Boundary Scan Cells

Note) The optional instructions IDCODE, USERCODE, INTEST and RUNBIST are not implemented in the TMP19A23.

20.2 JTAG Interface Signals

JTAG interface signals are as follows (see Fig. 20-2).

- TDI To input JTAG serial data
- TDO To output JTAG serial data
- TMS To select JTAG test mode
- TCK To input JTAG serial clock
- $\overline{\text{TRST}}$ To input JTAG test reset

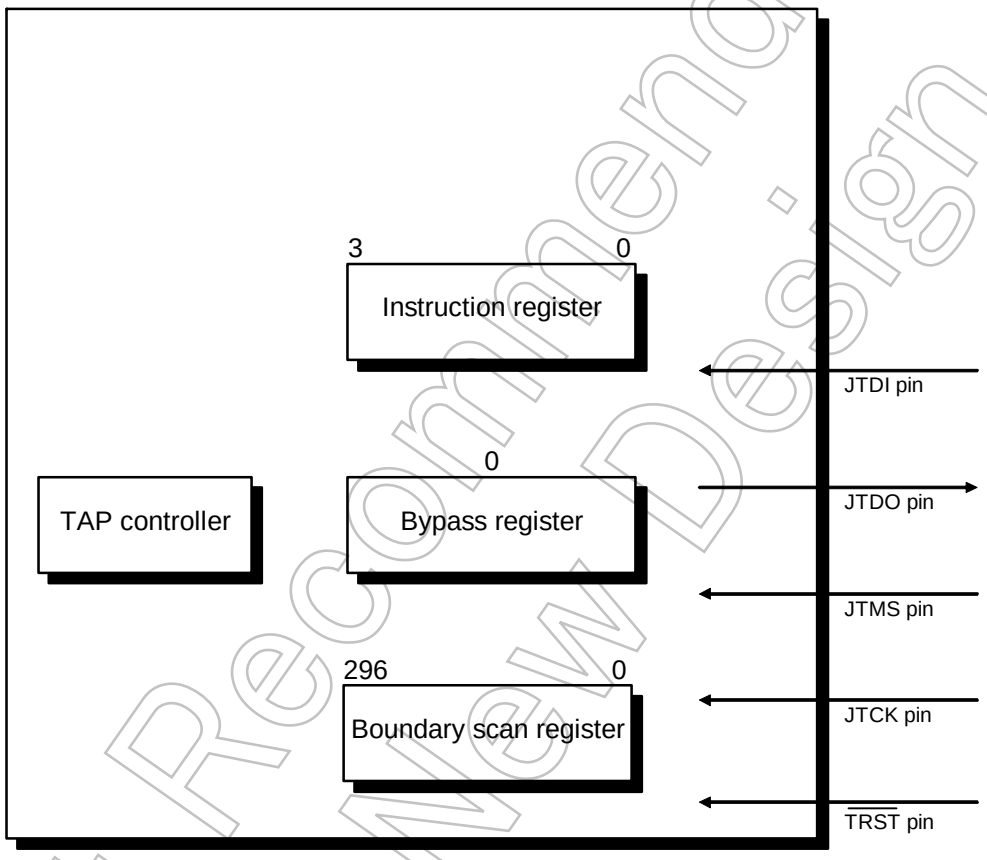


Fig. 20-2 JTAG Interface Signals and Registers

The JTAG boundary scan mechanism (hereafter called "JTAG mechanism") enables testing of the processor, printed circuit boards connected to the processor, and connections between other components on printed circuit boards.

The JTAG mechanism does not have a function of testing the processor itself.

20.3 JTAG Controller and Registers

The following JTAG controller and registers are built into the processor:

- Instruction register
- Boundary scan register
- Bypass register
- Device identification register
- Test Access Port (TAP) controller

In the JTAG basic mechanism, the TAP controller state machine monitors the signals input through the JTMS pin. As the JTAG mechanism starts operation, the TAP controller determines a test function to be executed by loading data into the JTAG instruction register (IR) and performing a serial data scan via the data register (DR), as shown in Table 20-3. When data is scanned, the state of the JTMS pin represents new specific data words and the end of data flow. The data register is selected according to data loaded into the instruction register.

20.3.1 Instruction Register

The JTAG instruction register consists of four cells, including shift registers. It is used to select either a test to be executed or a test data register to be accessed or to select both. Either the boundary scan register or the bypass register is selected according to combinations shown in Table 20-3.

Table 20-3 Bit Configurations of the JTAG Instruction Register

Instruction code Most significant to least significant bit	Instruction	Data register to be selected
0000	EXTEST	Boundary scan register
0001	SAMPLE/PRELOAD	Boundary scan register
0010 ~1110	Reserved	Reserved
1111	BYPASS	Bypass register

Fig. 20-4 shows the format of the instruction register.

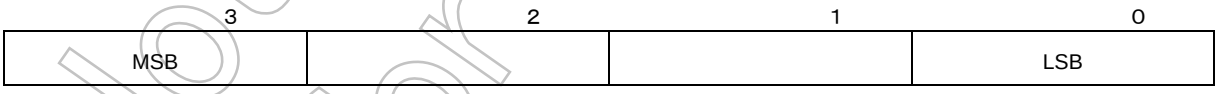


Fig. 20-4 Instruction Register

The instruction code is shifted from the least significant bit to the instruction register.

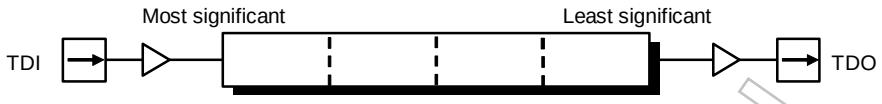


Fig. 20-5 Shift Direction of the Instruction Code to the Instruction Register

20.3.2 Bypass Register

The bypass register has a one-bit width. If the TAP controller is in the Shift-DR state (bypass state), data at the TDI pin is shifted into the bypass register, and the output from the bypass register is shifted out to the TDO output pin.

Simply put, the bypass register is a circuit for bypassing the devices in a serial boundary scan chain connected to the substrates that are not required for a test to be conducted. Fig. 20-6 shows the logical position of the bypass register in a boundary scan chain.

If the bypass register is used, the speed of access to boundary scan registers in an active IC in a data path used for substrate level testing can be increased.

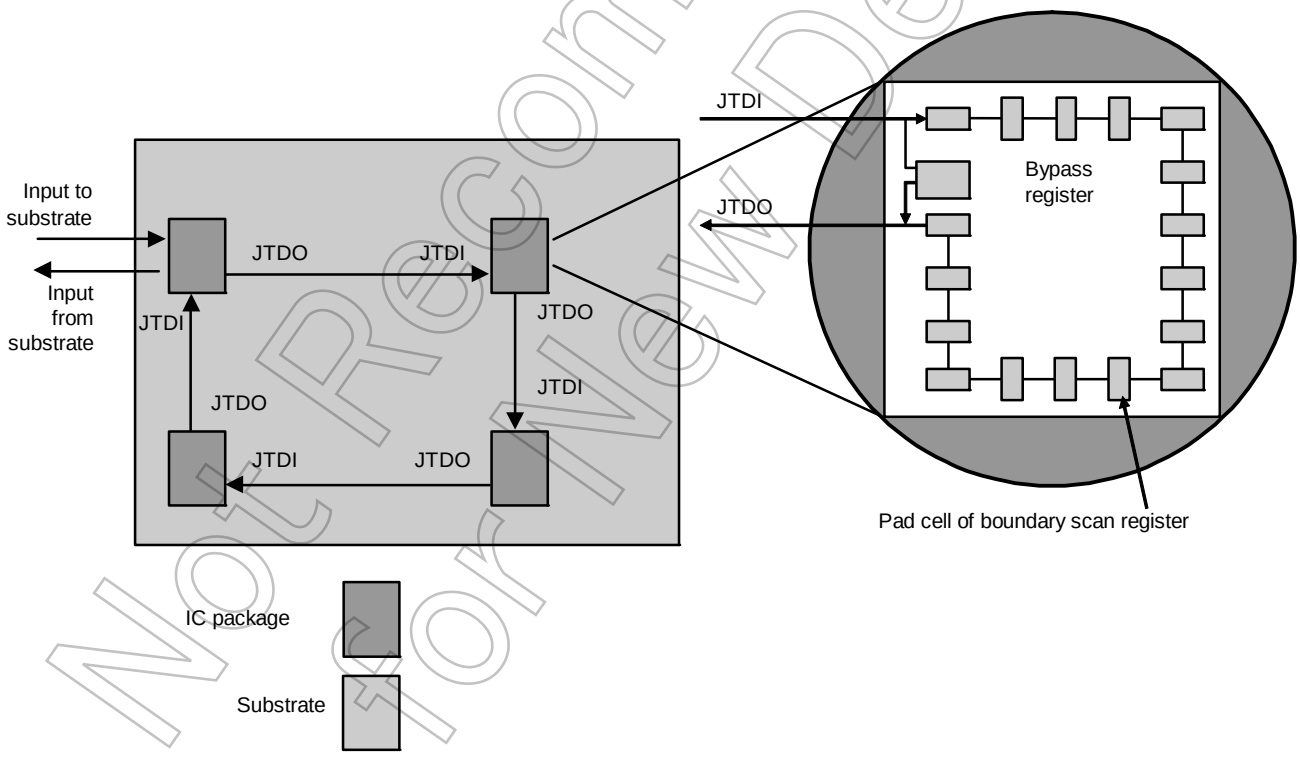


Fig. 20-6 Function of the Bypass Register

20.3.3 Boundary Scan Register

The boundary scan register has inputs and outputs for some analog output signals, as well as all signals from the TMP19A23 except control signals. Pins of the TMP19A23 can drive any test patterns by scanning data into the boundary scan register in the Shift-DR state. After the boundary scan register goes into the Capture-DR state, data enters the processor, is shifted, and inspected.

The boundary scan register forms a data path. It basically functions as a single shift register of 297-bit width. Cells in this data path are connected to all input and output pads of the TMP19A23.

The TDI input is introduced to the least significant bit (LSB) in the boundary scan register. The most significant bit in the boundary scan register is taken out of the TDO output.

20.3.4 Test Access Port (TAP)

The test access port (TAP) consists of five signal pins: $\overline{\text{TRST}}$, TDI, TDO, TMS, and TCK. Serial test data, instructions and test control signals are sent and received through these signal pins.

Data is serially scanned into one of three registers (instruction register, bypass register and boundary scan register) via the TDI pin or it is scanned out from one of these three registers into the TDO pin, as shown in Fig. 20-7.

The TMS input is used to control the state transitions of the main TAP controller state machine. The TCK input is a test clock exclusively for shifting serial JTAG data synchronously; it works independently of a chip core clock or a system clock.

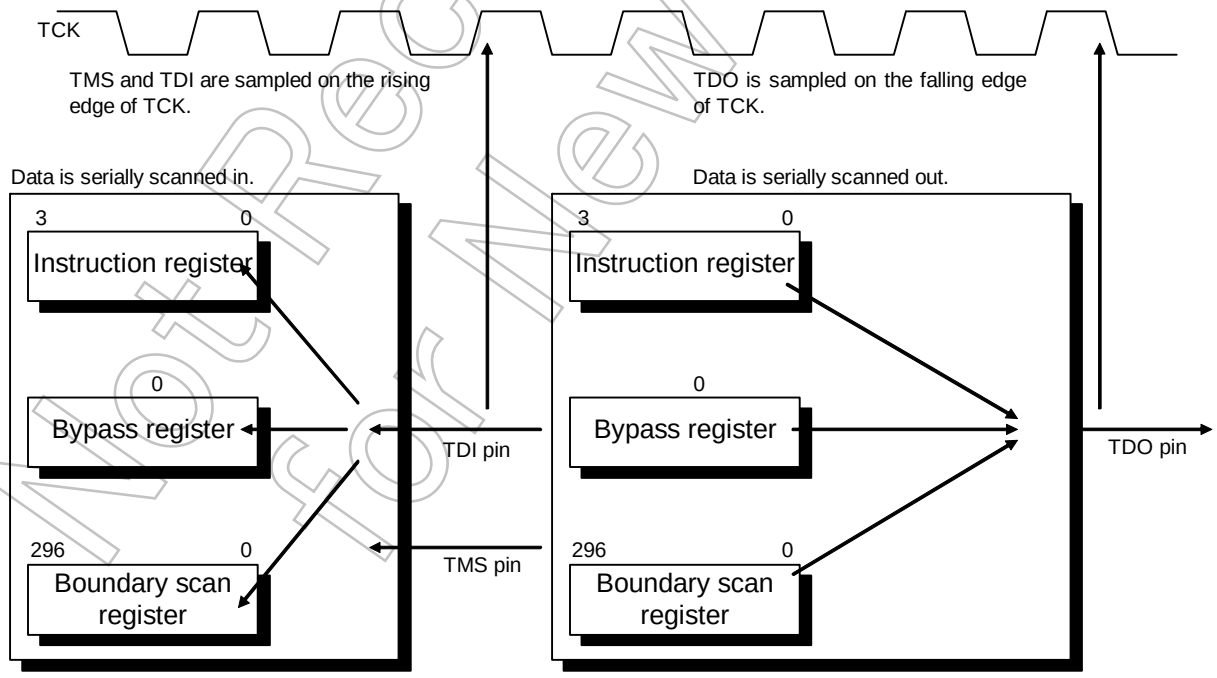


Fig. 20-7 JTAG Test Access Port

Data through the TDI and TMS pins are sampled on the rising edge of the input clock signal TCK. Data through the TDO pin changes on the falling edge of the clock signal TCK.

20.3.5 TAP Controller

In the processor, a 16-state TAP controller specified in the IEEE JTAG standard is implemented.

20.3.6 Controller Reset

To reset the state machine of the TAP controller,

- assert the $\overline{\text{TRST}}$ signal input (Low) to reset the TAP controller or
- continue to assert the input signal TMS by using the rising edge of the TCK input five times successively after clearing the reset state of the processor.

The reset state can be maintained by keeping TMS in an asserted state.

20.3.7 State Transitions of the TAP Controller

Fig. 20-8 shows the state transitions of the TAP controller. The state of the TAP controller changes depending on which value TMS will select on the rising edge of TCK, 0 or 1. In this figure, the arrow shows a state transition and the value that TMS selects to execute each state transition is shown alongside of the arrow.

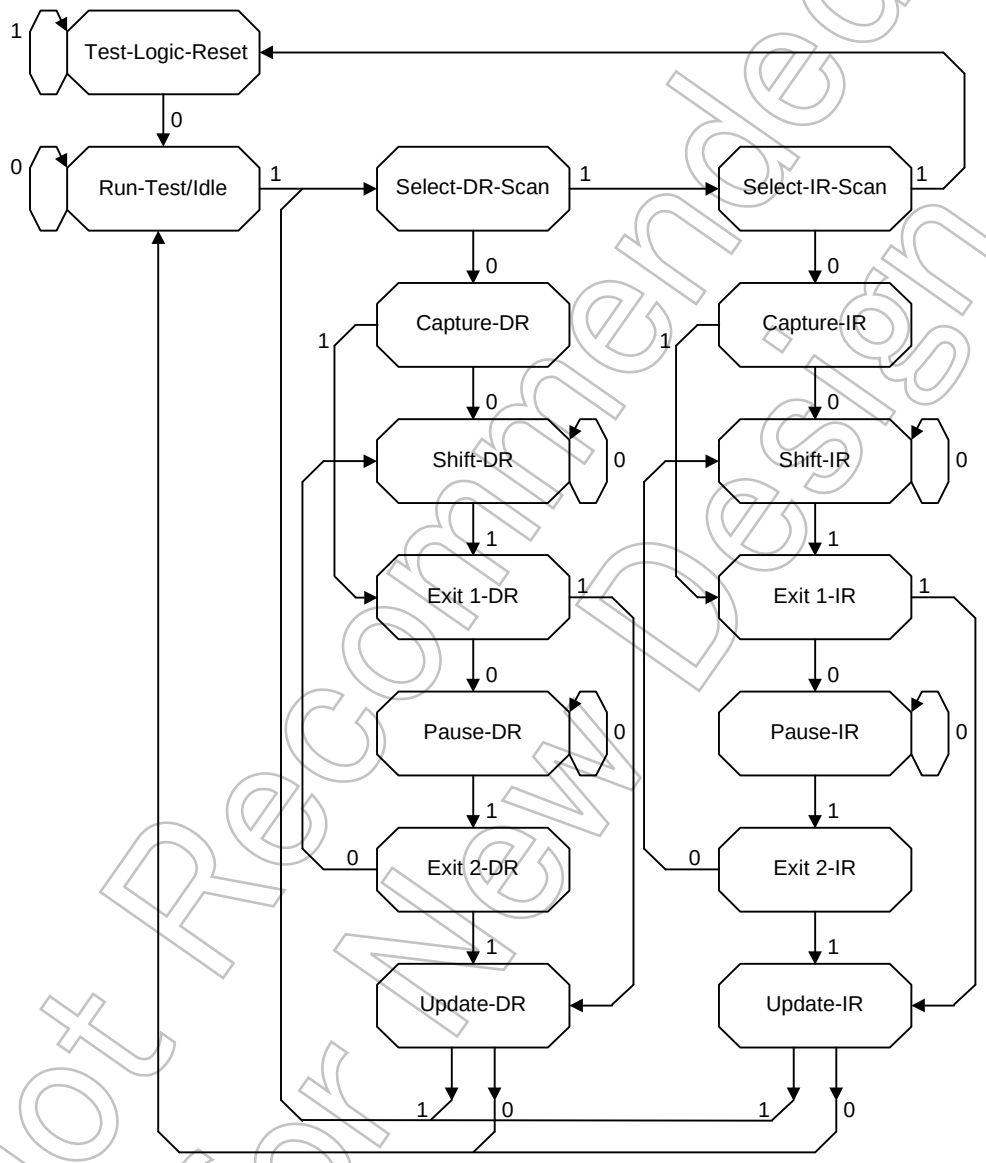


Fig. 20-8 State Transition Diagram of the TAP Controller

The TAP controller operates in each state described below. In Fig. 20-8, a column to the left is the data column and a column to the right is the instruction column. The data column represents the data register (DR), and the instruction column represents the instruction register (IR).

- Test-Logic-Reset
  If the TAP controller is in a reset state, the device identification register is selected by default. The most significant bit in the boundary scan register is cleared to "0," and the output is disabled. The TAP controller remains in the Test-Logic-Reset state if TMS is "1." If "0" is input into TMS in the Test-Logic-Reset state, the TAP controller goes into the Run-Test/Idle state.

- Run-Test/Idle
  In the Run-Test/Idle state, the IC goes into test mode only if a specific instruction, such as the built-in self test (BIST) instruction, is issued. If an instruction that cannot be executed in the Run-Test/Idle state has been issued, the test data register selected by the last instruction maintains the existing state.
  The TAP controller remains in the Run-Test/Idle state if TMS is "0." If "1" is input into TMS, the TAP controller goes into the Select-DR-Scan state.

- Select-DR-Scan
  The Select-DR-Scan state of the TAP controller is a transient state. In this state, the IC performs no operations. If "0" is input into TMS when the TAP controller is in the Select-DR-Scan state, the TAP controller goes into the Capture-DR state. If "1" is input into TMS, the instruction column goes into the Select-IR-Scan state.

- Select-IR-Scan
  The Select-IR-Scan state of the TAP controller is a transient state. In this state, the IC performs no operations.
  If "0" is input into TMS when the TAP controller is in the Select-IR-Scan state, the TAP controller goes into the Capture-IR state. If "1" is input into TMS, the TAP controller returns to the Test-Logic-Reset state.

- Capture-DR
  If the data register selected by the instruction register has parallel inputs when the TAP controller is in the Capture-DR state, data is loaded into the data register in a parallel fashion. If the data register does not have parallel inputs or if data does not need to be loaded into the selected test data register, the data register maintains the existing state.
  If "0" is input into TMS when the TAP controller is in the Capture-DR state, the TAP controller goes into the Shift-DR state. If "1" is input into TMS, the TAP controller goes into the Exit 1-DR state.

- Shift-DR
  If the TAP controller is in the Shift-DR state, data is serially shifted out by the data register connected between TDI and TDO.
  If the TAP controller is in the Shift-DR state, the Shift-DR state is maintained while TMS is "0." If "1" is input into TMS, the TAP controller goes into the Exit 1-DR state.

- Exit 1-DR
  The Exit 1-DR state of the TAP controller is a transient state.
  If "0" is input into TMS when the TAP controller is in the Exit 1-DR state, the TAP controller goes into the Pause-DR state. If "1" is input into TMS, it goes into the Update-DR state.

- Pause-DR
  In the Pause-DR state, the shift operation performed by the data register selected by the instruction register is temporarily suspended. The instruction register and the data register maintain their existing state.
  The TAP controller remains in the Pause-DR state while TMS is "0." If "1" is input into TMS, it goes into the Exit 2-DR state.

- Exit 2-DR
  The Exit 2-DR state of the TAP controller is a transient state.
  If "0" is input into TMS when the TAP controller is in the Exit 2-DR state, the TAP controller returns to the Shift-DR state. If "1" is input into TMS, it goes into the Update-DR state.

- Update-DR
  In the Update-DR state, data is output in a parallel fashion from the data register having a parallel output synchronously to the rising edge of TCK. The data register with a parallel output latch does not output data during the shift operation; it outputs data only in the Update-DR state.
  If "0" is input into TMS when the TAP controller is in the Update-DR state, the TAP controller goes into the Run-Test/Idle state. If "1" is input into TMS, it goes into the Select-DR-Scan state.

- Capture-IR
  In the Capture-IR state, data is loaded into the instruction register in a parallel fashion. Data loaded is 0001. The Capture-IR state is used to test the instruction register. A malfunction of the instruction register can be detected by shifting out the data loaded.
  If "0" is input into TMS when the TAP controller is in the Capture-IR state, the TAP controller goes into the Shift-IR state. If "1" is input into TMS, it goes into the Exit 1-IR state.

- Shift-IR
  In the Shift-IR state, the instruction register is connected between TDI and TDO, and data loaded synchronously to the rising edge of TCK is serially shifted out.
  The TAP controller remains in the Shift-IR state while TMS is "0." If "1" is input into TMS, the TAP controller goes into the Exit 1-IR state.

- Exit 1-IR
  The Exit 1-IR state of the TAP controller is a transient state.
  If "0" is input into TMS when the TAP controller is in the Exit 1-IR state, the TAP controller goes into the Pause-IR state. If "1" is input into TMS, it goes into the Update-IR state.

- Pause-IR
  In the Pause-IR state, the shift operation performed by the instruction register is temporarily suspended. The existing state of the instruction register and that of the data register are maintained. The TAP controller remains in the Pause-IR state while TMS is "0." If "1" is input into TMS, it goes into the Exit 2-IR state.

- Exit 2-IR
  The Exit 2-IR state of the TAP controller is a transient state.

  If "0" is input into TMS when the TAP controller is in the Exit 2-IR state, the TAP controller goes into the Shift-IR state. If "1" is input into TMS, it goes into the Update-IR state.

- Update-IR
  In the Update-IR state, an instruction is updated by outputting the instruction in a parallel fashion synchronously to the rising edge of TCK.

  If "0" is input into TMS when the TAP controller is in the Update-IR state, the TAP controller goes into the Run-Test/Idle state. If "1" is input into TMS, it goes into the Select-DR-Scan state.

Table 20-9 shows the boundary scan sequence relative to processor signals.

Table 20-9 JTAG Scan Sequence Relative to the TMP19A23 Processor Pins

1:PC0	2:PC1	3:PC2	4:PC3	5:BW1	6:PD7	7:PD6
8:PD5	9:PD4	10:PD3	11:PD2	12:PD1	13:PD0	14:PE5
15:PE4	16:PE3	17:PE2	18:PE1	19:PE0	20:DINT	21:PA0
22:PA1	23:PA2	24:PA3	25:PA4	26:PA5	27:PA6	28:PA7
29:PB7	30:PB6	31:PB5	32:PB4	33:PB3	34:PB2	35:PB1
36:PB0	37:P67	38:P66	39:P65	40:P64	41:P63	42:P62
43:P61	44:P60	45:P57	46:P56	47:P55	48:P54	49:BOOT
50:P53	51:P52	52:P51	53:P50	54:P27	55:P26	56:P25
57:P24	58:P23	59:P22	60:P21	61:P20	62:P17	63:P16
64:P15	65:P14	66:P13	67:P12	68:P11	69:P10	70:P07
71:P06	72:P05	73:P04	74:P03	75:P02	76:P01	77:P00
78:P40	79:P41	80:P42	81:P43	82:P44	83:P45	84:P46
85:P47	86:P30	87:P31	88:P32	89:P33	90:P34	91:P35
92:P36	93:P37	94:P70	95:P71	96:P72	97:P73	98:P74
99:P75	100:P76	101:P77	102:P80	103:P81	104:P82	105:P83
106:P84	107:P90	108:P91	109:P92	110:P93	111:P94	112:P95
113:P96	114:P97					

(Note) The pins that can be JTAG-scanned are shown above.

20.4 Instructions Supported by the JTAG Controller Cells

This section describes the instructions supported by the JTAG controller cells of the TMP19A23.

20.4.1 EXTEST instruction

The EXTEST instruction is used for external interconnect test. If this instruction is issued, the BSR cells at output pins output test patterns in the Update-DR state, and the BSR cells at input pins capture test results in the Capture-DR state.

Before the EXTEST instruction is selected, the boundary scan register is usually initialized using the SAMPLE/PRELOAD instruction. If the boundary scan register has not been initialized, there is the possibility that indeterminate data will be transmitted in the Update-DR state and bus conflicts may occur between ICs. Fig. 20-10 shows the flow of data while the EXTEST instruction is selected.

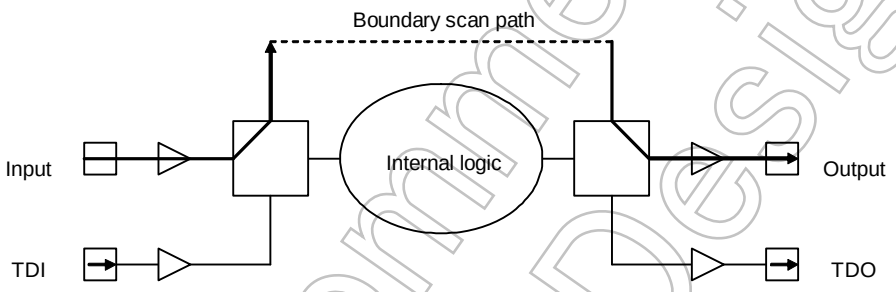


Fig. 20-10 Flow of Data While the EXTEST Instruction Is Selected

The basic external interconnect test procedure is as follows:

1. Initialize the TAP controller to put it in the Test-Logic-Reset state.
2. Load the SAMPLE/PRELOAD instruction into the instruction register. This allows the boundary scan register to be connected between TDI and TDO.
3. Initialize the boundary scan register by shifting in determinate data.
4. Load the initial test data into the boundary scan register.
5. Load the EXTEST instruction into the instruction register.
6. Capture the data applied to the input pin and input it into the boundary scan register.
7. Shift out the captured data while simultaneously shifting in the next test pattern.
8. Output the test pattern that was shifted into the boundary scan register for output to the output pin.

Repeat steps 6 through 8 for each test pattern.

When using EXTEST instruction, please note that malfunction may occur depending on the data input from terminal pin on ground that CPU is in operating state and make sure to execute the test after the system reset is released.

20.4.2 SAMPLE and PRELOAD Instructions

The SAMPLE and PRELOAD instructions are used to connect TDI and TDO by way of the boundary scan register. This instruction has two functions.

- The SAMPLE instruction is used to monitor the I/O pad of an IC. While SAMPLE is monitoring the I/O pads, the internal logic is not disconnected from the I/O pins of an IC. This instruction is executed in the Capture-DR state. A main function of SAMPLE is to read values of the I/O pins of an IC at the rising edge of TCK during normal functional operation. Fig. 20-11 shows the flow of data while the SAMPLE instruction is selected.

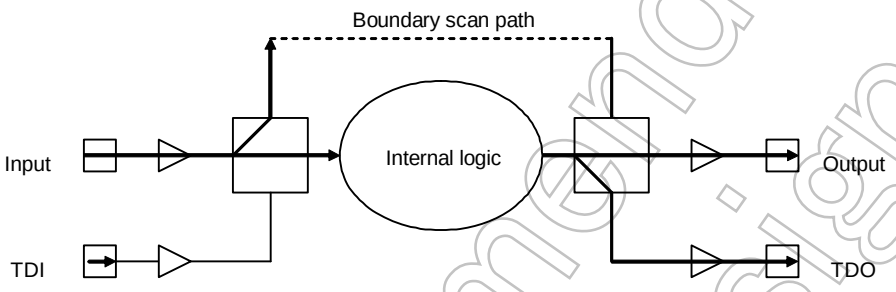


Fig. 20-11 Flow of Data While SAMPLE Is Selected

- The PRELOAD instruction is used to initialize the boundary scan register before selecting other instructions. For example, the boundary scan register is initialized using PRELOAD before selecting the EXTEST instruction, as previously explained. PRELOAD shifts data into the boundary scan register without affecting the normal operation of the system logic. Fig. 20-12 shows the flow of data while the PRELOAD instruction is selected.

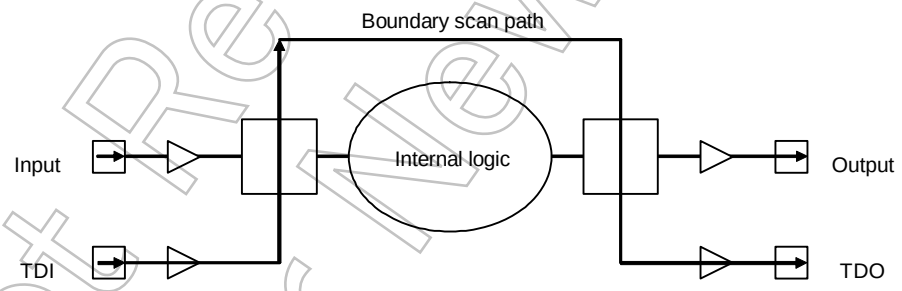


Fig. 20-12 Flow of Test Data While PRELOAD Is Selected

When using the SAMPLE instruction, complete the instruction update during the system reset. After the reset is released, do not switch the TAP instruction.

20.4.3 BYPASS Instruction

When conducting the type of test in which an IC does not need to be controlled or monitored, the BYPASS instruction is used to form the shortest serial path bypassing an IC by connecting the bypass register between JTDI and JTDO. The BYPASS instruction does not affect the normal operation of the system logic implemented on a chip. Data passes through the bypass register while the BYPASS instruction is selected, as shown in Fig. 20-13.

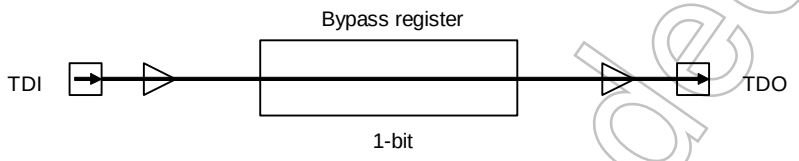


Fig. 20-13 Flow of Data While the Bypass Register Is Selected

20.5 Points to Note

This section describes the points to note regarding JTAG boundary scan operations implemented in this processor.

- The X2 and X1 signal pads do not comply with JTAG.

- To reset the JTAG circuit, execute either of the following:
  ① Initialize the JTAG circuit by asserting $\overline{\text{TRST}}$, and then negate $\overline{\text{TRST}}$.
- ② Set the TMS pin to "1," and supply TCK with more than 5 clocks.

21 Flash Memory Operation

This section describes the hardware configuration and operation of the flash memory.

21.1 Flash Memory

21.1.1 Features

1) Memory capacity
The TMP19A23FYFG/XBG device contains 2M bits (256 kB) of flash memory capacity. The memory area consists of 2 independent memory blocks (128 kB × 2) to enable independent write access to each block. When the CPU is to access the internal flash memory, 32-bit data bus width is used.

2) Flash memory access
Interleave access is used in this device.

3) Write/erase time
Write time: 1sec/Chip (Typ) 0.5sec/128Kbyte (Typ.)
Erase: 0.2sec/Chip (Typ) 0.1msec/128Kbyte (Typ.)
(Note) The above values are theoretical values not including data transfer time.
The write time per chip depends on the write method to be used by the user.

4) Programming method
The onboard programming mode is available for the user to program (rewrite) the device while it is mounted on the user's board.

・The onboard programming mode

4-1) User boot mode
The user's original rewriting method can be supported.

4-2) Single boot mode
The rewriting method to use serial data transfer (Toshiba's unique method) can be supported.

Rewriting method
The flash memory included in this device is generally compliant with the applicable JEDEC standards except for some specific functions. Therefore, if the user is currently using an external flash memory device, it is easy to implement the functions into this device. Furthermore, the user is not required to build his/her own programs to realize complicated write and erase functions because such functions are automatically performed using the circuits already built-in the flash memory chip.
This device is also implemented with a read-protect function to inhibit reading flash memory data from any external writer device. On the other hand, rewrite protection is available only through command-based software programming; any hardware setting method to apply +12VDC is not supported. The above described protection function is automatically enabled when the two areas are both configured for protection. To release the flash memory protection, you need to release ROM data protection before releasing the block protection. If not, all the internal data is automatically erased and then all the protections are released.

JEDEC compliant functions	Modified, added, or deleted functions
• Automatic programming • Automatic chip erase • Automatic block erase • Data polling/toggle bit	<Modified> Block protect (only software protection is supported) <Deleted> Erase resume - suspend function <Added> Automatic multiple block erase (supported to the chip level)

21.1.2 Block Diagram of the Flash Memory Section

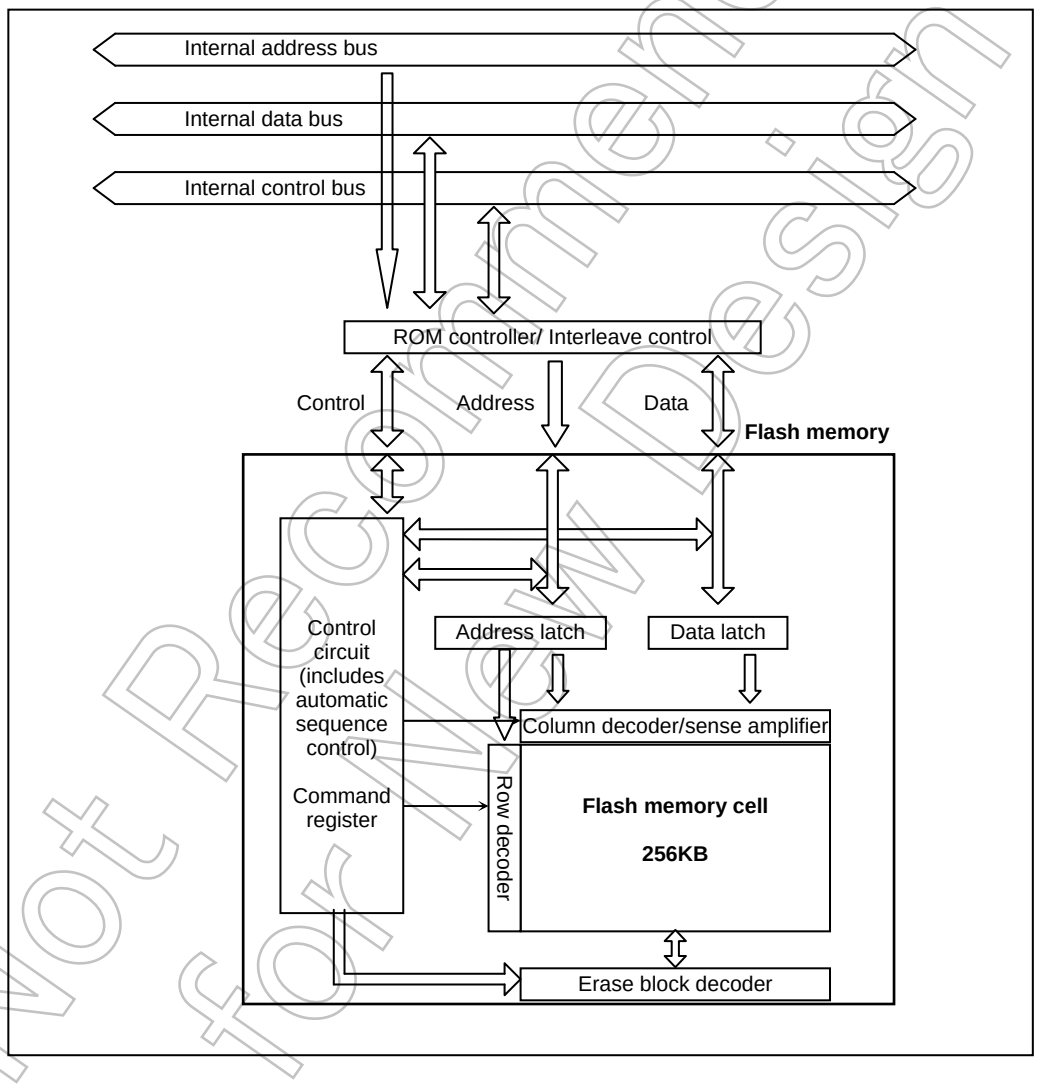


Fig. 21-1 Block Diagram of the Flash Memory Section

21.2 Operation Mode

This device has three operation modes including the mode not to use the internal flash memory.

Table 21-1 Operation Modes

Operation mode		Operation details
Single chip mode		After reset is cleared, it starts up from the internal flash memory.
	Normal mode	In this operation mode, two different modes, i.e., the mode to execute user application programs and the mode to rewrite the flash memory onboard the user's card, are defined. The former is referred to as "normal mode" and the latter "user boot mode.
	User boot mode	The user can uniquely configure the system to switch between these two modes. For example, the user can freely design the system such that the normal mode is selected when the port "00" is set to "1" and the user boot mode is selected when it is set to "0." The user should prepare a routine as part of the application program to make the decision on the selection of the modes.
Single boot mode		After reset is cleared, it starts up from the internal Boot ROM (Mask ROM). In the Boot ROM, an algorithm to enable flash memory rewriting on the user's set through the serial port of this device is programmed. By connecting to an external host computer through the serial port, the internal flash memory can be programmed by transferring data in accordance with predefined protocols.

Among the flash memory operation modes listed in the above table, the User Boot mode and the Single Boot mode are the programmable modes. These two modes, the User Boot mode and the Single Boot mode, are referred to as "Onboard Programming" modes where onboard rewriting of internal flash memory can be made on the user's card.

Either the Single Chip or Single Boot operation mode can be selected by externally setting the level of the $\overline{\text{BOOT}}$ input pin while the device is in reset status.

After the level is set, the CPU starts operation in the selected operation mode when the reset condition is removed. Regarding the BOOT pin, be sure not to change the levels during operation once the mode is selected.

The mode setting method and the mode transition diagram are shown below:

Table 21-2 Operation Mode Setting

Operation mode	Input pin	
	$\overline{\text{RESET}}$	$\overline{\text{BOOT}}$
Single chip mode	0 → 1	1
Single boot mode	0 → 1	0

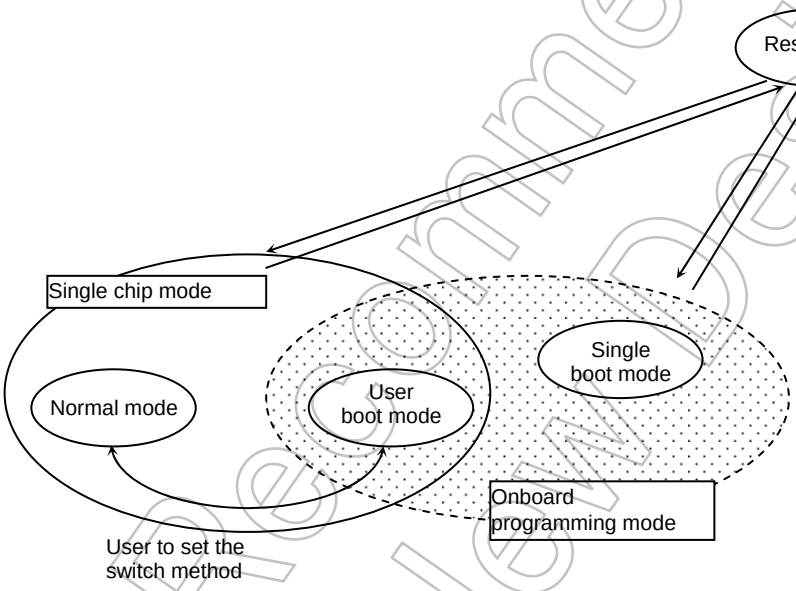


Fig. 21-2 Mode Transition Diagram

21.2.1 Reset Operation

To reset the device, ensure that the power supply voltage is within the operating voltage range, that the internal oscillator has been stabilized, and that the $\overline{\text{RESET}}$ input is held at "0" for a minimum duration of 12 system clocks (1.8 μs with 54MHz operation; the "1/8" clock gear mode is applied after reset).

(Note 1) Regarding power-on reset of devices with internal flash memory;
for devices with internal flash memory, it is necessary to apply "0" to the $\overline{\text{RESET}}$ inputs upon power on for a minimum duration of 500 microseconds regardless of the operating frequency.

(Note 2) While flash programming or deletion is in progress, at least 0.5 microseconds of reset period is required regardless of the system clock frequency.

21.2.2 DSU (EJTAG) - PROBE Interface

This interface is used when the DSU probe is used in debugging. This is the dedicated interface for connection to the DSU probe. Please refer to the operation manual for the DSU probe you are going to use for details of debugging procedures to use the DSU probe. Here, the function to enable/disable the DSU probe in the DSU (EJTAG) mode is described.

1) Protect function

This device allows use of on-board DSU probes for debugging. To facilitate this, the device is implemented with a protection function to prevent easy reading of the internal flash memory by a third party other than the authorized user. By enabling the protection function, it becomes impossible to read the internal flash memory from a DSU probe. Use this function together with the protection function of the internal flash memory itself as described later.

2) DSU probe enable/disable function

This device allows use of on-board DSU probes for debugging operations. To facilitate this, the device is implemented with the "DSU probe inhibit" function (hereafter referred to as the "DSU inhibit" function) to prevent easy reading of the internal flash memory by a third party other than the authorized user. By enabling the DSU inhibit function, use of any DSU probe becomes impossible.

3) DSU enable (Enables use of DSU probes for debugging)

In order to prevent the DSU inhibit function from being accidentally removed by system runaway, etc., the method to cancel the DSU inhibit function is in double protection structure so it is necessary to set SEQMOD <DSUOFF> to "0" and also write the protect code "0x0000_00C5" to the DSU inhibit control register SEQCNT to cancel the function. Then, debugging to use a DSU probe can be allowed. While power to the device is still applied, setting SEQMOD <SEQON> to "1" and writing "0x0000_00C5" to the SEQCNT register will enable the protection function again.

Table 21-3 DSU Protect Mode Register

SEQMOD (0xFFFF_E510)

	7	6	5	4	3	2	1	0
Bit Symbol								DSUOFF
Read/Write	R							R/W
After reset	0							1
Function	Always reads "0."							1: DSU disable 0: DSU enable
	15	14	13	12	11	10	9	8
Bit Symbol								
Read/Write	R							
After reset	0							
Function	Always reads "0."							
	23	22	21	20	19	18	17	16
Bit Symbol								
Read/Write	R							
After reset	0							
Function								
	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset	0							
Function	Always reads "0."							

(Note 1)This register must be 32-bit accessed.
(Note 2)This register is initialized only by power-on reset. It is not initialized in reset usually.

Table 21-4 DSU Protect Control Register

SEQCNT (0xFFFF_E514)

	7	6	5	4	3	2	1	0
Bit Symbol								
Read/Write	W							
After reset								
Function	Write "0x0000_00C5".							
	15	14	13	12	11	10	9	8
Bit Symbol								
Read/Write	W							
After reset								
Function	Write “0x0000_00C5”.							
	23	22	21	20	19	18	17	16
Bit Symbol								
Read/Write	W							
After reset								
Function	Write “0x0000_00C5”.							
	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	W							
After reset								
Function	Write “0x0000_00C5”.							

(Note) This register must be 32-bit accessed.

4) Example use by the user

An example to use a DSU probe together with this function is shown as follows:

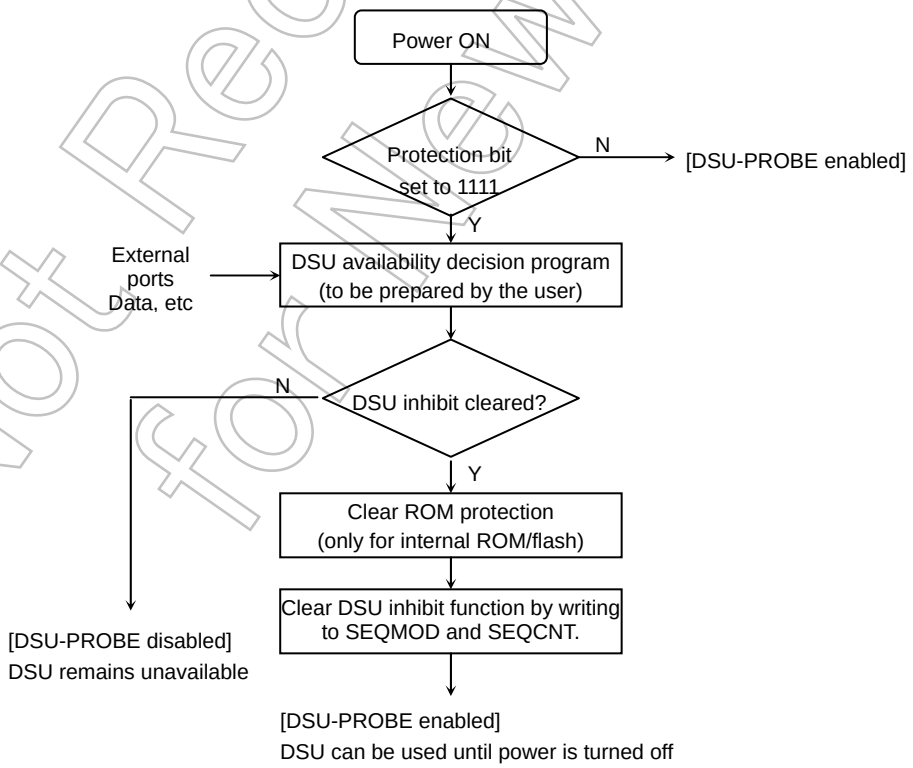


Fig. 21-3 Example Use of DSU Inhibit Function

21.2.3 User Boot Mode (ingle chip mode)

User Boot mode is to use flash memory programming routine defined by users. It is used when the data transfer buses for flash memory program code on the old application and for serial I/O are different. It operates at the single chip mode; therefore, a switch from normal mode in which user application is activated at the single chip mode to User Boot Mode for programming flash is required. Specifically, add a mode judgment routine to a reset program in the old application.

The condition to switch the modes needs to be set by using the I/O of 19A23 in conformity with the user's system setup condition. Also, flash memory programming routine that the user uniquely makes up needs to be set in the new application. This routine is used for programming after being switched to User Boot Mode. The execution of the programming routine must take place while it is stored in the area other than the flash memory since the data in the internal flash memory cannot be read out during delete/ writing mode. Once re-programming is complete, it is recommended to protect relevant flash blocks from accidental corruption during subsequent Single-Chip (Normal mode) operations. All the interruption including a non-maskable are inhibited at User Boot Mode.

(1-A) and (1-B) are the examples of programming with routines in the internal flash memory and in the external memory. For a detailed description of the erase and program sequence, refer to 21.3 On-board Programming of Flash Memory (Rewrite/Erase).

User Boot Mode

(1-A) Method 1: Storing a Programming Routine in the Flash Memory

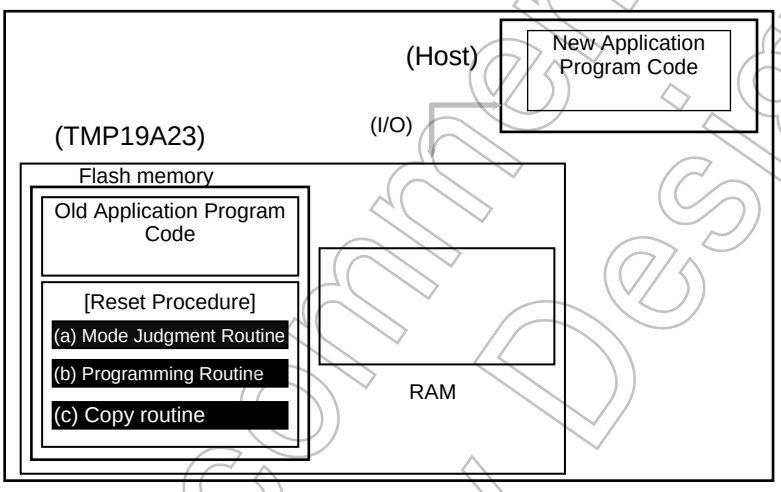
(Step-1)

Determine the conditions (e.g., pin states) required for the flash memory to enter User Boot mode and the I/O bus to be used to transfer new program code. Create hardware and software accordingly. Before installing the TMP19A23 on a printed circuit board, write the following program routines into an arbitrary flash block using programming equipment.

(a) Mode judgment routine: Code to determine whether or not to switch to User Boot mode

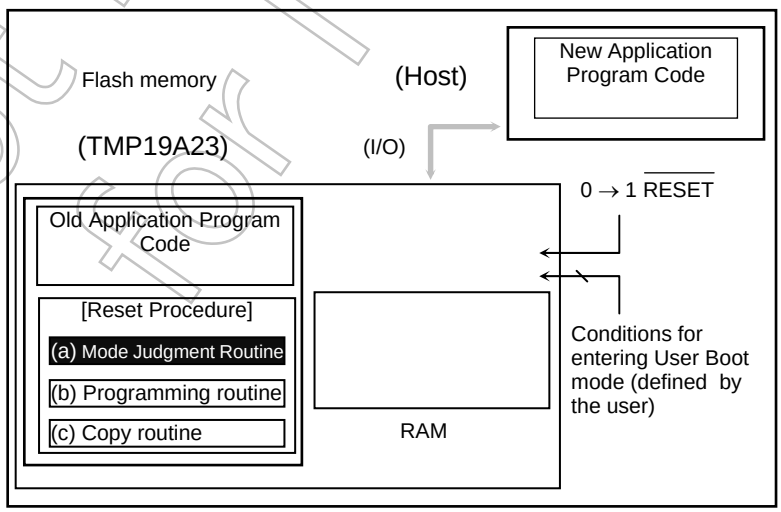
(b) Programming routine: Code to download new program code from a host controller and re-program the flash memory

(c) Copy routine: Code to copy the data described in (b) from the TMP19A23 flash memory to either the TMP19A423 on-chip RAM or external memory device.



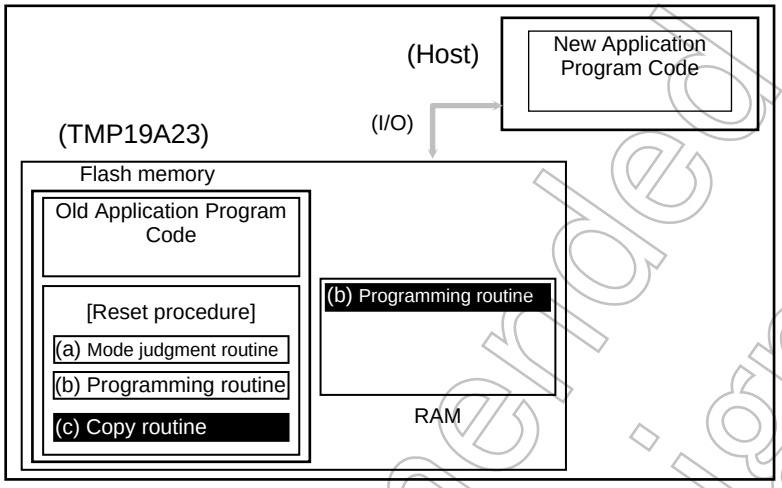
(Step-2)

After $\overline{\text{RESET}}$ is released, the reset procedure determines whether to put the TMP19A23 flash memory in User Boot mode. If mode switching conditions are met, the flash memory enters User Boot mode. (All interrupts including NMI must be disabled while in User Boot mode.)



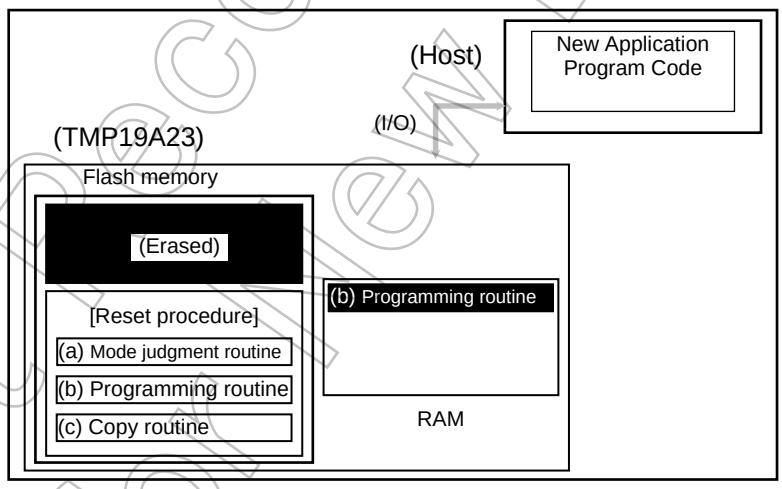
(Step-3)

Once transition to User Boot mode is occurred, execute the copy routine (c) to copy the flash programming routine (b) to either the TMP19A23 on-chip RAM or an external memory device. (In the following figure, the on-chip RAM is used).



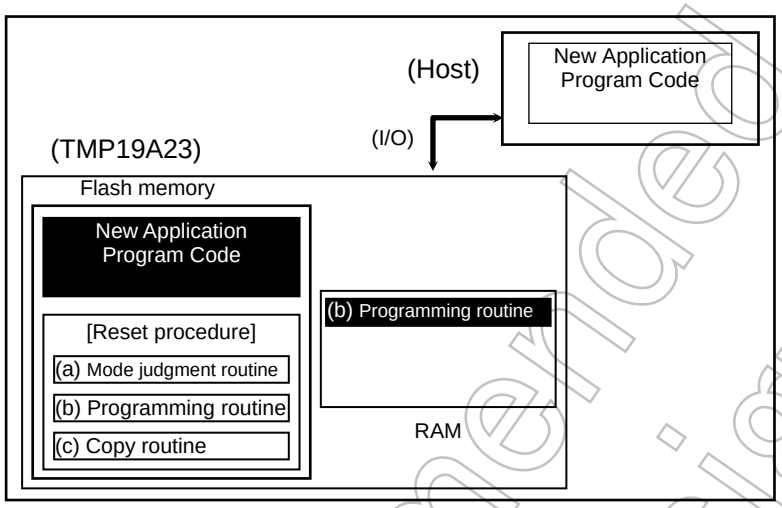
(Step-4)

Jump program execution to the flash programming routine in the on-chip RAM to erase a flash block containing the old application program code.



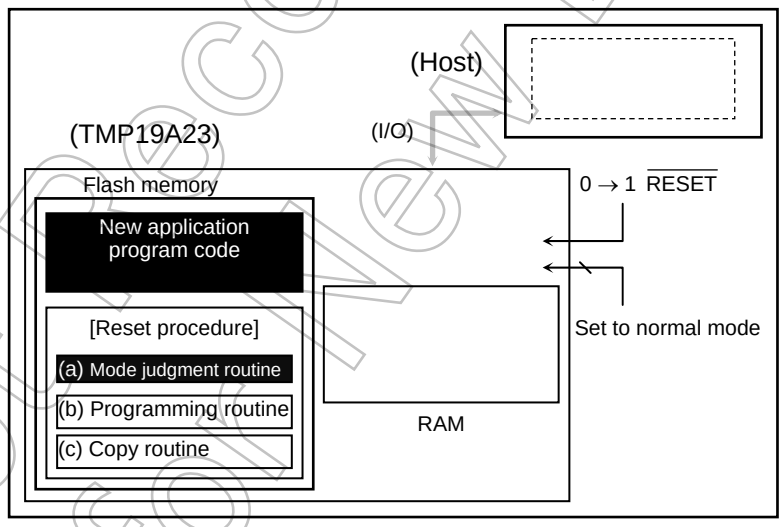
(Step-5)

Continue executing the flash programming routine to download new program code from the host controller and program it into the erased flash block. Once programming is complete, turn on the protection of that flash block.



(Step-6)

Set $\overline{\text{RESET}}$ to "0" to reset the TMP19A23. Upon reset, the on-chip flash memory is put in Normal mode. After $\overline{\text{RESET}}$ is released, the CPU will start executing the new application program code.



(1-B) Method 2: Transferring a Programming Routine from an External Host

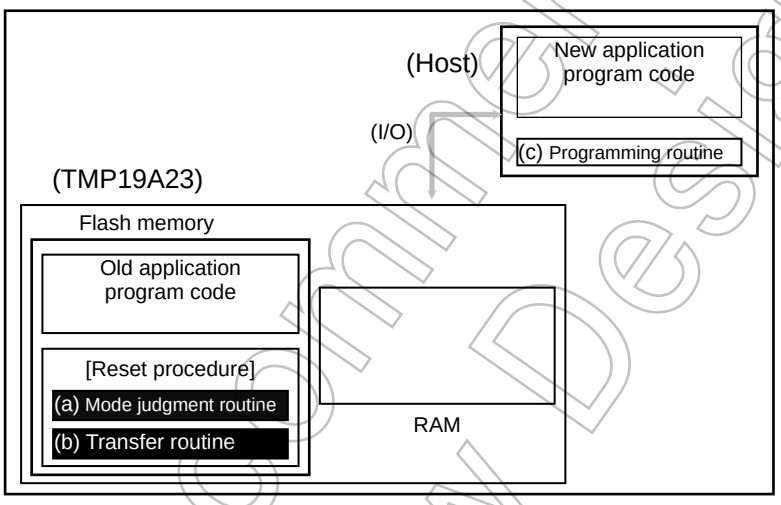
(Step-1)

Determine the conditions (e.g., pin states) required for the flash memory to enter User Boot mode and the I/O bus to be used to transfer new program code. Create hardware and software accordingly. Before installing the TMP19A23 on a printed circuit board, write the following program routines into an arbitrary flash block using programming equipment.

(a) Mode judgment routine: Code to determine whether or not to switch to User Boot mode
(b) Transfer routine: Code to download new program code from a host controller

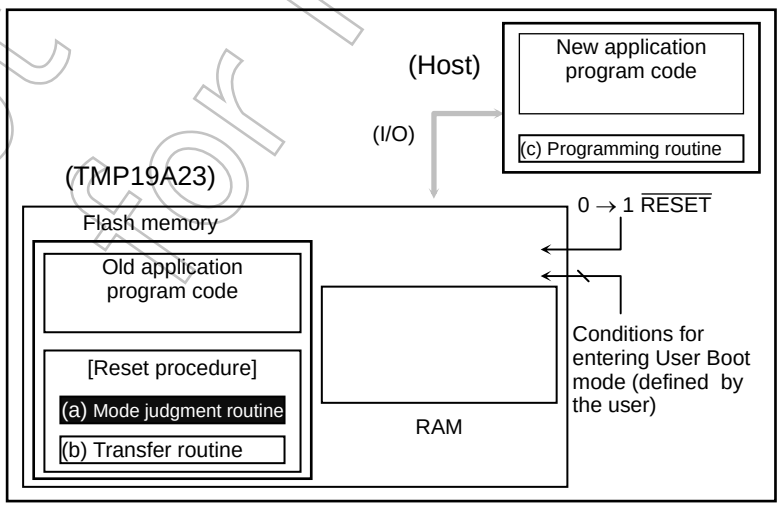
Also, prepare a programming routine shown below on the host controller:

(c) Programming routine: Code to download new program code from an external host controller and re-program the flash memory



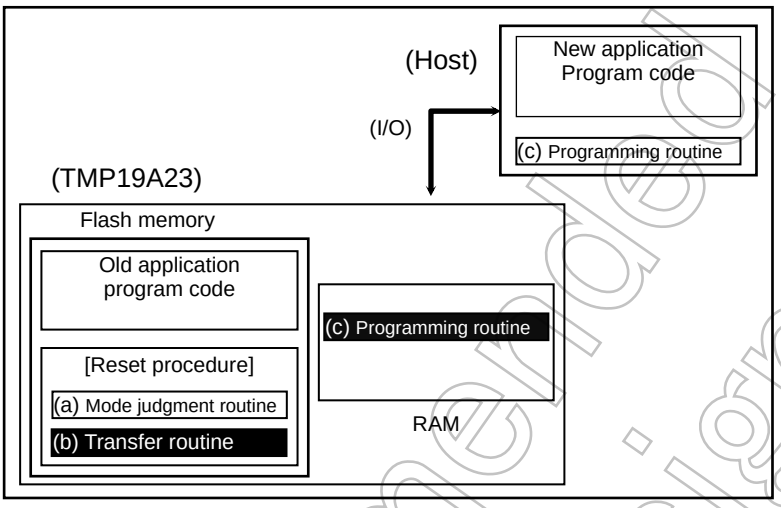
(Step-2)

After $\overline{\text{RESET}}$ is released, the reset procedure determines whether to put the TMP19A23 flash memory in User Boot mode. If mode switching conditions are met, the flash memory enters User Boot mode. (All interrupts including NMI must be disabled while in User Boot mode).



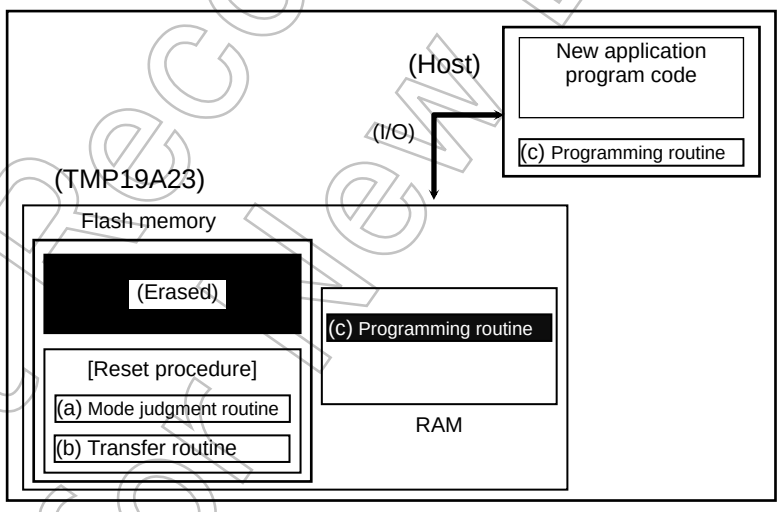
(Step-3)

Once User Boot mode is entered, execute the transfer routine (b) to download the flash programming routine (c) from the host controller to either the TMP19A23 on-chip RAM or an external memory device. (In the following figure, the on-chip RAM is used).



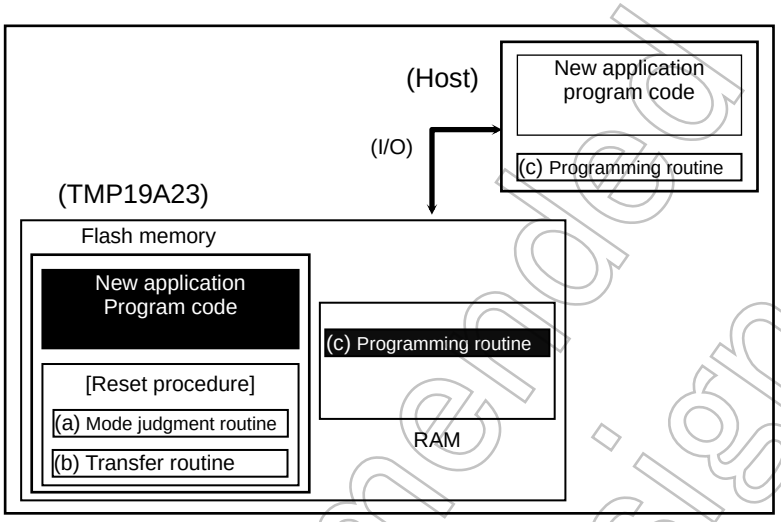
(Step-4)

Jump program execution to the flash programming routine in the on-chip RAM to erase a flash block containing the old application program code.



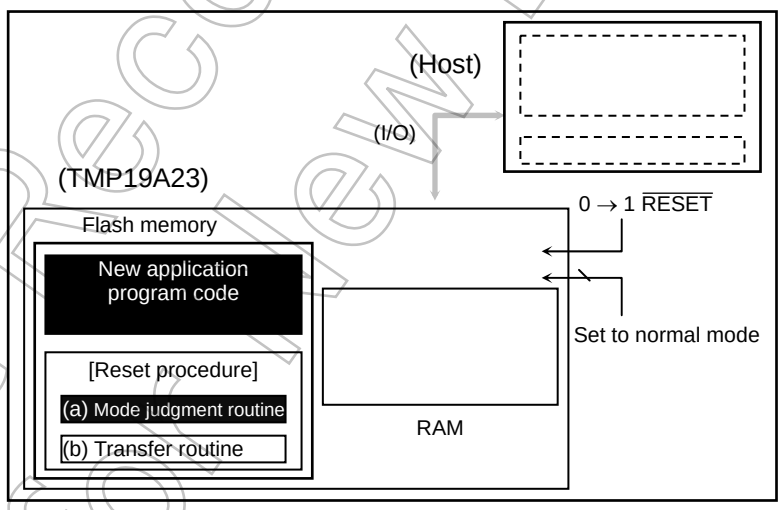
(Step-5)

Continue executing the flash programming routine to download new program code from the host controller and program it into the erased flash block. Once programming is complete, turn on the protection of that flash block.



(Step-6)

Set $\overline{\text{RESET}}$ to "0" low to reset the TMP19A23. Upon reset, the on-chip flash memory is put in Normal mode. After $\overline{\text{RESET}}$ is released, the CPU will start executing the new application program code.



21.2.4 Single Boot Mode

In Single Boot mode, the flash memory can be re-programmed by using a program contained in the TMP19A23 on-chip boot ROM. This boot ROM is a masked ROM. When Single Boot mode is selected upon reset, the boot ROM is mapped to the address region including the interrupt vector table while the flash memory is mapped to an address region different from it.

Single Boot mode allows for serial programming of the flash memory. Channel 0 of the SIO (SIO0) of the TMP19A23 is connected to an external host controller. Via this serial link, a programming routine is downloaded from the host controller to the TMP19A23 on-chip RAM. Then, the flash memory is re-programmed by executing the programming routine. The host sends out both commands and programming data to re-program the flash memory.

Communications between the SIO0 and the host must follow the protocol described later. To secure the contents of the flash memory, the validity of the application's password is checked before a programming routine is downloaded into the on-chip RAM. If password matching fails, the transfer of a programming routine itself is aborted.

As in the case of User Boot mode, all interrupts including the non-maskable interrupt (NMI) must be disabled in Single Boot mode while the flash memory is being erased or programmed. In Single Boot mode, the boot-ROM programs are executed in Normal mode.

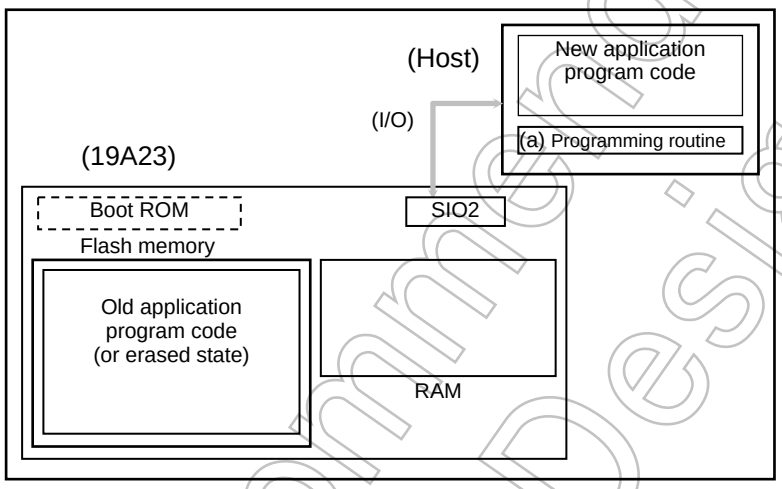
Once re-programming is complete, it is recommended to protect relevant flash blocks from accidental corruption during subsequent Single-Chip (Normal mode) operations.

Single Boot Mode

(2-A) Using the Program in the On-Chip Boot ROM

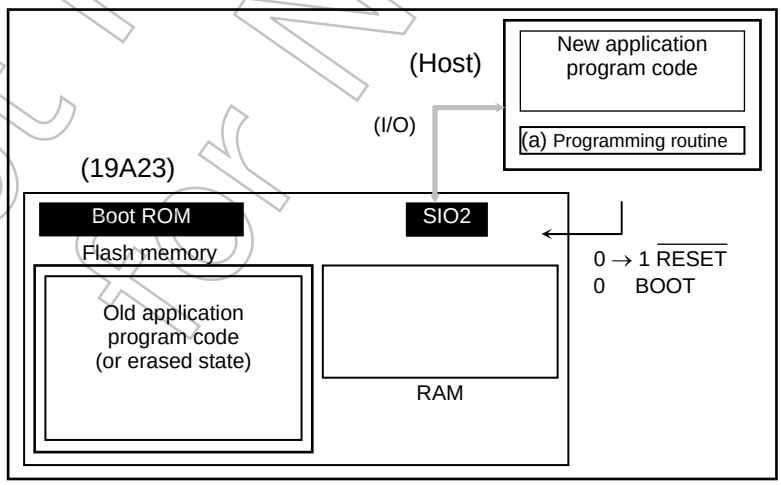
(Step-1)

The flash block containing the older version of the program code need not be erased before executing the programming routine. Since a programming routine and programming data are transferred via the SIO (SIO2), the SIO2 must be connected to a host controller. Prepare a programming routine (a) on the host controller.



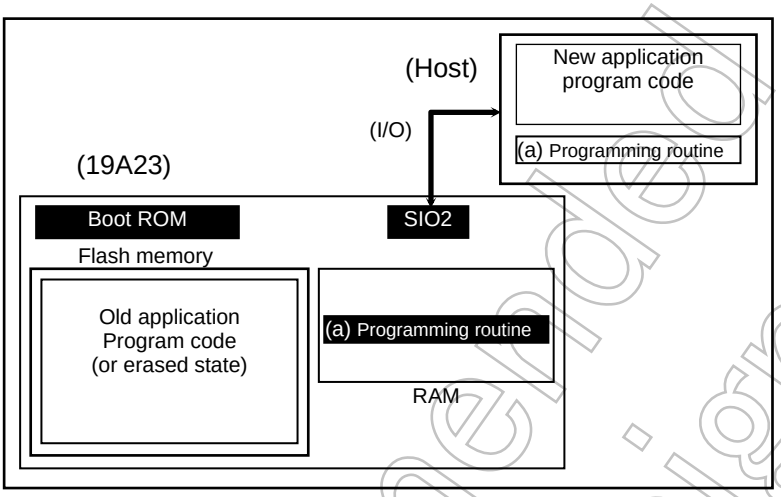
(Step-2)

Cancel the reset of the TMP19A23 by setting the Single Boot mode pin to “0”, so that the CPU re-boots from the on-chip boot ROM. The 12-byte password transferred from the host controller via SIO2 is first compared to the contents of the special flash memory locations. (If the flash block has already been erased, the password is 0xFFFF).



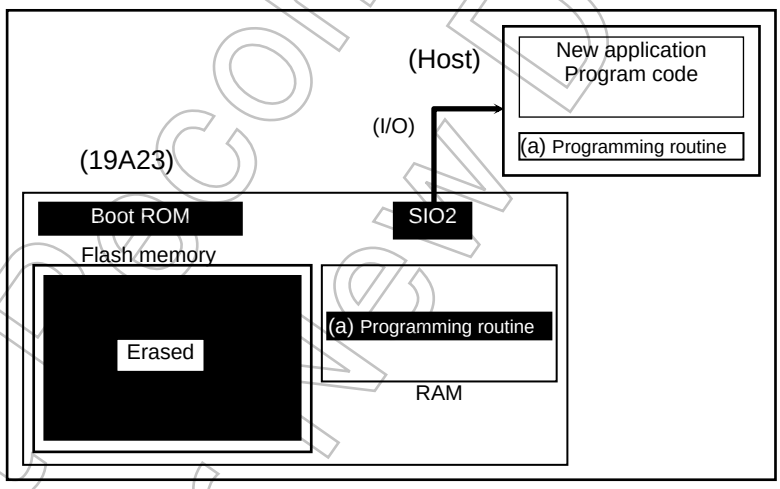
(Step-3)

If the password was correct, the boot program downloads, via the SIO0, the programming routine (a) from the host controller into the on-chip RAM of the TMP19A23. The programming routine must be stored in the address range 0xFFFF_8000 - 0xFFFF_DFFF.



(Step-4)

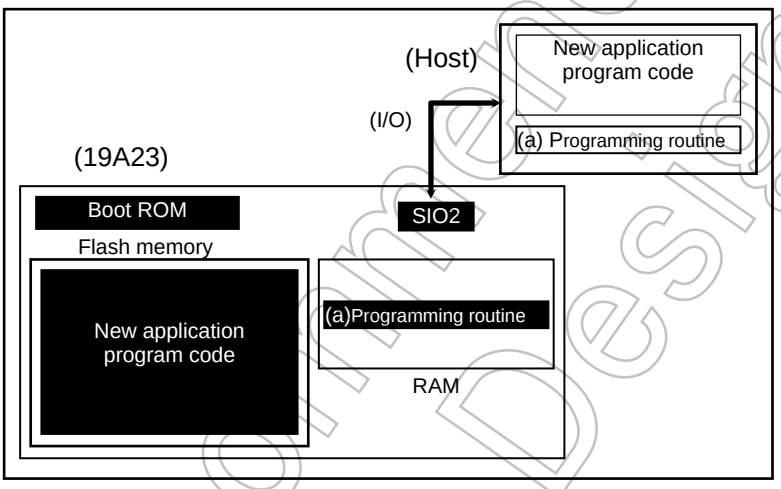
The CPU jumps to the programming routine (a) in the on-chip RAM to erase the flash block containing the old application program code. The Block Erase or Chip Erase command may be used.



(Step-5)

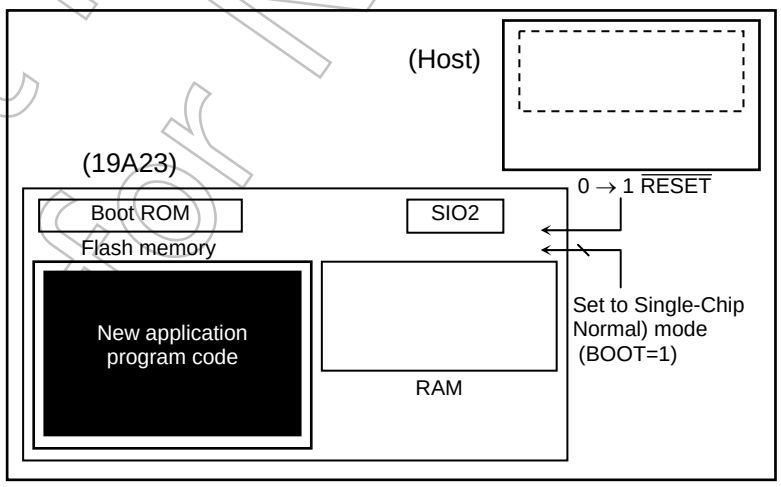
Next, the programming routine (a) downloads new application program code from the host controller and programs it into the erased flash block. Once programming is complete, protection of that flash block is turned on. It is not allowed to move program control from the programming routine (a) back to the boot ROM.

In the example below, new program code comes from the same host controller via the same SIO2 channel as for the programming routine. However, once the programming routine has begun to execute, it is free to change the transfer path and the source of the transfer. Create board hardware and a programming routine to suit your particular needs.



(Step-6)

When programming of the flash memory is complete, power off the board and disconnect the cable leading from the host to the target board. Turn on the power again so that the TMP19A23 re-boots in Single-Chip (Normal) mode to execute the new program.



(1) Configuration for Single Boot Mode

To execute the on-board programming, boot the TMP19A23 with Single Boot mode following the configuration shown below.

$\overline{BOOT}$ = 0

$\overline{RESET}$ = 0 → 1

Set the $\overline{RESET}$ input to 0, and set the each $\overline{BOOT}$ inputs to values shown above, and then release RESET (high).

(2) Memory Map

Fig. 21-4 shows a comparison of the memory maps in Normal and Single Boot modes. In Single Boot mode, the on-chip flash memory is mapped to physical addresses 0x4000_0000 through 0x4003_FFFF, virtual addresses 0x0000_0000 through 0x0003_FFFF, and the on-chip boot ROM (Mask ROM) is mapped to physical addresses 0x1FC0_0000 through 0x1FC0_1FFF.

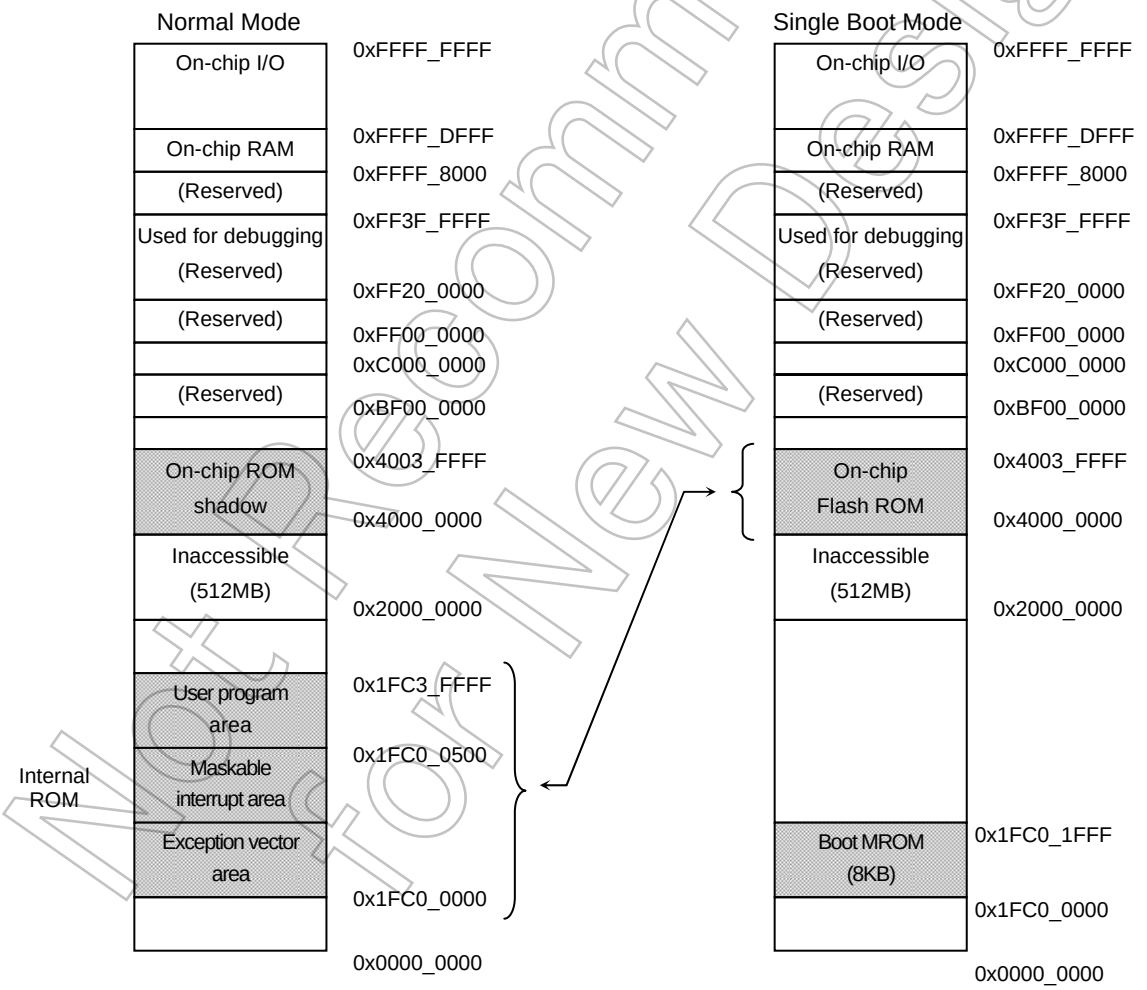


Fig. 21-4Memory Maps for Normal and Single Boot Modes (Physical Addresses)

(3) Interface specification

In Single Boot mode, an SIO channel is used for communications with a programming controller. The same configuration is applied to a communication format on a programming controller to execute the on-board programming. Both UART (asynchronous) and I/O Interface (synchronous) modes are supported. The communication formats are shown below.

- UART communication

  Communication channel : SIO channel 2
  Serial transfer mode : UART (asynchronous), full-duplex, LSB fast
  Data length : 8 bit
  Parity bits : None
  STOP bits : 1 bit
  Baud rate : Arbitrary baud rate

- I/O interface mode

  Communication channel : SIO channel 2
  Serial transfer mode : I/O interface mode, half-duplex, LSB fast
  Synchronization clock (SCLK2) : Input mode
  Handshaking signal : PB7 configured as an output mode
  Baud rate : Arbitrary baud rate

Table 21-5 Required Pin Connections

Pins			Interface	
			UART	I/O Interface Mode
Power supply pins	QFP	REGIN	○	○
		DVCC3	○	○
	BGA	DVCC15	○	○
		DVCC3	○	○
		DVSS	○	○
Mode-setting pin		$\overline{BOOT}$	○	○
Reset pin		$\overline{RESET}$	○	○
Communication pins		TXD2	○	○
		RXD2	○	○
		SCLK2	x	○ (Input mode)
		PB7	x	○ (Output mode)

(4) Data Transfer Format

Table 21-6 to Table 21-9 illustrate the operation commands and data transfer formats at each operation mode. In conjunction with this section, refer to (5) Operation of Boot Program.

Table 21-6 Single Boot Mode Commands

Code	Command
10H	RAM transfer
20H	Show Flash Memory SUM
30H	Show Product Information

Table 21-7 Transfer Format for the RAM Transfer Command

	Byte	Data Transferred from the Controller to the TMP19A23	Baud rate	Data Transferred from the TMP19A23 to the Controller
Boot ROM	1 byte	Serial operation mode and baud rate For UART mode 86H For I/O Interface mode 30H	Desired baud rate (Note 1)	-
	2 byte	-		ACK for the serial operation mode byte For UART mode -Normal acknowledge 86H (The boot program aborts if the baud rate can not be set correctly.) For I/O Interface mode -Normal acknowledge 30H
	3 byte	Command code (10H)		-
	4 byte	-		ACK for the command code byte (Note 2) -Normal acknowledge 30H -Negative acknowledge × 1H -Communication error × 8H
	5 byte - 16 byte	Password sequence (12 bytes) (0x0000_0474~0x0000_047F)		-
	17 byte	Check SUM value for bytes 5 - 16		-
	18 byte	-		ACK for the checksum byte (Note 2) -Normal acknowledge 10H -Negative acknowledge x1H -Communication error x8H
	19 byte	RAM storage start address 31 - 24		-
	20 byte	RAM storage start address 23 - 16		-
	21 byte	RAM storage start address 15 - 8		-
	22 byte	RAM storage start address 7 - 0		-
	23 byte	RAM storage byte count 15 - 8		-
	24 byte	RAM storage byte count 7 - 0		-
	25 byte	Check SUM value for bytes 19 - 24		-
	26 byte	-		ACK for the checksum byte (Note 2) -Normal acknowledge 10H -Negative acknowledge x1H -Communication error x8H
	27 byte ~ m byte	RAM storage data		-
	m + byte	Checksum value for bytes 27 - m		-
	m + byte	-		ACK for the checksum byte (Note 2) -Normal acknowledge 10H -Negative acknowledge x1H -Communication error x8H
RAM	m + byte	-		Jump to RAM storage start address

(Note 1) In I/O Interface mode, the baud rate for the transfers of the first and second bytes must be 1/16 of the desired baud rate.

(Note 2) In case of any negative acknowledge, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, if a communication error occurs, a negative acknowledge does not occur.

(Note 3) The 19th to 25th bytes must be within the RAM address range 0xFFFF_8000~0xFFFF_CFFF.

Table 21-8 Transfer Format for the Show Flash Memory Sum Command

	Byte	Data Transferred from the Controller to the TMP19A23	Baud rate	Data Transferred from the TMP19A23 to the Controller
Boot ROM	1 byte	Serial operation mode and baud rate For UART mode 86H For I/O Interface mode 30H	Desired baud rate (Note 1)	-
	2 byte	-		ACK for the serial operation mode byte For UART mode -Normal acknowledge 86H (The boot program aborts if the baud rate can not be set correctly.) For I/O Interface mode -Normal acknowledge 30H
	3 byte	Command code (20H)		-
	4 byte	-		ACK for the command code byte (Note 2) -Normal acknowledge 30H -Negative acknowledge x1H -Communication error x8H
	5 byte	-		SUM (upper byte)
	6 byte	-		SUM (lower byte)
	7 byte	-		Checksum value for bytes 5 and 6
	8 byte	(Wait for the next command code.)		-

(Note 1) In I/O Interface mode, the baud rate for the transfers of the first and second bytes must be 1/16 of the desired baud rate.

(Note 2) In case of any negative acknowledge, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, if a communication error occurs, a negative acknowledge does not occur.

Table 21-9 Transfer Format for the Show Product Information Command (1/2)

	Byte	Data Transferred from the Controller to the TMP19A23	Baud rate	Data Transferred from the TMP19A23 to the Controller
Boot ROM	1 byte	Serial operation mode and baud rate For UART mode 86H For I/O Interface mode 30H	Desired baud rate (Note 1)	-
	2 byte	-		ACK for the serial operation mode byte For UART mode -Normal acknowledge 86H (The boot program aborts if the baud rate can not be set correctly.) For I/O Interface mode -Normal acknowledge 30H
	3 byte	Command code (30H)		-
	4 byte	-		ACK for the command code byte (Note 2) -Normal acknowledge 10H -Negative acknowledge × 1H -Communication error × 8H
	5 byte	-		Flash memory data (at address 0x4000_0470)
	6 byte	-		Flash memory data (at address 0x4000_0471)
	7 byte	-		Flash memory data (at address 0x4000_0472)
	8 byte	-		Flash memory data (at address 0x4000_0473)
	9 byte - 20 byte	-		Product name (12-byte ASCII code) "TX1923FY" from the 9th byte
	21 byte - 24 byte	-		Password comparison start address (4 bytes) 74H, 04H, 00H and 00H from the 21st byte
	25 byte - 28 byte	-		RAM start address (4 bytes) 00H,80H, FFH and FFH from the 25th byte
	29 byte - 32 byte	-		Dummy data (4 bytes) FFH, 8FH, FFH and FFH from the 29th byte
	33 byte - 36 byte	-		RAM end address (4 bytes) FFH, DFH, FFH and FFH from the 33rd byte
	37 byte- 40 byte	-		Dummy data (4 bytes) 00H, 90H, FFH and FFH from the 37th byte.
	41 byte - 44 byte	-		Dummy data (4 bytes) FFH, CFH, FFH and FFH from the 41st byte
	45 byte – 46 byte	-		Fuse information (2 bytes) 00H and 00H from the 45th byte.
	47 byte - 50 byte	-		Flash memory start address (4 bytes) 00H, 00H, 00H and 00H from the 47th byte
	51 byte - 54 byte	-		Flash memory end address (4 bytes) FFH, FFH, 03H and 00H from the 51st byte
	55 byte- 56 byte	-		Flash memory block count (2 bytes) 02H and 00H from the 55th byte
	57 byte – 60 byte	-		Start address of a group of the same-size flash blocks (4 bytes) 00H, 00H, 00H and 00H from 57th byte

Table 21-9 Transfer Format for the Show Product Information Command (2/2)

	Byte	Data Transferred from the Controller to the TMP19A23	Baud rate	Data Transferred from the TMP19A23 to the Controller
Boot ROM	61 byte - 64 byte	-		Size (in halfwords) of the same-size flash blocks (4 bytes) 00H, 00H, 01H and 00H from the 61st byte
	65 byte	-		Number of flash blocks of the same size (1 byte) 02H
	66 byte	-		Checksum value for bytes 5 - 65
	67 byte	(Wait for the next command code.)		-

(Note 1) In I/O Interface mode, the baud rate for the transfers of the first and second bytes must be 1/16 of the desired baud rate.

(Note 2) In case of any negative acknowledge, the boot program returns to a state in which it waits for a command code (3rd byte). In I/O Interface mode, if a communication error occurs, a negative acknowledge does not occur.

(5) Operation of Boot Program

When Single Boot mode is selected, the boot program is automatically executed on startup. The boot program offers these three commands, of which the details are provided on the following subsections. The addresses described in this section are the virtual unless otherwise noted.

1. RAM Transfer command

The RAM Transfer command stores program code transferred from a host controller to the on-chip RAM and executes the program once the transfer is successfully completed. The maximum program size is 24 kbytes. The RAM storage start address must be set within the range.

The RAM Transfer command can be used to download a flash programming routine of your own; this provides the ability to control on-board programming of the flash memory in a unique manner. The programming routine must utilize the flash memory command sequences described in Section 21.3.

Before initiating a transfer, the RAM Transfer command checks a password sequence coming from the controller against that stored in the flash memory.

2. Show Flash Memory Sum command

The Show Flash Memory Sum command adds the contents of the 256 kbytes of the flash memory together. The boot program does not provide a command to read out the contents of the flash memory. Instead, the Flash Memory Sum command can be used for software revision management.

3. Show Product Information command

The Show Product Information command provides the product name, on-chip memory configuration and the like. This command also reads out the contents of the flash memory locations at addresses 0x0000_0470 through 0x0000_0473. In addition to the Show Flash Memory Sum command, these locations can be used for software revision management.

1) RAM Transfer Command (See Table 21-7)

1. The 1st byte specifies which one of the two serial operation modes is used. For a detailed description of how the serial operation mode is determined, see Determination of a Serial Operation Mode described later. If it is determined as UART mode, the boot program then checks if the SIO0 is programmable to the baud rate at which the 1st byte was transferred. During the first-byte interval, the RXE bit in the HSC0MOD register is cleared.

   - To communicate in UART mode
     Send, from the controller to the target board, 86H in UART data format at the desired baud rate. If the serial operation mode is determined as UART, then the boot program checks if the SIO0 can be programmed to the baud rate at which the first byte was transferred. If that baud rate is not possible, the boot program aborts, disabling any subsequent communications.

   - To communicate in I/O Interface mode
     Send, from the controller to the target board, 30H in I/O Interface data format at 1/16 of the desired baud rate. Also send the 2nd byte at the same baud rate. Then send all subsequent bytes at a rate equal to the desired baud rate.

     In I/O Interface mode, the CPU sees the serial receive pin as if it were a general input port in monitoring its logic transitions. If the baud rate of the incoming data is high or the chip's operating frequency is high, the CPU may not be able to keep up with the speed of logic transitions. To prevent such situations, the 1st and 2nd bytes must be transferred at 1/16 of the desired baud rate; then the boot program calculates 16 times that as the desired baud rate. When the serial operation mode is determined as I/O Interface mode, the SIO0 is configured for SCLK Input mode. Beginning with the third byte, the controller must ensure that its AC timing restrictions are satisfied at the selected baud rate. In the case of I/O Interface mode, the boot program does not check the receive error flag; thus there is no such thing as error acknowledge (bit 3, x8H).

2. The 2nd byte, transmitted from the target board to the controller, is an acknowledge response to the 1st byte. The boot program echoes back the first byte: 86H for UART mode and 30H for I/O Interface mode.

   UART mode
   If the SIO0 can be programmed to the baud rate at which the 1st byte was transferred, the boot program programs the BR0CR and sends back 86H to the controller as an acknowledge. If the SIO0 is not programmable at that baud rate, the boot program simply aborts with no error indication. Following the 1st byte, the controller should allow for a time-out period of five seconds. If it does not receive 86H within the allowed time-out period, the controller should give up the communication. The boot program sets the RXE bit in the HSC0MOD register to enable reception (1) before loading the SIO transmit buffer with 86H.

- I/O Interface mode
  The boot program programs the HSC0MOD0 and HSC0CR registers to configure the SIO0 in I/O Interface mode (clocked by the rising edge of SCLK0), writes 30H to the HSC0BUF. Then, the SIO0 waits for the SCLK0 signal to come from the controller. Following the transmission of the 1st byte, the controller should send the SCLK clock to the target board after a certain idle time (several microseconds). This must be done at 1/16 the desire baud rate. If the 2nd byte, which is from the target board to the controller, is 30H, then the controller should take it as a go-ahead. The controller must then deliver the 3rd byte to the target board at a rate equal to the desired baud rate. The boot program sets the RXE bit in the HSC0MOD register to enable reception before loading the SIO transmit buffer with 30H.

3. The 3rd byte transmitted from the controller to the target board is a command. The code for the RAM Transfer command is 10H.

4. The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits x8H (bit 3) and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command. When the SIO0 is configured for I/O Interface mode, the boot program does not check for a receive error.

   If the 3rd byte is equal to any of the command codes listed in Table 21-6, the boot program echoes it back to the controller. When the RAM Transfer command was received, the boot program echoes back a value of 10H and then branches to the RAM Transfer routine. Once this branch is taken, a password check is done. Password checking is detailed in a later section “Password”. If the 3rd byte is not a valid command, the boot program sends back x1H (bit 0) to the controller and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

5. The 5th to 16th bytes transmitted from the controller to the target board, are a 12-byte password. The 5th byte is compared to the contents of address 0x0000_0474 in the flash memory; the 6th byte is compared to the contents of address 0x0000_0475 in the flash memory; likewise, the 16th byte is compared to the contents of address 0x0000_047F in the flash memory. If the password checking fails, the RAM Transfer routine sets the password error flag.

6. The 17th byte is a checksum value for the password sequence (5th to 16th bytes). To calculate the checksum value for the 12-byte password, add the 12 bytes together, drop the carries and take the two’s complement of the total sum. Transmit this checksum value from the controller to the target board. The checksum calculation is described in details in a later section “Checksum Calculation”.

7. The 18th byte, transmitted from the target board to the controller, is an acknowledge response to the 5th to 17th bytes. First, the RAM Transfer routine checks for a receive error in the 5th to 17th bytes. If there was a receive error, the boot program sends back 18H (bit 3) and returns to the state in which it waits for a command (i.e., the 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the previously issued command (i.e., all 1s). When the SIO0 is configured for I/O Interface mode, the RAM Transfer routine does not check for a receive error.

   Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the series of the 5th to 17th bytes must result in 00H (with the carry dropped). If it is not 00H, one or more bytes of data has been corrupted. In case of a checksum error, the RAM Transfer routine sends back 11H to the controller and returns to the state in which it waits for a command (i.e., the 3rd byte) again.

   Finally, the RAM Transfer routine examines the result of the password check. The following two cases are treated as a password error. In these cases, the RAM Transfer routine sends back 11H (bit 0) to the controller and returns to the state in which it waits for a command (i.e., the 3rd byte) again.

   - Irrespective of the result of the password comparison, all the 12 bytes of a password in the flash memory are the same value other than FFH.
   - Not the entire password bytes transmitted from the controller matched those contained in the flash memory.

   When all the above checks have been successful, the RAM Transfer routine returns a normal acknowledge response (10H) to the controller.

8. The 19th to 22nd bytes, transmitted from the controller the target board, indicate the start address of the RAM region where subsequent data (e.g., a flash programming routine) should be stored. The 19th byte corresponds to bits 31–24 of the address and the 22nd byte corresponds to bits 7–0 of the address.

9. The 23rd and 24th bytes, transmitted from the controller to the target board, indicate the number of bytes that will be transferred from the controller to be stored in the RAM. The 23rd byte corresponds to bits 15–8 of the number of bytes to be transferred, and the 24th byte corresponds to bits 7–0 of the number of bytes.

10. The 25th byte is a checksum value for the 19th to 24th bytes. To calculate the checksum value, add all these bytes together, drop the carries and take the two's complement of the total sum. Transmit this checksum value from the controller to the target board. The checksum calculation is described in details in a later section "Checksum Calculation".

11. The 26th byte, transmitted from the target board to the controller, is an acknowledge response to the 19th to 25th bytes of data. First, the RAM Transfer routine checks for a receive error in the 19th to 25th bytes. If there was a receive error, the RAM Transfer routine sends back 18H and returns to the command wait state (i.e., the 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the previously issued command (i.e., all 1s). When the SIO0 is configured for I/O Interface mode, the RAM Transfer routine does not check for a receive error.

    Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the series of the 19th to 25th bytes must result in 00H (with the carry dropped). If it is not 00H, one or more bytes of data has been corrupted. In case of a checksum error, the RAM Transfer routine sends back 11H to the controller and returns to the state in which it waits for a command (i.e., the 3rd byte) again.

    - The RAM storage start address must be within the range 0xFFFD_8000~0xFFFD_EFFF.

    When the above checks have been successful, the RAM Transfer routine returns a normal acknowledge response (10H) to the controller.

12. The 27th to mth bytes from the controller are stored in the on-chip RAM of the TMP19A23. Storage begins at the address specified by the 19th–22nd bytes and continues for the number of bytes specified by the 23rd–24th bytes.

13. The (m+1) th byte is a checksum value. To calculate the checksum value, add the 27th to mth bytes together, drop the carries and take the two's complement of the total sum. Transmit this checksum value from the controller to the target board. The checksum calculation is described in details in a later section "Checksum Calculation".

14. The (m+2) th byte is a acknowledge response to the 27th to (m+1) th bytes.

    First, the RAM Transfer routine checks for a receive error in the 27th to (m+1) th bytes. If there was a receive error, the RAM Transfer routine sends back 18H (bit 3) and returns to the state in which it waits for a command (i.e., the 3rd byte) again. In this case, the upper four bits of the acknowledge response are the same as those of the previously issued command (i.e., all 1s). When the SIO0 is configured for I/O Interface mode, the RAM Transfer routine does not check for a receive error.

    Next, the RAM Transfer routine performs the checksum operation to ensure data integrity. Adding the series of the 27th to (m+1) th bytes must result in 00H (with the carry dropped). If it is not 00H, one or more bytes of data has been corrupted. In case of a checksum error, the RAM Transfer routine sends back 11H (bit 0) to the controller and returns to the command wait state (i.e., the 3rd byte) again. When the above checks have been successful, the RAM Transfer routine returns a normal acknowledge response (10H) to the controller.

15. If the (m+2) th byte was a normal acknowledge response, a branch is made to the address specified by the 19th to 22nd bytes in 32-bit ISA mode.

2) Show Flash Memory Sum Command (See Table 21-8)

1. The processing of the 1st and 2nd bytes are the same as for the RAM Transfer command.

2. The 3rd byte, which the target board receives from the controller, is a command. The code for the Show Flash Memory Sum command is 20H.

3. The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits x8H (bit 3) and returns to the command wait state again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command. When the SIO0 is configured for I/O Interface mode, the boot program does not check for a receive error.

   If the 3rd byte is equal to any of the command codes listed in Table 21-6, the boot program echoes it back to the controller. When the Show Flash Memory Sum command was received, the boot program echoes back a value of 20H and then branches to the Show Flash Memory Sum routine. If the 3rd byte is not a valid command, the boot program sends back x1H (bit 0) to the controller and returns to the command wait state (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

4. The Show Flash Memory Sum routine adds all the bytes of the flash memory together. The 5th and 6th bytes, transmitted from the target board to the controller, indicate the upper and lower bytes of this total sum, respectively. For details on sum calculation, see a later section “Calculation of the Show Flash Memory Sum Command”.

5. The 7th byte is a checksum value for the 5th and 6th bytes. To calculate the checksum value, add the 5th and 6th bytes together, drop the carry and take the two’s complement of the sum. Transmit this checksum value from the controller to the target board.

6. The 8th byte is the next command code.

3) Show Product Information Command (See Table 21-9)

1. The processing of the 1st and 2nd bytes are the same as for the RAM Transfer command.

2. The 3rd byte, which the target board receives from the controller, is a command. The code for the Show Product Information command is 30H.

3. The 4th byte, transmitted from the target board to the controller, is an acknowledge response to the 3rd byte. Before sending back the acknowledge response, the boot program checks for a receive error. If there was a receive error, the boot program transmits x8H (bit 3) and returns to the command wait state again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command. When the SIO0 is configured for I/O Interface mode, the boot program does not check for a receive error.

   If the 3rd byte is equal to any of the command codes listed in Table 21-6, the boot program echoes it back to the controller. When the Show Flash Memory Sum command was received, the boot program echoes back a value of 30H and then branches to the Show Flash Memory Sum routine. If the 3rd byte is not a valid command, the boot program sends back x1H (bit 0) to the controller and returns to the state in which it waits for a command (the third byte) again. In this case, the upper four bits of the acknowledge response are undefined - they hold the same values as the upper four bits of the previously issued command.

4. The 5th to 8th bytes, transmitted from the target board to the controller, are the data read from addresses 0x0000_0470–0x0000_0473 in the flash memory. Software version management is possible by storing a software ID in these locations.

5. The 9th to 20th bytes, transmitted from the target board to the controller, indicate the product name, which is “TX1923FY _ _” in ASCII code (where _ is a space).

6. The 21st to 24th bytes, transmitted from the target board to the controller, indicate the start address of the flash memory area containing the password, i.e., 74H, 04H, 00H, and 00H.

7. The 25th to 28th bytes, transmitted from the target board to the controller, indicate the start address of the on-chip RAM, i.e., 00H, 80H, FDH, FFH.

8. The 29th to 32nd bytes, transmitted from the target board to the controller, are dummy data (FFH, 8FH, FDH, FFH).

9. The 33rd to 36th bytes, transmitted from the target board to the controller, indicate the end address of the on-chip RAM, i.e., FFH, FFH, FDH, FFH.

10. The 37th to 40th bytes, transmitted from the target board to the controller, are 00H, 90H, FDH and FFH. The 41st to 44th bytes, transmitted from the target board to the controller, are FFH, EFH, FDH and FFH.

11. The 45th and 46th bytes, transmitted from the target board to the controller, indicate the presence or absence of the security and protect bits and whether the flash memory is divided into blocks. Bit 0 indicates the presence or absence of the security bit; it is 0 if the security bit is available. Bit 1 indicates the presence or absence of the protect bits; it is 0 if the protect bits are available. If bit 2 is 0, it indicates that the flash memory is divided into blocks. The remaining bits are undefined. The 45th and 46th bytes are 01H, 00H.

12. The 47th to 50th bytes, transmitted from the target board to the controller, indicate the start address of the on-chip flash memory, i.e., 00H, 00H, 00H, and 00H.

13. The 51st to 54th bytes, transmitted from the target board to the controller, indicate the end address of the on-chip flash memory, i.e., FFH, FFH, 03H, and 00H.

14. The 55th to 56th bytes, transmitted from the target board to the controller, indicate the number of flash blocks available, i.e., 02H, 00H.

15. The 57th to 92nd bytes, transmitted from the target board to the controller, contain information about the flash blocks. Flash blocks of the same size are treated as a group. Information about the flash blocks indicate the start address of a group, the size of the blocks in that group (in halfwords) and the number of the blocks in that group.

    The 57th to 65th bytes are the information about the 128-kbyte blocks (Block 0 to Block 1). See Table 21-9 for the values of bytes transmitted.

16. The 66th byte, transmitted from the target board to the controller, is a checksum value for the 5th to 65th bytes. The checksum value is calculated by adding all these bytes together, dropping the carry and taking the two’s complement of the total sum.

17. The 67th byte is the next command code.

4) Acknowledge Responses

The boot program represents processing states with specific codes. Table 21-10 to Table 21-12 show the values of possible acknowledge responses to the received data. The upper four bits of the acknowledge response are equal to those of the command being executed. Bit 3 of the code indicates a receive error. Bit 0 indicates an invalid command error, a checksum error or a password error. Bit 1 and bit 2 are always 0. Receive error checking is not done in I/O Interface mode.

Table 21-10 ACK Response to the Serial Operation Mode Byte

Return Value	Meaning
0x86	The SIO can be configured to operate in UART mode. (See Note)
0x30	The SIO can be configured to operate in I/O Interface mode.

(Note) If the serial operation mode is determined as UART, the boot program checks if the SIO can be programmed to the baud rate at which the operation mode byte was transferred. If that baud rate is not possible, the boot program aborts, without sending back any response.

Table 21-11 ACK Response to the Command Byte

Return Value	Meaning
0x?8 (See Note)	A receive error occurred while getting a command code.
0x?1 (See Note)	An undefined command code was received. (Reception was completed normally.)
0x10	The RAM Transfer command was received.
0x20	The Show Flash Memory Sum command was received.
0x30	The Show Product Information command was received.

(Note) The upper four bits of the ACK response are the same as those of the previous command code.

Table 21-12 ACK Response to the Checksum Byte

Return Value	Meaning
0xN8 (See Note)	A receive error occurred.
0xN1 (See Note)	A checksum or password error occurred.
0xN0 (See Note)	The checksum was correct.

(Note) The upper four bits of the ACK response are the same as those of the operation command code. It is 1 (N＝RAM transfer command data [7:4]) when password error occurs.

5) Determination of a Serial Operation Mode

The first byte from the controller determines the serial operation mode. To use UART mode for communications between the controller and the target board, the controller must first send a value of 86H at a desired baud rate to the target board. To use I/O Interface mode, the controller must send a value of 30H at 1/16 the desired baud rate. Figure 21-5 shows the waveforms for the first byte.

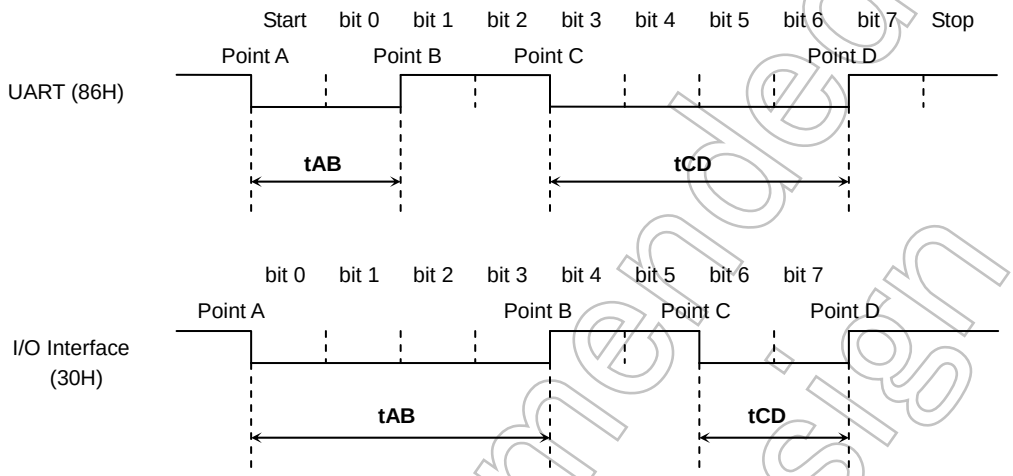


Figure 21-5 Serial Operation Mode Byte

After $\overline{\text{RESET}}$ is released, the boot program monitors the first serial byte from the controller, with the SIO reception disabled, and calculates the intervals of tAB, tAC and tAD. Figure 21-6 shows a flowchart describing the steps to determine the intervals of tAB, tAC and tAD. As shown in the flowchart, the boot program captures timer counts each time a logic transition occurs in the first serial byte. Consequently, the calculated tAB, tAC and tAD intervals are bound to have slight errors. If the transfer goes at a high baud rate, the CPU might not be able to keep up with the speed of logic transitions at the serial receive pin. In particular, I/O Interface mode is more prone to this problem since its baud rate is generally much higher than that for UART mode. To avoid such a situation, the controller should send the first serial byte at 1/16 the desired baud rate.

The flowchart in Figure 21-6 shows how the boot program distinguishes between UART and I/O Interface modes. If the length of tAB is equal to or less than the length of tCD, the serial operation mode is determined as UART mode. If the length of tAB is greater than the length of tCD, the serial operation mode is determined as I/O Interface mode. Bear in mind that if the baud rate is too high or the timer operating frequency is too low, the timer resolution will be coarse, relative to the intervals between logic transitions. This becomes a problem due to inherent errors caused by the way in which timer counts are captured by software; consequently the boot program might not be able to determine the serial operation mode correctly. To prevent this problem, reset UART mode within the programming routine.

For example, the serial operation mode may be determined to be I/O Interface mode when the intended mode is UART mode. To avoid such a situation, when UART mode is utilized, the controller should allow for a time-out period within which it expects to receive an echo-back (86H) from the target board. The controller should give up the communication if it fails to get that echo-back within the allowed time. When I/O Interface mode is utilized, once the first serial byte has been transmitted, the controller should send the SCLK clock after a certain idle time to get an acknowledge response. If the received acknowledge response is not 30H, the controller should give up further communications.

When the intended mode is I/O interface mode, the first byte does not have to be 0x30 as long as tAB is greater than tCD as shown above. 0x91, 0xA1 or 0xB1 can be sent as the first byte code to determine the falling edges of Point A and Point C and the rising edges of Point B and Point D. If tAB is greater than tCD and SIO is selected by the resolution of the operation mode determination, the second byte code is 0x30 even though the transmitted code on the first byte is not 0x30 (The first byte code to determine I/O interface mode is described as 0x30).

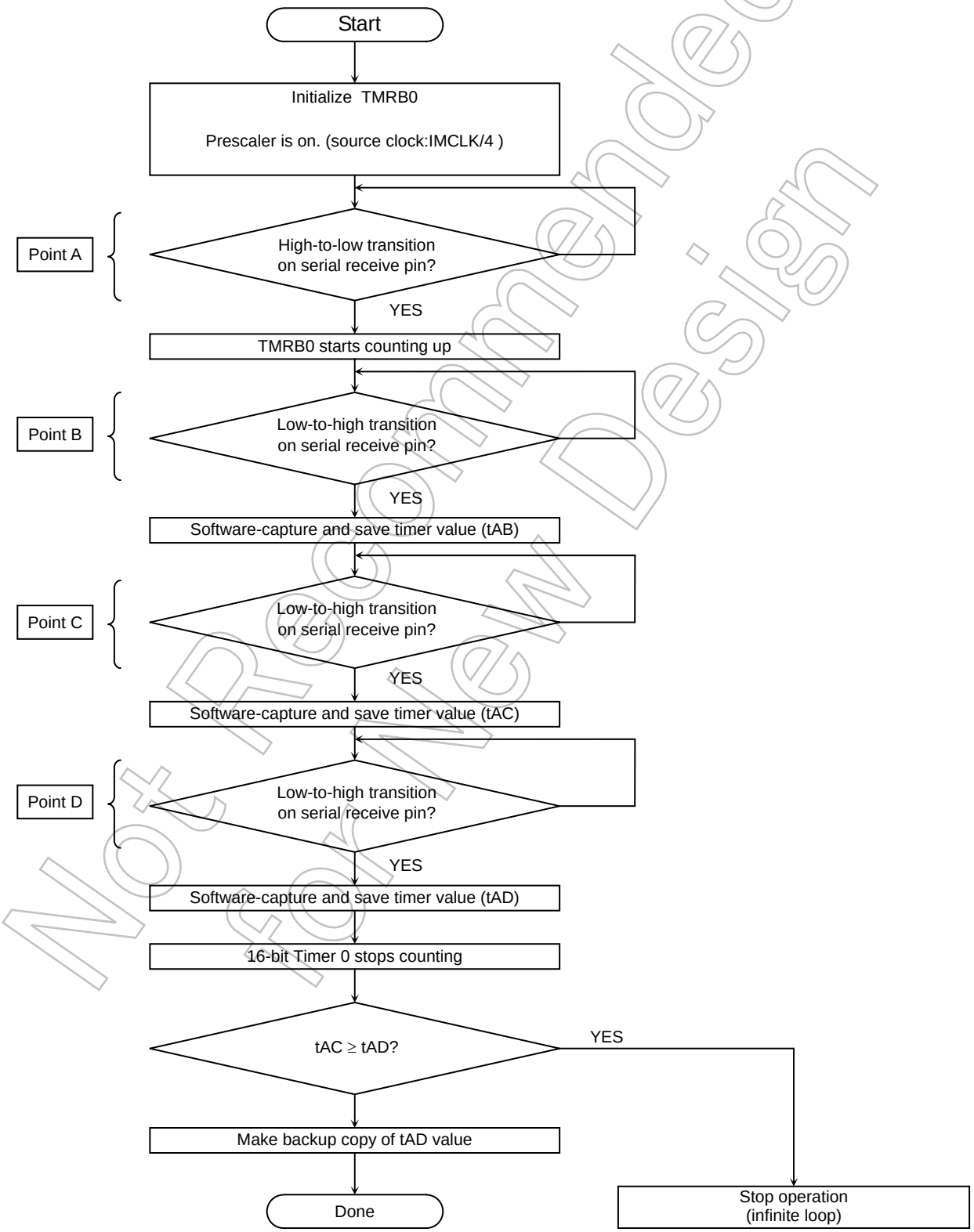


Figure 21-6 Serial Operation Mode Byte Reception Flow

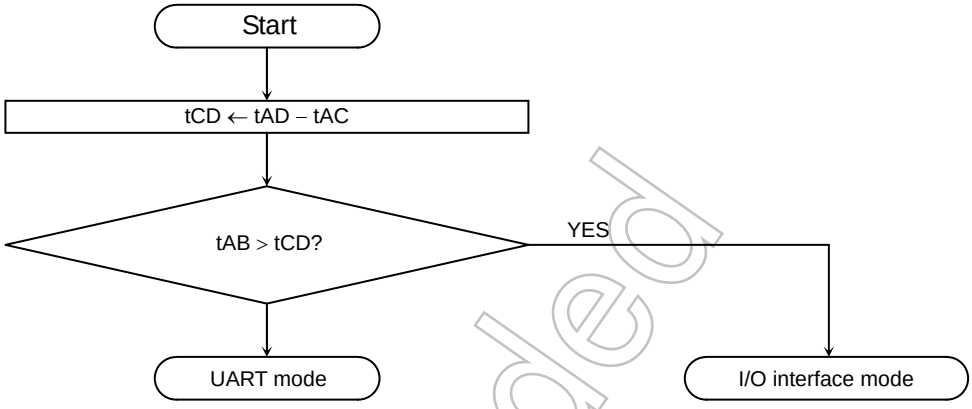


Figure 21-7 Serial Operation Mode Determination Flow

6) Password

The RAM Transfer command (10H) causes the boot program to perform a password check. Following an echo-back of the command code, the boot program checks the contents of the 12-byte password area (0x4000_0474 to 0x4000_047F) within the flash memory. If all these address locations contain the same bytes of data other than FFH, a password area error occurs as shown in Figure 21-8. In this case, the boot program returns an error acknowledge (11H) in response to the checksum byte (the 17th byte), regardless of whether the password sequence sent from the controller is all FFHs.

The password sequence received from the controller (5th to 16th bytes) is compared to the password stored in the flash memory. Table 21-13 shows how they are compared byte-by-byte. All of the 12 bytes must match to pass the password check. Otherwise, a password error occurs, which causes the boot program to return an error acknowledge in response to the checksum byte (the 17th byte).

When ROM is under protection, the masked address data code is used as the password instead of actual password data code (see Table 21-13). ROM protection must be released before using the data code in the password area.

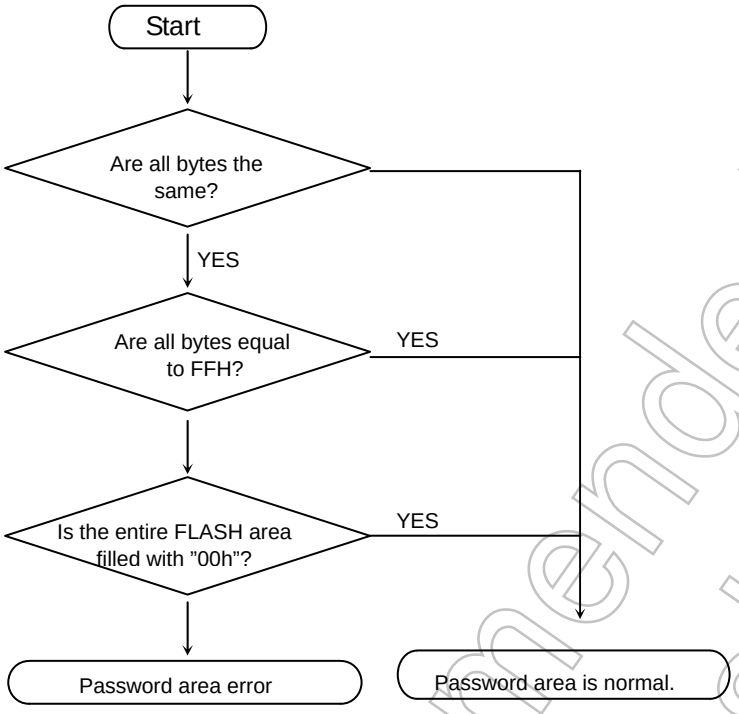


Figure 21-8 Password Area Check Flow

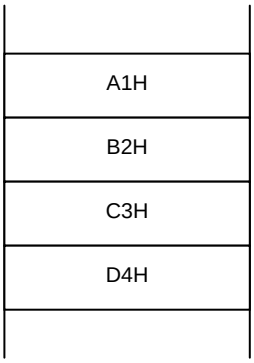
Table 21-13 Relationship between Received Bytes and Flash Memory Locations

	Data to refer	
Received Byte	Compared Flash Memory Data	ROM Protect
5th byte	Address 0x0000_0474	Address 0x0000_0004
6th byte	Address 0x0000_0475	Address 0x0000_0005
7th byte	Address 0x0000_0476	Address 0x0000_0006
8th byte	Address 0x0000_0477	Address 0x0000_0007
9th byte	Address 0x0000_0478	Address 0x0000_0000
10th byte	Address 0x0000_0479	Address 0x0000_0001
11th byte	Address 0x0000_047A	Address 0x0000_0002
12th byte	Address 0x0000_047B	Address 0x0000_0003
13th byte	Address 0x0000_047C	Address 0x0000_0004
14th byte	Address 0x0000_047D	Address 0x0000_0005
15th byte	Address 0x0000_047E	Address 0x0000_0006
16th byte	Address 0x0000_047F	Address 0x0000_0007

7) Calculation of the Show Flash Memory Sum Command

The result of the sum calculation “byte + byte + byte + …“ is responded by a word quantity. The Show Flash Memory Sum command adds all 256 kbytes of the flash memory together and provides the total sum as a word quantity. The sum is sent to the controller, with the upper eight bits first, followed by the lower eight bits.

Example)



For the interest of simplicity, assume the depth of the flash memory is four locations. Then the sum of the four bytes is calculated as:

A1H + B2H + C3H + D4H = 02EAH

Hence, 02H is first sent to the controller, followed by EAH.

8) Checksum Calculation

The checksum byte for a series of bytes of data is calculated by adding the bytes together, dropping the carries, and taking the two’s complement of the total sum. The Show Flash Memory Sum command and the Show Product Information command perform the checksum calculation. The controller must perform the same checksum operation in transmitting checksum bytes.

Example)

Assume the Show Flash Memory Sum command provides the upper and lower bytes of the sum as E5H and F6H. To calculate the checksum for a series of E5H and F6H:

Add the bytes together

E5H + F6H = 1DBH

Take the two’s complement of the sum, and that is the checksum byte.

0 − DBH = 25H

(9) General Boot Program Flowchart

Figure 21-9 shows an overall flowchart of the boot program.

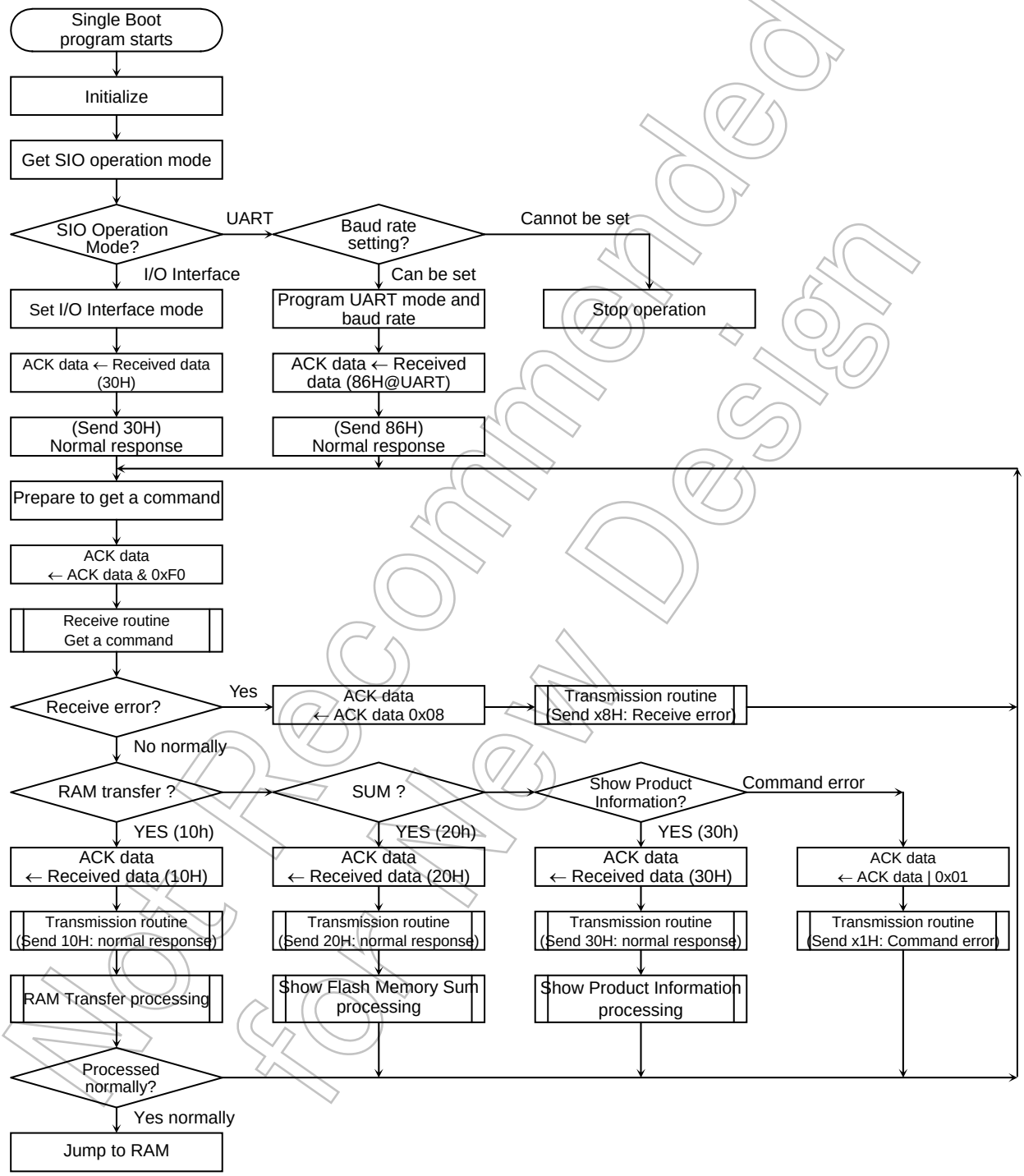


Fig. 21-9 Overall Boot Program Flow

21.3 On-board Programming of Flash Memory (Rewrite/Erase)

In on-board programming, the CPU is to execute software commands for rewriting or erasing the flash memory. The rewrite/erase control program should be prepared by the user beforehand. Because the flash memory content cannot be read while it is being written or erased, it is necessary to run the rewrite/erase program from the internal RAM or from an external memory device after shifting to the user boot mode. In this section, flash memory addresses are represented in virtual addresses unless otherwise noted.

21.3.1 Flash Memory

Except for some functions, writing and erasing flash memory data are in accordance with the standard JEDEC commands. In writing or erasing, use the SW command of the CPU to enter commands to the flash memory. Once the command is entered, the actual write or erase operation is automatically performed internally.

Table 21-14 Flash Memory Functions

Major functions	Description
Automatic page program	Writes data automatically.
Automatic chip erase	Erases the entire area of the flash memory automatically.
Automatic block erase	Erases a selected block automatically. (128 kB at a time)
Protect function	By writing a 4-bit protection code, the write or erase function can be individually inhibited for each block. When all the blocks are under protection, the entire flash memory area is also protected. By releasing the flash protection, the entire data stored in the flash memory is automatically erased.

Note that addressing of operation commands is different from the case of standard commands due to the specific interface arrangements with the CPU as detailed operation of the user boot mode and RAM transfer mode is described later. Also note that the flash memory is written in 32-bit blocks. So, 32-bit (word) data transfer commands must be used in writing the flash memory.

(1) Block configuration

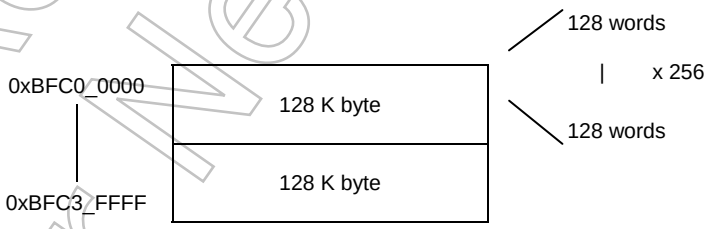


Fig. 21-10 Block Configuration of Flash Memory

(2) Basic operation

Generally speaking, this flash memory device has the following two operation modes:

- The mode to read memory data (Read mode)
- The mode to automatically erase or rewrite memory data (Automatic operation)

Transition to the automatic mode is made by executing a command sequence while it is in the memory read mode. In the automatic operation mode, flash memory data cannot be read and any commands stored in the flash memory cannot be executed. In the automatic operation mode, any interrupt or exception generation cannot set the device to the read mode except when a hardware reset is generated. During automatic operation, be sure not to cause any exceptions other than debug exceptions and reset while a DSU probe is connected. Any interrupt or exception generation cannot set the device to the read mode except when a hardware reset is generated.

1) Read

When data is to be read, the flash memory must be set to the read mode. The flash memory will be set to the read mode immediately after power is applied, when CPU reset is removed, or when an automatic operation is normally terminated. In order to return to the read mode from other modes or after an automatic operation has been abnormally terminated, either the Read/reset command (a software command to be described later) or a hardware reset is used. The device must also be in the read mode when any command written on the flash memory is to be executed.

- **Read/reset command and Read command (software reset)**

When an automatic operation is abnormally terminated, the flash memory cannot return to the read mode by itself (When FLCS<RDY/BSY> = 0, data read from the flash memory is undefined.) In this case, the Read/reset command can be used to return the flash memory to the read mode. Also, when a command that has not been completely written has to be canceled, the Read/reset command must be used to return to the read mode. The Read command is used to return to the read mode after executing the SW command to write the data "0x0000_00F0" to an arbitrary address of the flash memory.

- **With the Read/reset command, the device is returned to the read mode after completing the third bus write cycle.**

2) Command write

This flash memory uses the command control method. Commands are executed by executing a command sequence to the flash memory. The flash memory executes automatic operation commands according to the address and data combinations applied (refer to Command Sequence).

If it is desired to cancel a command write operation already in progress or when any incorrect command sequence has been entered, the Read/reset command is to be executed. Then, the flash memory will terminate the command execution and return to the read

While commands are generally comprised of several bus cycles, the operation to apply the SW command to the flash memory is called "bus write cycle." The bus write cycles are to be in a specific sequential order and the flash memory will perform an automatic operation when the sequence of the bus write cycle data and address of a command write operation is in accordance with a predefined specific sequence. If any bus write cycle does not follow a predefined command write sequence, the flash memory will terminate the command execution and return to the read mode. The address [31:21] in each bus write cycle should be the virtual address [31:21] of command execution. It will be explained later for the address bits [20:8].

(Note 1) Command sequences are executed from outside the flash memory area.
(Note 2) The interval between bus write cycles for this device must be 15 system clock cycles or longer. The command sequencer in the flash memory device requires a certain time period to recognize a bus write cycle. If more than one bus write cycles are executed within this time period, normal operation cannot be expected. For adjusting the applicable bus write cycle interval using a software timer to be operated at the operating frequency, use the section 10) "ID-Read" to check for the appropriateness.
(Note 3) Between the bus write cycles, never use any load command (such as LW, LH, or LB) to the flash memory or perform a DMA transmission by specifying the flash area as the source address. Also, don't execute a Jump command to the flash memory. While a command sequence is being executed, don't generate any interrupt such as maskable interrupts (except debug exceptions when a DSU probe is connected). If such an operation is made, it can result in an unexpected read access to the flash memory and the command sequencer may not be able to correctly recognize the command. While it could cause an abnormal termination of the command sequence, it is also possible that the written command is incorrectly recognized.
(Note 4) The SYNC command must be executed immediately after the SW command for each bus write cycle.
(Note 5) For the command sequencer to recognize a command, the device must be in the read mode prior to executing the command. Be sure to check before the first bus write cycle that the FLCS[0] RDY/BSY bit is set to "1." It is recommended to subsequently execute a Read command.
(Note 6) Upon issuing a command, if any address or data is incorrectly written, be sure to perform a system reset operation or issue a reset command to return to the read mode again.

3) Reset

Hardware reset

The flash memory has a reset input as the memory block and it is connected to the CPU reset signal. Therefore, when the RESET input pin of this device is set to V_{IL} or when the CPU is reset due to any overflow of the watch dog timer, the flash memory will return to the read mode terminating any automatic operation that may be in progress. The CPU reset is also used in returning to the read mode when an automatic operation is abnormally terminated or when any mode set by a command is to be canceled. It should also be noted that applying a hardware reset during an automatic operation can result in incorrect rewriting of data. In such a case, be sure to perform the rewriting again.

Refer to Section 21.2.1 "Reset Operation" for CPU reset operations. After a given reset input, the CPU will read the reset vector data from the flash memory and starts operation after the reset is removed.

4) Automatic Page Programming

Writing to a flash memory device is to make "1" data cells to "0" data cells. Any "0" data cell cannot be changed to a "1" data cell. For making "0" data cells to "1" data cells, it is necessary to perform an erase operation.

The automatic page programming function of this device writes data in 128 word blocks. A 128 word block is defined by a same [31:9] address and it starts from the address [8:0] = 0 and ends at the address [8:0] = 0x1FF. This programming unit is hereafter referred to as a "page."

Writing to data cells is automatically performed by an internal sequencer and no external control by the CPU is required. The state of automatic page programming (whether it is in writing operation or not) can be checked by the FLCS [0] <RDY/BSY> register.

Also, any new command sequence is not accepted while it is in the automatic page programming mode. If it is desired to interrupt the automatic page programming, use the hardware reset function. If the operation is stopped by a hardware reset operation, it is necessary to once erase the page and then perform the automatic page programming again because writing to the page has not been normally terminated.

The automatic page programming operation is allowed only once for a page already erased. No programming can be performed twice or more times irrespective of the data cell value whether it is "1" or "0." Note that rewriting to a page that has been once written requires execution of the automatic block erase or automatic chip erase command before executing the automatic page programming command again. Note that an attempt to rewrite a page two or more times without erasing the content can cause damages to the device.

No automatic verify operation is performed internally to the device. So, be sure to read the data programmed to confirm that it has been correctly written.

The automatic page programming operation starts when the fourth bus write cycle of the command cycle is completed. On and after the fifth bus write cycle, data will be written sequentially starting from the next address of the address specified in the fourth bus write cycle (in the fourth bus write cycle, the page top address will be command written) (32 bits of data is input at a time). Be sure to use the SW command in writing commands on and after the fourth bus cycle. In this, any SW command shall not be placed across word boundary. On and after the fifth bus write cycle, data is command written to the same page area. Even if it is desired to write the page only partially, it is required to perform the automatic page programming for the entire page. In this case, the address input for the fourth bus write cycle shall be set to the top address of the page. Be sure to perform command write operation with the input data set to "1" for the data cells not to be set to "0." For example, if the top address of a page is not to be written, set the input data of the fourth bus write cycle to 0xFFFFFFFF to command write the data.

Once the fourth bus cycle is executed, it is in the automatic programming operation. This condition can be checked by monitoring the register bit FLCS [0] <RDY/BSY> (See Table 21-26). Any new command sequence is not accepted while it is in automatic page programming mode. If it is desired to stop operation, use the hardware reset function. Be careful in doing so because data cannot be written normally if the operation is interrupted. When a single page has been command written normally terminating the automatic page writing process, the FLCS [0] <RDY/BSY> bit is set to "1" and it returns to the read mode.

When multiple pages are to be written, it is necessary to execute the page programming command for each page because the number of pages to be written by a single execution of the automatic page program command is limited to only one page. It is not allowed for automatic page programming to process input data across pages.

Data cannot be written to a protected block. When automatic programming is finished, it automatically returns to the read mode. This condition can be checked by monitoring FLCS [0] <RDY/BSY> (See Table 21-26). If automatic programming has failed, the flash memory is locked in the mode and will not return to the read mode. For returning to the read mode, it is necessary to use the reset command or hardware reset to reset the flash memory or the device. In this case, while writing to the address has failed, it is recommended not to use the device or not to use the block that includes the failed address.

(Note) Software reset becomes ineffective in bus write cycles on and after the fourth bus write cycle of the automatic page programming command.

5) Automatic chip erase

The automatic chip erase operation starts when the sixth bus write cycle of the command cycle is completed.

This condition can be checked by monitoring FLCS [0] <RDY/BSY> (See Table 21-26). While no automatic verify operation is performed internally to the device, be sure to read the data to confirm that data has been correctly erased. Any new command sequence is not accepted while it is in an automatic chip erase operation. If it is desired to stop operation, use the hardware reset function. If the operation is forced to stop, it is necessary to perform the automatic chip erase operation again because the data erasing operation has not been normally terminated.

Also, any protected blocks cannot be erased. If all the blocks are protected, the automatic chip erase operation will not be performed and it returns to the read mode after completing the sixth bus read cycle of the command sequence. When an automatic chip erase operation is normally terminated, it automatically returns to the read mode. If an automatic chip erase operation has failed, the flash memory is locked in the mode and will not return to the read mode.

For returning to the read mode, it is necessary to use the reset command or hardware reset to reset the flash memory or the device. In this case, the failed block cannot be detected. It is recommended not to use the device anymore or to identify the failed block by using the block erase function for not to use the identified block anymore.

6) Automatic block erase (128 kB at a time)

The automatic block erase operation starts when the sixth bus write cycle of the command cycle is completed.

This status of the automatic block erase operation can be checked by monitoring FLCS [0] <RDY/BSY> (See Table 21-26). While no automatic verify operation is performed internally to the device, be sure to read the data to confirm that data has been correctly erased. Any new command sequence is not accepted while it is in an automatic block erase operation. If it is desired to stop operation, use the hardware reset function. In this case, it is necessary to perform the automatic block erase operation again because the data erasing operation has not been normally terminated.

Also, any protected blocks cannot be erased. If an automatic block erase operation has failed, the flash memory is locked in the mode and will not return to the read mode. In this case, use the reset command or hardware reset to reset the flash memory or the device.

7) Automatic programming of protection bits

This device is implemented with four protection bits. The protection bits can be individually set in the automatic programming. The applicable protection bit is specified in the seventh bus write cycle. By automatically programming the protection bits, write and/or erase functions can be inhibited (for protection) individually for each block. The protection status of each block can be checked by the FLCS <BLPRO 3:0> register to be described later. This status of the automatic programming operation to set protection bits can be checked by monitoring FLCS <RDY/BSY> (See Table 21-26). Any new command sequence is not accepted while automatic programming is in progress to program the protection bits. If it is desired to stop the programming operation, use the hardware reset function. In this case, it is necessary to perform the programming operation again because the protection bits may not have been correctly programmed. If all the protection bits have been programmed, the flash memory cannot be read from any area outside the flash memory such as the internal RAM. In this condition, the FLCS <BLPRO 3:0> bits are set to "0 x F" indicating that it is in the protected state (See Table 21-26). After this, no command writing can be performed.

(Note) Software reset is ineffective in the seventh bus write cycle of the automatic protection bit programming command. The FLCS <RDY/BSY> bit turns to "0" after entering the seventh bus write cycle.

8) Automatic erasing of protection bits

Different results will be obtained when the automatic protection bit erase command is executed depending on the status of the protection bits. It depends on the status of FLCS <BLPRO 3:0> before the command execution whether it is set to "0 x F" or to any other values. Be sure to check the value of FLCS <BLPRO 3:0> before executing the automatic protection bit erase command.

・When FLCS <BLPRO 3:0> is set to "0 x F" (all the protection bits are programmed):

When the automatic protection bit erase command is command written, the flash memory is automatically initialized within the device. When the seventh bus write cycle is completed, the entire area of the flash memory data cells is erased and then the protection bits are erased. This operation can be checked by monitoring FLCS <RDY/BSY>. If the automatic operation to erase protection bits is normally terminated, FLCS will be set to "0x01." While no automatic verify operation is performed internally to the device, be sure to read the data to confirm that it has been correctly erased. For returning to the read mode while the automatic operation after the seventh bus cycle is in progress, it is necessary to use the hardware reset to reset the flash memory or the device. If this is done, it is necessary to check the status of protection bits by FLCS <BLPRO 3:0> after retuning to the read mode and perform either the automatic protection bit erase, automatic chip erase, or automatic block erase operation, as appropriate.

・When FLCS <BLPRO 3:0> is other than "0 x F" (not all the protection bits are programmed):

The protection condition can be canceled by the automatic protection bit erase operation. With this device, protection bits can be erased handling two bits at a time. The target bits are specified in the seventh bus write cycle and when the command is completed, the device is in a condition the two bits are erased. The protection status of each block can be checked by FLCS <BLPRO 3:0> to be described later. This status of the programming operation for automatic protection bits can be checked by monitoring FLCS <RDY/BSY>. When the automatic operation to erase protection bits is normally terminated, the two protection bits of FLCS <BLPRO 3:0> selected for erasure are set to "0."

In any case, any new command sequence is not accepted while it is in an automatic operation to erase protection bits. If it is desired to stop the operation, use the hardware reset function. When the automatic operation to erase protection bits is normally terminated, it returns to the read mode.

The FLCS <RDY/BSY> bit is "0" while in automatic operation and it turns to "1" when the automatic operation is terminated.

9) Flash control/ status register

This resister is used to monitor the status of the flash memory and to indicate the block protection status.

Table 21-15 Flash Control Register

		7	6	5	4	3	2	1	0
FLCS	Bit Symbol	BLPRO3	BLPRO2	BLPRO1	BLPRO0				RDY/BSY
(0xFFFF_E520)	Read/Write	R				R	R	R	R
	After reset	1	1	0	0	0	0	0	1
	Function	Protection area setting (for each 128 kB) 0000: No blocks are protected xxx1: Block 0 is protected xx1x : Block 1 is protected				"0" can be read.	"0" can be read.	"0" can be read.	Ready/Busy 0: In operation 1: Operation terminated

	15	14	13	12	11	10	9	8
Bit Symbol								
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function								

	23	22	21	20	19	18	17	16
Bit Symbol								
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function								

	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function								

Bit 0: Ready/Busy flag bit

The RDY/BSY output is provided as a means to monitor the status of automatic operation. This bit is a function bit for the CPU to monitor the function. When the flash memory is in automatic operation, it outputs "0" to indicate that it is busy. When the automatic operation is terminated, it returns to the ready state and outputs "1" to accept the next command. If the automatic operation has failed, this bit maintains the "0" output. By applying a hardware reset, it returns to "1."

(Note) Please issue it after confirming the command issue is always a ready state. A normal command not only is sent when the command is issued to a busy inside but also there is a possibility that the command after that cannot be input. In that case, please return by system reset or the reset command.

Bit [7:4]: Protection status bits (can be set to any combination of blocks)

Each of the protection bits (4 bits) represents the protection status of the corresponding block. When a bit is set to "1," it indicates that the block corresponding to the bit is protected. When the block is protected, data cannot be written to it.

10) ID-Read

Using the ID-Read command, you can obtain the type and other information on the flash memory contained in the device. The data to be loaded will be different depending on the address [15:14] of the fourth and subsequent bus write cycles (any input data other than 0xF can be used). On and after the fourth bus write cycle, when an LW command (to read an arbitrary flash memory area) is executed after an SW command, the ID value will be loaded (execute a SYNC command immediately after the LW command). Once the fourth bus write cycle of an ID-Read command has passed, the device will not automatically return to the read mode. In this condition, the set of the fourth bus write cycle and LW/SYNC commands can be repetitively executed. For returning to the read mode, reset the system or use the Read or Read/reset command.

(Important) The "interval between bus write cycles" between successive command sequences must be 15 system clock cycles or longer irrespective of the operating frequency used. This device doesn't have any function to automatically adjust the interval between bus write cycles regarding execution of multiple SW commands to the flash memory. Therefore, if an inadequate interval is used between two sets of bus write cycles, the flash memory cannot be written as expected. Prior to setting the device to work in the onboard programming mode, adjust the bus write cycle interval using a software timer, etc., to verify that the ID-Read command can be successfully executed at the operating frequency of the application program. In the onboard programming mode, use the bus write cycle interval at which the ID-Read command can be operated normally to execute command sequences to rewrite the flash memory.

(3) List of Command Sequences

Table 21-16 Flash Memory Access from the Internal CPU

Command sequence	First bus cycle	Second bus cycle	Third bus cycle	Fourth bus cycle	Fifth bus cycle	Sixth bus cycle	Seventh bus cycle
	Addr.	Addr.	Addr.	Addr.	Addr.	Addr.	Addr.
	Data	Data	Data	Data	Data	Data	Data
Read	0xXX	RA					
	0xF0	RD					
Read/Reset	0x55XX	0xAAXX	0x55XX	RA			
	0xAA	0x55	0xF0	RD			
ID-Read	0x55XX	0xAAXX	0x55XX	IA	0xXX	–	
	0xAA	0x55	0x90	0x00	ID	–	
Automatic page programming (note)	0x55XX	0xAAXX	0x55XX	PA	PA	PA	PA
	0xAA	0x55	0xA0	PD0	PD1	PD2	PD3
Automatic chip erase	0x55XX	0xAAXX	0x55XX	0x55XX	0xAAXX	0x55XX	–
	0xAA	0x55	0x80	0xAA	0x55	0x10	–
Auto Block erase (note)	0x55XX	0xAAXX	0x55XX	0x55XX	0xAAXX	BA	–
	0xAA	0x55	0x80	0xAA	0x55	0x30	–
Protection bit programming	0x55XX	0xAAXX	0x55XX	0x55XX	0xAAXX	0x55XX	PBA
	0xAA	0x55	0x9A	0xAA	0x55	0x9A	0x9A
Protection bit erase	0x55XX	0xAAXX	0x55XX	0x55XX	0xAAXX	0x55XX	PBA
	0xAA	0x55	0x6A	0xAA	0x55	0x6A	0x6A

(4) Supplementary explanation

- RA: Read address
- RD: Read data
- IA: ID address
- ID: ID data
- PA: Program page address
  PD: Program data (32 bit data)
  After the fourth bus cycle, enter data in the order of the address for a page.
- BA: Block address
- PBA: Protect bit address

(Note 1) Always set "0" to the address bits [1:0] in the entire bus cycle. (Setting values to bits [7:2] are undefined.)
(Note 2) Bus cycles are "bus write cycles" except for the second bus cycle of the Read command, the fourth bus cycle of the Read/reset command, and the fifth bus cycle of the ID-Read command. Bus write cycles are executed by SW commands. "Data" in the table is to be stored for SW commands. The address [31:16] in each bus write cycle should be the target flash memory address [31:16] of the command sequence. Use "Addr." in the table for the address [15:0].
(Note 3) In executing the bus write cycles, the interval between each bus write cycle shall be 15 system clocks or more.
(Note 4) The "Sync command" must be executed immediately after completing each bus write cycle.
(Note 5) Execute the "Sync command" immediately following the "LW command" after the fourth bus write cycle of the ID-Read command.

(5) Address bit configuration for bus write cycles

Table 21-17 Address Bit Configuration for Bus Write Cycles

Address	Addr [31:21]	Addr [20]	Addr [19]	Addr [18:17]	Addr [16]	Addr [15]	Addr [14]	Addr [13]	Addr [12:9]	Addr [8]	Addr [7:0]
Normal commands	**Normal bus write cycle address configuration**										
	Flash area	"0" is recommended				Command					Addr[1:0]=0 (fixed) Others:0 (recommended)
Block erase	**BA: Block address (Set the sixth bus write cycle address for block erase operation)**										
	Flash area	"0" is recommended		Block selection		Addr[1:0]=0 (fixed), Others: 0 (recommended)					
Auto page programming	**PA: Program page address (Set the fourth bus write cycle address for page programming operation)**										
	Flash area	"0" is recommended		Block selection		Page selection					Addr[1:0]=0 (fixed), Others: 0 (recommended)
ID-READ	**IA: ID address (Set the fourth bus write cycle address for ID-Read operation)**										
	Flash area	"0" is recommended				ID address					Addr[1:0]=0 (fixed), Others: 0 (recommended)
Protection bit programming	**PBA: Protection bit address (Set the seventh bus write cycle address for protection bit programming)**										
	Flash area	"0" is recommended				Protection bit write 00: Block 0 01: Block 1					Addr[1:0]=0 (fixed), Others: 0 (recommended)
Protection bit erase	**PBA: Protection bit address (Set the seventh bus write cycle address for protection bit erasure)**										
	Flash area	"0" is recommended				Erase protection for 0: Block 0 1: Block 1					Addr[1:0]=0 (fixed), Others: 0 (recommended)

(Note 1) Table 24-17 "Flash Memory Access from the Internal CPU" can also be used.

(Note 2) Address setting can be performed according to the "Normal bus write cycle address configuration" from the first bus cycle.

(Note 3) "0" is recommended" can be changed as necessary.

Table 21-18 Block Erase Address Table

BA	Address range		Size
	Flash memory address	When applied to the projected area	
Block 0	0xBFC0_0000~0xBFC1_FFFF	0x0000_0000~0x0001_FFFF	128 Kbyte
Block 1	0xBFC2_0000~0xBFC3_FFFF	0x0002_0000~0x0003_FFFF	128 Kbyte

Example: When BA0 is to be selected, any single address in the range 0xBFC0_0000 to 0xBFC1_FFFF may be entered.

As for the addresses from the first to the sixth bus cycles, specify the upper 4 bit with the corresponding flash memory addresses of the blocks to be erased.

Table 21-19 Protection Bit Programming Address Table

OPBA	The seventh bus write cycle address [15:14]	
	Address [15]	Address [14]
Block 0	0	0
Block 1	0	1

Table 21-20 Protection Bit Erase Address Table

OPBA	The seventh bus write cycle address [15:14]	
	Address [15]	Address [14]
Block 0	0	X
Block 1	0	X

The protection bit erase command will erase bits 0 and 1 together.
The bits 2 and 3 are also erased together.

Table 21-21 The ID-Read command's fourth bus write cycle ID address (IA) and the data to be read by the following LW command (ID)

IA [15:14]	ID [7: 0]	Code
00b	0x98	Manufacturer code
01b	0x5A	Device code
10b	Reserved	---
11b	0x09	Macro code

(6) Flowchart

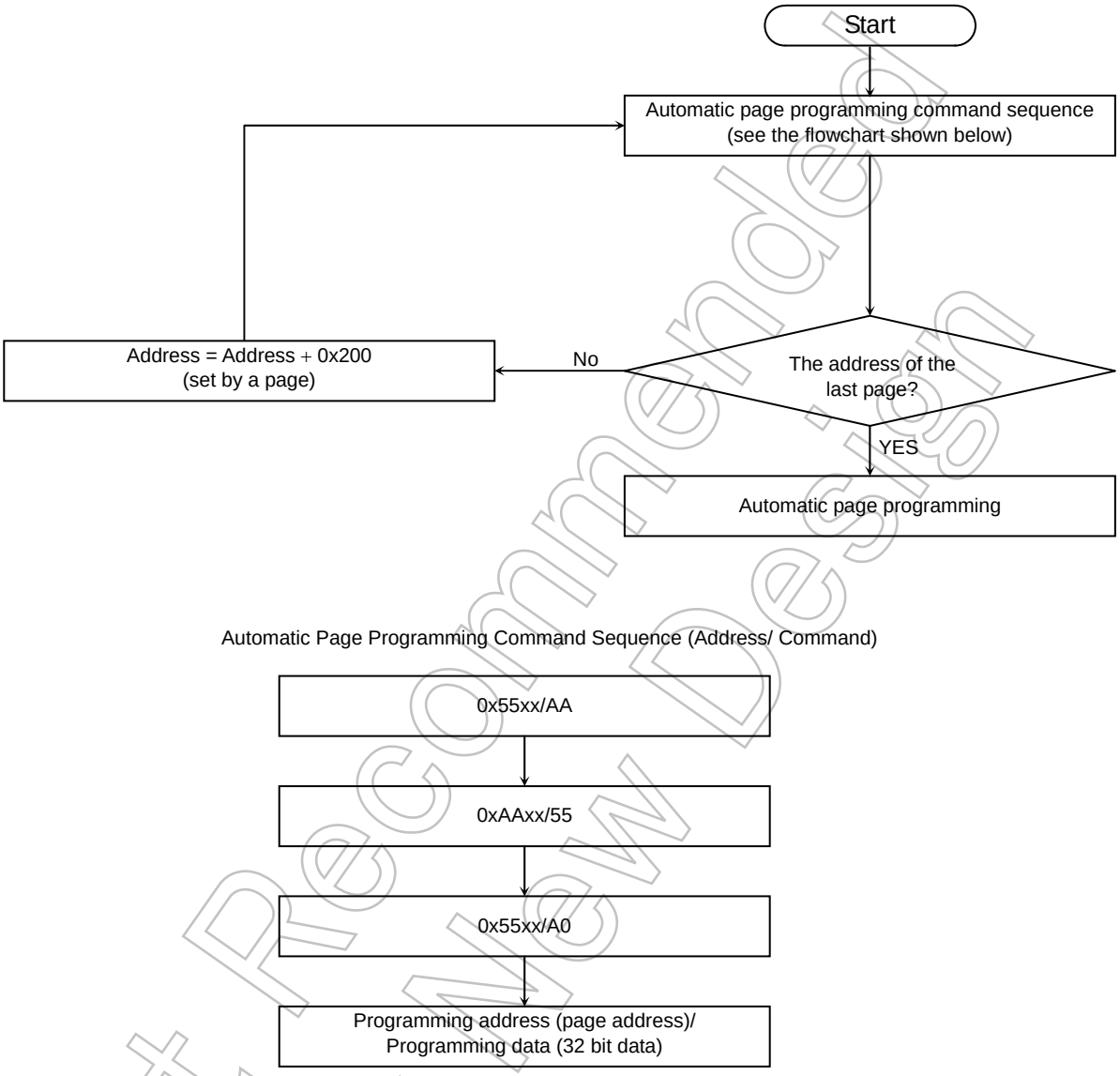


Fig. 21-11 Automatic Programming

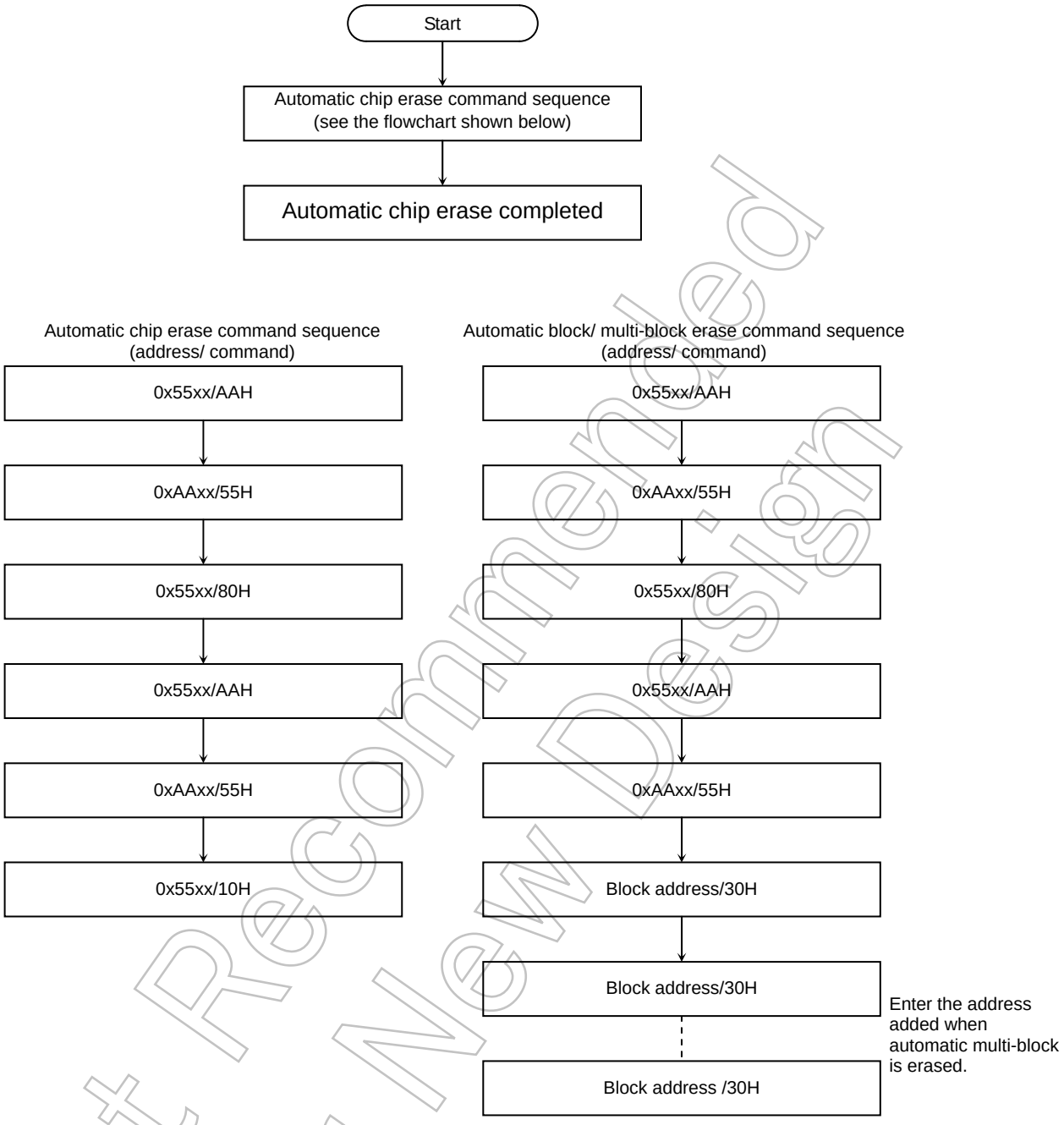


Fig. 21-12 Automatic Erase

22. Various protecting functions

22.1 Overview

The ROM protect function for designating the internal ROM (Flash) area as a read-protected area and the DSU protecting function for prohibiting the use of DSU (DSU-Probe) are built into the TMP19A23. The read protecting functions specifically include the following:

- Flash protecting function
- ROM data protecting function
- DSU protecting function

22.2 Features

22.2.1 Flash Protecting Function

A built-in Flash can prohibit the operation of writing and the deletion at every the block of every 128 Kbyte. This function is called the block protecting.
To make the block protecting function effective, set the protect bit, which corresponds to the block to protect, to “1”. The block protecting can be released by making the protecting bit "0". (Please see the chapter of the Flash operation explanation about the program method.) The protecting bit can be monitored by FLCS register < BLPRO3:0 > bit.
The state to put protecting on all blocks is called the Flash protecting. Please note that all the protecting bits become "0" after automatically deleting all data of the Flash when the protecting release operates after it puts it into the state of the Flash protection (operation that makes the protecting bit "0").

The Flash protecting must be activated to validate the “ROM data protecting” and “DSU protecting” described later.

22.2.2 ROM data Protecting

ROM data protecting restrict the on-chip RAM from reading out the data.
It also prohibits the Flash from executing commands. When ROM protecting register ROMSEC1<RSECON> bit is "1", ROM data protecting becomes effective with Flash protected.

The default setting of RSECON bit is “1”.
It never goes into ROM data protecting state unless all the blocks of Flash are not protected. When it goes in to the Flash protecting state with the entire Flash blocks protected, the ROM data protecting state is set as the default.

(Note) Under the ROM data protecting condition, only the command in the internal ROM is accessible to RSECON bit. Please note that the protection releasing program is needed to be stored in the internal ROM.

If instructions in the ROM area have been replaced with instructions in the RAM area in a PC by using the ROM correction function, a PC shows the instructions as residing in the flash ROM area. Because they actually reside in the RAM area, data cannot be read in a ROM protected state. To read data by using instructions held in the overwritten RAM area, it is necessary to write data to RAM by using a program available in the ROM area or to use other means.

If the ROM area is put in a protected state, the following operations cannot be performed:

- Using instructions placed in areas other than the ROM area to load or store the data taken from the ROM area
- Store to DMAC register (NMI by the bus error is generated.)
- Loading or storing the data taken from the ROM area in accordance with EJTAG
- Using BOOT-ROM to load or store the data taken from the ROM area (FLASH only)
- Executing flash writer to load or store the data taken from the ROM area(FLASH only)
- Using instructions placed in areas other than the ROM area to access the registers (ROMSEC1, ROMSEC2) that concern the protection of the ROM area
- Executing the command to unprotect automatic blocking in writer mode, performing the flash command sequences other than the automatic blocking unprotect command sequence, and performing the flash command sequence in single or boot mode by specifying an address in the ROM area (FLASH only).

The following operations can be performed even if the ROM area is in a protected state:

- Using instructions placed in the ROM area to load the data taken from the ROM area
- Using instructions placed in all areas to load the data taken from areas other than the ROM area
- Using instructions placed in all areas to make instructions branch off to the ROM area
- Performing PC trace (there are restrictions) or break on the ROM area in accordance with EJTAG
- Data transfer of ROM area by DMAC

22.2.3 DSU Protecting

The DSU protecting function is a function for invalidating the connection of DSU-probe to prohibit the third parties other than the user from reading the data of a built-in flash easily.

When SEQMOD register < DSUOFF > bit is "1", the DSU protecting becomes effective with Flash protected.

The default of the DSUOFF bit is set to "1". It enters the state of the DSU protecting as long as the FLASH protecting is always effective and the DSUOFF bit is not set to "0" by the user program. Unless all the blocks in the Flash are protected, DSU protecting is not activated. If all the blocks in the Flash are protected (Flash protecting state), DSU protecting is the default setting in the DSUOFF bit.

(Note) The DSUOFF bit can be accessed only with the instruction put on built-in ROM in the state of ROM data protecting. Please note that it is necessary to put the program of the DSU protecting release on internal ROM.

22.3 Protect Configuration and Protect Statuses

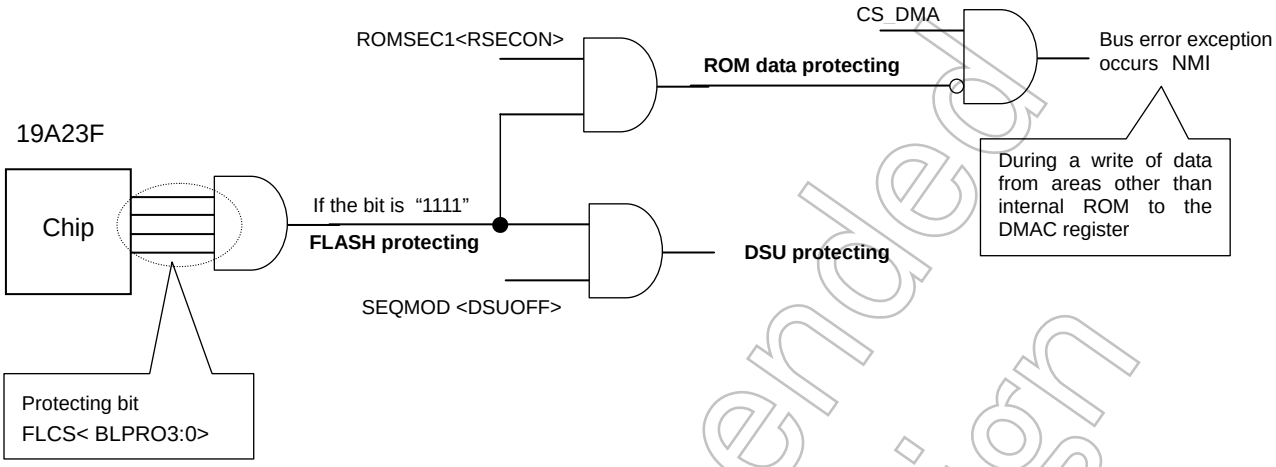


Fig. 22-1 Various Protect Statuses

Table 22-1 Protect Statuses in Each Mode

	FLASH protecting	ON				OFF
	ROM data protecting	ON		OFF		OFF
	DSU protecting	ON	OFF	ON	OFF	OFF
Single /single boot mode	Read from internal ROM	○	○	○	○	○
	Read from areas other than internal ROM	x *1	x *1	○	○	○
	Internal ROM read by DSU-probe	x	○	x	○	○
	RSECON rewrite (from internal ROM)	○	○	○	○	○
	RSECON rewrite (from areas other than internal ROM)	x *2	x *2	○	○	○
	DSUOFF rewrite (from internal ROM)	○	○	○	○	○
	DSUOFF rewrite (from areas other than internal ROM)	x *3	x *3	○	○	○
	Issuing of the command to erase protect bits	x *4	x *4	○ *7	○ *7	○
	Issuing of commands other than the command to erase protect bits	x *4	x *4	x *6	x *6	△ *8
	Writing of data to the DMAC setting register (from ROM)	○	○	○	○	○
	Writing of data to the DMAC setting register (from areas other than ROM)	x *5	x *5	○	○	○

*1 : The data of address "0xBFC0_0000" or "0xBFC0_0002" can be read.
*2 : Cannot write on (clear) the RSECON bit.
*3 : Cannot write on (clear) the DSUOFF bit.
*4 : Flash memory does not recognize commands.
*5 : Non-maskable interruption occurs.
*6 : Flash memory does not recognize commands.
*7 : Issued commands are converted to the command for erasing the whole flash memory area and the command for erasing all protect bits.
*8 : Due to the protecting bit status, commands to the protected bit are rejeced.
*9 : Data is always read as 0x00000098.

22.4 Register

Flash control/status register

This register shows the status of flash memory being monitored and the block protect status of flash memory.

Table 22-2 Flash Control Register

FLCS (0xFFFF_E520)

	7	6	5	4	3	2	1	0
Bit Symbol	BLPRO3	BLPRO2	BLPRO1	BLPRO0				RDY/BSY
Read/Write	R				R	R	R	R
After reset by power-on	1	1	0	0	0	0	0	1
Function	Protect area setting (in units of 128 KB) 0000：All blocks unprotected xxx1：Block 0 protected xx1x：Block 1 protected				“0” can be read.	“0” can be read.	“0” can be read.	Ready/Busy 0: In auto operation 1: Auto operation completed

	15	14	13	12	11	10	9	8
Bit Symbol								
Read/Write	R							
After reset by power-on	0	0	0	0	0	0	0	0
Function								

	23	22	21	20	19	18	17	16
Bit Symbol								
Read/Write	R							
After reset by power-on	0	0	0	0	0	0	0	0
Function								

	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset by power-on	0	0	0	0	0	0	0	0
Function								

Bit 0: Ready/Busy flag bit

The RDY/BSY output is provided to identify the status of auto operation. This bit is a functional bit for monitoring this function by communicating with the CPU. If flash memory is in auto operation, "0" is output to show that flash memory is busy. As flash memory completes auto operation and goes into a ready state, "1" is output and the next command will be accepted. If the result of auto operation is faulty, this bit continues to output "0." It returns to "1" upon a hardware reset.

(Note) Before issuing a command, make sure that flash memory is in a ready state. If a command is issued when flash memory is busy, a right command cannot be generated and there is the possibility that subsequent commands may not be able to be input. In this case, you must return to a normal functional state by executing a system reset or issuing a reset command.

Bit [7:4]: Protect bit (x: A combination setting can be made for each block)

The protect bit (4-bit) value corresponds to the protect status of each block. If this bit is "1," the corresponding block is in a protected state. A protected block cannot be overwritten.

Table 22-3 ROM Protect Register

ROMSEC1
(0xFFFF_E518)

	7	6	5	4	3	2	1	0
Bit Symbol								RSECON
Read/Write	R							R/W
After reset by power-on	0							1
Function	"0" is always read.							ROM protecting 1: ON 0: OFF (see note)

	15	14	13	12	11	10	9	8
Bit Symbol								
Read/Write	R							
After reset by power-on	0							
Function	"0" is always read.							

	23	22	21	20	19	18	17	16
Bit Symbol								
Read/Write	R							
After reset by power-on	0							
Function								

	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset by power-on	0							
Function	"0" is always read.							

(Note 1) This register is initialized only by power-on reset. To access this register, 32-bit access is required.

Table 22-4 ROM Protect Lock Register

ROMSEC2
(0xFFFF_E51C)

	7	6	5	4	3	2	1	0
Bit Symbol								
Read/Write	W							
After reset	Undefined							
Function	(See note)							
	15	14	13	12	11	10	9	8
Bit Symbol								
Read/Write	W							
After reset	Undefined							
Function	(See note)							
	23	22	21	20	19	18	17	16
Bit Symbol								
Read/Write	W							
After reset	Undefined							
Function	(See note)							
	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	W							
After reset	Undefined							
Function	(See note)							

(Note) If this register is set to "0x0000_003D" after ROMSEC1<RSECON> is set, appropriate bit values are automatically set in ROMSEC1<RSECON>.
If the ROM area is protected, the registers ROMSEC1 and ROMSEC2 can be accessed only by using the instructions residing in the ROM area.
To access this register, 32-bit access is required.
This register is a write-only register. If it is read, values will be undefined.

Table 22-5 DSU Protect Mode Register

SEQMOD (0xFFFF_E510)

	7	6	5	4	3	2	1	0
Bit Symbol								DSUOFF
Read/Write	R							R/W
After reset	0							1
Function	"0" is always read.							1: DSU disabled 0: DSU enabled

	15	14	13	12	11	10	9	8
Bit Symbol								
Read/Write	R							
After reset	0							
Function	"0" is always read.							

	23	22	21	20	19	18	17	16
Bit Symbol								
Read/Write	R							
After reset	0							
Function								

	31	30	29	28	27	26	25	24
Bit Symbol								
Read/Write	R							
After reset	0							
Function	"0" is always read.							

(Note) This register is initialized only by power-on reset.
To access this register, 32-bit access is required.

Table 22-6 DSU Protect Control Register

SEQCNT (0xFFFF_E514)

	7	6	5	4	3	2	1	0
Bit Symbol	DSECODE 07	DSECODE 06	DSECODE 05	DSECODE 04	DSECODE 03	DSECODE 02	DSECODE 01	DSECODE 00
Read/Write	W							
After reset	0							
Function	Write "0x0000_00C5".							

	15	14	13	12	11	10	9	8
Bit Symbol	DSECODE 15	DSECODE 14	DSECODE 13	DSECODE 12	DSECODE 11	DSECODE 10	DSECODE 09	DSECODE 08
Read/Write	W							
After reset	0							
Function	Write "0x0000_00C5".							

	23	22	21	20	19	18	17	16
Bit Symbol	DSECODE 23	DSECODE 22	DSECODE 21	DSECODE 20	DSECODE 19	DSECODE 18	DSECODE 17	DSECODE 16
Read/Write	W							
After reset	0							
Function	Write "0x0000_00C5".							

	31	30	29	28	27	26	25	24
Bit Symbol	DSECODE 31	DSECODE 30	DSECODE 29	DSECODE 28	DSECODE 27	DSECODE 26	DSECODE 25	DSECODE 24
Read/Write	W							
After reset	0							
Function	Write "0x0000_00C5".							

(Note) To access this register, 32-bit access is required.
This register is a write-only register. If it is read, values will be undefined.

22.5 Proted-related / Release Settings

If it is necessary to overwrite Flash memory or protect bits in a protected state, "automatic protect bit deletion" must be executed or the ROM protect function must be disabled. DSU cannot be used if it is in a protected state.

Flash memory may go into a read-protected state after the automatic protect bit program is executed. In this case, it is necessary to set DSU-PROBE to "enable" before the automatic protect bit program is executed.

If "automatic protect bit deletion" is executed when Flash memory is in a read-protected state, Flash memory is automatically initialized inside this device. Therefore, release the ROM data protection in advance to rewrite the data in Flash memory.

22.5.1 Flash Protect Function

The Flash protecting function becomes effective by putting the block protecting on all the blocks. The flash memory protecting bit program commands are used to enable or disable the flash read protect function. For further information, refer to the command sequence explained in the chapter describing the operations of flash memory.

The protecting bit is cleared after all the data in the flash is deleted when the protecting bit release command is executed with the flash protected, and the flash protecting is released.
In the state of ROM data protecting, explains as follows, the command execution to the flash is disregarded. It is necessary to release ROM data protecting first clearing the RSECON bit of ROM protecting register when the flash protecting is released with ROM protected.

22.5.2 ROM data Protecting

ROM data protecting is effective the flash protecting and becomes effective at ROM protecting register ROMSEC1<RSECON>="1".
After releasing reset, the RSECON bit is initialized by "1". The flash protecting is sure to enter the state of ROM data protecting in the mask version after releasing reset because it is always effective. The condition whether to enter the state of ROM data protecting is defined depending on the state of the flash protecting.

When ROM protecting register is rewritten with ROM data protected, rewriting can be executed only from the program put on internal ROM. Therefore, it is necessary to prepare the release program of ROM data protecting on internal ROM.

RSECON bit consists of the 2 data paths to prevent the unintended release due to overdrive.

Fig. 22-2 shows the schematic.

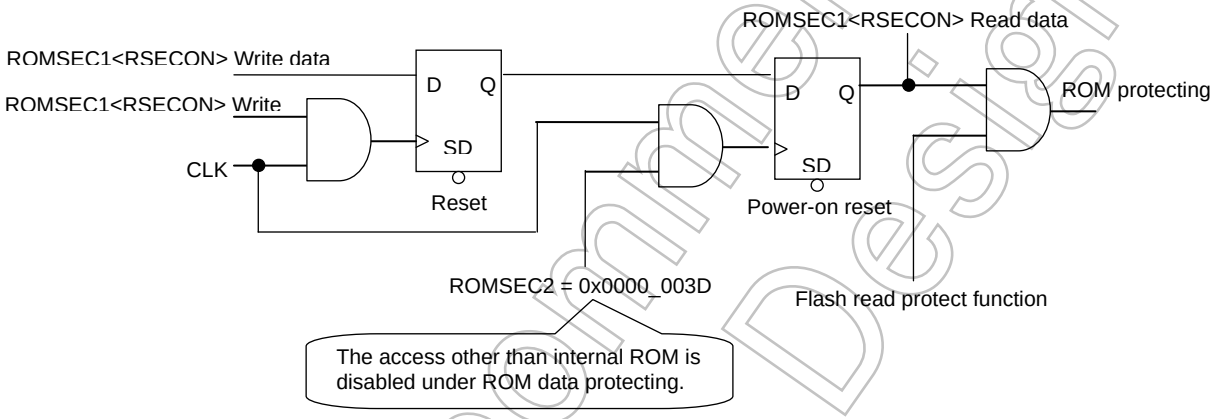


Fig. 22-2 ROM data protecting

ROM data protecting is released by setting ROM protecting register ROMSEC1<RSECON>"0" and writing protecting code "0x0000_003D" in ROM protecting lock register ROMSEC2. Moreover, ROM data protecting function can be set again by similarly setting ROM protecting register ROMSEC1<RSECON> "1" and writing protecting code "0x0000_003D" in ROM protecting lock register ROMSEC2.

Please note the reading data is different from original write data since the ROMSEC2 register is only for writing.

ROM protecting register is initialized by power-on reset, and the value doesn't change by a normal reset.

22.5.3 DSU Protecting

22.5.4 DSU enable/ disable (enabling/ disabling debug function with DSU-PROBE)

DSU data protecting is effective the flash protecting and becomes effective at DSU protecting register SEQMOD< DSUOFF >=“1".
After releasing reset, the DSUOFF bit is initialized by "1". The condition whether to enter the state of ROM data protecting is defined depending on the state of the flash protecting.

When DSU protecting register is rewritten with ROM data protected, rewriting can be executed only from the program put on internal ROM. Therefore, it is necessary to prepare the release program of DSU data protecting on internal ROM.

DSUOFF bit consists of the 2 data paths to prevent the unintended release due to overdrive.

Fig. 22-3 shows the schematic.

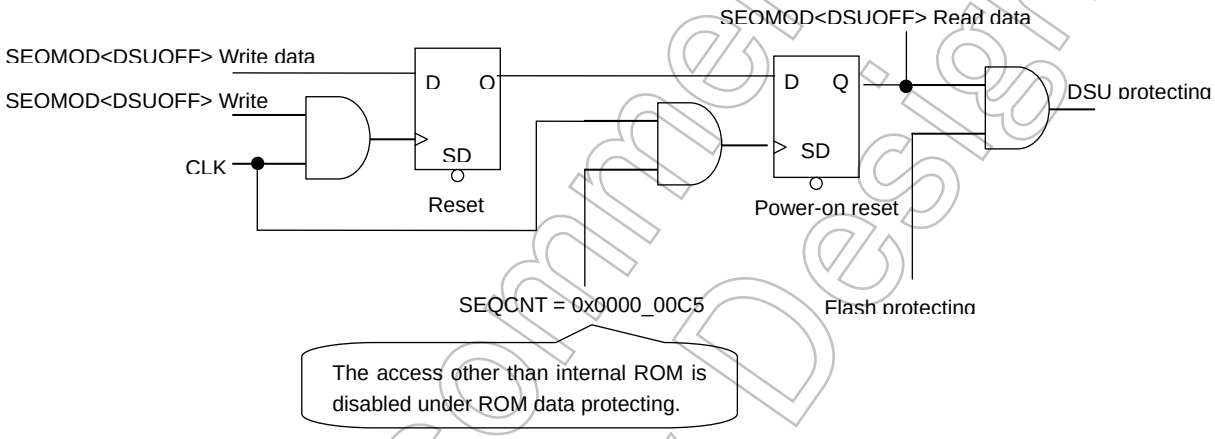


Fig. 22-3 DSU protecting

DSU protecting is released by setting DSU protecting mode register SEQMOD<DSUOFF>“0" and writing protecting code “0x0000_00C5" in DSU protecting control register SEQCNT. Moreover, DSU protecting function can be set again by similarly setting ROM protecting mode register SEQMOD<DSUOFF>“1" and writing protecting code “0x0000_00C5" in DSU protecting control register SEQCNT.

Please note the reading data is different from original write data because of the SEQCNT register is only for writing.

It provides with the power-on reset circuit. DSU protecting register is initialized by power-on reset, and the value doesn't usually change in reset.

22.5.5 ROM Protect Register: ROMSEC1<RSECON>

The ROM protect register is equipped with a power-on reset circuit. Caution must be exercised as data read from the ROMSEC1<RSECON> bit is different from the actually written data. How data is processed is shown below. It is initialized by power-on reset.

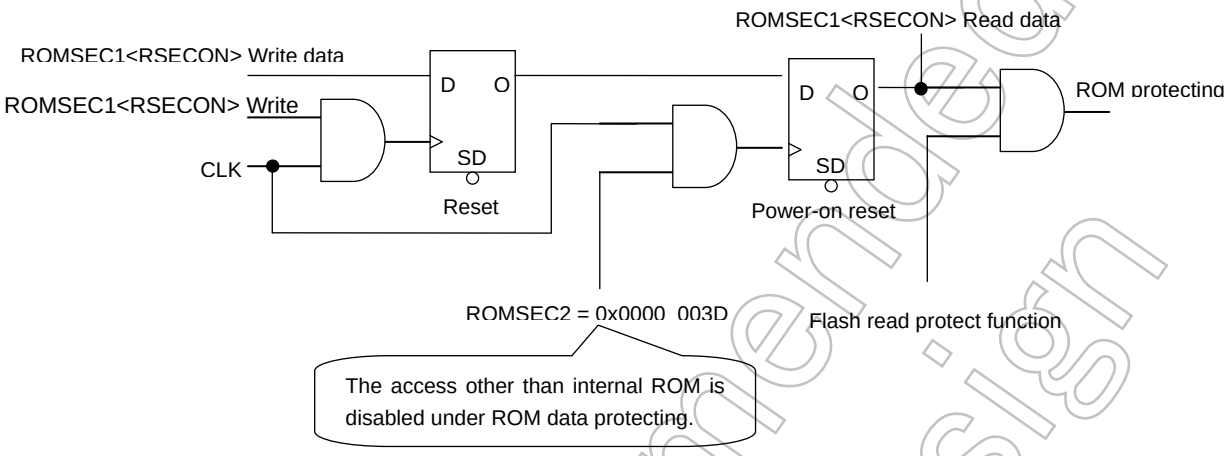


Fig. 22-4 ROMSEC1 Power-on reset circuit

22.5.6 DSU Protect Mode Register: SEQMOD <DSUOFF>

The DSU protect mode register is equipped with a power-on reset circuit. Caution must be exercised as data read from the SEQMOD <DSUOFF> bit is different from the actually written data. How data is processed is shown below. It is initialized by power-on reset.

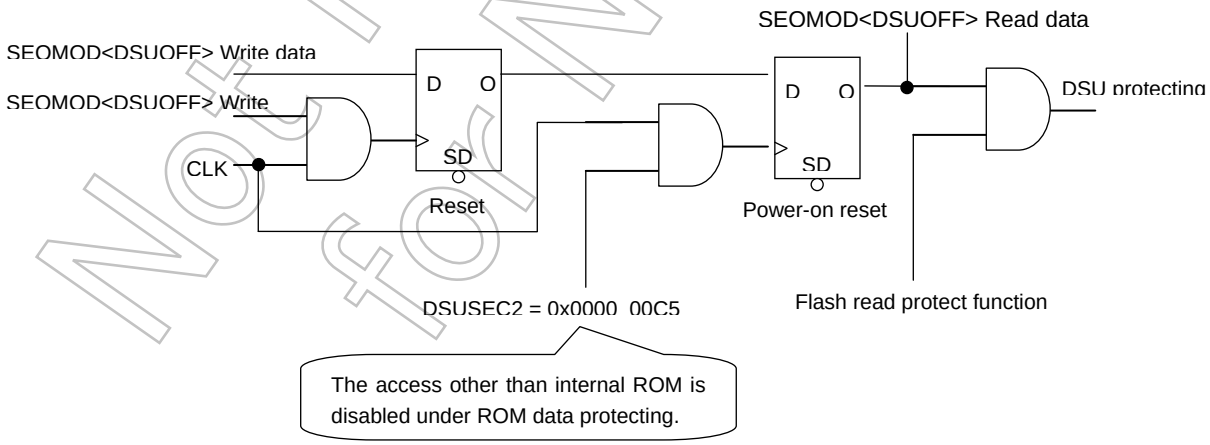


Fig. 22-5 SEQMOD power-on reset circuit

23 Electrical Characteristics

23.1 Absolute Maximum Ratings

Parameter		Symbol	Rating	Unit
Supply voltage		Vcc15 (Core)	− 0.3 ~ 3.0	V
		Vcc3(I/O)	− 0.3 ~ 3.9	
		AVCC3 (A/D)	− 0.3 ~ 3.9	
		DVCC3	− 0.3 ~ 3.9	
Input voltage	Other than 5V tolerant pin	V_{IN}	− 0.3 ~ V_{CC} + 0.3	V
	5V tolerant pin		− 0.3 ~ 5.5	
Low-level output current	Per pin	I_{OL}	5	mA
	Total	ΣI_{OL}	50	
High-level output current	Per pin	I_{OH}	− 5	
	Total	ΣI_{OH}	50	
Power dissipation (Ta = 85°C)		PD	600	mW
Soldering temperature (10s)		T_{SOLDER}	260	°C
Storage temperature		T_{STG}	− 40 ~ 125	°C
Operating Temperature	Except during Flash W/E	T_{OPR}	− 20 ~ 85	°C
	During Flash W/E		0 ~ 70	
Write/erase cycles		N_{EW}	100	cycle

V_{CC15}=DVCC15=CVCC15, V_{CC}3=DVCC3,

V_{SS}=DVSS=AVSS=CVSS

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 with respect to current, voltage, power consumption, temperature, etc. 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.

23.2 DC Electrical Characteristics (1/3)

[1] BGA

Ta=-20~85°C

Parameter		Symbol	Rating	Min.	Typ. (Note 1)	Max.	Unit
Supply voltage	AVCC3 = 3.3V CVCC15=DVCC15 DVCC3	DVCC15 CVCC15	fosc = 8~10MHz fsys = 4MHz~40MHz	1.35		1.65	V
	CVSS=DVSS= AVSS= 0V	DVCC3		2.7		3.6	
Low-level input voltage	P7~P8	V_{IL1}	2.7V≤AVCC3≤3.6V	−0.3		0.3 AVCC3	V
	Normal port	V_{IL2}	2.7V≤DVCC3≤3.6V			0.3 DVCC3	
	Schmitt-Triggered port	V_{IL3}	2.7V≤DVCC3≤3.6V (Other than 5V tolerant pin)			0.2 DVCC3	
			2.7V~5.5V (5V tolerant pin)				
	X1	V_{IL4}	1.35V≤CVCC15≤1.65V			0.1 CVCC15	

(Note 1)Ta = 25°C, DVCC15=1.5V, DVCC3= AVCC3=3.3V, unless otherwise noted.

Ta=-20~85°C

Parameter		Symbol	Rating		Min.	Typ. (Note 1)	Max.	Unit
High-level input voltage	P7~P8	V_{IH1}	2.7V≤AVCC3≤3.6V		0.7 AVCC3		DVCC3+0.3 DVCC15+0.2 CVCC15+0.2	V
	Normal port	V_{IH2}	2.7V≤DVCC3≤3.6V		0.7 DVCC3			
	Schmitt-Triggered port	V_{IH3}	2.7V≤DVCC3≤3.6V (Other than 5V tolerant pin)		0.8 DVCC3			
			2.7V~5.5V (5V tolerant pin)					
	X1	V_{IH4}	1.35V≤CVCC15≤1.65V		0.9 CVCC15			
Low-level output voltage		V_{OL}	I_{OL}=2mA	DVCC3≥2.7V			0.4	V
High-level output voltage		V_{OH}	I_{OH}=−2mA	DVCC3≥2.7V	2.4			

(Note 1) Ta=25°C, DVCC15=1.5V, DVCC3=AVCC3=3.3V, unless othrwise noted.

[2] LQFP

Ta=-20~85°C

Parameter		Symbol	Rating	Min.	Typ. (Note 1)	Max.	Unit
Supply voltage	AVCC3 = 3.3V DVCC3=REGVIN CVSS=DVSS=AVSS= 0V	DVCC3 REGVIN	fosc = 8~13.5MHz fsys = 4MHz~54MHz	3.0		3.6	V
Low level input voltage	P7~P8	V_{IL1}	3.0V≤AVCC3≤3.6V	−0.3		0.3 AVCC3	V
	Normal port	V_{IL2}	3.0V≤DVCC3≤3.6V			0.3 DVCC3	
	Schmitt-Triggered port	V_{IL3}	3.0V≤DVCC3≤3.6V (Other than 5V tolerant pin)			0.3 DVCC3	
			3.0V~5.5V (5V tolerant pin)			0.2 DVCC3	

(Note 1) Ta=25°C, DVCC3=REGVIN=AVCC3=3.3V, unless otherwise noted.

(Note 2) REGVOUT, DVCC15 and CVCC15 must be wired outside the device.

Ta=-20~85°C

Parameter		Symbol	Rating		Min.	Typ. (Note 1)	Max.	Unit
High level input voltage	P7~P8	V_{IH1}	3.0V≤AVCC3≤3.6V		0.7 AVCC3		DVCC3+0.3	V
	Normal port	V_{IH2}	3.0V≤DVCC3≤3.6V		0.7 DVCC3			
	Schmitt-Triggered port	V_{IH3}	3.0V≤DVCC3≤3.6V (Other than 5V tolerant pin)		0.8 DVCC3			
			3.0V~5.5V (5V tolerant pin)					
Low-level output voltage		V_{OL}	I_{OL}=2mA	DVCC3≥3.0V			0.4	V
High-level output voltage		V_{OH}	I_{OH}=−2mA	DVCC3≥3.0V	2.4			

(Note 1) Ta=25°C, DVCC3=REGVIN=AVCC3=3.3V unless otherwise noted.

(Note 2) REGVOUT, DVCC15 and CVCC15 must be wired outside the device.

23.3 DC Electrical Characteristics (2/3)

[1] BGA

Ta=-20~85°C

Parameter	Symbol	Rating	Min.	Typ. (Note 1)	Max.	Unit
Input leakage current	I_{LI}	$0.0 \le V_{IN} \le$DVCC15 $0.0 \le V_{IN} \le$DVCC3 $0.0 \le V_{IN} \le$AVCC3		0.02	± 5	μA
Output leakage current	I_{LO}	$0.2 \le V_{IN} \le$ DVCC15 – 0.2 $0.2 \le V_{IN} \le$ DVCC3 – 0.2 $0.2 \le V_{IN} \le$ AVCC3 – 0.2		0.05	± 10	
Pull up resister at Reset	RRST	DVCC3 = 2.7V~3.6V	30	55	100	kΩ
Schmitt-Triggered port	VTH	2.7V≤DVCC3≤3.6V	0.3	0.6		V
Programmable pull up/ pull down resistor	PKH	DVCC3 = 2.7V~3.6V	40	55	100	kΩ
Pin capacitance (Except power supply pins)	C_{IO}	fc = 1MHz			10	pF

(Note 1) Ta=25°C, DVCC15=1.5V, DVCC3=AVCC3=3.3V, unless otherwise noted.

[2] LQFP

Ta=-20~85°C

Parameter	Symbol	Rating	Min.	Typ. (Note 1)	Max.	Unit
Input leakage current	I_{LI}	$0.0 \le V_{IN} \le$DVCC3 $0.0 \le V_{IN} \le$AVCC3		0.02	± 5	μA
Output leakage current	I_{LO}	$0.2 \le V_{IN} \le$ DVCC3 – 0.2 $0.2 \le V_{IN} \le$ AVCC3 – 0.2		0.05	± 10	
Pull up resister at Reset	RRST	DVCC3 = 3.0V~3.6V	30	55	100	kΩ
Schmitt-Triggered port	VTH	3.0V≤DVCC3≤3.6V	0.3	0.6		V
Programmable pull up/ pull down resistor	PKH	DVCC3 = 3.0V~3.6V	40	55	100	kΩ
Pin capacitance (Except power supply pins)	C_{IO}	fc = 1MHz			10	pF

(Note 1) Ta=25°C, DVCC3=REGVIN=AVCC3=3.3V, unless otherwise noted.

23.4 DC Electrical Characteristics (3/3)

[1] BGA

DVCC15=CVCC15=1.35V~1.65V, DVCC3=AVCC3=2.7V~3.6V, Ta=-20~85°C

Parameter	Symbol	Rating	Min.	Typ. (Note 1)	Max.	Unit
NORMAL(Note 2) Gear 1/1	I_{CC}	f_{sys} = 40 MHz (f_{osc} = 10 MHz)		36 (note 4)	58	mA
IDLE(Doze) (Note 3)				9	20	
IDLE(Halt) (Note 3)				8	19	
STOP				50	3500	μA

(Note 1) Ta=25°C, DVCC15=1.5V , DVCC3=AVCC3=3.3 V, unless otherwise noted.
(Note 2) I_{CC} NORMAL:
Measured with the CPU dhrystone operating (excluding DSU), RAM, FLASH.
All functions operating excluding A/D.
(Note 3) I_{CC} IDLE:
Measured with all functions stopped.
The currents flow through DVCC15, DVCC3, CVCC15 and AVCC3 are included.
(Note 4) The current value exceeds the values shown above by approx. 10 mA in executing the program with internal RAM.

[2] LQFP

DVCC3=REGVIN=AVCC3=3.0V~3.6V, Ta=-20~85°C

Parameter	Symbol	Rating	Min.	Typ. (Note 1)	Max.	Unit
NORMAL (Note 2) Gear 1/1	I_{CC}	f_{sys} = 54 MHz (f_{osc} = 13.5 MHz)		46	69	mA
IDLE(Doze) (Note 3)				12	23	
IDLE(Halt) (Note 3)				11	21	
STOP				300	3500	μA

(Note 1) Ta=25°C, DVCC3=REGVIN=AVCC3=3.3 V, unless otherwise noted.
(Note 2) I_{CC} NORMAL:
Measured with the CPU dhrystone operating (excluding DSU), RAM, FLASH.
All functions operating excluding A/D.
(Note 3) I_{CC} IDLE:
Measured with all functions stopped.
The currents flow through DVCC3, REGVIN and AVCC3 are included.

23.5 DC Electrical Characteristics (4/4) For LQFP only

[1] Recommended circuit

TMP19A23 incorporates a regulator. Fig. 23-1 illustrate a recommended circuit foraregulator power unit. Capacitors are required for stable operation on input/ output processing.

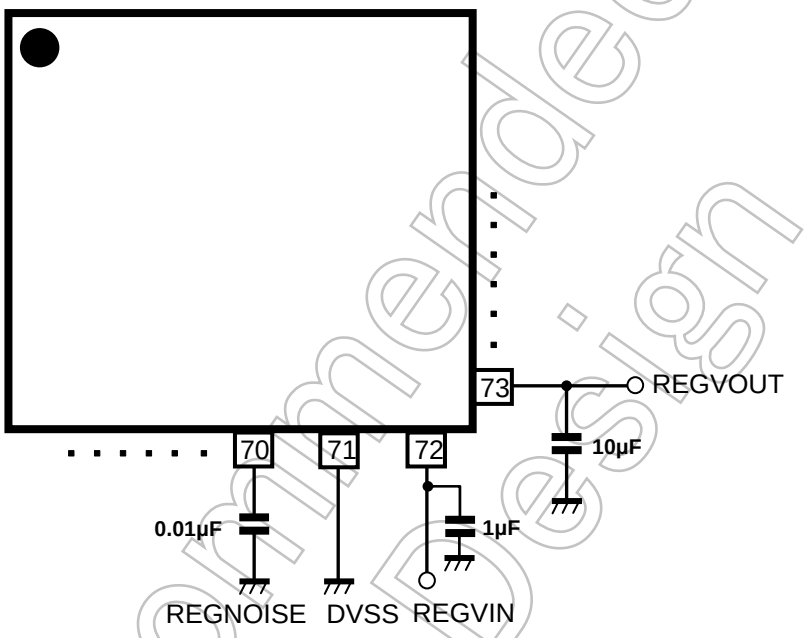


Fig. 23-1 Recommended circuit for a regulator power unit

[2] REGNOISE pin

TMP19A23 incorporates REGNOISE pin for decreasing output noise voltage. Output noise is decreased by connecting a capacitor inbetween the pin and GND pin. Capacitors with a capacity of 0.0047μF or more are required inbetween REGNOISE pin and GND pin for stable operation.
The time required for an initial rise varies depending on the capacity of the capacitors connected to REGNOISE pin.

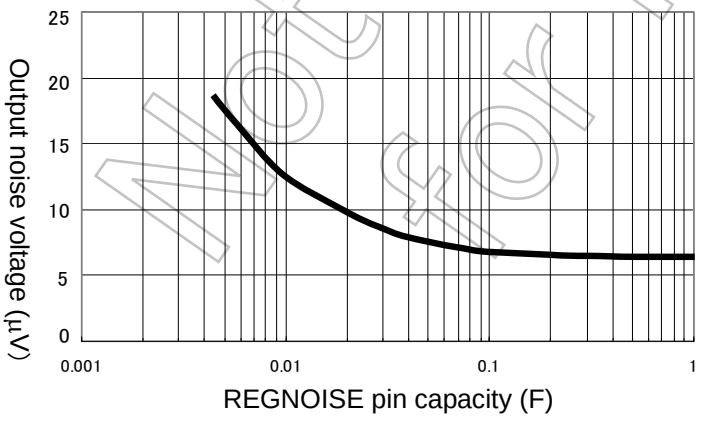


Fig. 23-2 REGNOISE-V_N characteristic

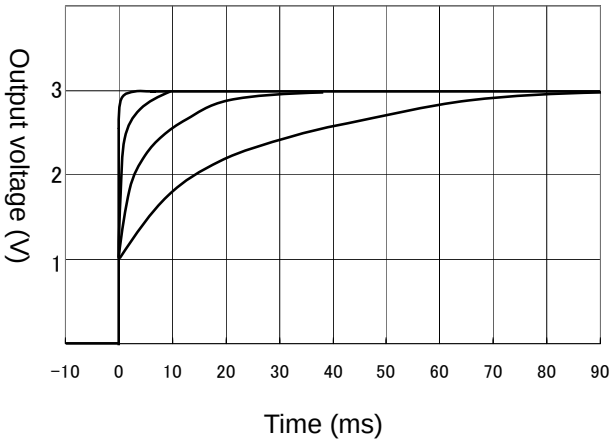


Fig. 23-3 Rising edge waveform of output

23.6 10-bit ADC Electrical Characteristics

[1] BGA

DVCC15=CVCC15=1.35V~1.65V, DVCC3=AVCC3=VREFH=2.7V~3.6V,
AVSS=DVSS, Ta=-20~85°C
AVCC3 load capacitance ≥3.3μF, VREFH load capacitance ≥3.3μF

Parameter		Symbol	Rating	Min	Typ	Max	Unit
Analog reference voltage (+)		VREFH		2.7	3.3	3.6	V
Analog reference voltage (–)		VREFL		AVSS	AVSS	AVSS	V
Analog input voltage		VAIN		VREFL		VREFH	V
Analog supply current	A/D conversion	IREF	DVSS = AVSS = VREFL		4.5	5.5	mA
	Non-A/D conversion		DVSS = AVSS = VREFL		±0.02	±5	μA
Supply current	A/D conversion	-	Non-IREF			3	mA
INL error		-	AIN resistance ≤600Ω AIN load capacitance ≤30pF Conversion time ≥1.15μs		±2	±3	LSB
DNL error					±1	±2	
Offset error					±2	±4	
Full-scale error					±2	±4	
INL error		-	AIN resistance ≤600Ω AIN load capacitance ≥0.1μF Conversion time ≥1.15μs		±2	±3	
DNL error					±1	±2	
Offset error					±2	±4	
Full-scale error					±2	±4	
INL error		-	AIN resistance ≤1.3kΩ AIN load capacitance ≥0.1μF Conversion time ≥1.15μs		±2	±3	
DNL error					±1	±2	
Offset error					±2	±4	
Full-scale error					±2	±4	
INL error		-	AIN resistance ≤10kΩ AIN load capacitance ≥0.1μF Conversion time ≥2.30μs		±2	±3	
DNL error					±1	±2	
Offset error					±2	±4	
Full-scale error					±2	±4	

(Note 1) 1LSB= (VREFH–VREFL) / 1024[V]

[2] LQFP

DVCC3=REGVIN=AVCC3=VREFH=3.0V~3.6V,
AVSS=DVSS, Ta=-20~85°C
AVCC3 load capacitance ≥3.3μF, VREFH load capacitance ≥3.3μF

Parameter		Symbol	Rating	Min	Typ	Max	Unit
Analog reference voltage (+)		VREFH		3.0	3.3	3.6	V
Analog reference voltage (–)		VREFL		AVSS	AVSS	AVSS	V
Analog input voltage		VAIN		VREFL		VREFH	V
Analog supply current	A/D conversion	IREF	DVSS = AVSS = VREFL		4.5	5.5	mA
	Non-A/D conversion		DVSS = AVSS = VREFL		±0.02	±5	μA
Supply current	A/D conversion	-	Non-IREF			3	mA
INL error		-	AIN resistance ≤600Ω AIN load capacitance ≤30pF Conversion time ≥1.22μs		±2	±3	LSB
DNL error					±1	±2	
Offset error					±2	±4	
Full-scale error					±2	±4	
INL error		-	AIN resistance ≤600Ω AIN load capacitance ≥0.1μF Conversion time ≥1.22μs		±2	±3	
DNL error					±1	±2	
Offset error					±2	±4	
Full-scale error					±2	±4	
INL error		-	AIN resistance ≤1.3kΩ AIN load capacitance ≥0.1μF Conversion time ≥1.22μs		±2	±3	
DNL error					±1	±2	
Offset error					±2	±4	
Full-scale error					±2	±4	
INL error		-	AIN resistance ≤10kΩ AIN load capacitance ≥0.1μF Conversion time ≥2.44μs		±2	±3	
DNL error					±1	±2	
Offset error					±2	±4	
Full-scale error					±2	±4	

(Note 1) 1LSB= (VREFH–VREFL) / 1024[V]

23.7 AC Electrical Characteristics

23.7.1 Multiplex Bus mode

[1] BGA (fsys=40MHz)

DVCC15=CVCC15=1.35V~1.65V, DVCC3=AVCC3=2.7V~3.6V,Ta =-20~85°C

[2] LQFP (fsys=54MHz)

DVCC3=REGVIN=AVCC3= 3.0V~3.6V, Ta =-20~85°C

ALE = 1 clock cycle, 2 programmed wait state

No.	Parameter	Symbol	Equation		40 MHz (fsys)(Note)		54MHz (fsys)(Note)		Unit
			Min	Max	Min	Max			
1	System clock period (x)	t_{SYS}	x		25		18.5		ns
2	A0-A15 valid to ALE fall	t_{AL}	x –11		14.0		7.5		ns
3	A0-A15 hold after ALE falling	t_{LA}	x – 8		17.0		10.5		ns
4	ALE pulse width high	t_{LL}	x – 6		19.0		12.5		ns
5	$\overline{RD}$ / $\overline{WR}$ /$\overline{HWR}$ low after ALE falling	t_{LC}	x – 8		17.0		10.5		ns
6	ALE high after $\overline{RD}$ / $\overline{WR}$ /$\overline{HWR}$ rising	t_{CL}	x – 8		17.0		10.5		ns
7	A0-15 valid to $\overline{RD}$ / $\overline{WR}$ /$\overline{HWR}$ fall	t_{ACL}	2x – 11		39.0		26		ns
8	A16-23 valid to $\overline{RD}$ / $\overline{WR}$ /$\overline{HWR}$ fall	t_{ACH}	2x – 11		39.0		26		ns
9	A16 – 23 hold to $\overline{RD}$ / $\overline{WR}$ /$\overline{HWR}$ fall	t_{CAR}	x – 11		14.0		7.5		ns
10	A0 – 15 valid to D0 – 15 input	t_{ADL}		x (2 + TW + ALE) – 43		82.0		49.5	ns
11	A16 – 23 valid to D0 – 15 input	t_{ADH}		x (2 + TW + ALE) – 43		82.0		49.5	ns
12	D0 – 15 input after $\overline{RD}$ falling	t_{RD}		x (1 + TW) – 40		35.0		15.5	ns
13	$\overline{RD}$ width low	t_{RR}	x (1 + TW) – 6		69		49.5		ns
14	D0 – 15 hold after $\overline{RD}$ rising	t_{HR}	0		0		0		ns
15	A0 – 15 input after $\overline{RD}$ rising	t_{RAE}	x – 6		19.0		12.5		ns
16	$\overline{WR}$ /$\overline{HWR}$ width low	t_{WW}	x (1 + TW) – 6		69.0		49.5		ns
17	D0-15 valid to $\overline{WR}$ /$\overline{HWR}$ rise	t_{DW}	x (1 + TW) – 11		64.0		44.5		ns
18	D0 – 15 hold after $\overline{WR}$ /$\overline{HWR}$ rising	t_{WD}	x – 11		14.0		7.5		ns
19	A16 – 23 valid to $\overline{WAIT}$ input	t_{AWH}		x + x (ALE) + x (TW-1) – 32		43.0		23.5	ns
20	A0 – 15 valid to $\overline{WAIT}$ input	t_{AWL}		x + x (ALE) + x (TW-1) – 32		43.0		23.5	ns
21	$\overline{WAIT}$ hold after $\overline{RD}$ / $\overline{WR}$ /$\overline{HWR}$	t_{CW}	x (TW – 3) – 9	x (TW – 1) – 29	20.0	46.0	13.5	26.5	ns

(Note)

Internal 2 wait insertion , ALE “1” Clock, @40MHz/54MHz

TW = (W+ 2N)

W : Number of Auto wait insertion , 2N : Number of external wait insertion

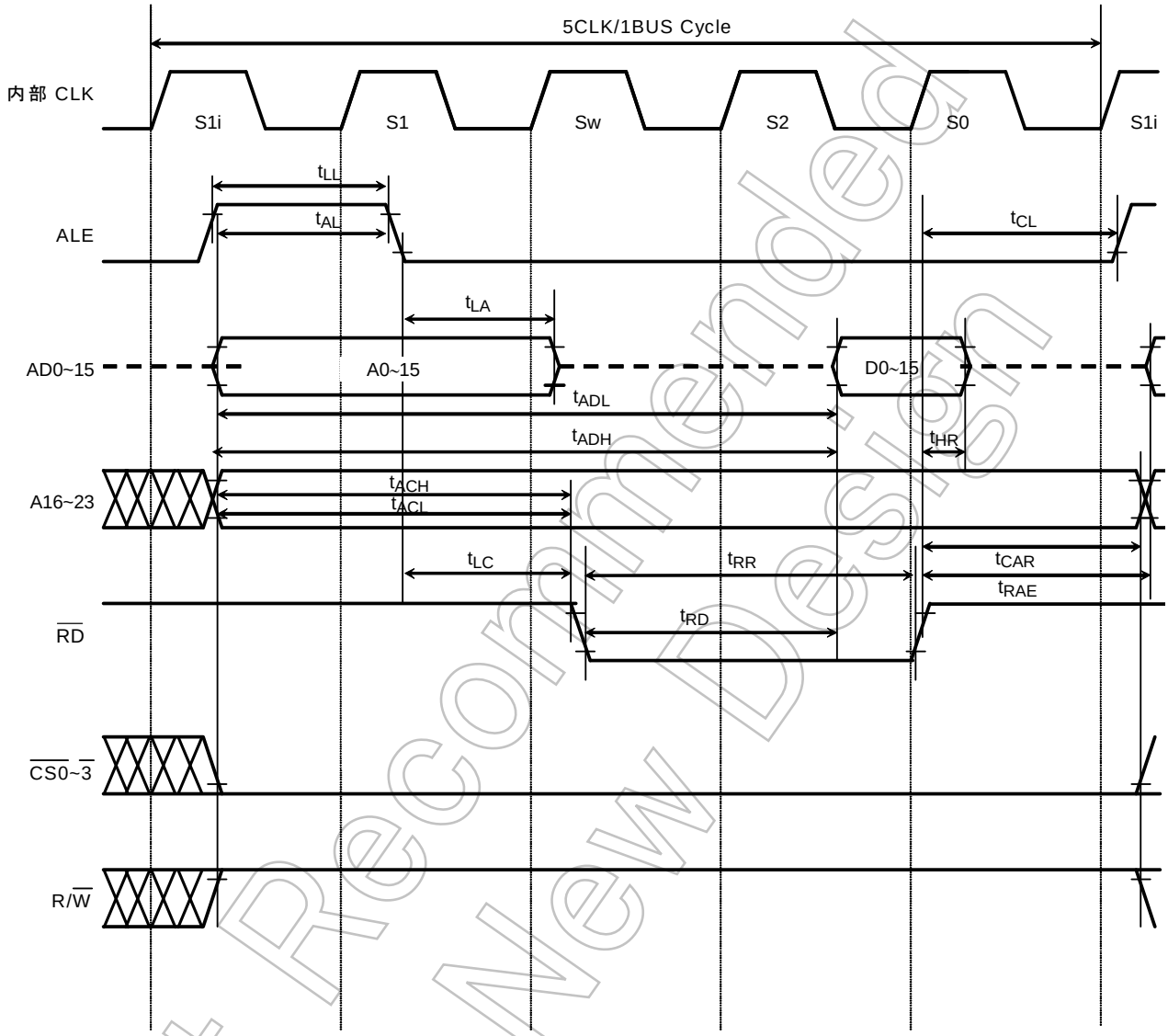
TW = 2 + 2*1 = 4

AC measurement condition

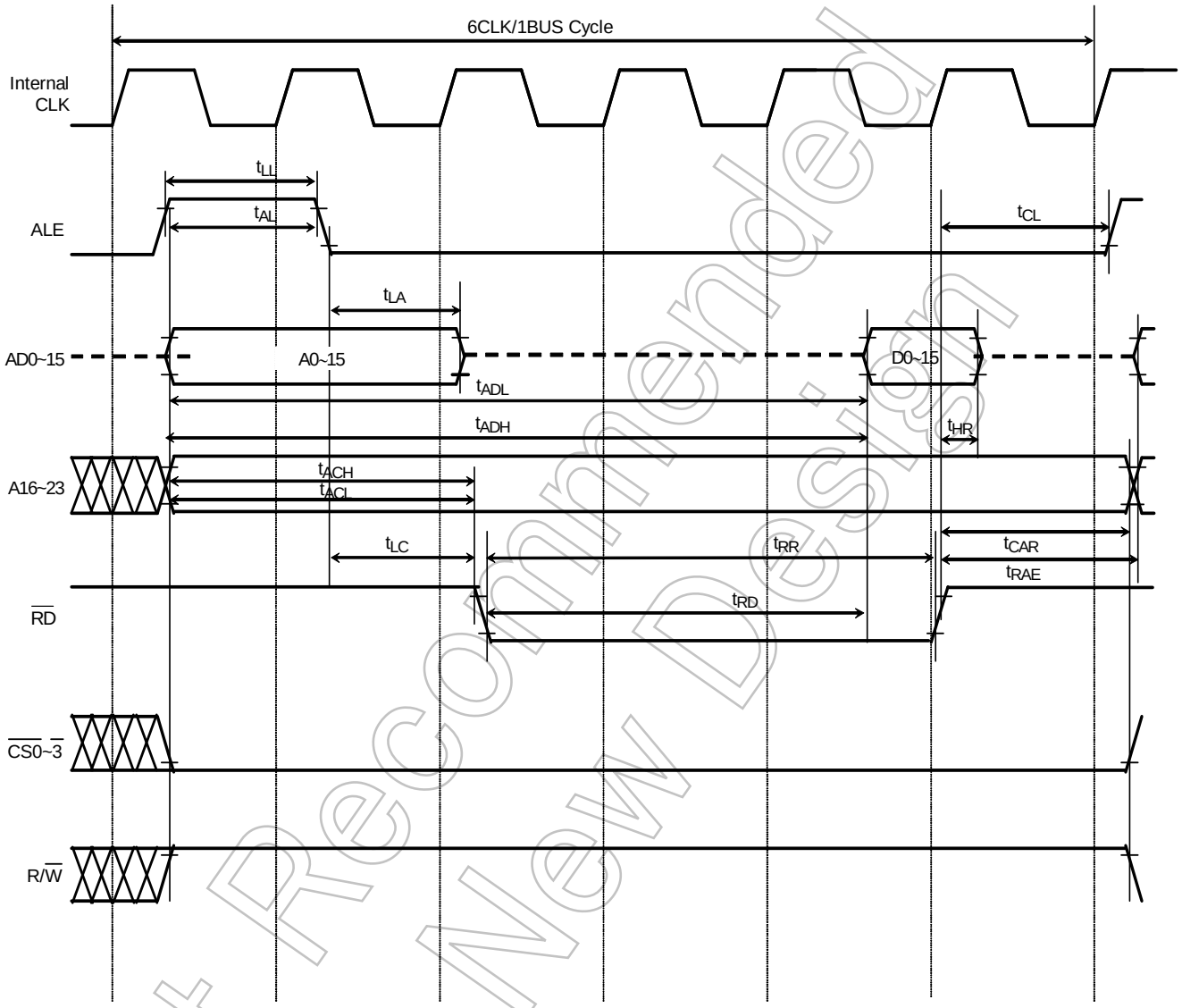
/Output levels: High 0.8DVCC3 V/Low 0.2DVCC3V, CL=30 pF

/Input levels: High 0.7DVCC3 V/Low 0.3DVCC3V

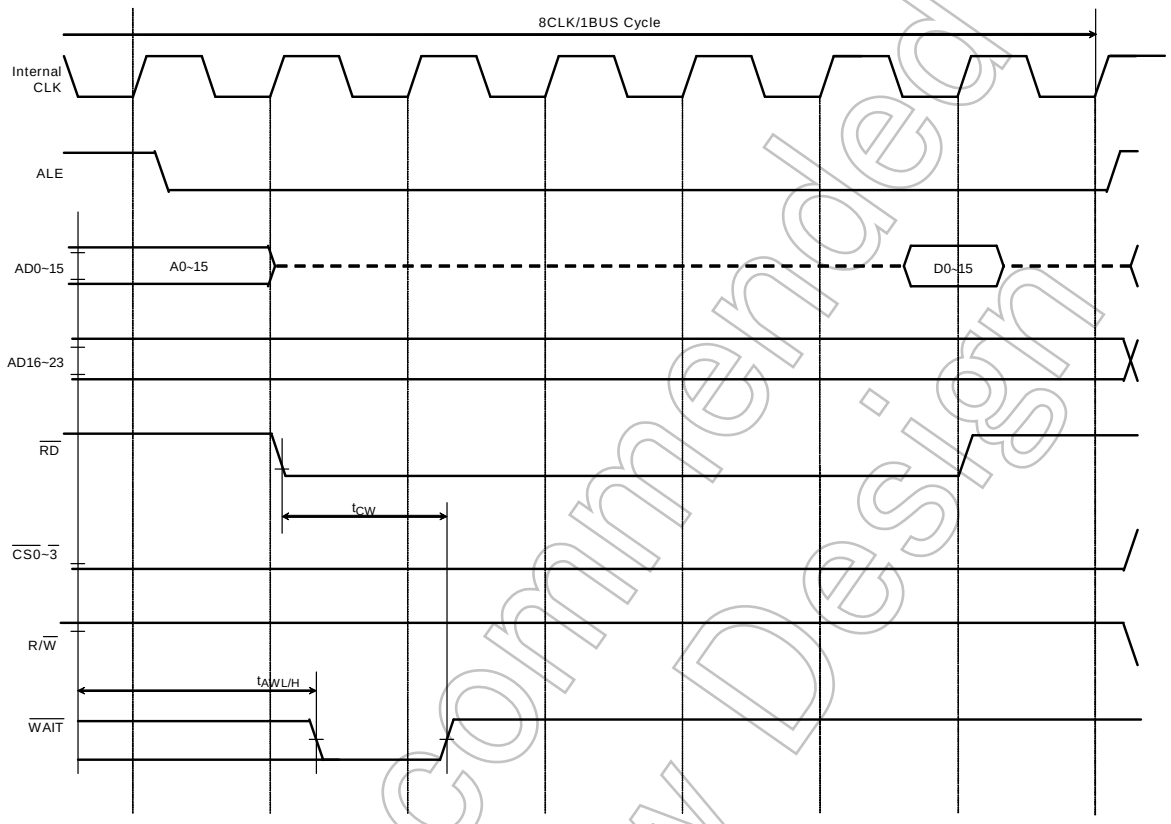
(1) Read cycle timing, ALE width = 1 clock cycle, 1 programmed wait state



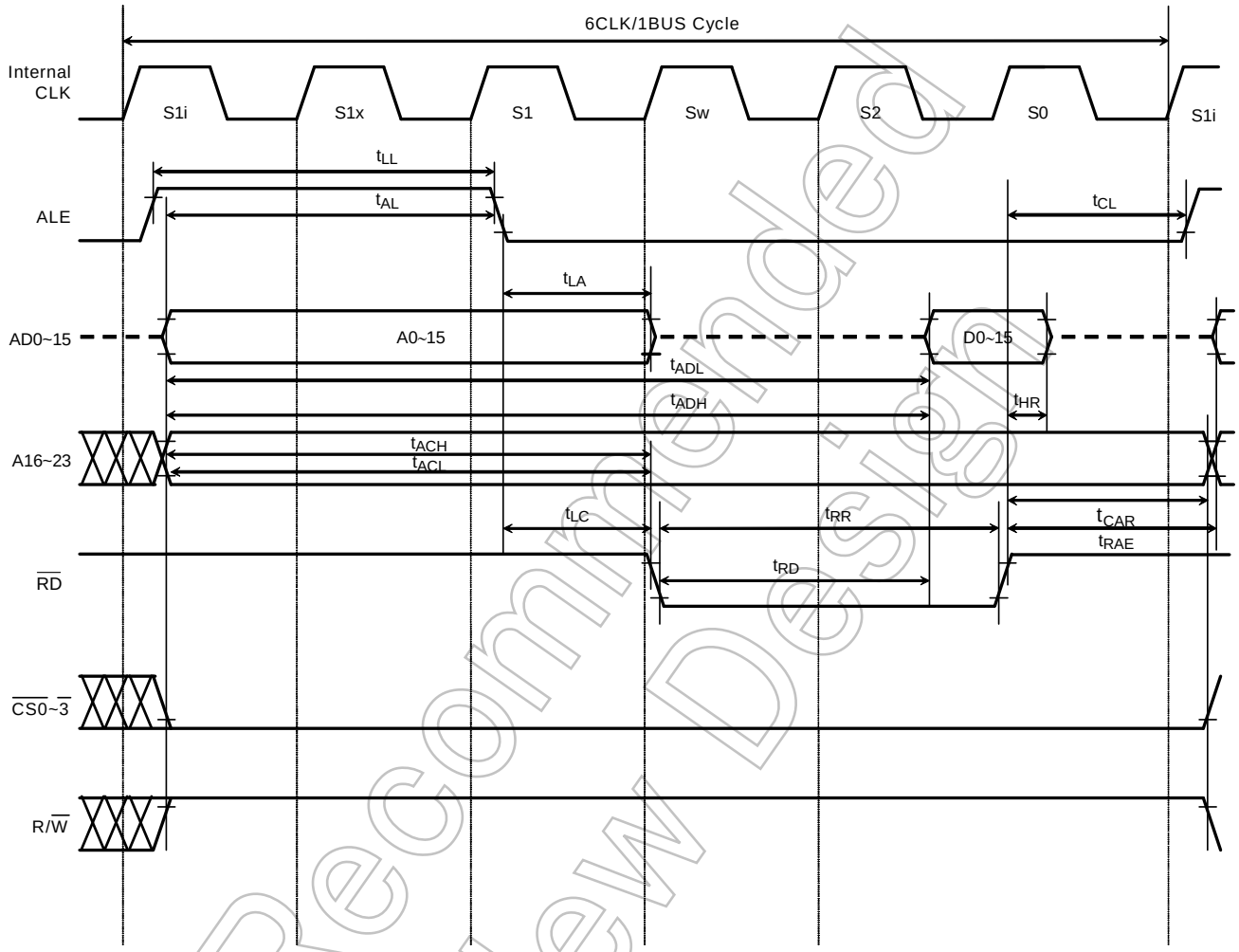
(2) Read cycle timing, ALE width = 1 clock cycle, 2 programmed wait state



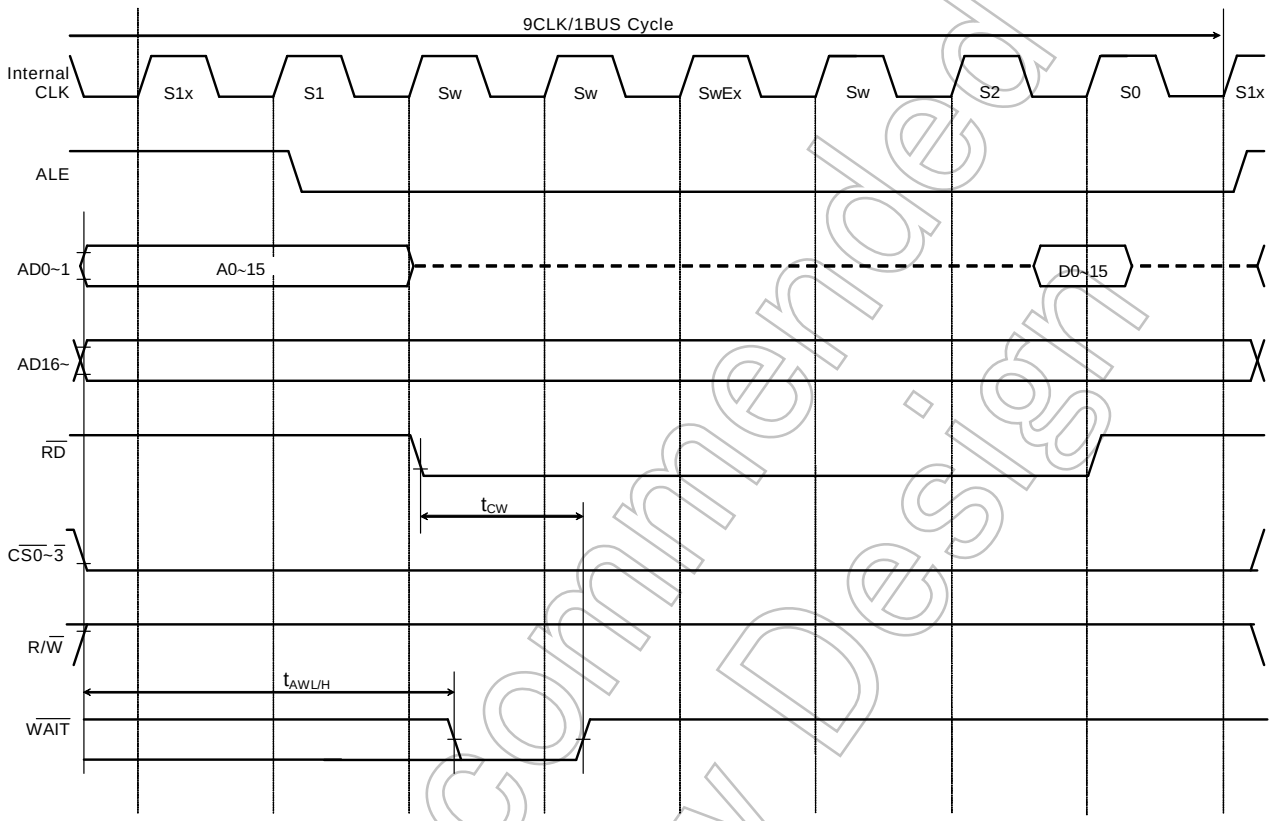
(3) Read cycle timing, ALE width = 1 clock cycle, 4 programmed wait state
(externally generated wait states with 2+2N, N = 1)



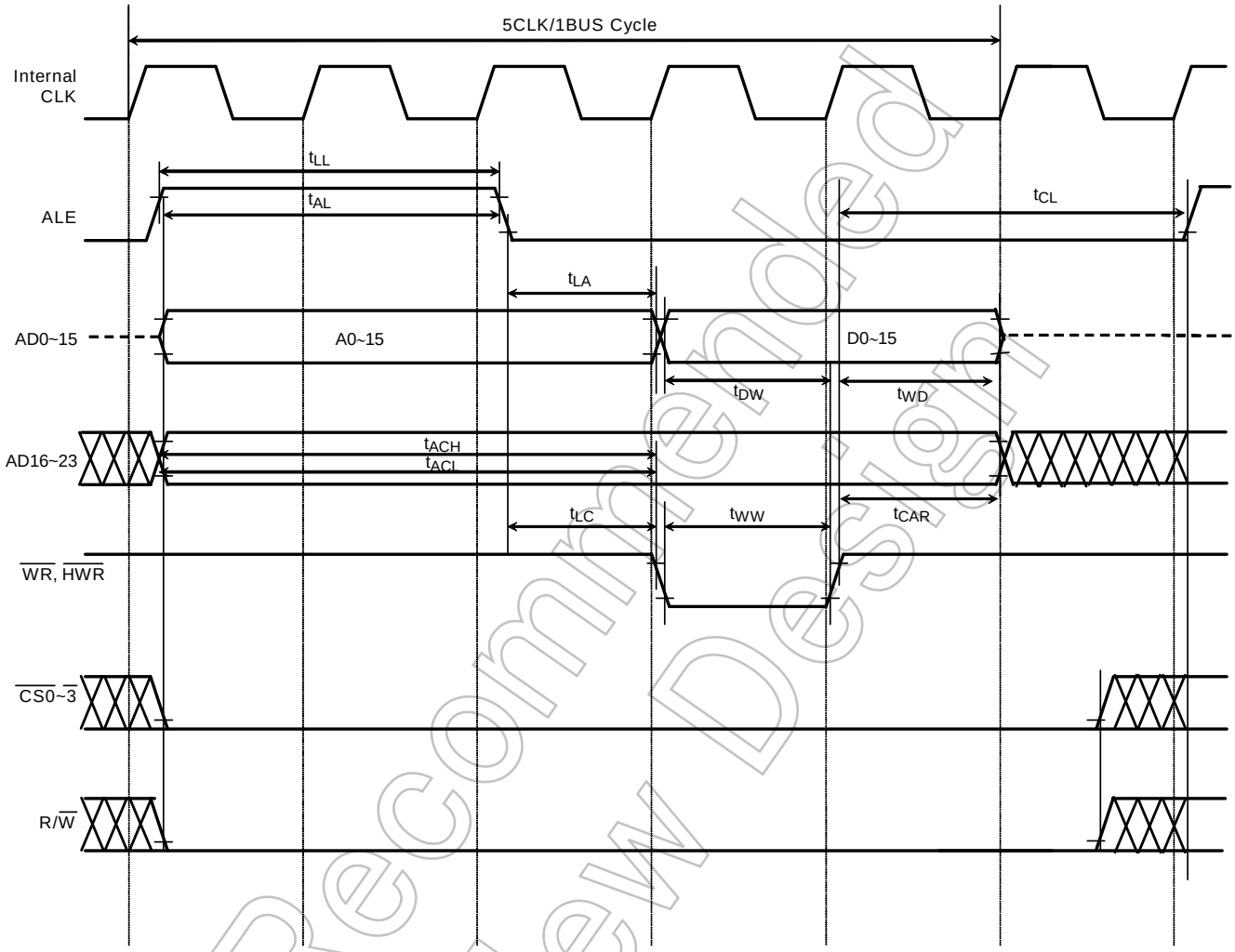
(4) Read cycle timing, ALE width = 2 clock cycle, 1 programmed wait state



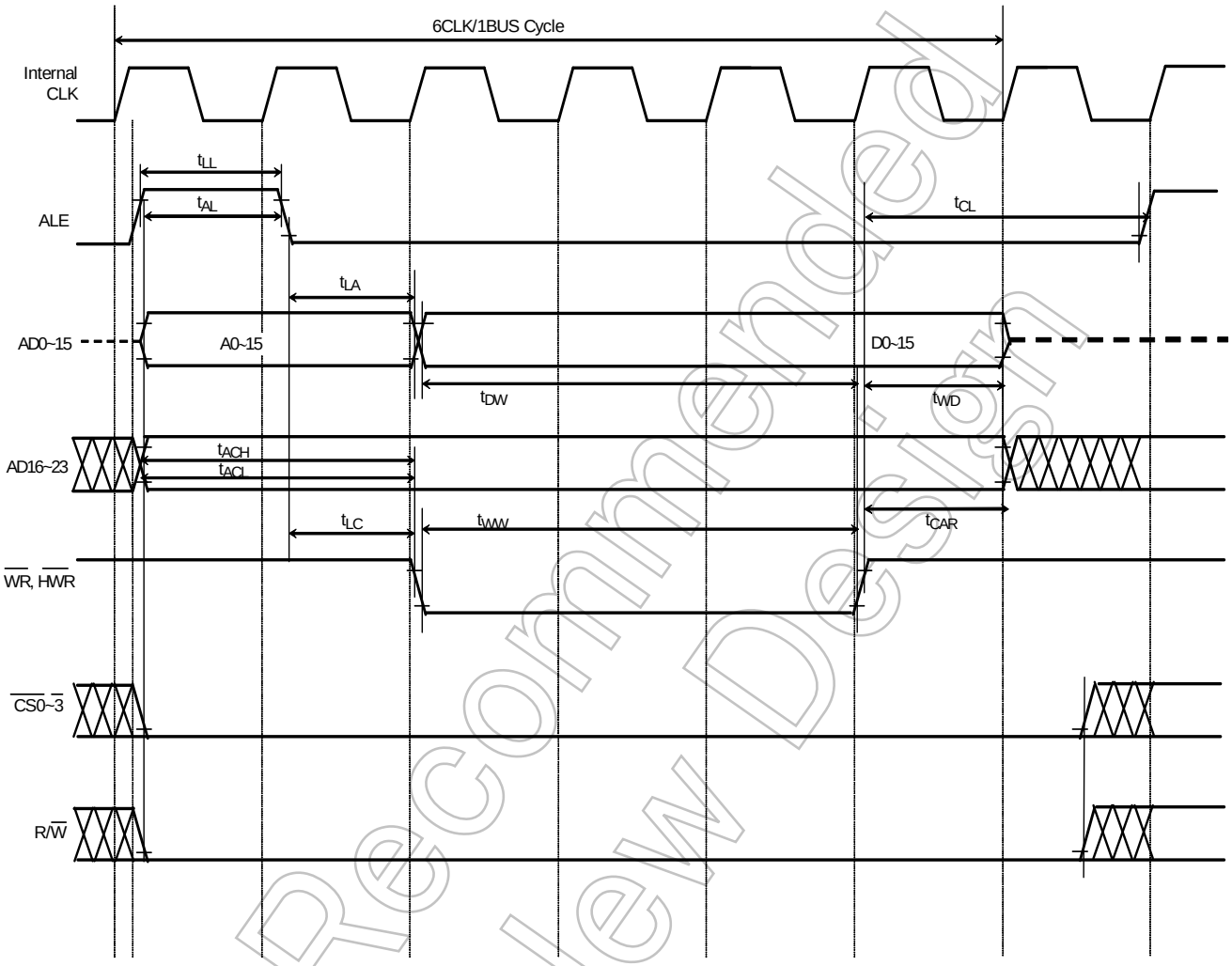
(5) Read cycle timing, ALE width = 2 clock cycle, 4 programmed wait state
(externally generated wait states with 2+2N, N = 1)



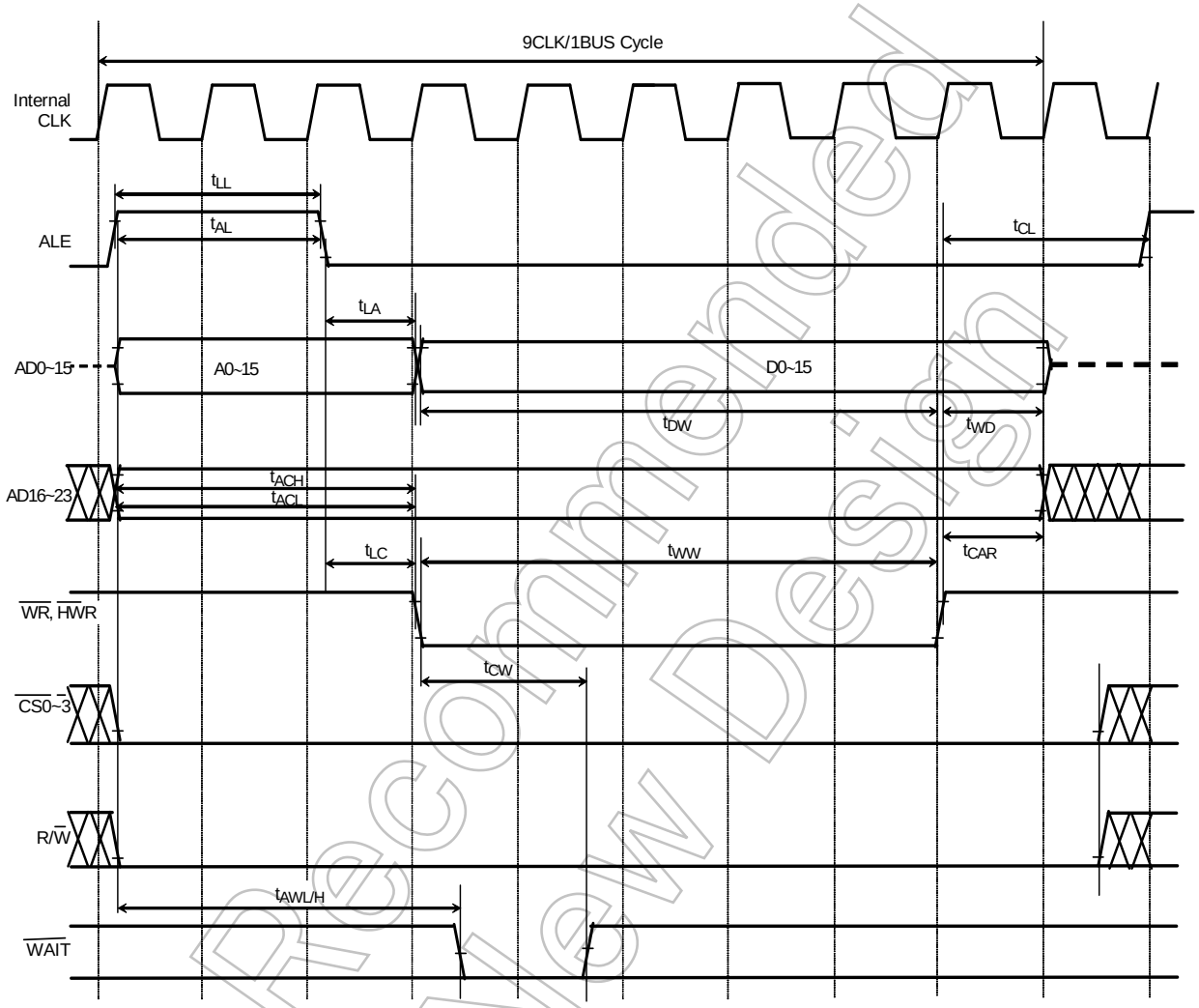
(6) Write cycle timing, ALE width = 2 clock cycles, zero wait state



(7) Write cycle timing, ALE width = 1 clock cycles, 2 wait state



(8) Write cycle timing, ALE width = 2 clock cycles, 4 wait state
(externally generated wait states with 2+2N, N = 1)



23.7.2 Separate Bus mode

[1] BGA (fsys=40MHz)

DVCC15=CVCC15=1.35V~1.65V, DVCC3=AVCC3=2.7V~3.6V,Ta =-20~85°C

[2] LQFP (fsys=54MHz)

DVCC3=REGVIN=AVCC3= 3.0V~3.6V, Ta =-20~85°C

SYSCR3<ALESEL> = “0”, 2 programmed wait state

No.	Parameter	Symbol	Equation		40 MHz (fsys)(Note)		54 MHz (fsys)(Note)		Unit
			Min	Max	Min	Max	Min	Max	
1	System clock period (x)	t_{SYS}	x		25		18.5		ns
2	A0-23 valid to $\overline{RD}$ / $\overline{WR}$ /$\overline{HWR}$ fall	t_{AC}	x(1+ ALE) –11		39.0		26		ns
3	A0 – 23 hold after $\overline{RD}$ / $\overline{WR}$ /$\overline{HWR}$ rising	t_{CAR}	x – 11		14.0		7.5		ns
4	A0 – 23 valid to D0 – 15 input	t_{AD}		x (2 + TW + ALE) – 43		82.0		49.5	ns
5	D0 – 15 input after $\overline{RD}$ falling	t_{RD}		x (1 + TW) – 40		35.0		15.5	ns
6	$\overline{RD}$ width low	t_{RR}	x (1 + TW) –6		69.0		49.5		ns
7	D0 – 15 hold after $\overline{RD}$ rising	t_{HR}	0		0		0		ns
8	A0 – 23 output after $\overline{RD}$ rising	t_{RAE}	x – 6		19.0		12.5		ns
9	$\overline{WR}$ /$\overline{HWR}$ width low	t_{WW}	x (1 + TW) –6		69.0		49.5		ns
10	D0-15 valid after $\overline{WR}$ /$\overline{HWR}$ falling	t_{DO}		9.7		9.7		9.7	ns
11	D0-15 valid to $\overline{WR}$ /$\overline{HWR}$ rise	t_{DW}	x (1 + TW) – 11		64.0		44.5		ns
12	D0 – 15 hold to $\overline{WR}$ /$\overline{HWR}$ rise	t_{WD}	x – 11		14.0		7.5		ns
13	A0 – 23 valid to $\overline{WAIT}$ input	t_{AW}		x + x (ALE) + x(TW – 1) – 32		43.0		23.5	ns
14	$\overline{WAIT}$ hold after $\overline{RD}$ / $\overline{WR}$ /$\overline{HWR}$	t_{CW}	x(TW – 3) – 5	x(TW – 1) – 29	20.0	46.0	13.5	26.5	ns

(Note)

Internal 2 wait insertion， ALE “1” Clock, @40MHz/54MHz

TW = (W+ 2N)

W : Number of Auto wait insertion , 2N : Number of external wait insertion

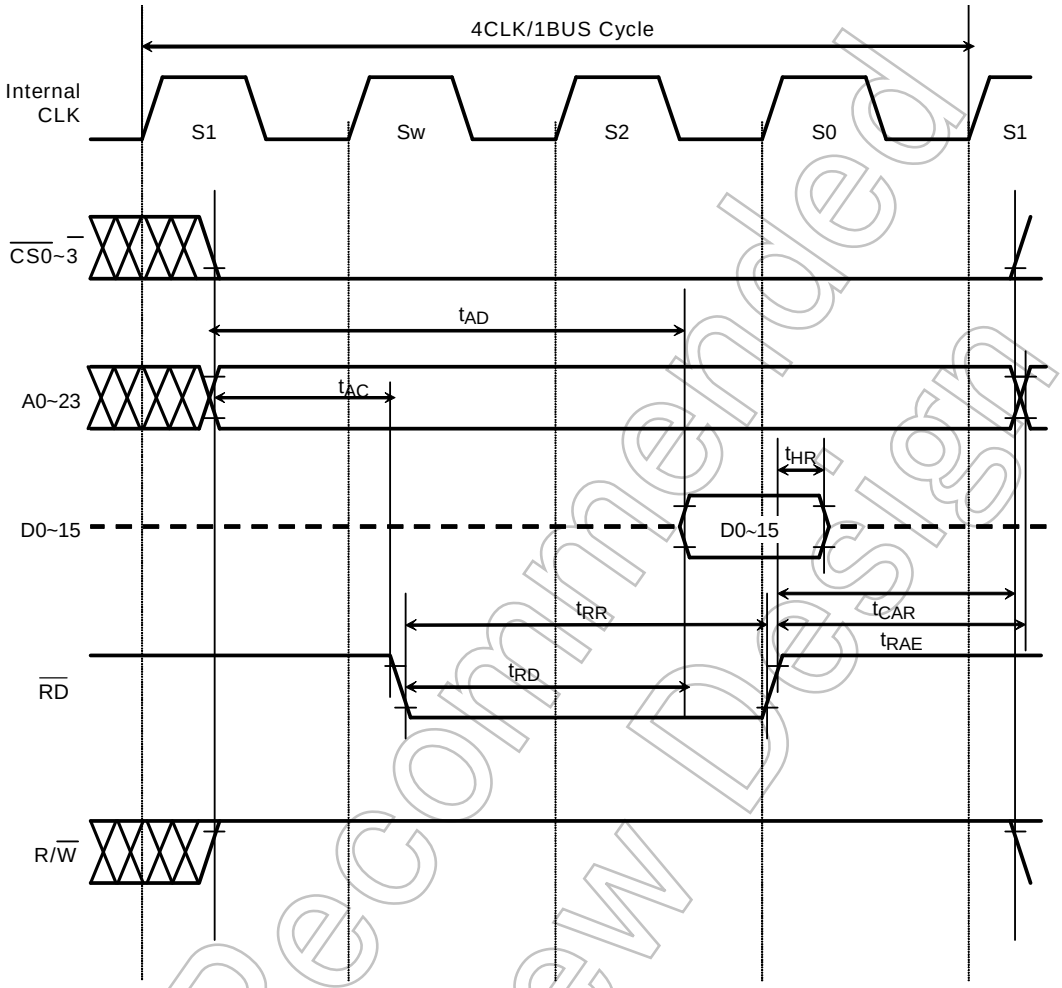
TW = 2 + 2*1 = 4

AC measurement conditions:

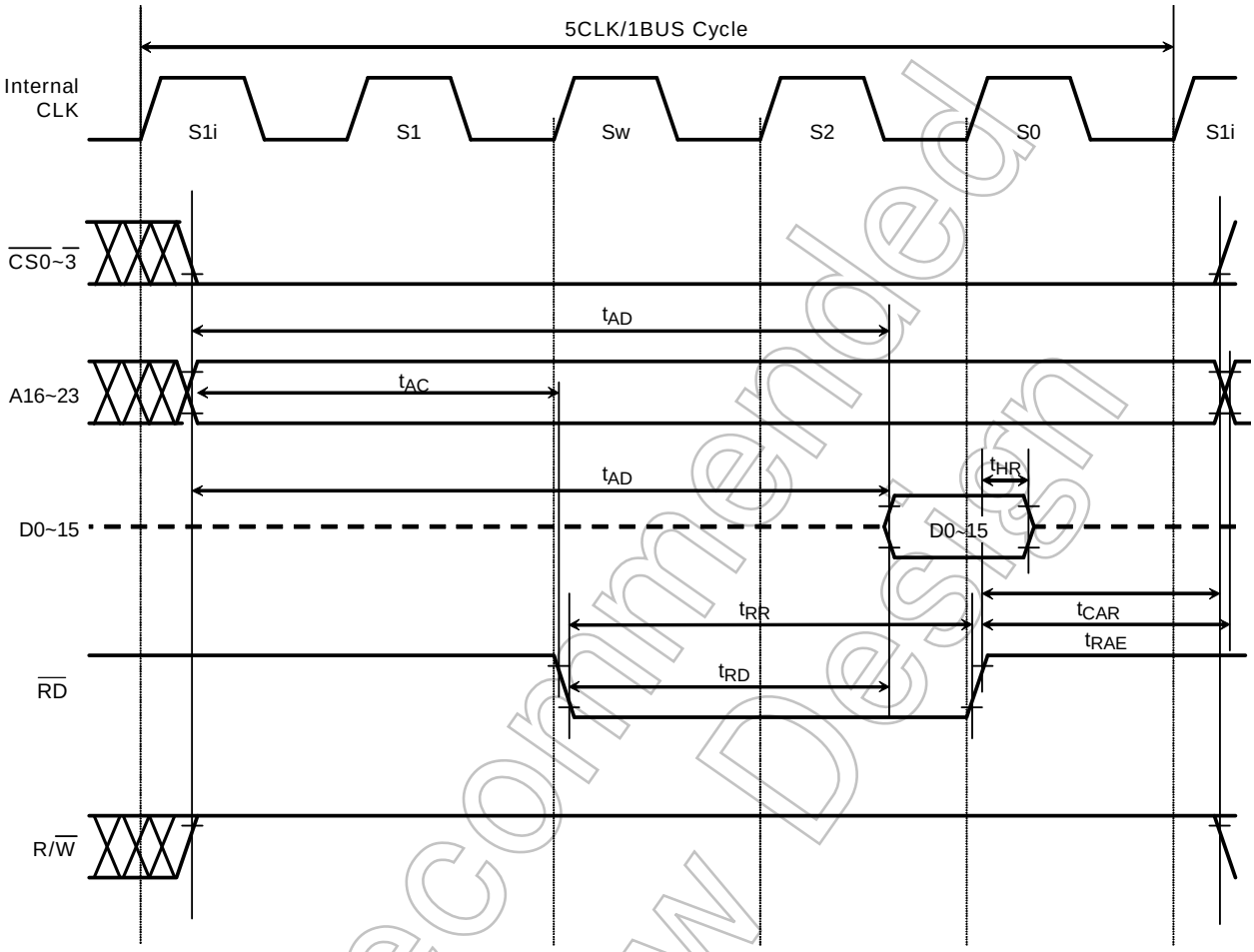
Output levels: High = 0.8DVCC3 V/Low 0.2DVCC3 V, CL = 30 pF

Input levels: High = 0.7DVCC3 V/Low 0.3DVCC3 V

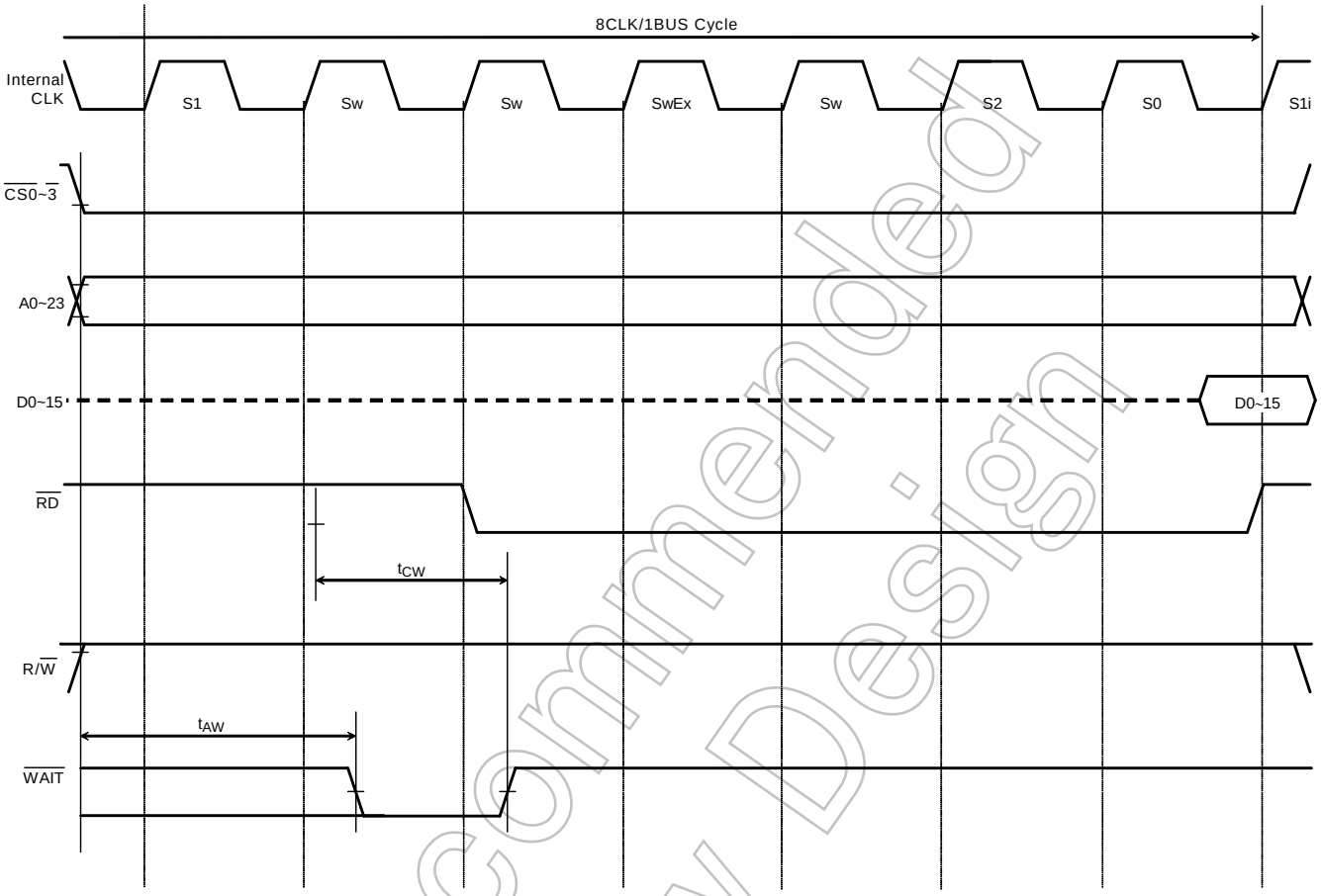
(1) Read cycle timing (SYSCR3<ALESEL> = 0, 1 programmed wait state)



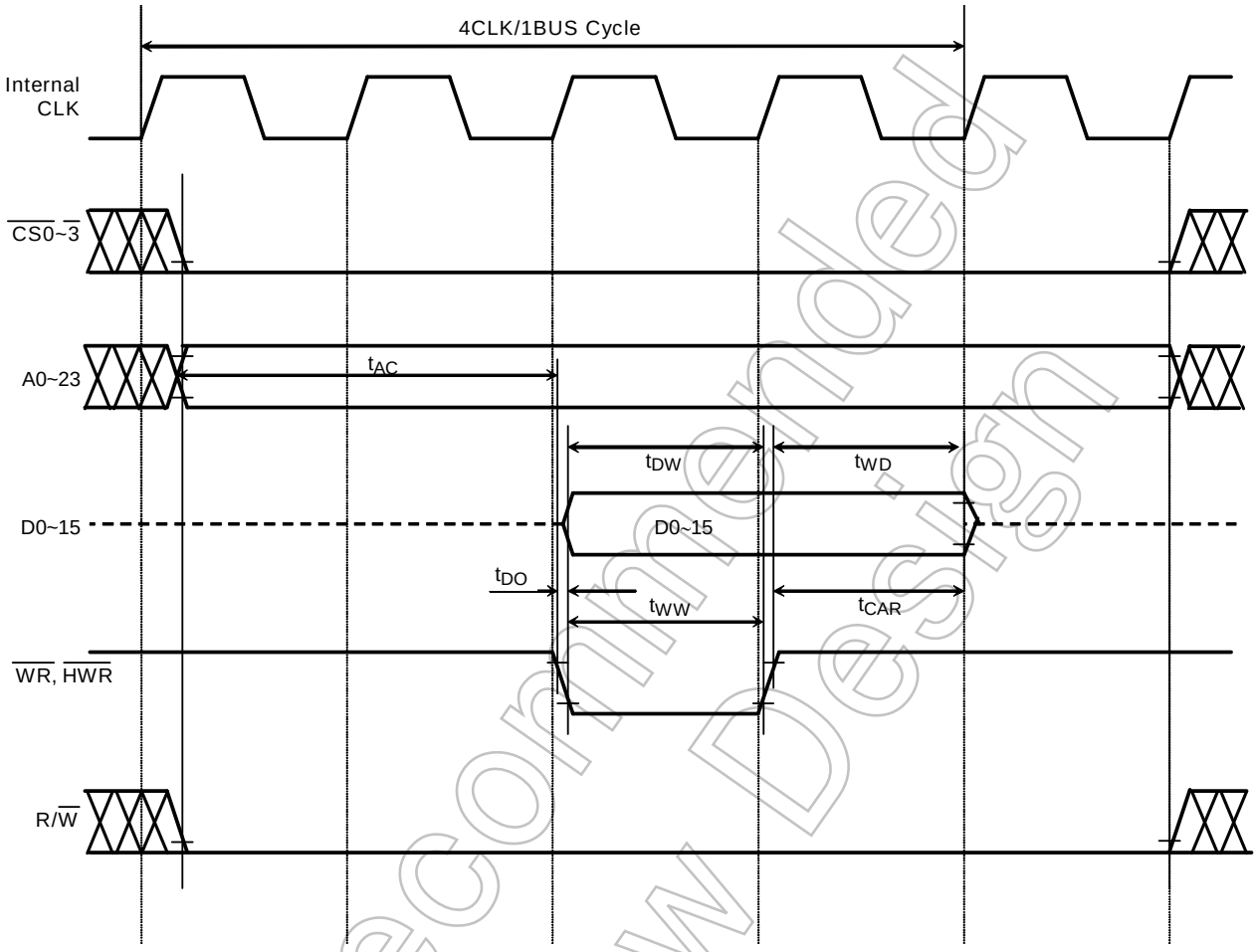
(2) Read cycle timing (SYSCR3<ALESEL> = “1”, 1 programmed wait state)



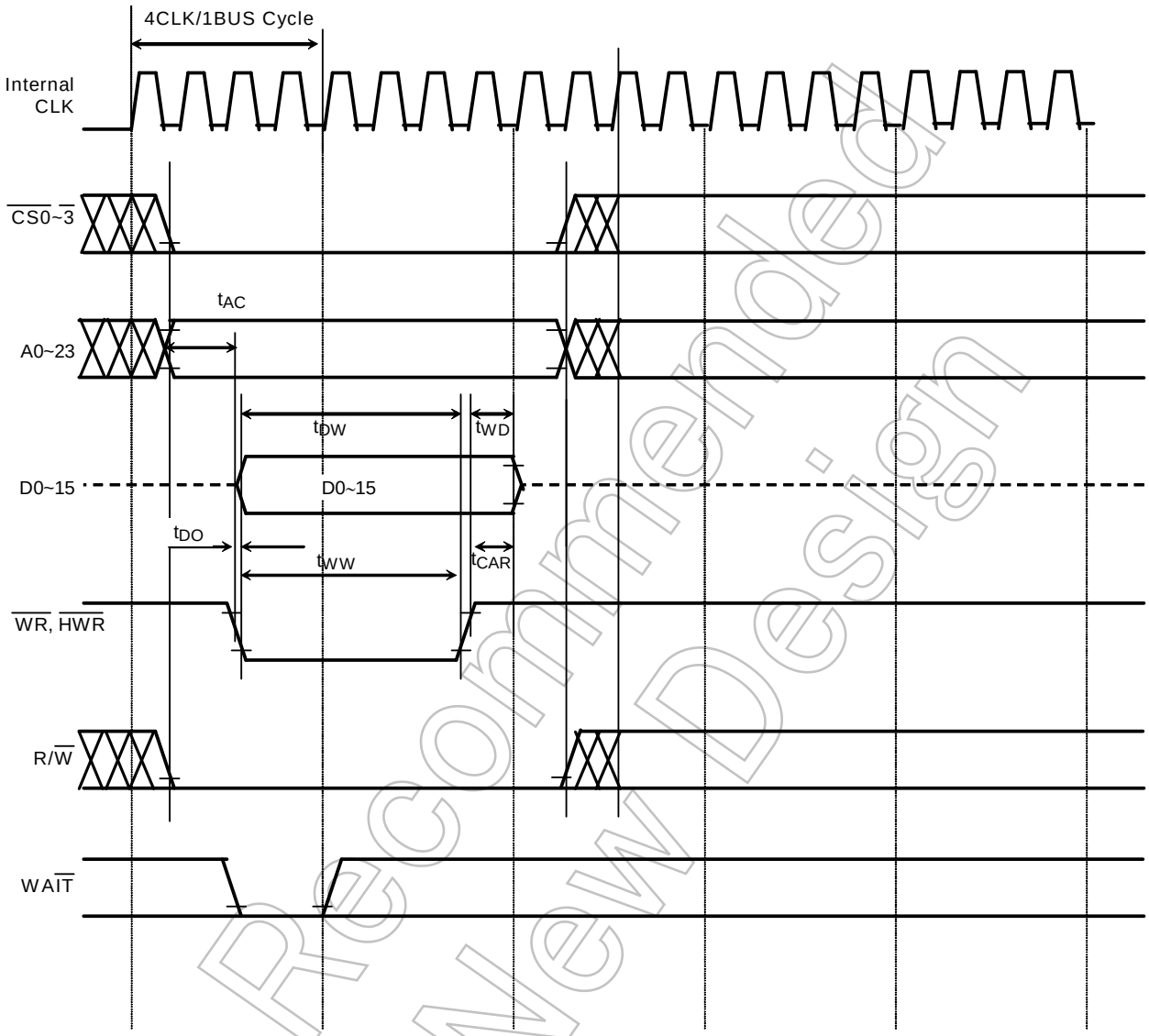
(3) Read cycle timing SYSCR3<ALESEL> = “1”, 4 externally generated wait states with 2+2N, N = 1)



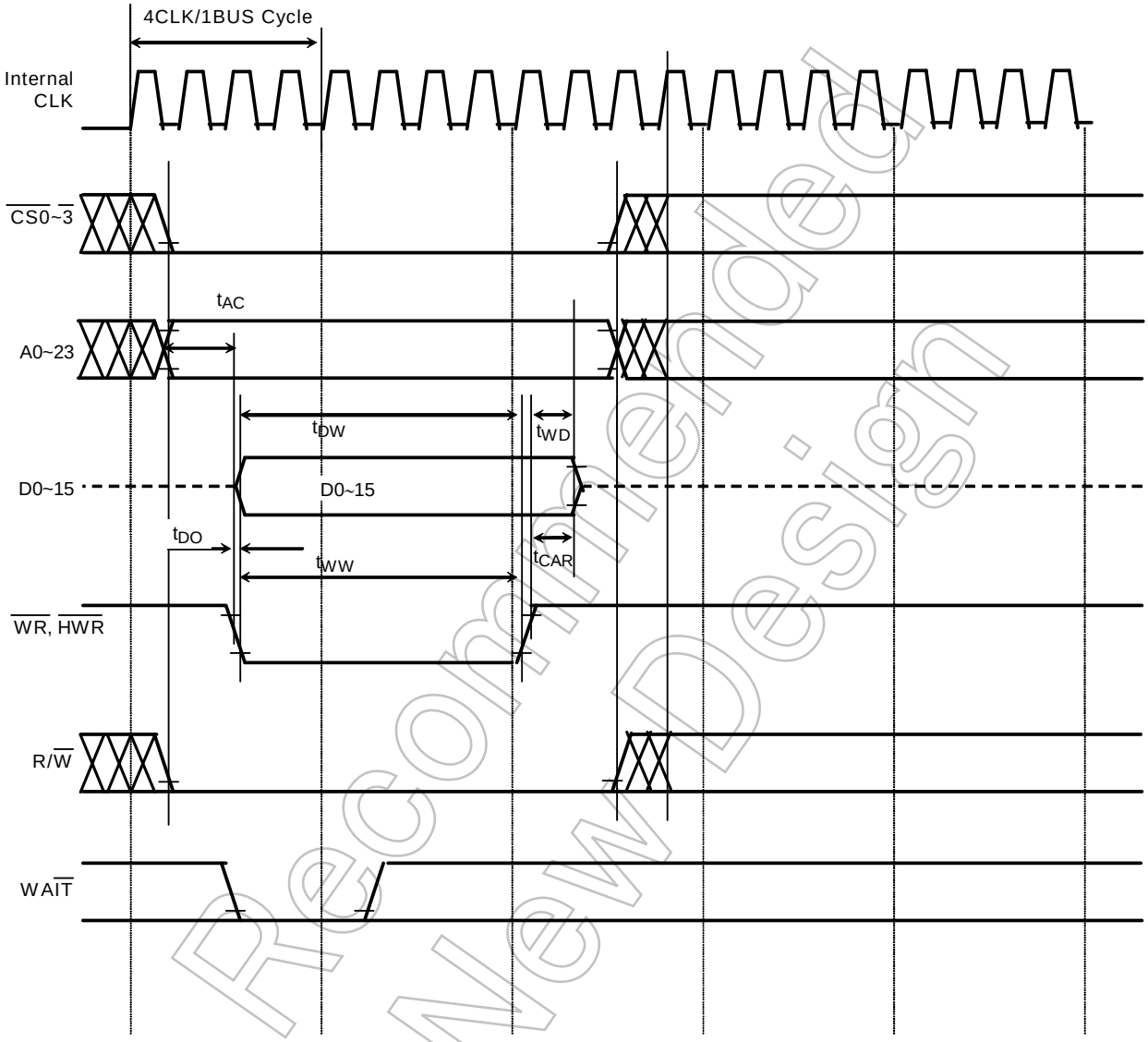
(4) Write cycle timing (SYSCR3<ALESEL> = “1”, zero wait sate)



(5) Write cycle timing (SYSCR3<ALESEL> =”1”, 2 programmed wait states +2N (N=1))



(6) Write cycle timing (SYSCR3<ALESEL> = “1”, 3 programmed wait states +2N (N=1)

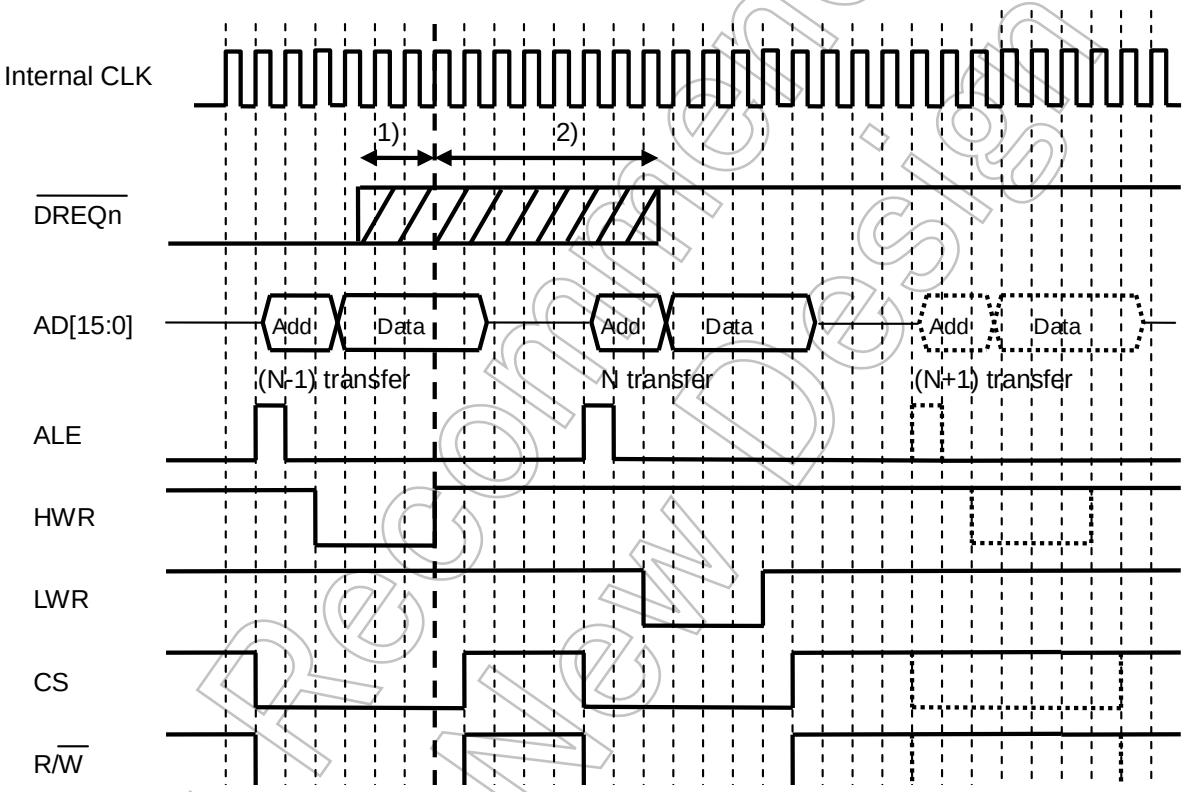


23.8 Transfer with DMA Request

The following shows an example of a transfer between the on-chip RAM and an external device in multiplex bus mode.

- 16-bit data bus width, non-recovery time
- Level data transfer mode
- Transfer size (TrSiz) of 16 bits, device port size (DPS) of 16 bits
- Source/destination: internal RAM/external device

The following shows transfer operation timing of the internal RAM to an external bus during write operation (memory-to-memory transfer).



1) Indicates the condition under which Nth transfer is performed successfully.
2) Indicates the condition under which (N + 1)th transfer is not performed.

[1] BGA (fsys=40MHz)
DVCC15=CVCC15=1.35V~1.65V, DVCC3=AVCC3=2.7V~3.6V,Ta =-20~85°C

[2] LQFP (fsys=54MHz)
DVCC3=REGVIN=AVCC3= 3.0V~3.6V, Ta =-20~85°C

Parameter	Symbol	Equation		40 MHz (fsys)		54 MHz (fsys)		Unit
		1) Min	2) Max	Min	Max	Min	Max	
$\overline{RD}$ assert to $\overline{DREQn}$ negate (external device to internal RAM transfer)	tDREQ_r	(W+1)x	(2W+ALE+8)x −51	50	224	36	147	ns
$\overline{WR}$ / $\overline{HWR}$ rise to $\overline{DREQn}$ negate (Internal RAM to external device transfer)	tDREQ_w	−(W+2)x	(5+WAIT)x −51.8	−75	98.2	−54	56.2	ns

23.9 Serial Channel Timing

(1) I/O Interface mode (BGA DVCC3=2.7V~3.6V, LQFP DVCC3=3.0V~3.6V)

In the table below, the letter x represents the fsys cycle period, which varies depending on the programming of the clock gear function.

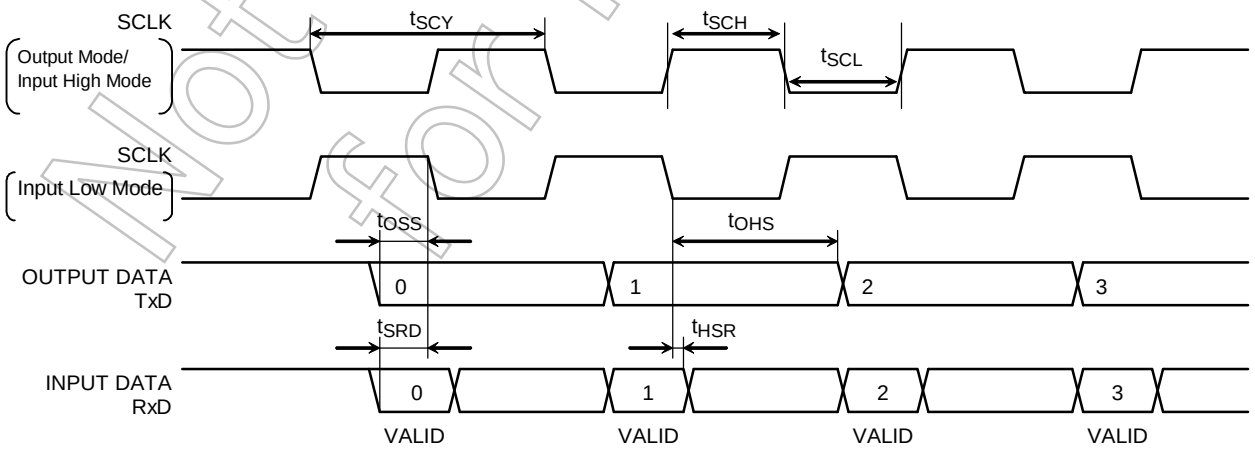
1) SCLK input mode (SIO0~SIO2)

Parameter	Symbol	Equation		40 MHz		54 MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK period	t_{SCY}	12x		300		222		ns
SCLK Clock High width (input)	tsc_H	6x		150		111		ns
SCLK Clock Low width (input)	tsc_L	6x		150		111		ns
TxD to SCLK rise or fall*	t_{OSS}	2x−30		20		6		ns
TxD hold or fall after SCLK rising*	t_{OHS}	8x−15		185		129		ns
RxD valid to SCLK rise or fall*	t_{SRD}	30		30		30		ns
RxD hold or fall after SCLK rising*	t_{HSR}	2x+30		80		66		ns

*: SCLK rise or fall: Measured relative to the programmed active edge of SCLK.

2) SCLK output mode (SIO0~SIO2)

Parameter	Symbol	Equation		40 MHz		54MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK period (programmable)	t_{SCY}	8x		200		144		ns
TxD to SCLK rise	t_{OSS}	4x−10		90		62		ns
TxD hold after SCLK rising	t_{OHS}	4x−10		90		62		ns
RxD valid to SCLK rise	t_{SRD}	45		45		45		ns
RxD hold after SCLK rising	t_{HSR}	0		0		0		ns



23.10 High Speed Serial Channel Timing

(1) I/O Interface mode (BGA DVCC3=2.7V~3.6V, LQFP DVCC3=3.0V~3.6V)

In the table below, the letter x represents the fsys cycle period, which varies depending on the programming of the clock gear function.

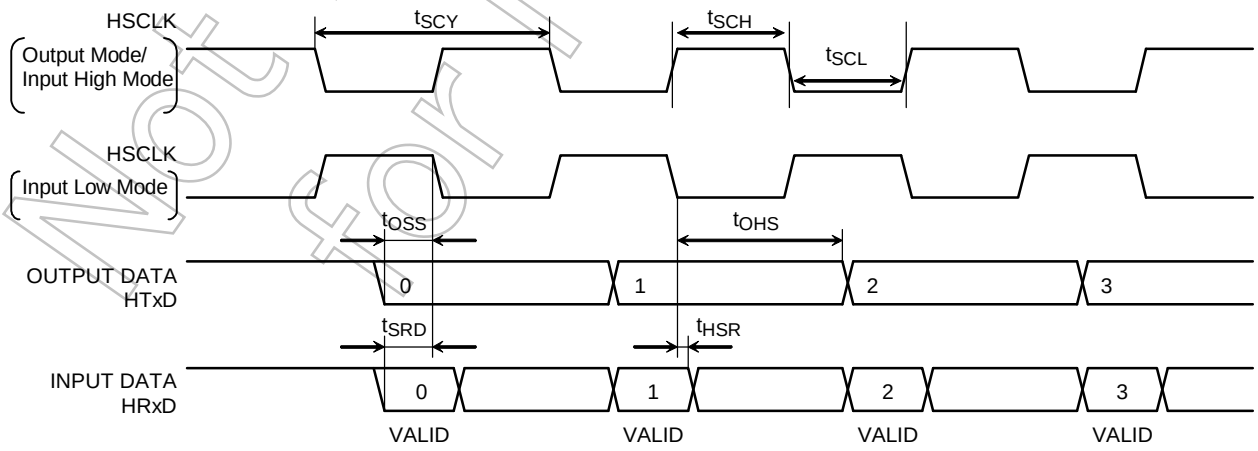
1) HSCLK Input Mode (HSIO0)

Parameter	Symbol	Equation		40 MHz		54MHz		Unit
		Min	Max	Min	Max	Min	Max	
HSCLK period	t_{SCY}	12(x/2)		150		108		ns
HSCLK Clock High width (input)	tsc_H	3x		75		54		ns
HSCLK Clock Low width (input)	tsc_L	3x		75		54		ns
TxD to HSCLK rise or fall*	t_{OSS}	2(x/2)－30		-5		-12		ns
TxD hold or fall after HSCLK rising*	t_{OHS}	8(x/2)－15		85		57		ns
RxD valid to HSCLK rise or fall*	t_{SRD}	30		30		30		ns
RxD hold or fall after HSCLK rising*	t_{HSR}	2(x/2) + 30		55		48		ns

*: HSCLK rise or fall: Measured relative to the programmed active edge of HSCLK.

2) HSCLK Output Mode (HSIO0)

Parameter	Symbol	Equation		40 MHz		54MHz		Unit
		Min	Max	Min	Max	Min	Max	
HSCLK period (programmable)	t_{SCY}	8(x/2)		100		72		ns
TxD to HSCLK rise	t_{OSS}	4(x/2) －10		40		26		ns
TxD hold after HSCLK rising	t_{OHS}	4(x/2) －10		40		26		ns
RxD valid to HSCLK rise	t_{SRD}	45		45		45		ns
RxD hold after HSCLK rising	t_{HSR}	0		0		0		ns



23.11 SBI Timing

(1) I2C Mode

In the table below, the letters x and t represent the fsys periods and ϕT1 respectively.

n denotes the value of n programmed into the SCK (SCL output frequency select) field in the SBInCR.

Parameter	Symbol	Equation		Standard Mode		Fast Mode		Unit
		Min	Max	Min	Max	Min	Max	
SCL clock frequency	t_{SCL}	0		0	100	0	400	kHz
Hold time for START condition	$t_{HD:STA}$			4.0		0.6		μs
SCL low width (Input) (Note 1)	t_{LOW}			4.7		1.3		μs
SCL high width (Input) (Note 2)	t_{HIGH}			4.0		0.6		μs
Setup time for a repeated START condition	$t_{SU;STA}$	(Note 5)		4.7		0.6		μs
Data hold time (Input) (Note 3, 4)	$t_{HD;DAT}$			0.0		0.0		μs
Data setup time	$t_{SU;DAT}$			250		100		ns
Setup time for STOP condition	$t_{SU;STO}$			4.0		0.6		μs
Bus free time between STOP and START conditions	t_{BUF}	(Note 5)		4.7		1.3		μs

Note 1) SCL clock low width (output) is calculated with: $(2^{n-1}+58)/(fsys/2)$

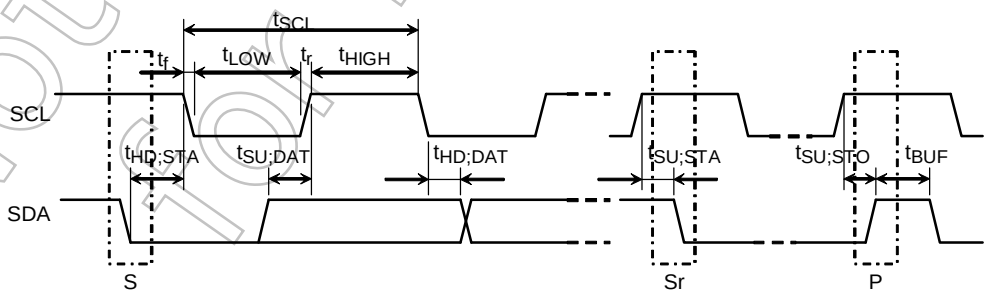
Note 2) SCL high width (output) is calculated with: $(2^{n-1}+12)/(fsys/2)$

Notice: On I²C-bus specification, Maximum Speed of Standard Mode is 100KHz, Fast mode is 400Khz. Internal SCL Frequency setting should comply with Note1 & Note2 shown above.

Note 3) The output data hold time is equal to 12x of internal SCL.

Note 4) The Philips I²C-bus specification states that a device must internally provide a hold time of at least 300 ns for the SDA signal to bridge the undefined region of the falling edge of SCL. However, this SBI does not satisfy this requirement. Also, the output buffer for SCL does not incorporate slope control of the falling edges; therefore, the equipment manufacturer should design so that the input data hold time shown in the table is satisfied, including tr/tf of the SCL and SDA lines.

Note 5) Software-dependent.



S: START condition
Sr: Repeated START condition
P; STOP condition

(2) Clock-Synchronous 8-Bit SIO mode

In the tables below, the letters x and T represent the fsys cycle periods and ϕT1 respectively. The letter n denotes the value of n programmed into the SCK (SCL output frequency select) field in the SBI0CR1.

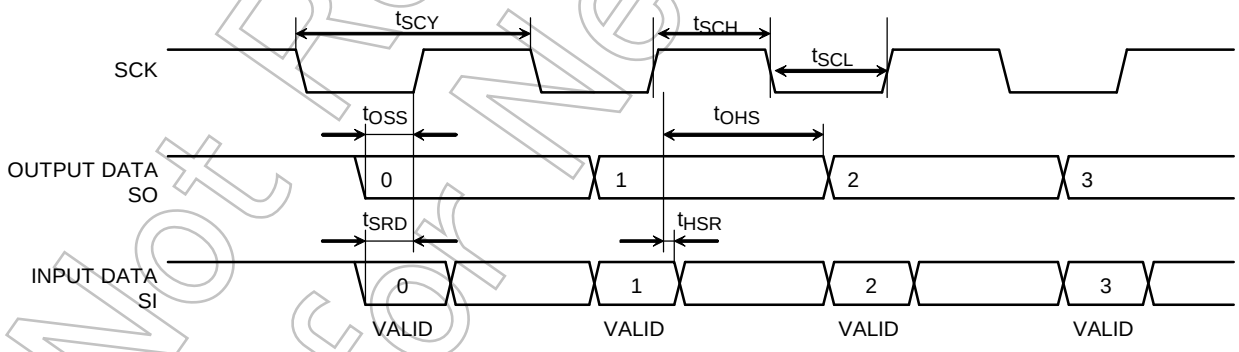
The electrical specifications below are for an SCK signal with a 50% duty cycle.

3) SCK Input Mode

Parameter	Symbol	Equation		40 MHz		54MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCK period	t_{SCY}	16x		400		296		ns
SCK Clock High width(input)	tsc_H	8x		200		148		ns
SCK Clock Low width(input)	tsc_H	8x		200		148		ns
SO to SCK rise	t_{OSS}	$(t_{SCY}/2) - (6x + 20)$		30		128		ns
SO hold after SCK rising	t_{OHS}	$(t_{SCY}/2) + 4x$		300		222		ns
SI valid to SCK rise	t_{SRD}	0		0		0		ns
SI hold after SCK rising	t_{HSR}	4x + 10		110		84		ns

4) SCK Output Mode

Parameter	Symbol	Equation		40 MHz		54MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCK period (programmable)	t_{SCY}	16x		400		296		ns
SO to SCK rise	t_{OSS}	$(t_{SCY}/2) - 20$		180		128		ns
SO hold after SCK rising	t_{OHS}	$(t_{SCY}/2) - 20$		180		128		ns
SI valid to SCK rise	t_{SRD}	2x + 30		80		67		ns
SI data hold after SCK rising	t_{HSR}	0		0		0		ns



23.12 Event Counter

In the table below, the letter x represents the fsys cycle period.

Parameter	Symbol	Equation		40 MHz		54MHz		Unit
		Min	Max	Min	Max	Min	Max	
Clock low pulse width	t_{VCKL}	2X + 100		150		137		ns
Clock high pulse width	t_{VCKH}	2X + 100		150		137		ns

23.13 Capture

In the table below, the letter x represents the fsys cycle period.

Parameter	Symbol	Equation		40 MHz		54MHz		Unit
		Min	Max	Min	Max	Min	Max	
Low pulse width	t_{CPL}	2X + 100		150		137		ns
High pulse width	t_{CPH}	2X + 100		150		137		ns

23.14 TMRD PPG Output

Parameter	Symbol	Equation		40 MHz		54MHz		Unit
		Min	Max	Min	Max	Min	Max	
High pulse width	t_{PPGH}	100		100		100		ns
Low pulse width	t_{PPGL}	100		100		100		ns

23.15 General Interrupt (INTC)

In the table below, the letter x represents the fsys cycle period.

Parameter	Symbol	Equation		40 MHz		54MHz		Unit
		Min	Max	Min	Max	Min	Max	
Low pulse width for INT0-INTF	t_{INTAL}	X + 100		125		118.5		ns
High pulse width for INT0-INTF	t_{INTAH}	X + 100		125		118.5		ns

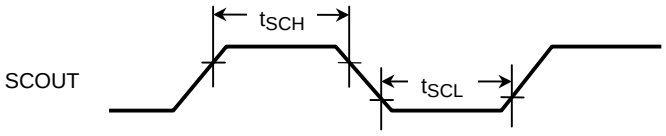
23.16 STOP Release Interrupt

Parameter	Symbol	Equation		40 MHz		54MHz		Unit
		Min	Max	Min	Max	Min	Max	
Low pulse width for $\overline{NMI}$ and INT0-INTF	t_{INTBL}	100		100		100		ns
High pulse width for INT0-INTF	t_{INTBH}	100		100		100		ns

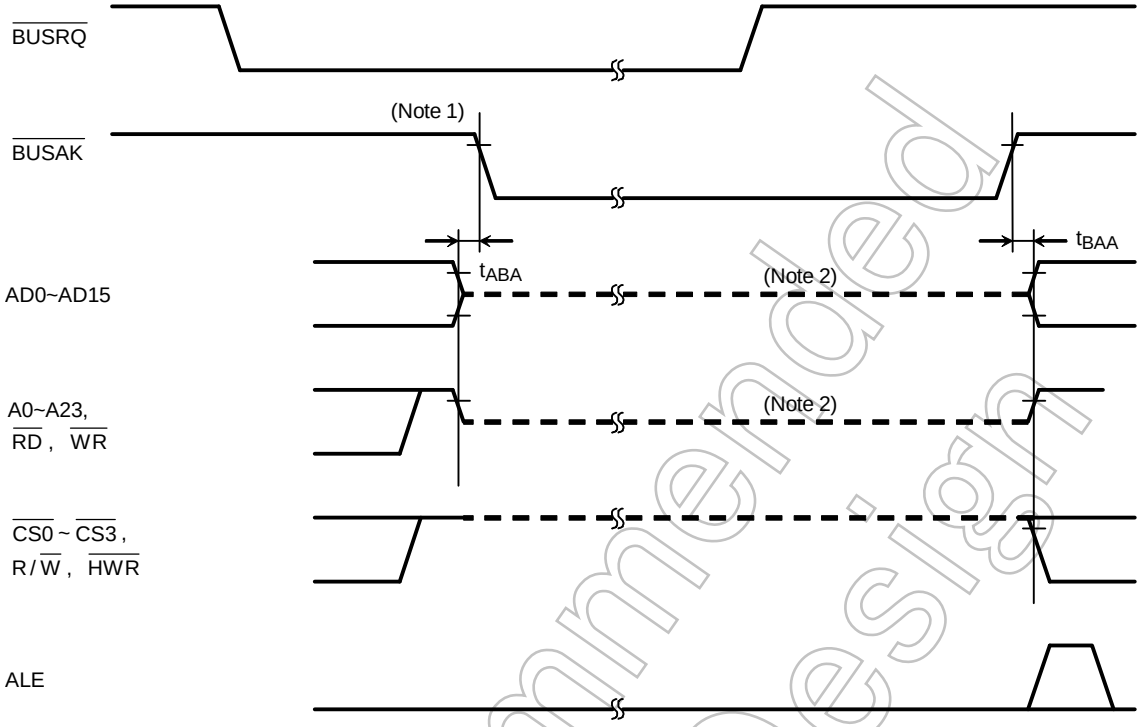
23.17 SCOUT Pin AC Characteristic

Parameter	Symbol	Equation		40 MHz		54MHz		Unit
		Min	Max	Min	Max	Min	Max	
Clock high pulse width	t_{SCH}	0.5T – 5		7.5		4.3		ns
Clock low pulse width	t_{SCL}	0.5T – 5		7.5		4.3		ns

Note: In the above table, the letter T represents the cycle period of the SCOUT output clock.



23.18 Bus Request and Bus Acknowledge Signals



Parameter	Symbol	Equation		40 MHz		54MHz		Unit
		Min	Max	Min	Max	Min	Max	
Bus float to $\overline{BUSAK}$ fall	t_{ABA}	0	80	0	80	0	80	ns
Bus float to $\overline{BUSAK}$ rise	t_{BAA}	0	80	0	80	0	80	ns

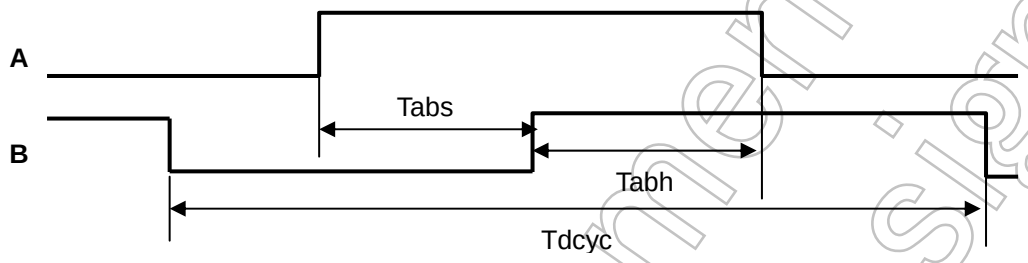
(Note 1) If the current bus cycle has not terminated due to wait-state insertion, the TMP19A23 does not respond to $\overline{BUSRQ}$ until the wait state ends.

(Note 2) This broken line indicates that output buffers are disabled, not that the signals are at indeterminate states. The pin holds the last logic value present at that pin before the bus is relinquished. This is dynamically accomplished through external load capacitances. In case of using the external load capacitance to maintain the bus at a predefined state, the equipment manufacturer needs to consider the additional time (determined by the CR constant) required for the signal transmission through the external load capacitances. The on-chip, integrated programmablepull up/pulldown resistors remain active, depending on internal signal states.

23.19 Dual Pulse Input

Parameter	Symbol	Equation		40 MHz		54MHz		Unit
		Min	Max	Min	Max	Min	Max	
Dual input pulse period	Tdcyc	8Y		400		296		ns
Dual input pulse setup	Tabs	Y+20		70		57		ns
Dual input pulse hold	Tabh	Y+20		70		57		ns

Y: Sampling clock (fsys/2)

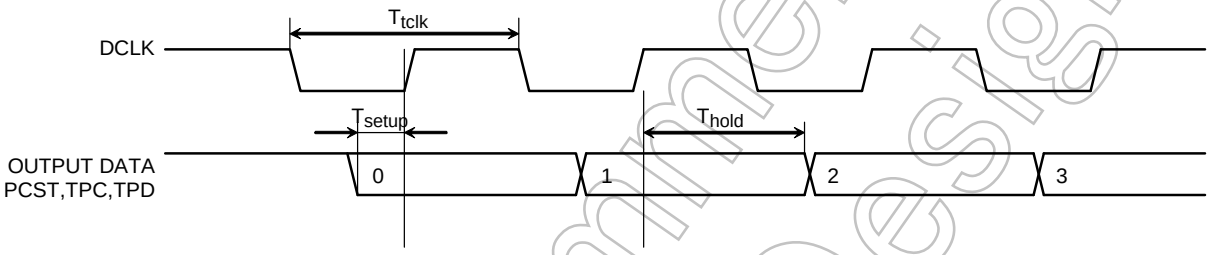


23.20 ADTRG Input

Parameter	Symbol	Equation		40 MHz		54MHz		Unit
		Min	Max	Min	Max	Min	Max	
Low pulse width for ADTRG	Tad_L	fsysy/2+20		32.5		29.3		ns
High pulse width for ADTRG	Tadh	fsysy/2+20		32.5		29.3		ns

23.21 DSU

Parameter	Symbol	Equation		40 MHz/54MHz		Unit
		Min	Max	Min	Max	
PCST valid to DCLK rise	Tsetup	11		11		ns
PCST hold after DCLK rising	Thold	0.5		0.5		ns
TPC valid to DCLK rise	Tsetup	11		11		ns
TPC hold after DCLK rising	Thold	0.5		0.5		ns
TPD valid to DCLK rise	Tsetup	11		11		ns
TPD hold after DCLK rising	Thold	0.5		0.5		ns



23.22 EJTAG

Parameter	Symbol			Unit
		Min	Max	
TCK rise to TMS/TDI input	Ttsetup	40		ns
TMS/TDI hold after TCK rising	Tthold	50		ns
TDO hold after TCK falling	Ttout		10	ns