TOSHIBA

32bit TX System RISC
TX19 family

TMP1942CYUE
TMP1942CZUE/XBG

Rev1.0   March 29, 2007

32-Bit RISC Microprocessor TX19 Family
TMP1942CYUE/CZUE/CZXBG

1.   Outline and Features

The TX19 is a family of high-performance 32-bit microprocessors that offers the speed of a 32-bit RISC solution with the added advantage of a significantly reduce code size of a 16-bit architecture. The instruction set of the TX19 includes as a subset the 32-bit instructions of the TX39, which is based on the MIPS R3000A[TM] architecture. Additionally, the TX19 supports the MIPS16[TM] Application-Specific Extensions (ASE) for improved code density.

The TMP1942 is built on a TX19 core processor and a selection of intelligent peripherals. The TMP1942 is suitable for low-voltage, low-power applications.

Features of the TMP1942 include the following:

(1) TX19 core processor

    1) Two instruction set architecture (ISA) modes: 16-bit ISA for code density and 32-bit ISA for speed

- The 16-bit ISA is object-code compatible with the code-efficient MIPS16^{TM} ASE.
- The 32-bit ISA is object-code compatible with the high-performance TX39 family.

    2) Combines high performance with low power consumption.

    - High performance

- Single clock cycle execution for most instructions
- 3-operand computational instructions for high instruction throughput
- 5-stage pipeline
- On-chip high-speed memory
- DSP function: Executes 32-bit x 32-bit multiplier operations with a 64-bit accumulation in a single clock cycle.

    - Low power consumption

- Optimized design using a low-power cell library
- Programmable standby modes in which processor clocks are stopped

    3) Fast interrupt response suitable for real-time control

- Distinct starting locations for each interrupt service routine
- Automatically generated vectors for each interrupt source
- Automatic updates of the interrupt mask level

(2) Internal RAM: FDUE/FDXBG: 20KB,CYUE/CZUE/CZXBG: 16 KB

    Internal ROM: FDUE/FDXBG: 512KB,CYUE/CZXBG: 384KB,CYUE: 256 KB

        ROM correction function (8 words x 4 blocks)

        (For FDUE/FDXBG, only registers are available; data is not replaced.)

(3) External memory expansion

- 16-Mbyte off-chip address space for code and data
- External bus interface with dynamic bus sizing for 8-bit and 16-bit data ports

(4) 4-channel DMA controller

- Interrupt- or software-triggered

(5) 6 channel 8-bit PWM timer

    (12 channel 8-bit interval timer, 6 channel 16-bit interval timer, 6 channel 8-bit PPG output)

(6) 14 channel 16-bit timer

    (2 channels support 2-phase input pulse counter mode.)

(7) 1 channel real-time counter (RTC)

(8) 5 channel general-purpose serial interface

    (Supports both UART and synchronous transfer modes)

(9) 1 channel serial bus interface

    Either I^{2}C bus mode or clock-synchronous mode can be selected.

(10) 16 channel 10-bit A/D converter (with internal sample/hold)

    Conversion time: 2 μs (throughput), 4 to 5 μs (latency)

(11) 3 channel 10-bit D/A converter

(12) Watchdog timer

(13) 4 channel chip select/wait controller

(14) Interrupt sources

- 4 CPU interrupts: software interrupt instruction

- 45 internal interrupts: 7 priority levels, with the exception of the watchdog timer interrupt

- 29 external interrupts: 7 priority levels, with the exception of the NMI interrupt

    The external sources include 14 KWUP sources, which are all assigned to a single interrupt vector, and 4 extended interrupts (INTB, INTC, INTD, and INTE), which are all assigned to a single interrupt vector with an identification flag. Thus, the actual number of external interrupt sources is 13.

(15) 108 pin input/output ports

(16) Three standby function

- IDLE, SLEEP, and STOP

(17) Dual clocks

- RTC clock: Low-speed clock (32.768 kHz)

(18) Clock generator

- On-chip PLL (x4)

- Clock gear: Divides the operating speed of the CPU by 1/2, 1/4 or 1/8

(19) Operating voltage range: 2.7 to 3.6 V

PC and PF are 2.7 to 3.6 V or 4.5 to 5.25 V for 5 V-enabled ports.

(20) Operating frequency

- 32 MHz (Vcc $\geq$ 3.0 V)

- 28 MHz (Vcc $\geq$ 2.7 V)

(21) Package

- 144-pin QFP (16 x 16 x 1.4 (t) mm, 0.4-mm pitch): FDUE/CZUE/CYUE

- 177-pin CSP (13 x 13 x 1.4 (t) mm, 0.8-mm pitch): FDXBG/CZXBG

Note: TMP1942FDXBG (Package: 177-pin CSP) is under development.
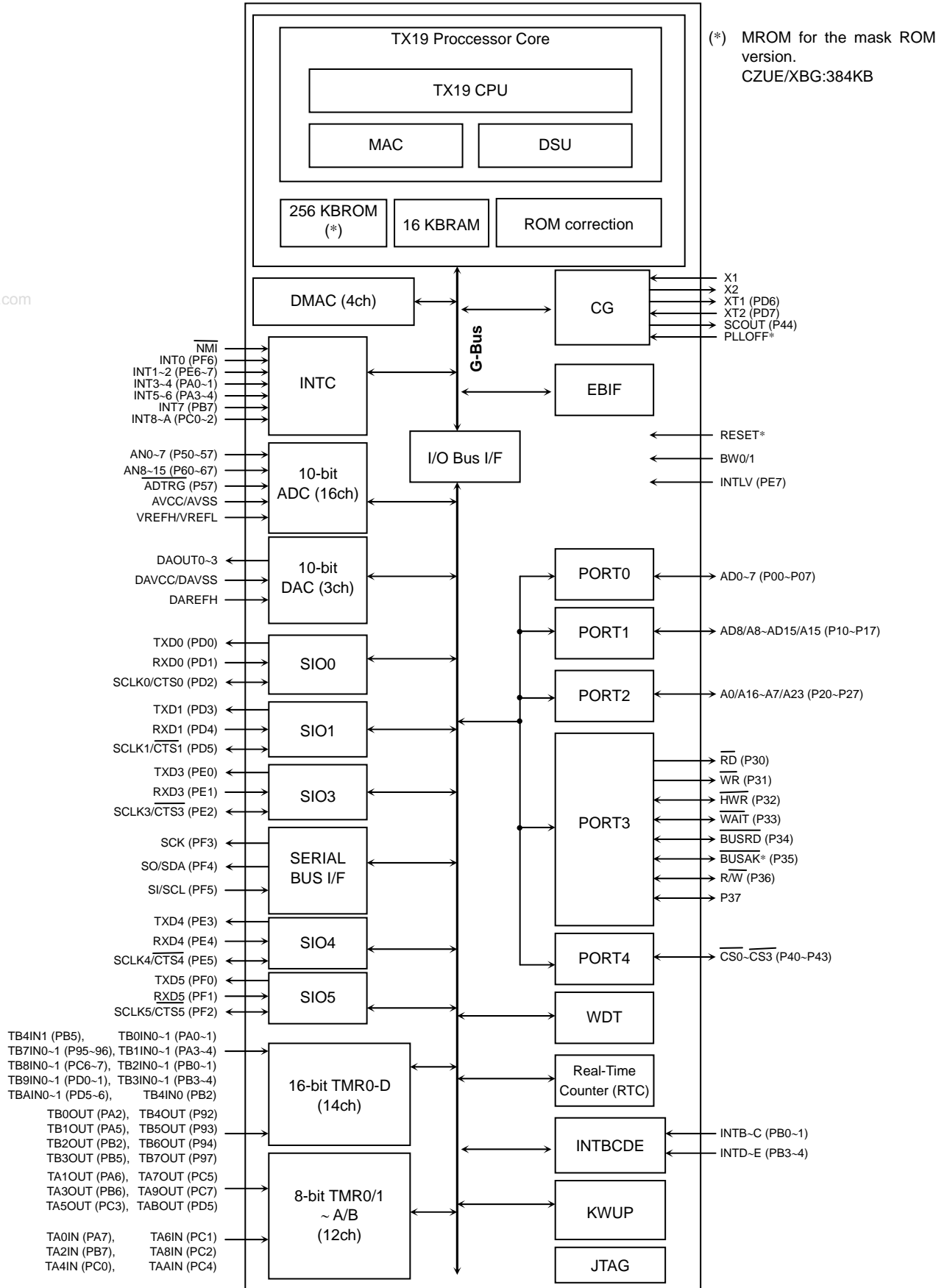
Figure 1.1 TMP1942 Block Diagram

2. Signal Descriptions

This section contains pin assignments for the TMP1942 as well as brief descriptions of the functions of the TMP1942 input and output pins.

2.1 Pin Assignment

Table 2.1.1 shows TMP1942 pin assignment.
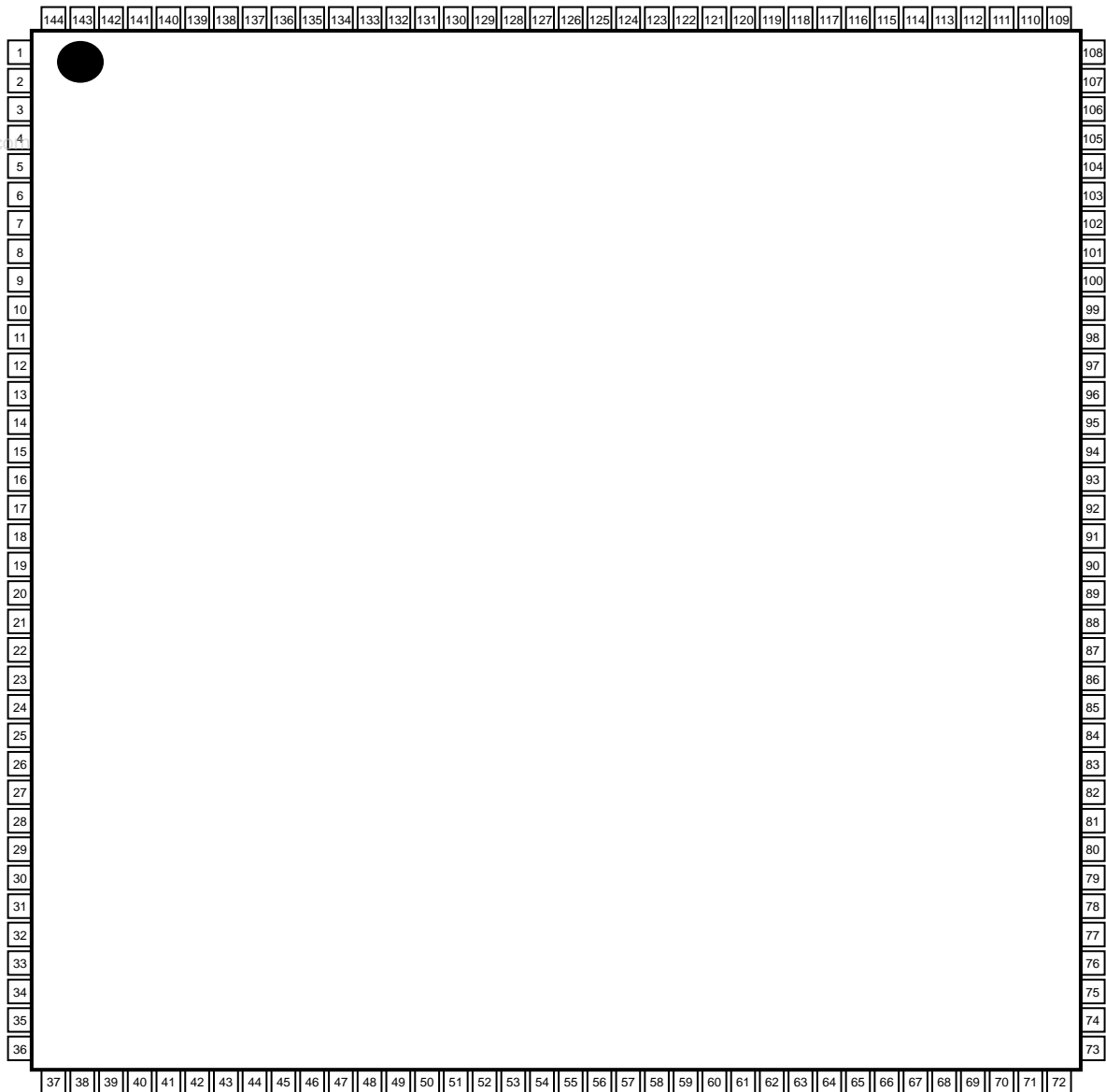
Figure 2.1.1 144-Pin LQFP Pin Assignment

Table 2.1.1 Pin Assignment (144-pin LQFP)

Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name
1	VREFH	37	P11/AD9/A9	73	P90/KEY8/DCLK	109	CVCC
2	VREFL	38	P12/AD10/A10	74	P91/KEY9/PCST2	110	X2
3	P50/AN0	39	P13/AD11/A11	75	P92/TB4OUT/PCST1	111	CVSS
4	P51/AN1	40	P14/AD12/A12	76	P93/TB5OUT/PCST0	112	X1
5	P52/AN2	41	P15/AD13/A13	77	P94/TB6OUT/SDSA0/TPC	113	TEST1
6	P53/AN3	42	P16/AD14/A14	78	P95/TB7IN0/$\overline{DBGE}$	114	$\overline{RESET}$
7	DAVCC	43	P17/AD15/A15	79	P96/TB7IN1/$\overline{DINT}$	115	PD6/XT1
8	DAVSS	44	P20/A0/A16	80	P97/TB7OUT/DRESET	116	PD7/XT2
9	DAREH	45	P21/A1/A17	81	DVCC3	117	$\overline{NMI}$
10	DAOUT0	46	P22/A2/A18	82	PA0/TB0IN0/INT3	118	BW0
11	DAOUT1	47	P23/A3/A19	83	PA1/TB0IN1/INT4	119	PB0/TB2IN0/INTB
12	DAOUT2	48	P24/A4/A20	84	PA2/TB0OUT	120	PB1/TB2IN1/INTC
13	P54/AN4	49	P25/A5/A21	85	PA3/TB1IN0/INT5	121	PB2/TB2OUT/TB4IN0
14	P55/AN5	50	P26/A6/A22	86	PA4/TB1IN1/INT6	122	PB3/TB3IN0/INTD
15	P56/AN6	51	P27/A7/A23	87	PA5/TB1OUT	123	PB4/TB3IN1/INTE
16	P57/AN7/$\overline{ADTRG}$	52	TEST0	88	PA6/TA1OUT	124	PB5/TB3OUT/TB4IN1
17	P60/AN8/KEY0	53	$\overline{PLLOFF}$	89	PA7/TA0IN/KEYA	125	PB6/TA3OUT
18	DVSS	54	DVSS	90	DVSS	126	DVSS
19	P61/AN9/KEY1	55	ALE	91	RSTPUP	127	DVCC3
20	P62/AN10/KEY2	56	DVCC3	92	PC0/TA4IN/INT8	128	PB7/TA2IN/INT7/KEYB
21	P63/AN11/KEY3	57	BW1	93	PC1/TA6IN/INT9	129	PD0/TXD0/TB9IN0
22	P64/AN12/KEY4	58	P30/$\overline{RD}$	94	PC2/TA8IN/INTA	130	PD1/RXD0/TB9IN1
23	P65/AN13/KEY5	59	P31/$\overline{WR}$	95	PC3/TA5OUT	131	PD2/SCLK0/$\overline{CTS0}$
24	P66/AN14/KEY6	60	P32/$\overline{HWR}$	96	PC4/TAAIN	132	PD3/TXD1/TBAIN0
25	P67/AN15/KEY7	61	P33/$\overline{WAIT}$	97	PC5/TA7OUT	133	PD4/RXD1/TBAIN1
26	DVCC3	62	P34/$\overline{BUSRQ}$	98	PC6/TB8IN0/KEYC	134	PD5/SCLK1/$\overline{CTS1}$/TABOUT
27	P00/AD0	63	P35/$\overline{BUSAK}$	99	PC7/TB8IN1/TA9OUT	135	PE0/TXD3
28	P01/AD1	64	P36/R$\overline{W}$	100	DVCC52	136	PE1/RXD3
29	P02/AD2	65	P37/$\overline{DSU}$	101	PF0/TXD5	137	PE2/SCLK3/$\overline{CTS3}$
30	P03/AD3	66	DVSS	102	PF1/RXD5/KEYD	138	PE3/TXD4
31	P04/AD4	67	DVCC3	103	PF2/SCLK5/$\overline{CTS5}$	139	PE4/RXD4
32	P05/AD5	68	P40/$\overline{CS0}$	104	PF3/SCK	140	PE5/SCLK4/$\overline{CTS4}$
33	P06/AD6	69	P41/$\overline{CS1}$	105	PF4/SO/SDA	141	PE6/INT1/$\overline{BOOT}$
34	P07/AD7	70	P42/$\overline{CS2}$	106	PF5/SI/SCL	142	PE7/INT2/INTLV
35	DVSS	71	P43/$\overline{CS3}$	107	PF6/INT0	143	AVCC
36	P10/AD8/A8	72	P44/SCOUT	108	DVCC51	144	AVSS

Figure 2.1.2 shows pin assignment for the 177-pin model of the TMP1942.

A1	A2	A3	A4	A5	A6	A7	A8	A9	A10	A11	A12	A13	A14	A15
B1	B2	B3	B4	B5	B6	B7	B8	B9	B10	B11	B12	B13	B14	B15
C1	C2	C3	C4	C5	C6	C7	C8	C9	C10	C11	C12	C13	C14	C15
D1	D2	D3	D4	D5	D6	D7	D8	D9	D10	D11	D12	D13	D14	D15
E1	E2	E3	E4								E12	E13	E14	E15
F1	F2	F3	F4								F12	F13	F14	F15
G1	G2	G3	G4								G12	G13	G14	G15
H1	H2	H3	H4								H12	H13	H14	H15
J1	J2	J3	J4								J12	J13	J14	J15
K1	K2	K3	K4								K12	K13	K14	K15
L1	L2	L3	L4								L12	L13	L14	L15
M1	M2	M3	M4	M5	M6	M7	M8	M9	M10	M11	M12	M13	M14	M15
N1	N2	N3	N4	N5	N6	N7	N8	N9	N10	N11	N12	N13	N14	N15
P1	P2	P3	P4	P5	P6	P7	P8	P9	P10	P11	P12	P13	P14	P15
R1	R2	R3	R4	R5	R6	R7	R8	R9	R10	R11	R12	R13	R14	R15

Figure 2.1.2 177-Pin CSP Pin Assignment

Table 2.1.2 Pin Assignment (177-pin CSP)

Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name	Pin No.	Pin Name
A1	VREFL	D1	P50/AN0	H13	NC	N4	P16/AD14/A14
A2	AVSS	D2	DAVSS	H14	NC	N5	P21/A1/A17
A3	AVCC	D3	P52/AN2	H15	DVSS	N6	P25/A5/A21
A4	PE7/INT2/INTLV	D4	P51/AN1	J1	P67/AN15/KEY7	N7	DVSS
A5	PE3/TXD4	D5	PE0/TXD3	J2	P65/AN13/KEY5	N8	TEST0
A6	TCK (JTAG)	D6	PD3/TXD1/TBAIN0	J3	P66/AN14/KEY6	N9	P30/$\overline{\text{RD}}$
A7	PD2/SCLK0/$\overline{\text{CTS0}}$	D7	PB7/TA2IN/INT7/KEYB	J4	P64/AN12/KEY4	N10	P32/$\overline{\text{HWR}}$
A8	PB5/TB3OUT/TB4IN1	D8	DVSS	J12	PA6/TA1OUT	N11	P37
A9	PB1/TB2IN1/INTC	D9	PB2/TB2OUT/TB4IN0	J13	PA7/TA0IN/KEYA	N12	DVSS
A10	PD7/TX2	D10	$\overline{\text{NMI}}$	J14	NC	N13	P41/$\overline{\text{CS1}}$
A11	PD6/TX1	D11	NC	J15	PA5/TB1OUT	N14	P91/KEY9
A12	X1	D12	NC	K1	P01/AD1	N15	NC
A13	X2	D13	PF1/RXD5/KEYD	K2	DVCC3	P1	NC
A14	CVCC	D14	PF3/SCK	K3	NC	P2	P10/AD8/A8
A15	NC	D15	PF6/INT0	K4	NC	P3	P12/AD10/A10
B1	NC	E1	DAVCC	K12	PA2/TB0OUT	P4	P20/A0/A16
B2	NC	E2	DAOUT0	K13	PA3/TB1IN0/INT5	P5	P22/A2/A18
B3	PE6/INT1	E3	DAREFH	K14	PA4/TB1IN1/INT6	P6	P26/A6/A22
B4	PE4/RXD4	E4	P53/AN3	K15	PA1/TB0IN1/INT4	P7	TDO (JTAG)
B5	$\overline{\text{TRST}}$ (JTAG)	E5	NC (Bonding not applied)	L1	P04/AD4	P8	ALE
B6	PD5/SCLK1/$\overline{\text{CTS1}}$/TABOUT	E12	PC6/TB8IN0/KEYC	L2	P02/AD2	P9	BW1
B7	PD0/TXD0/TB9IN0	E13	DVCC52	L3	TMS (JTAG)	P10	P33/$\overline{\text{WAIT}}$
B8	DVCC3	E14	PF0/TXD5	L4	P00/AD0	P11	TDI (JTAG)
B9	PB4/TB3IN1/INTE	E15	PF2/SCLK5/$\overline{\text{CTS5}}$	L12	P97/TB7OUT	P12	P40/$\overline{\text{CS0}}$
B10	PB0/TB2IN0/INTB	F1	DAOUT1	L13	DVCC3	P13	P42/$\overline{\text{CS2}}$
B11	NC	F2	P55/AN5	L14	PA0/TB0IN0/INT3	P14	P44/SCOUT
B12	$\overline{\text{RESET}}$	F3	P54/AN4	L15	P96/TB7IN1	P15	NC
B13	CVSS	F4	DAOUT2	M1	P07/AD7	R1	P11/AD9/A9
B14	DVCC51	F12	PC2/TA8IN/INTA	M2	P05/AD5	R2	NC
B15	NC	F13	PC4/TAAIN	M3	P03/AD3	R3	NC
C1	VREFH	F14	PC5/TA7OUT	M4	P14/AD12/A12	R4	P13/AD11/A11
C2	NC	F15	PC7/TB8IN1/TA9OUT	M5	P15/AD13/A13	R5	P17/AD15/A15
C3	PE5/SCLK4/$\overline{\text{CTS4}}$	G1	P56/AN6	M6	P24/A4/A20	R6	P23/A3/A19
C4	PE2/SCLK3/$\overline{\text{CTS3}}$	G2	P61/AN9/KEY1	M7	$\overline{\text{PLLOFF}}$	R7	P27/A7/A23
C5	PE1/RXD3	G3	NC	M8	NC	R8	NC
C6	PD4/RXD1/TBAIN1	G4	P60/AN8/KEY0	M9	DVCC3	R9	P31/$\overline{\text{WR}}$
C7	PD1/RXD0/TB9IN1	G12	PC0/TA4IN/INT8	M10	P34/$\overline{\text{BUSRQ}}$	R10	P35/$\overline{\text{BUSAK}}$
C8	PB6/TA3OUT	G13	PC1/TA6IN/INT9	M11	P36/R/$\overline{\text{W}}$	R11	DVCC3
C9	PB3/TB3IN0/INTD	G14	NC	M12	P93/TB5OUT	R12	NC
C10	BW0	G15	PC3/TA5OUT	M13	P94/TB6OUT	R13	P43/$\overline{\text{CS3}}$
C11	NC	H1	DVSS	M14	P95/TB7IN0	R14	NC
C12	TEST1	H2	P63/AN11/KEY3	M15	P92/TB4OUT	R15	P90/KEY8
C13	PF4/SO/SDA	H3	P57/AN7/$\overline{\text{ADTRG}}$	N1	NC		
C14	PF5/SI/SCL	H4	P62/AN10/KEY2	N2	DVSS		
C15	NC	H12	RSTPUP	N3	P06/AD6		

2.2   Pin Usage Information

Table 2.2.1 lists the names and functions of the TMP1942's input/output pins.

Table 2.2.1 Pin Names and Functions

Pin Name	# of Pins	Type	Function
P00~P07	8	Input/output	Port 0: Individually programmable as input or output
AD0~AD7		Input/output	Address (Lower): Bits 0-7 of the address/data bus
P10~P17	8	Input/output	Port 1: Individually programmable as input or output
AD8~AD15		Input/output	Address/Data (Upper): Bits 8-15 of the address/data bus
A8~A15		Output	Address: Bits 8-15 of the address bus
P20~P27	8	Input/output	Port 2: Individually programmable as input or output
A0~A7		Output	Address: Bits 0-7 of the address bus
A16~A23		Output	Address: Bits 16-23 of the address bus
P30	1	Output	Port 30: Output-only
$\overline{\text{RD}}$		Output	Read Strobe: Asserted during a read operation from an external memory device
P31	1	Output	Port 31: Output-only
$\overline{\text{WR}}$		Output	Write Strobe: Asserted during a write operation on D0-D7
P32	1	Input/output	Port 32: Programmable as input or output (with internal pull-up resister)
$\overline{\text{HWR}}$		Output	Higher Write Strobe: Asserted during a write operation on D8-D15
P33	1	Input/output	Port 33: Programmable as input or output (with internal pull-up resister)
$\overline{\text{WAIT}}$		Input	Wait: Causes the CPU to suspend external bus activity
P34	1	Input/output	Port 34: Programmable as input or output (with internal pull-up resister)
$\overline{\text{BUSRQ}}$		Input	Bus Request: Asserted by an external bus master to request bus mastership
P35	1	Input/output	Port 35: Programmable as input or output (with internal pull-up resister)
$\overline{\text{BUSAK}}$		Output	Bus Acknowledge: Indicates that the CPU has relinquished the bus in response to $\overline{\text{BUSRQ}}$.
P36	1	Input/output	Port 36: Programmable as input or output (with internal pull-up resister)
R/$\overline{\text{W}}$		Output	Read/Write: Indicates the direction of data transfer on the bus: 1 = read or dummy cycle, 0 = write cycle
P37	1	Input/output	Port 37: Programmable as input or output (with internal pull-up resister)
$\overline{\text{DSU}}$		Input	This pin is used to select the operating mode during reset. The TMP1940CYAF enters NORMAL mode when this pin is sampled high at the rising edge of $\overline{\text{RESET}}$. This pin should not be pulled down to a logic 0 during a reset sequence. The TMP1940FDBF, which has an on-chip flash, uses this pin as an interface to the DSU tool. For details, refer to Part 4, *TMP1940FDBF*.
P40	1	Input/output	Port 40: Programmable as input or output (with internal pull-up resister)
$\overline{\text{CS0}}$		Output	Chip Select 0: Asserted low to enable external devices at programmed addresses
P41	1	Input/output	Port 41: Programmable as input or output (with internal pull-up resister)
$\overline{\text{CS1}}$		Output	Chip Select 1: Asserted low to enable external devices at programmed addresses
P42	1	Input/output	Port 42: Programmable as input or output (with internal pull-up resister)
$\overline{\text{CS2}}$		Output	Chip Select 2: Asserted low to enable external devices at programmed addresses
P43	1	Input/output	Port 43: Programmable as input or output (with internal pull-up resister)
$\overline{\text{CS3}}$		Output	Chip Select 3: Asserted low to enable external devices at programmed addresses
P44	1	Input/output	Port 44: Programmable as input or output
SCOUT		Output	System Clock Output: Drives out a clock signal at the same frequency as the CPU clock (high-speed or low-speed)
P50~P57	8	Input	Port 5: Input-only
AN0~AN7		Input	Analog input: Input to the A/D converter
$\overline{\text{ADTRG}}$		Input	External start request for the A/D converter (multiplexed with P57)
P60~P67	1	Input/output	Port 6: Input-only
AN8~AN15		Input	Analog input: Input to the A/D converter
KEY0-KEY7		Output	Key on wake-up input (with internal pull-up resister) (dynamic pull-up selectable)
P90	1	Input/output	Port 90: Programmable as input or output
DSU (DCLK)		Output	DSU pin
KEY8		Input	Key on wake-up input (with internal pull-up resister) (dynamic pull-up selectable)

Pin Name	# of Pins	Type	Function
P91	1	Input/output	Port 91: Programmable as input or output
DSU (PCST2)		Output	DSU pin
KEY9		Input	Key on wake-up input (with internal pull-up resister) (dynamic pull-up selectable)
P92	1	Input/output	Port 92: Programmable as input or output
DSU (PCST1)		Output	DSU pin
TB4OUT		Output	16-Bit Timer 4 Output: Output from 16-bit Timer 4
P93	1	Input/output	Port 93: Programmable as input or output
DSU (PCST0)		Output	DSU pin
TB5OUT		Output	16-Bit Timer 5 Output: Output from 16-bit Timer 5
P94	1	Input/output	Port 94: Programmable as input or output
DSU		Output	DSU pin
(SDSA0/TPC)			
TB6OUT		Output	16-Bit Timer 6 Output: Output from 16-bit Timer 6
P95	1	Input/output	Port 95: Programmable as input or output
DSU (DBGE∗)		Input	DSU pin
TB7IN0			16-Bit Timer 7 Input 0: Count/capture trigger input to 16-bit Timer 7
P96	1	Input/output	Port 96: Programmable as input or output
DSU (DINT∗)		Input	DSU pin
TB7IN1			16-Bit Timer 7 Input 1: Capture trigger input to 16-bit Timer 7
P97	1	Input/output	Port 97: Programmable as input or output
DSU		Input	DSU pin
(DRESET)			
TB7OUT		Output	16-Bit Timer 7 Output: Output from 16-bit Timer 7
PA0	1	Input/output	Port A0: Programmable as input or output
TB0IN0		Input	16-Bit Timer 0 Input 0: Count/capture trigger input to 16-bit Timer 0
INT3		Input	Interrupt Request 3: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive
PA1	1	Input/output	Port A1: Programmable as input or output
TB0IN1		Input	16-Bit Timer 0 Input 1: Capture trigger input to 16-bit Timer 0
INT4		Input	Interrupt Request 4: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive
PA2	1	Input/output	Port A2: Programmable as input or output
TB0OUT		Output	16-Bit Timer 0 Output: Output from 16-bit Timer 0
PA3	1	Input/output	Port A3: Programmable as input or output
TB1IN0		Input	16-Bit Timer 1 Input 0: Count/capture trigger input to 16-bit Timer 1
INT5		Input	Interrupt Request 5: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive
PA4	1	Input/output	Port A4: Programmable as input or output
TB1IN1		Input	16-Bit Timer 1 Input 1: Capture trigger input to 16-bit Timer 1
INT6		Input	Interrupt Request 6: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive
PA5	1	Input/output	Port A5: Programmable as input or output
TB1OUT		Output	16-Bit Timer 1 Output: Output from 16-bit Timer 1
PA6	1	Input/output	Port A6: Programmable as input or output
TA1OUT		Output	8-Bit Timer 0/1 Output: Output from 8-bit Timer 0 or 1
PA7	1	Input/output	Port A7: Programmable as input or output
TA0IN		Input	8-Bit Timer 0 Input: Input to 8-bit Timer 0
KEYA		Input	Key on wake-up input (with internal pull-up resister) (dynamic pull-up selectable)
PB0	1	Input/output	Port B0: Programmable as input or output
TB2IN0		Input	16-Bit Timer 2 Input 0: Count/capture trigger input/2-phase input pulse counter input to 16-bit Timer 2
INTB		Input	Interrupt Request B: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive

Pin Name	# of Pins	Type	Function
PB1 TB2IN1 INTC	1	Input/output Input Input	Port B1: Programmable as input or output 16-Bit Timer 2 Input 1: Capture trigger input/2-phase input pulse counter input to 16-bit Timer 2 Interrupt Request C: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive
PB2 TB2OUT TB4IN0	1	Input/output Output Input	Port B2: Programmable as input or output 16-Bit Timer 2 Output: Output from 16-bit Timer 2 16-Bit Timer 4 Input 0: Count/capture trigger input to 16-bit Timer 4
PB3 TB3IN0 INTD	1	Input/output Input Input	Port B3: Programmable as input or output 16-Bit Timer 3 Input 0: Count/capture trigger input/2-phase input pulse counter input to 16-bit Timer 3 Interrupt Request D: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive
PB4 TB3IN1 INTE	1	Input/output Input Input	Port B4: Programmable as input or output 16-Bit Timer 3 Input 1: Capture trigger input/2-phase input pulse counter input to 16-bit Timer 3 Interrupt Request E: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive
PB5 TB3OUT TB4IN1	1	Input/output Output Input	Port B5: Programmable as input or output 16-Bit Timer 3 Output: Output from 16-bit Timer 3 16-Bit Timer 4 Input 1: Capture trigger input to 16-bit Timer 4
PB6 TA3OUT	1	Input/output Output	Port B6: Programmable as input or output 8-Bit Timer 2/3 Output: Output from 8-bit Timer 2 or 3
PB7 TA2IN INT7 KEYB	1	Input/output Input Input Input	Port B7: Programmable as input or output 8-Bit Timer 2 Input: Input to 8-bit Timer 2 Interrupt Request 7: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive Key on wake-up input (with internal pull-up resister) (dynamic pull-up selectable)
PC0 TA4IN INT8	1	Input/output Input Input	Port C0: Programmable as input or output 8-Bit Timer 4 Input: Input to 8-bit Timer 4 Interrupt Request 8: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive
PC1 TA6IN INT9	1	Input/output Input Input	Port C1: Programmable as input or output 8-Bit Timer 6 Input: Input to 8-bit Timer 6 Interrupt Request 9: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive
PC2 TA8IN INTA	1	Input/output Input Input	Port C2: Programmable as input or output 8-Bit Timer 8 Input: Input to 8-bit Timer 8 Interrupt Request A: Programmable to be high-level, low-level, rising-edge or falling-edge sensitive
PC3 TA5OUT	1	Input/output Output	Port C3: Programmable as input or output 8-Bit Timer 4/5 Output: Output from 8-bit Timer 4 or 5
PC4 TAAIN	1	Input/output Input	Port C4: Programmable as input or output 8-Bit Timer A Input: Input to 8-bit Timer A
PC5 TA7OUT	1	Input/output Output	Port C5: Programmable as input or output 8-Bit Timer 6/7 Output: Output from 8-bit Timer 6 or 7
PC6 TB8IN0 KEYC	1	Input/output Input Input	Port C6: Programmable as input or output 16-Bit Timer 8 Input 0: Count/capture trigger input to 16-bit Timer 8 Key on wake-up input (with internal pull-up resister) (dynamic pull-up selectable)
PC7 TB8IN1 TA9OUT	1	Input/output Input Output	Port C7: Programmable as input or output 16-Bit Timer 8 Input 1: Capture trigger input to 16-bit Timer 8 8-Bit Timer 8/9 Output: Output from 8-bit Timer 8 or 9
PD0 TXD0 TB9IN0	1	Input/output Output Input	Port D0: Programmable as input or output Serial Transmit Data 0 Programmable as an open-drain output 16-Bit Timer 9 Input 0: Count/capture trigger input to 16-bit Timer 9

TMP1942CY/CZ-11

Pin Name	# of Pins	Type	Function
PD1	1	Input/output	Port D1: Programmable as input or output
RXD0		Input	Serial Receive Data 0
TB9IN1		Input	16-Bit Timer 9 Input 1: Capture trigger input to 16-bit Timer 9
PD2	1	Input/output	Port D2: Programmable as input or output
SCLK0		Input/output	Serial Clock Input/Output 0
CTS0*		Input	Serial Clear-to-Send 0
			Programmable as an open-drain output
PD3	1	Input/output	Port D3: Programmable as input or output
TXD1		Output	Serial Transmit Data 1
			Programmable as an open-drain output
TBAIN0		Input	16-Bit Timer A Input 0: Count/capture trigger input to 16-bit Timer A
PD4	1	Input/output	Port D4: Programmable as input or output
RXD1		Input	Serial Receive Data 1
TBAIN1		Input	16-Bit Timer A Input 1: Capture trigger input to 16-bit Timer A
PD5	1	Input/output	Port D5: Programmable as input or output
SCLK1		Input/output	Serial Clock Input/Output 1
CTS1		Input	Serial Clear-to-Send 1
			Programmable as an open-drain output
TABOUT		Output	8-Bit Timer A/B Output: Output from 8-bit Timer A or B
PD6	1	Input/output	Port D6: Programmable as input or open-drain output
XT1		Input	Connection pin for a low-speed crystal
PD7	1	Input/output	Port D7: Programmable as input or open-drain output
XT2		Output	Connection pin for a low-speed crystal
PE0	1	Input/output	Port E0: Programmable as input or output
TXD3		Output	Serial Transmit Data 3
			Programmable as an open-drain output
PE1	1	Input/output	Port E1: Programmable as input or output
RXD3		Input	Serial Receive Data 3
PE2	1	Input/output	Port E2: Programmable as input or output
CTS3*		Input/output	Serial Clock Input/Output 3
		Input	Serial Clear-to-Send 3
			Programmable as an open-drain output
PE3	1	Input/output	Port E3: Programmable as input or output
TXD4		Output	Serial Transmit Data 4
			Programmable as an open-drain output
PE4	1	Input/output	Port E4: Programmable as input or output
RXD4		Input	Serial Receive Data 4
PE5	1	Input/output	Port E5: Programmable as input or output
SCLK4		Input/output	Serial Clock Input/Output 4
CTS4		Input	Serial Clear-to-Send 4
			Programmable as an open-drain output
PE6	1	Input/output	Port E6: Programmable as input or output
INT1		Input	Interrupt request 1: Individually programmable to be high-level, low-level, rising-edge or falling-edge sensitive.
BOOT			Single-boot mode setting pin: Used when rewriting built-in flash memory (low active). During normal operation, this pin should be pulled up. This pin should always be pulled up for the mask ROM version.
PE7	1	Input/output	Port E7: Programmable as input or output
INT2		Input	Interrupt request 2: Individually programmable to be high-level, low-level, rising-edge or falling-edge sensitive.
INTLV			Interleave mode setting pin: This pin should be pulled up when using interleave mode. Otherwise, it should be pulled down.
PF0	1	Input/output	Port F0: Programmable as input or output
TXD5		Output	Serial Transmit Data 5
			Programmable as an open-drain output

Pin Name	# of Pins	Type	Function
PF1	1	Input/output	Port F1: Programmable as input or output
RXD5		Input	Serial Receive Data 5
KEYD		Input	Key on wake-up input (with internal pull-up resister) (dynamic pull-up selectable)
PF2	1	Input/output	Port F2: Programmable as input or output
SCLK5		Input/output	Serial Clock Input/Output 5
$\overline{\text{CTS5}}$		Input	Serial Clear-to-Send 5
			Programmable as an open-drain output
PF3	1	Input/output	Port F3: Programmable as input or output
SCK		Input/output	Clock input/output pin when the serial bus interface is in SIO mode
PF4	1	Input/output	Port F4: Programmable as input or output
SO		Output	Data transmission pin when the serial bus interface is in SIO mode
SDA		Input/output	Data transmission/reception pin when the serial bus interface is in I^{2}C mode
			Programmable as an open-drain output
PF5	1	Input/output	Port F5: Programmable as input or output
SI		Input	Data reception pin when the serial bus interface is in SIO mode
SCL		Input/output	Clock input/output pin when the serial bus interface is in I^{2}C mode
			Programmable as an open-drain output
PF6		Input/output	Port F6: Programmable as input or output
INT0		Input	Interrupt request 0: Individually programmable to be high-level, low-level, rising-edge or falling-edge sensitive.
ALE	1	Output	Address Latch Enable (This signal is driven out only when external memory is accessed)
TEST0	1	Input	Test pin
TEST1	1	Input	Test pin
RSTPUP	1	Input	When this pin is driven high (upon reset), pull-up for ports 3 and 4 is enabled. When this pin is driven low, pull-up is disabled.
DAOUT0-2	3	Output	D/A converter output
$\overline{\text{NMI}}$	1	Input	Non-maskable Interrupt Request: Causes an NMI interrupt on the falling edge
BW0~1	2	Input	Set both AM0 and AM1 to 1.
$\overline{\text{PLLOFF}}$	1	Input	This pin should be tied to logic 1 when the frequency multiplied clock from the PLL is used; otherwise, it should be tied to logic 0.
$\overline{\text{RESET}}$	1	Input	Reset (with internal pull-up resister): Initializes the whole TMP1940CYAF
VREFH	1	Input	Input pin for high reference voltage for the A/D converter.
VREFL	1	Input	Input pin for low reference voltage for the A/D converter.
AVCC	1	—	Power supply pin for the A/D converter. This pin should always be connected to power supply even when the A/D converter is not used.
AVSS	1	—	Ground pin for the A/D converter. This pin should always be connected to ground even when the A/D converter is not used.
DAVCC	1	—	Power supply pin for the D/A converter. This pin should always be connected to power supply even when the D/A converter is not used.
DAVSS	1	—	Ground pin for the D/A converter. This pin should always be connected to ground even when the D/A converter is not used.
DAREFH	1	—	Reference voltage input pin for the D/A converter
X1/X2	2	Input/output	Resonator connecting pin
CVCC	1	—	Power supply pin for the oscillator
CVSS	1	—	Ground pin for the oscillator (0 V)
DVCC3	4	—	Power supply pins
DVCC51	1	—	Power supply pin (port F)
DVCC52	1	—	Power supply pin (port C)
DVSS	5	—	Ground pins (0 V)

Port C becomes a 5 V port when a 5 V power supply is connected to DVCC52.

Port F becomes a 5 V port when a 5 V power supply is connected to DVCC51.

Note: When the DSU is enabled, port 9 functions as the processor probe interfacing signal regardless of the setting of the port 9 control register (P9CR).

The following table lists the JTAG specific pins added to the CSP package:

Pin Name	# of Pins	Type	Function
$\overline{\text{TRST}}$	1	Input	JTAG reset pin (with internal pull-up resistor)
TCK	1	Input	JTAG clock pin (with internal pull-up resistor)
TDI	1	Input	JTAG data input pin (with internal pull-up resistor)
TDO	1	Output	JTAG data output pin
TMS	1	Input	JTAG mode switching input pin (with internal pull-up resistor)

3. Functional Description

This section describes the functions and basic operation of each individual circuit block in the TMP1942 series devices.

3.1 Processor Core

The TX1942 contains a high-performance 32-bit processor core (the TX19 processor core). For details of the operation of the processor core, refer to "TX19 Family Architecture".

Functions unique to the TMP1942, which are not explained in "TX19 Family Architecture", are described below.

Recommended power-on sequence:

In powering up this device, it is recommended that the DVCC3 be turned on first.

At power-on, the pull-up resistors and input & output buffers pull-down resistors attached to

the I/O ports of the 5V supply domain may rail become unstable or a through current may pass

through the port until the DVCC3 has stabilized, when an injection order is not kept.

3.1.1 Reset Operation

To reset the TMP1942, $\overline{\text{RESET}}$ must be input Low (at 0) for at least 12 system clock cycles while the power supply voltage is within the rated operating range and the internal high-frequency oscillator is oscillating stably. (With the device operating at 32 MHz, this period is equal to 3 μs if the PLL is being used and 6 μs if the PLL is not being used.) After a reset the PLL-multiplied clock is specified by the setting of the $\overline{\text{PLLOFF}}$ pin and the clock gear is initialized to 1/8 mode.

To reset the TMP1942, $\overline{\text{RESET}}$ must be asserted for at least 12 system clock periods after the power supply voltage and the internal high-frequency oscillator have stabilized. This time is typically 3 μs at 32 MHz when the on-chip PLL is utilized, and 6μs otherwise. After a reset, either the PLL-multiplied clock or an external clock is selected, depending on the logic state of the $\overline{\text{PLLOFF}}$ pin. By default, the selected clock is geared down to 1/8 for internal operation.

The following occurs as a result of a reset:

- The System control coprocessor (CP0) registers within the TX19 core processor are initialized. For details, refer to the Architecture manual.
- The Reset exception is taken. Program control is transferred to the exception handler at a predefined address. This predefined location is called an exception vector, which directly indicates the start of the actual exception handler routine. The Reset exception is always vectored to virtual address 0xBFC0_0000 (which is the same as for the Nonmaskable Interrupt exception).
- All on-chip I/O peripheral registers are initialized.
- All port pins, including those multiplexed with on-chip peripheral functions, are configured as either general-purpose inputs or general-purpose outputs.

3.2 Memory Map
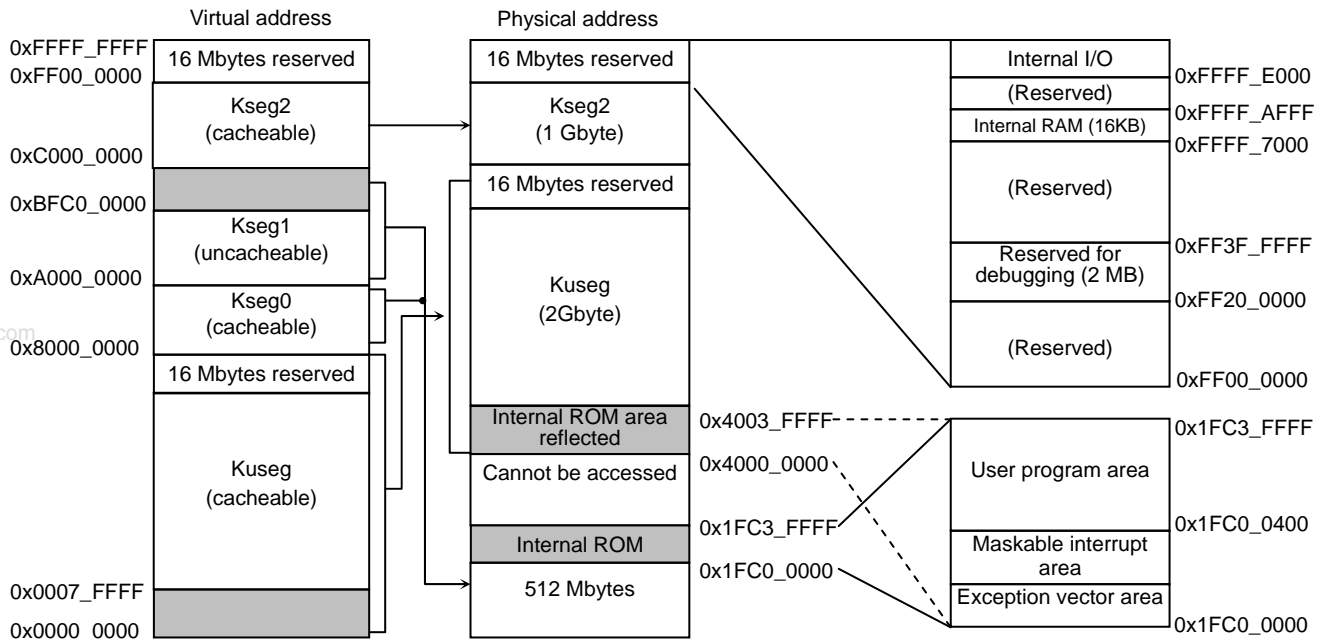
Figure 3.2.1 shows a memory map of the TMP1942.

Figure 3.2.1 Memory Map

Note 1: The internal ROM is mapped into the memory space from 0x1FC0_0000 to 0x1FC3_FFFF (for a 256-KB ROM) or
0x1FC0_0000 to 0x1FC5_FFFF (for a 384-KB ROM). The internal RAM is mapped into the memory space from
0xFFFF_8000 to 0xFFFF_BFFF (for a 16-KB RAM).

Note 2: The memory space from 0xFFFF_4000 to 0xFFFF_BFFF is a reserved RAM area. Any area other than those shown above,
where physical memory is located, should not be accessed.

Note 3: The internal memory data is stored in contiguous physical address locations starting at 0x1FC0_0000.
If exception vector addresses are placed in internal ROM, the system control coprocessor (CP0) Status register's BEV bit
must be set to 1 (the default). (This is because exception vector addresses are dispersed if BEV = 0.) If memory is added
externally, the BEV bit can be set to 0.
However, since a virtual address space of 0x0000_0000 ±32 KB is easier to access for reasons of code efficiency, this area
is reflected in the contiguous physical address space from 0x4000_0000 upwards (as indicated by the shaded area) which
corresponds to a virtual address space starting at 0x0000_0000 and which is equal in size to the internal memory. Hence,
accessing this area is equivalent to accessing the internal memory.

Example: Using 32-bit ISA
- Access to the 0x0000_0000 ±32 KB area
    ADDIU   r2, r0, 7   ; r 2 ← (0x0000_0007)
    SW   r2, Io (_t) (r0) ; 0x0000_xxxx ← (r2)
                    ↑
                Can be accessed using a single instruction.
- Access to areas other than 0x0000_0000 ±32 KB
    LUI   r3, hi (_f)   ; ← Upper address is set to r3.
    ADDIU   r2, r0, 8   ; r2 ← (0x0000_0008)
    SW   r2, Io (_f) (r3) ; Memory is accessed after lower address has been set.

Note 4: The TX1942 supports access to only 16 Mbytes of physical space as external address space. A 16-Mbyte physical address
space can be placed in any chip-select area within the CPU's 3.5 Gbytes of physical address space.
However, when access to the internal memory, internal I/O space or a reserved area is performed, the external address
space cannot be accessed simultaneously, since the other types of access have priority.

Note 5: Do not place an instruction in the last four words of the physical area.
- The relevant area of the internal ROM is 0x1FC3_FFF0 to 0x1FC3_FFFF (for a 256-KB ROM) or 0x1FC5_FFF0 to
0x1FC5_FFFF (for a 384-KB ROM).
- If ROM is added externally, this restriction applies to the last four words of the installed memory
(system-dependent).

3.3   Clock/Standby Control

There are essentially two modes of clock operation: single-clock mode (which uses only the X1 and X2 pins) and dual-clock mode (which uses the X1 and X2 pins as well as the XT1 and XT2 pins).

Figure 3.3.1 shows the state transition diagram for each operation mode.
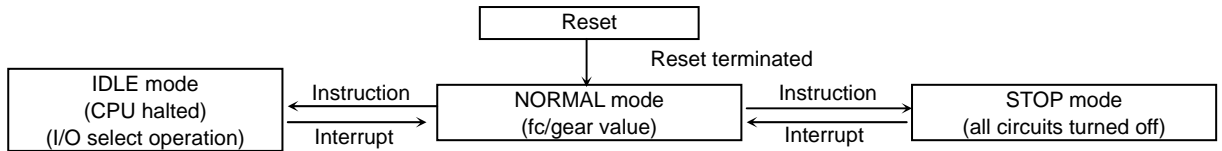
(a) State transition in single-clock mode
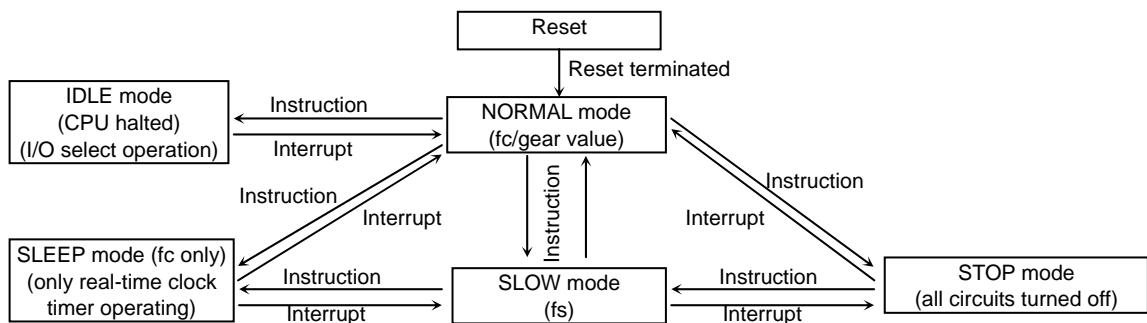
Note 1:   Before transition to SLOW/SLEEP mode can occur, the low-speed oscillator (fs) must be oscillating stably.

Note 2:   When SLEEP mode is terminated, the device returns to the state in which it was placed before entering SLEEP mode.

Note 3:   The state to which the device returns when STOP mode is terminated can be specified using system control register SYSCR0.

(b) State transition in dual-clock mode

Figure 3.3.1 State Transition Diagrams for Different Modes
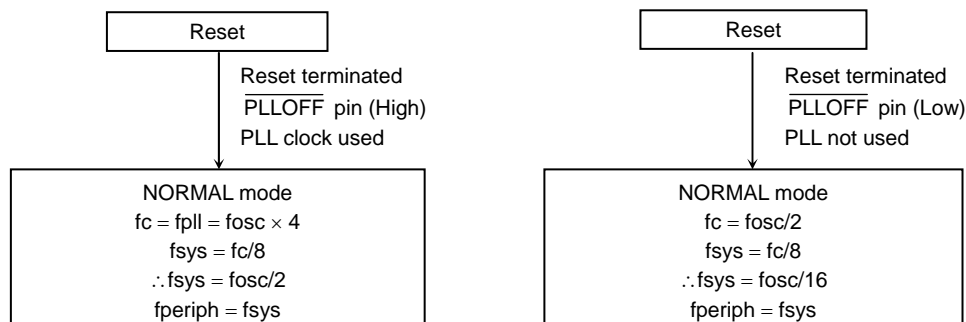
A. When a clock generated by the PLL is used   B. When the PLL is not used

Figure 3.3.32 Default States When the PLL is Used and Those When the PLL is Not Used

fosc:	Clock frequency input via X1 and X2 pins
fs:	Clock frequency input via XT1 and XT2 pins
fpll:	Clock frequency multiplied (x4) by PLL
fc:	Clock frequency selected by setting of $\overline{\text{PLLOFF}}$ pin
fgear:	Clock frequency selected by SYSCR1<GEAR1:GEAR0>
System clock fsys:	Clock frequency selected by SYSCR1<SYSCK>
fperiph:	Input clock for peripheral I/O prescaler

3.3.1 Block Diagram of Clock Circuits

1. Main system clock

   - A crystal can be connected between X1 and X2, or X1 can be externally driven with a clock.

   - $\overline{\text{PLLOFF}}$ The on-chip PLL can be enabled or disabled (bypassed) during reset by using the $\overline{\text{PLLOFF}}$ pin. When the PLL is enabled, the input clock frequency is multiplied by four.

   - The clock gear can be programmed to divide the clock by 2, 4 or 8. (The default is 1/8 on reset.)

   - Input clock frequency

		Input Frequency Range	fmax	fmin
PLLON (for both resonator and external input)		5~8 (MHz)	32 MHz	2.5 MHz
PLLOFF	Resonator	16~20 (MHz)	20 MHz	1 MHz
	External input	16~20 (MHz)	20 MHz	1 MHz
		20~32 (MHz)	16 MHz [*1]	1.25 MHz

*1. SYSCR1<DFOSC> must be 0. The default is 0.

2. Sub-system clock

   - Generated using a 32.768-kHz resonator (external input also accepted).

   - SLOW mode: The CPU runs at low speed.

   - SLEEP mode: Only the timer for real-time clock, 2-phase pulse input counter, and dynamic pull-up operate.

3. Block diagram
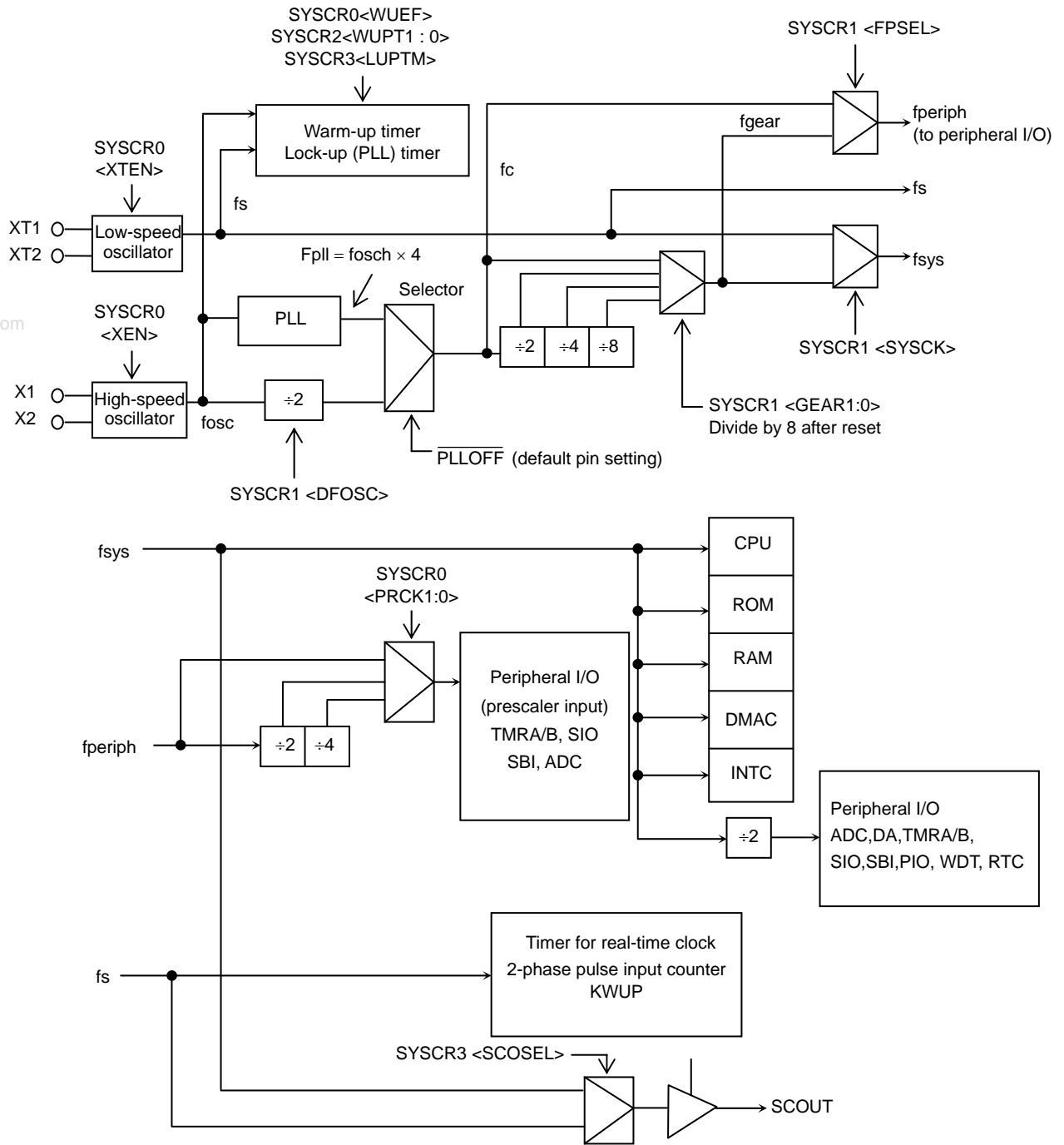
Note 1: When using the clock gear to reduce the system clock frequency, make sure that φTn of the prescaler output for each peripheral I/O block satisfies the following relationship:

   φTn<fsys/2

   To this end, set the clock-related registers so that φTn is slower than fsys/2.
   When selecting a low-speed system clock (fs), only the timer for real-time clock, 2-phase pulse input counter, and dynamic pull-up can operate.

Figure 3.3.3 Block Diagram of Dual-Clock and Standby Functions

3.3.2 Clock Generator (CG) Registers

(1) Clock-related registers

SYSCR0 (0xFFFF_EE00)

	7	6	5	4	3	2	1	0
Bit Symbol	XEN	XTEN	RXEN	RXTEN	RSYSCK	WUEF	PRCK1	PRCK0
Read/Write	R/W							
After reset	1	0	1	0	0	0	0	0
Function	High-speed oscillator 0: Turned off 1: Oscillating	Low-speed oscillator 0: Turned off 1: Oscillating	High-speed oscillator after exit from STOP mode 0: Turned off 1: Oscillating	Low-speed oscillator after exit from STOP mode 0: Turned off 1: Oscillating	Clock selection after exit from STOP mode 0: High speed 1: Low speed	Oscillator warm-up timer (WUP) control Write 0: Don't care Write 1: WUP start Read 0: WUP finished Read 1: WUP operating	Prescaler clock selection 00: fperiph/4 01: fperiph/2 10: fperiph 11: (reserved)	

SYSCR1 (0xFFFF_EE01)

	15	14	13	12	11	10	9	8
Bit Symbol			SYSCK	FPSEL	DFOSC		GEAR1	GEAR0
Read/Write	R/W						R/W	
After reset	-	-	0	0	0	-	1	1
Function			System clock selection 0: High speed (fc) 1: Low speed (fs)	fperiph selection 0: fgear 1: fc	High-speed oscillator frequency division selection 0: Divide by 2 1: Divide by 1		High-speed clock (fc) gear selection 00: fc 01: fc/2 10: fc/4 11: fc/8	

SYSCR2 (0xFFFF_EE02)

	23	22	21	20	19	18	17	16
Bit Symbol	DRVOSCH	DRVOSCL	WUPT1	WUPT0	STBY1	STBY0		DRVE
Read/Write	R/W						-	R/W
After reset	0	0	1	0	1	1	-	0
Function	High-speed oscillator driving capability control 0: Normal 1: Weak	Low-speed oscillator driving capability control 0: Normal 1: Weak	Oscillator warm-up time selection 00: 2^2/input frequency 01: 2^8/input frequency 10: 2^{14}/input frequency 11: 2^{16}/input frequency		Standby mode selection 00: Reserved 01: STOP mode 10: SLEEP mode 11: IDLE mode			1: Pins are also driven in STOP mode.

SYSCR3 (0xFFFF_EE03)

	31	30	29	28	27	26	25	24
Bit Symbol		SCOSEL		ALESEL			LUPFG	LUPTM
Read/Write	-	R/W	-	R/W	-	-	R	R/W
After reset	-	0	-	1	-	-	0	0
Function		SCOUT output selection 0: fs 1: fsys		ALE output width selection 0: fsys × 0.5 1: fsys × 1.5			Lock-up flag 0: LUP finished 1: LUP in operation	Lock-up time selection 0: 2^{16}/input frequency 1: 2^{12}/input frequency

Note 1: Standby mode selection depends on the settings of the Doze and Halt bits in the CP0's internal Config register. If the Halt bit = 1, the device will enter the mode selected by STBY[1:0].

If the Doze bit = 1, the device will always enter IDLE mode.

Note 2: When the PLL is not used, set the LUPTM bit in the SYSCR3 register to 1 (i.e., select 2^{12}/input frequency).

Note3: The WURT1-WUPT0 bitys in the SYSCR2 must be not be changed during the oscillator warm-up event (e.g. SLEEP-NORMAL-SLEEP)

Note 4: Do as follows to change the operating mode immediately after the device has warmed up from the clock stop state (e.g., from SLEEP mode to NORMAL mode to SLEEP mode).

- Warming up by hardware

   (1) Moving from STOP or SLEEP mode to NORMAL mode

      1) When the PLL is used

         Before moving to the next operating mode, ensure that the lock-up bit, LUPFG, in the SYSCR3 register has been cleared to zero and wait for five or more instructions to complete (including the instruction to check the LUPFG flag).

      2) When the PLL is not used

         - When the oscillator warm-up time (SYSCR2<WUPT1:0>) is programmed as "01" (i.e., 2^8/input frequency). Before moving to the next operating mode, ensure that the lock-up bit, LUPFG, in the SYSCR3 register has been cleared to zero and wait for five or more instructions to complete.

         - When the oscillator warm-up time (SYSCR2<WUPT1:0>) is programmed as "10" (2^{14}/input frequency) or "11" (2^{16}/input frequency). Before moving to the next operating mode, wait for five or more instructions to complete.

   (2) Moving from STOP or SLEEP mode to SLOW mode

      It is possible to move to SLOW mode immediately after the device has warmed up from STOP or SLEEP mode.

- Warming up by software

   (1) Moving from SLOW mode to NORMAL mode

      1) When the PLL is used

         It is possible to move to NORMAL mode immediately after the device has warmed up. However, to move to another mode after that, ensure that the lock-up bit, LUPFG, in the SYSCR3 register has been cleared to zero and wait for five or more instructions to complete (including the instruction to check the LUPFG flag).

      2) When the PLL is not used

         - When the oscillator warm-up time (SYSCR2<WUPT1:0>) is programmed as "01" (i.e., 2^8/input frequency). It is possible to move to NORMAL mode immediately after the device has warmed up. However, to move to another mode after that, ensure that the lock-up bit, LUPFG, in the SYSCR3 register has been cleared to zero and wait for five or more instructions to complete.

         - When the oscillator warm-up time (SYSCR2<WUPT1:0>) is programmed as "10" (2^{14}/input frequency) or "11" (2^{16}/input frequency). It is possible to move to NORMAL mode immediately after the device has warmed up. However, to move to another mode after that, wait for five or more instructions to complete.

   (2) Moving from NORMAL mode to SLOW mode

      Before moving to SLOW mode, ensure that the warm-up end flag (i.e., the WUEF bit in the SYSCR0 register) is cleared and wait for five or more instructions to complete.

(2) Standby (STOP/SLEEP mode) termination interrupts

IMCGA0
(0xFFFF_EE10)

	7	6	5	4	3	2	1	0
Bit Symbol			EMCG01	EMCG00				INT0EN
Read/Write	—	—	R/W		—	—	—	R/W
After reset	—	—	1	0	—	—	—	0
Function			Active state setting for INT0 standby termination request 00: Low level 01: High level 10: Falling edge 11: Rising edge					INT0 request input 0: Disable 1: Enable

	15	14	13	12	11	10	9	8
Bit Symbol			EMCG11	EMCG10	DFOSC			INT1EN
Read/Write	—	—	R/W		—	—	—	R/W
After reset	—	—	1	0	—	—	—	0
Function			Active state setting for INT1 standby termination request 00: Low level 01: High level 10: Falling edge 11: Rising edge					INT1 request input 0: Disable 1: Enable

	23	22	21	20	19	18	17	16
Bit Symbol			EMCG21	EMCG20				INT2EN
Read/Write	—	—	R/W		—	—	—	R/W
After reset	—	—	1	0	—	—	—	0
Function			Active state setting for INT2 standby termination request 00: Low level 01: High level 10: Falling edge 11: Rising edge					INT2 request input 0: Disable 1: Enable

	31	30	29	28	27	26	25	24
Bit Symbol			EMCG31	EMCG30				INT3EN
Read/Write	—	—	R/W		—	—	—	R/W
After reset	—	—	1	0	—	—	—	0
Function			Active state setting for INT3 standby termination request 00: Low level 01: High level 10: Falling edge 11: Rising edge					INT3 request input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
IMCGB0	Bit Symbol			EMCG41	EMCG40				INT4EN
(0xFFFF_EE14)	Read/Write	—	—	R/W		—	—	—	R/W
	After reset	—	—	1	0	—	—	—	0
	Function			Active state setting for INT4 standby termination request 00: Low level 01: High level 10: Falling edge 11: Rising edge					INT4 request input 0: Disable 1: Enable
		15	14	13	12	11	10	9	8
	Bit Symbol			EMCG51	EMCG50				KWUPEN
	Read/Write	—	—	R/W		—	—	—	R/W
	After reset	—	—	1	0	—	—	—	0
	Function			These bits should always be set to 01.					KWUP request input 0: Disable 1: Enable
		23	22	21	20	19	18	17	16
	Bit Symbol			EMCG61	EMCG60				INTBCDEEN
	Read/Write	—	—	R/W		—	—	—	R/W
	After reset	—	—	1	0	—	—	—	0
	Function			These bits should always be set to 01.					INTBCDE request input 0: Disable 1: Enable
		31	30	29	28	27	26	25	24
	Bit Symbol			EMCG71	EMCG70				INTRTCEN
	Read/Write	—	—	R/W		—	—	—	R/W
	After reset	—	—	1	0	—	—	—	0
	Function			These bits should always be set to 11.					INTRTCEN request input 0: Disable 1: Enable

Note 1: When enabling an interrupt source as a means of terminating a standby mode, always set the active state for the corresponding interrupt request.

Note 2: When using an interrupt, always perform the following steps in order:
  (1) Enable the input for the interrupt if the corresponding pin is also used for a general-purpose port or any other purpose.
  (2) Set the active state for the interrupt during initialization.
  (3) Clear the interrupt request.
  (4) Enable the interrupt.

Note 3: The TMP1942 has eight interrupt sources (INT0~INT4, INTRTC, INTB/INTC/INTD/INTE, and KWUP0-KWUPD) which can be used as a means of terminating a standby mode. For INT0 to INT4, use the CG block to specify whether they are used to terminate a standby mode and to specify their active edge or level. For INTB/INTC/INTD/INTE and KWUP0-KWUPD, use the CG block to specify whether they are used to terminate a standby mode and use INTBCDEST and KWUPSTn, respectively, to specify their active edge or level. Set the active state for the corresponding interrupt source to High in the INTC block.
Example: Enable the INT0 interrupt

IMCGA0<EMCG01:00> = "10" ⎫ CG block
IMCGA0<INT0EN> = "1" ⎭ (Input is enabled on the falling edge.)
IMC0L<EIM11:10> = "01" ⎫ INTC block
IMC0L<IL12:10> = "101" ⎭ (A High-level interrupt is active and the interrupt level is 5.)

All interrupt sources other than those which are used to terminate STOP/SLEEP mode are set in the INTC circuit block.

Note 4: Among the above eight interrupt sources used to request the termination of a standby mode, INT0 to INT4 do not require settings in the CG block if they are used as normal interrupts. They still, however, require level or edge specification in the INTC. If INTB/INTC/INTD/INTE and KWUP0-KWUPD are used as normal interrupts, specify the active level or edge using INTBCDEST/KWUPSTn and specify the High level in the INTC. Settings in the CG are not required. INTRTC always requires settings in both the CG and INTC even if it is used as a normal interrupt.

All interrupt sources other than those which are used to terminate a standby mode are set in the INTC circuit block.

(3) Interrupt request clear register

	7	6	5	4	3	2	1	0
Bit Symbol						ICRCG2	ICRCG1	ICRCG0
Read/Write	—	—	—	—	—		W	
After reset	—	—	1	0	—	—	—	—
Function						Clear interrupt request 000: INT0 100: INT4 001: INT1 101:KWUP 010: INT2 110: INTB/C/D/E 011: INT3 111: INTRTC		

EICRCG
(0xFFFF_EE20)

Note : To clear any of the eight interrupt sources which are used for terminating a standby mode:
   (1) For KWUP, use KWUPCLR.
   (2) For extended interrupts INTB/INTC/INTD/INTE, use INTFLG.
   (3) For INT0 to INT4 and INTRTC, perform the clearing operation twice, first in the CG block and then in the INTC block.
   (4) For all other interrupt sources, use the INTC block.

3.3.3   System clock control unit

When reset, the device enters single-clock mode with the result that XEN = 1, XTEN = 0 and GEAR1:0 = 11; the system clock fsys is set to fc/8 (= fc × 1/8). (Since the PLL multiplies the original oscillation frequency by 4, fc equals to fosc × 4, where fosc is the original oscillation frequency.) For example, if the X1 and X2 pins are connected to an 8-MHz resonator, a reset will set fsys to 4 MHz (= 8 MHz × 4 × 1/8).

To disable the system from using a PLL-multiplied clock as the system clock by default, drive the $\overline{\text{PLLOFF}}$ pin Low. In this case, too, the system clock fsys will be set to fc/8 (= fc × 1/8) by a reset. However, since SYSCR1<DFOSC> is initialized to 0 by a reset (so that fc = fosc × 1/2), if the X1 and X2 pins are connected to a 25-MHz resonator, fsys will be 1.25 MHz. Also, if the device is clocked by an external oscillator and no internal resonator is connected, fc = fosc can be selected by setting SYSCR1<DFOCS> to 1 after a reset, so that the system clock frequency fsys is twice the frequency obtained with an internal resonator.

(1) Oscillation settling time (switchover between NORMAL and SLOW modes)

If a resonator is connected to the resonator-connecting pins, the device uses the built-in warm-up timer to check whether resonator oscillation has settled. The warm-up time can be set to suit the characteristics of the resonator using SYSCR2<WUPT1:WUPT0>. The value of SYSCR0<WUEF> must be checked in software (using instructions) to determine the start and completion of the warm-up time.

Table 3.3.1 shows warm-up times for mode switching.

> Note 1: Warm-up is unnecessary when the clock generator uses an oscillator so that its oscillation is stable.
>
> Note 2: Since the warm-up timer is clocked by an oscillating clock, it will not be exact if the oscillation frequency fluctuates. The warm-up time should, therefore, be considered to be an approximate value.
>
> Note 3: Before starting the warm-up timer, first confirm that the PLL lock-up flag <LUPFG> is 0.
>
> Note 4: The following precautions must be observed when a low-speed oscillator is being used:
> When a low-speed oscillator is connected to ports PD6 and PD7, the corresponding register must be set as shown below in order to reduce the device's power consumption.
>
> (When using a resonator)
> Set PDCR<PD6C, PD7C> to 11 and PD<PD6, PD7> to 00.
>
> (When using an external clock)
> Set PDCR<PD6C, PD7C> to 11 and PD<PD6, PD7> to 10.

Table 3.3.1 Warm-Up Time

Warm-Up Time Selection SYSCR2<WUPT1:0>	High-Speed Clock (fosc)	Low-Speed Clock (fs)
(2^2/oscillation frequency)	0.5 [μs]	122 [μs]
(2^8/oscillation frequency)	32 [μs]	7.8 [ms]
(2^{14}/oscillation frequency)	2.048 [ms]	500 [ms]
(2^{16}/oscillation frequency)	8.192 [ms]	2000 [ms]

The values calculated are for when
fosc = 8 MHz
and fs = 32.768 kHz.

> Note: When returning from STOP/SLEEP mode to NORMAL or SLOW mode, set the warm-up time to 122 μs or greater beforehand.
>
> Example: If the device will return from SLEEP mode to SLOW mode, set SYSCR2<WUPT1:0> to 00, that is, a warm-up time of 122 μs, before entering SLEEP mode.

(2) Outputting the system clock from a pin

The system clock fsys or fs can be output from the P44/SCOUT pin to an external device. The P44/SCOUT pin can be set to function as the SCOUT pin by setting the registers which relate to port 4 as follows: P4CR<P44C> = 1 and P4FC<P44F> = 1. Use SYSCR3<SCOSEL> to select which clock will be output from this pin.

Table 3.3.2 shows the pin state for each standby mode when the P44/SCOUT pin is set to function as SCOUT.

Table 3.3.2 SCOUT Output State for Each Standby Mode

Mode / SCOUT Selection	NORMAL, SLOW	Standby Mode		
		IDLE	SLEEP	STOP
<SCOSEL> = "0"	Outputs fs clock.	Fixed to 0 or 1		
<SCOSEL> = "1"	Outputs fsys clock.			

> Note: This function does not guarantee a particular phase difference (AC timing) between the internal clock and the system clock output from SCOUT.

(3) Reducing the driving capability of oscillators

If a resonator is connected to the resonator-connecting pins of an oscillator, this function can suppress oscillation noise output from the oscillator, while reducing power consumption by the oscillator.

Setting SYSCR2<DRVOSCH> to 1 causes the driving capability of the high-speed oscillator to degrade (Weak). Similarly, setting SYSCR2<DRVOSCL> to 1 causes the driving capability of the low-speed oscillator to degrade (Weak).

Because both bits are initialized to 0 upon a system reset, both oscillators start oscillating with their normal driving capability (Normal) when the power is turned on. The oscillators must be placed in the Normal state (<DRVOSCL> or <DRVOSCH> = 0) when they start oscillating in any other cases, such as when STOP/SLEEP mode is terminated.

1) Reducing the driving capability of the high-speed oscillator
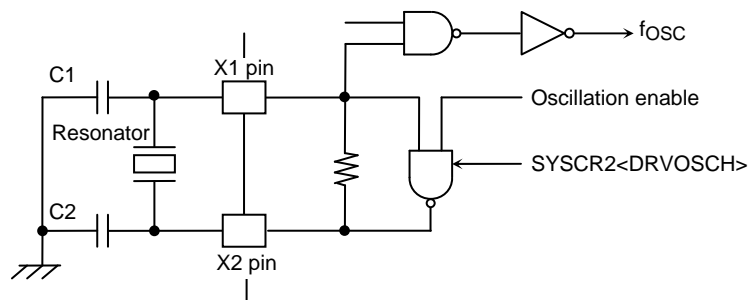
2) Reducing the driving capability of the low-speed oscillator
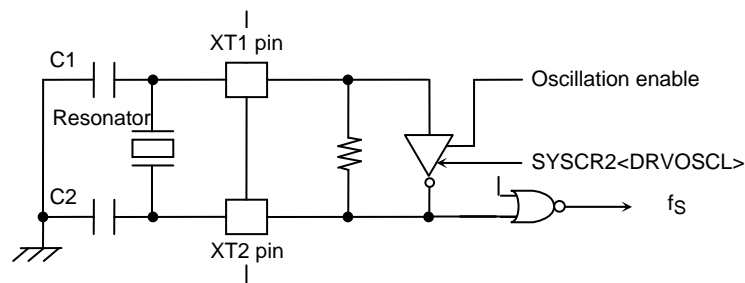
3.3.4   Prescaler clock control unit

The internal I/O blocks (TMRA01 to TMRAAB, TMRB0 to TMRBD, SIO0 to SIO5, SBI, and ADC) each incorporate a prescaler for dividing the clock frequency. The clock $\phi T0$ fed into these prescalers is derived from the clock fperiph. fperiph is either fgear or fc (as specified by the value of SYSCR1<FPSEL>) divided by either 4 or 2, or not divided (as specified by the value of SYSCR0<PRCK1:PRCK0>. By default, fperiph is set to fgear and $\phi T0$ to fperiph/4.

3.3.5   Clock multiplication circuit (PLL)

This circuit multiplies the high-speed oscillator output clock, fosc, by 4 and outputs the result as the clock fpll. This enables the oscillator to yield a fast internal clock with a low oscillator frequency. The PLL is halted by a reset. To use the PLL, hold the $\overline{\text{PLLOFF}}$ pin High when terminating a reset.

Note: If a reset is terminated while the $\overline{\text{PLLOFF}}$ pin is held Low, the PLL will not work and the internal clock chosen will be the original oscillating clock (i.e., it will not be multiplied by 4).

Since the PLL is configured as an analog circuit, it requires a certain settling time (a lock-up time) after it has been activated, as does the oscillator.

The same timer is used for both warm-up and lock-up. The lock-up time must be set using SYSCR3<LUPTM> so that it satisfies the following relationship:

Lock-up time $\geq$ warm-up time

By default, the lock-up time is 2^{16}/input frequency.

The lock-up timer is initiated as the high-speed oscillator starts warm-up, and the lock-up flag SYSCR3<LUPTM> remains 1 until the PLL is locked in phase and cleared to 0 upon the completion of lock-up.

If, for example, the PLL gets out of lock in a standby mode and control which depends on the software's execution speed, such as real-time processing, is to be performed, the software must check the lock-up flag after operation has started (i.e., after warm-up has been completed) to ensure that the clock has settled, before it starts processing.

On the other hand, various hardware settings and static processing, such as register and memory initialization, can be executed before the lock-up flag has been cleared.

> Note: The LUPFG bit is undefined when the $\overline{\text{PLLOFF}}$ pin is Low (the PLL is not used).

Precautions to be observed when switching clock gear:

Clock gear switchover is performed by writing a value to SYSCR1<GEAR1:GEAR0>. The clock gear is not switched immediately after the write: a execution time equal to several clock cycles is required. Therefore, one or more instructions following the clock gear switchover instruction may be executed using the old clock gear value. If these instructions need to be executed using the new clock gear value, insert a dummy instruction (which executes a write cycle only) after the clock gear switchover instruction.

When using a clock gear, make sure that the prescaler output ϕTn in each peripheral I/O block satisfies the following relationship:

ϕTn < fsys/2

For this purpose set the clock-related registers so that ϕTn is slower than fsys/2.

3.3.6   Standby control unit

If the Halt bit in the TX19 processor core's Config register is set in NORMAL mode, the device enters one of the standby modes - IDLE, SLEEP or STOP - as determined by the contents of SYSCR2<STBY1:STBY0>. If the Config register's Doze bit is set, the device enters IDLE mode regardless of the setting of SYSCR2<STBY1:STBY0>.

Features of the IDLE, SLEEP and STOP modes are described below.

1) IDLE: In this mode, only the CPU stops.

In the register corresponding to each module there is an IDLE mode run/stop setup bit for internal I/O. This allows each module to be set independently to run or stop while the device is in IDLE mode. Table 3.3.3 lists the IDLE setup registers available for each internal I/O module.

Table 3.3.3 IDLE Mode Internal I/O Setup Registers

Internal I/O	IDLE Mode Setup Register
TMRA01	TA01RUN<I2TA01>
TMRA23	TA23RUN<I2TA23>
TMRA45	TA45RUN<I2TA45>
TMRA67	TA67RUN<I2TA67>
TMRA89	TA89RUN<I2TA89>
TMRAAB	TAABRUN<I2TAAB>
TMRB0	TB0RUN<I2TB0>
TMRB1	TB1RUN<I2TB1>
TMRB2	TB2RUN<I2TB2>
TMRB3	TB3RUN<I2TB3>
TMRB4	TB4RUN<I2TB4>
TMRB5	TB5RUN<I2TB5>
TMRB6	TB6RUN<I2TB6>
TMRB7	TB7RUN<I2TB7>
TMRB8	TB8RUN<I2TB8>
TMRB9	TB9RUN<I2TB9>
TMRBA	TBARUN<I2TBA>
TMRBB	TBBRUN<I2TBB>
TMRBC	TBCRUN<I2TBC>
TMRBD	TBDRUN<I2TBD>
SIO0	SC0MOD1<I2S0>
SIO1	SC1MOD1<I2S1>
SIO3	SC3MOD1<I2S3>
SIO4	SC3MOD1<I2S4>
SIO5	SC4MOD1<I2S5>
SBI	SBI0BR1<I2SBI0>
A/D converter	ADMOD1<I2AD>
WDT	WDMOD<I2WDT>

Note 1: In Halt mode (entered when the Halt bit in the Config Register is set), the TX19 processor core stops processor operation while maintaining the pipeline status. Since it does not respond to requests for control of the bus from internal DMA, it retains control of the bus.

Note 2: In Doze mode (entered when the Doze bit in the Config Register is set), the TX19 processor core stops processor operation while maintaining the pipeline status. In this mode, it can respond to requests for control of the bus from devices external to the processor core.

2) SLEEP: Only the internal low-speed oscillator, timer for real-time clock, 2-phase pulse input counter, and KWUP (dynamic pull-up) operate.

3) STOP: The CPU runs with the low-speed clock. The INTC, timer for real-time clock, WDT, 2-phase pulse input counter, KWUP (dynamic pull-up), PIO, and EBIF can operate. Operation of other peripheral functions is not guaranteed.

4) SLOW: All of the internal circuits stop.

(1) Operating status in each mode

Table 3.3.4 Operating Status in Each Mode

Operation Mode	Operating Status
NORMAL	The TX19 processor core and peripheral I/O both operate at the maximum frequency.
IDLE (Halt)	The TX19 processor core, INTC, timer for real-time clock, WDT, 2-phase pulse input counter, KWUP (dynamic pull-up), PIO, and EBIF operate with the low-speed clock.
IDLE (Doze)	Processor operation stops and peripheral I/O operates as specified.
SLEEP	Processor operation stops. Only the internal low-speed oscillator, timer for real-time clock, 2-phase pulse input counter, and KWUP (dynamic pull-up) operate (fs).
STOP	Processor and peripheral I/O operation stops completely.

(2) CG operation in each mode

Table 3.3.5 CG Status in Each Operating Mode

Clock Source	Mode	Oscillator	PLL	Clock Supply to Peripheral I/O	Clock Supply to the CPU
Resonator	NORMAL	○	○	○	○
	SLOW	○	×	Partially supplied (Note)	○
	IDLE (Halt)	○	○	Selectable	×
	IDLE (Doze)	○	○	Selectable	×
	SLEEP	fs only	×	Timer for real-time clock, 2-phase pulse input counter, and dynamic pull-up	×
	STOP	×	×	×	×
External input	NORMAL	×	○	○	○
	SLOW	×	×	Partially supplied(Note)	○
	IDLE (Halt)	×	○	Selectable	×
	IDLE (Doze)	×	○	Selectable	×
	SLEEP	×	×	Timer for real-time clock, 2-phase pulse input counter, and dynamic pull-up	×
	STOP	×	×	×	×

Note: This includes the INTC, EBIF (external bus interface), I/O ports, WDT, and timer for real-time clock.

(3) Operation of circuit blocks in each mode (○: Operating, ×: Idle)

Table 3.3.6 Circuit Block Operating Status in Each Mode

Circuit Block	Clock Source	IDLE (Doze)	IDLE (Halt)	SLEEP	STOP
TX19 processor core	fsys	×	×	×	×
DMAC		○	×	×	×
INTC		○	○	×	×
EBIF		○	○	×	×
External bus right		○	○	×	×
PIO		○	×	×	×
DA		○	○	× (*1)	× (*1)
ADC				×	×
SIO				×	×
I2C		Can be selected to run or stop for each module independently.		×	×
Timer counter				×	×
WDT				×	×
2-phase pulse input counter	Fsys/fs			(fs only)	×
Dynamic pull-up	Fs	○	○	○	×
Timer for real-time clock	fs	○	○	○	×
CG	—	○	○	○	×

*1: DAC output is controlled with the OP bit for each channel.

(4) Terminating a standby mode

The device can be freed from a standby mode by an interrupt request or a reset. The combination of the interrupt mask register <CMask15:13> setting and the current standby mode determines which interrupt source will be used to terminate the standby mode. The interrupt mask register is part of the Status register in the TX19 processor core's system control coprocessor (CP0). Details are given in Table 3.3.7.

- Termination by an interrupt request

    The operation performed when the device is released from a standby mode by an interrupt request varies according to the interrupt enable status. If the interrupt level which was set before the device entered the standby mode is greater than or equal to the value in the interrupt mask register, the processor services the requested interrupt after exiting the standby mode and then begins executing instructions starting with the one following the instruction to enter the standby mode (i.e., the instruction which specified the appropriate Config register bit). If the interrupt request level is less than the value in the interrupt mask register, the processor immediately begins executing instructions starting with the one following the instruction to enter the standby mode (i.e., the instruction which specified the appropriate Config register bit) without servicing the requested interrupt. (The interrupt request flag remains 1.)

    Non-maskable interrupts are always serviced after standby mode has terminated, irrespective of the value of the mask register.

- Termination by a reset

    The device can be released from any standby mode by a reset. However, after release from STOP mode, a certain reset time is required for oscillator operation to settle. The reset selects a warm-up time of 2^{14}/oscillation frequency.

    After release by a reset, the internal RAM data can be retained in the state in which it was placed immediately before the standby mode was entered; however, all other settings will be initialized. (After released by an interrupt, other settings are also retained in the state in which they were placed immediately before the standby mode was entered.)

Table 3.3.7 Standby Termination Sources and Standby Termination Operation

Interrupt Acceptance State			Interrupt Enabled (Interrupt level) > (Interrupt mask)			Interrupt Enabled (Interrupt level) ≤ (Interrupt mask)		
Standby mode			IDLE (programmable)	SLEEP	STOP	IDLE (programmable)	SLEEP	STOP
Standby mode termination source	Interrupt	NMI	◎	◎	◎*1	◎	◎	◎
		INTWDT	◎	×	×	◎	×	×
		INT0~4, INTB~E	◎	◎	◎*1	○	○	○*1
		KWUP0~D	◎	◎	◎*1	○	○	○*1
		INTRTC	◎	◎	×	○	○	×
		INT5~A	◎	×	×	○	×	×
		INTTA0~B	◎	×	×	○	×	×
		INTTB0~D	◎	◎ (*2)	×	○	○ (*2)	×
		INTRX0~5, TX0~5	◎	×	×	○	×	×
		INTS2	◎	×	×	○	×	×
		INTAD/ADHP/ADM	◎	×	×	○	×	×
RESET			◎	◎		◎	◎	◎	◎

◎: After exiting the standby mode, the processor starts servicing the interrupt. (RESET initializes the LSI.)
○: After exiting the standby mode, the processor begins executing instructions starting with the one following the instruction to enter the standby mode, without servicing the interrupt.

×: Cannot be used to exit from a standby mode.

*1: The device is actually released from the standby mode after the warm-up time has passed.

*2: Only INTTB2 and INTTB3 can be used when 2-phase pulse input counter mode is selected.

Note 1: When using a level-sensitive interrupt to terminate a standby mode, be sure to hold the level until the processor starts servicing the interrupt. If the level is changed before that time, the interrupt cannot be serviced properly.

Note 2: If the interrupts are disabled in the CPU, use the interrupt controller (INTC) to disable only the interrupts other than those used for terminating standby, before placing the device in any of the standby modes.

---

(5) STOP mode

In STOP mode all internal circuits, including the internal oscillator, stop operating. The pin state in STOP mode varies according to the setting of SYSCR2<DRVE>, as shown in Table 3.3.10. Once released from STOP mode, the device waits for a while (until the warm-up time ends) before starting to output the system clock; the warm-up time is counted by the warm-up counter. This delay is to ensure that the internal oscillator settles properly. After exiting STOP mode the device starts operating according to the settings of SYSCRO<RXEN, RXTEN, RSYSCK>, which select the operating mode (NORMAL mode or SLOW mode) to be entered on exit from STOP mode.

These settings must be made before the instruction to enter standby mode is executed. The warm-up time is determined by the setting of SYSCR2<WUPT1:WUPT0>.

(6) Timing of terminating STOP/SLEEP mode

1) Operation mode transition from Normal through Stop to Normal
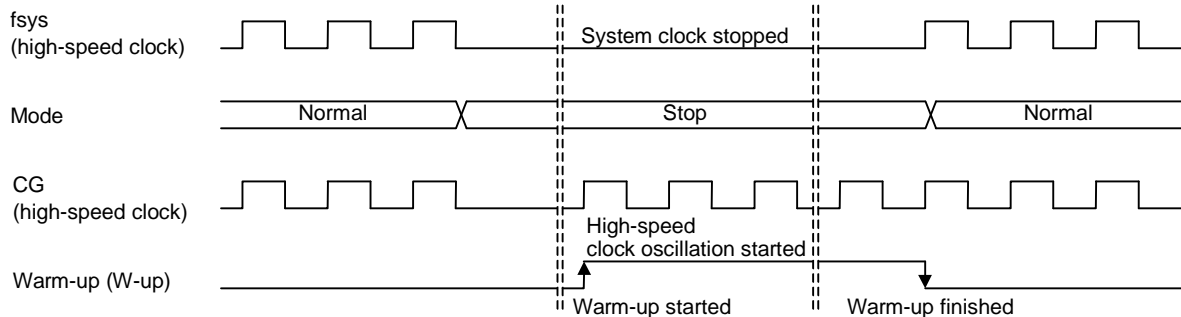
When fosc = 8 MHz

W-up time selection SYSCR2<WUPT1:0>	W-up time (fc)
00(2^2/fosc)	Not allowed
01(2^8/fosc)	Not allowed
10(2^{14}/fosc)	2.048 ms
11(2^{16}/fosc)	8.192 ms

Note: <WUPT1:WUPT0> must not be set to 00 or 01 resuming time requirements for the internal system.

2) Operation mode transition from Normal through Sleep to Normal
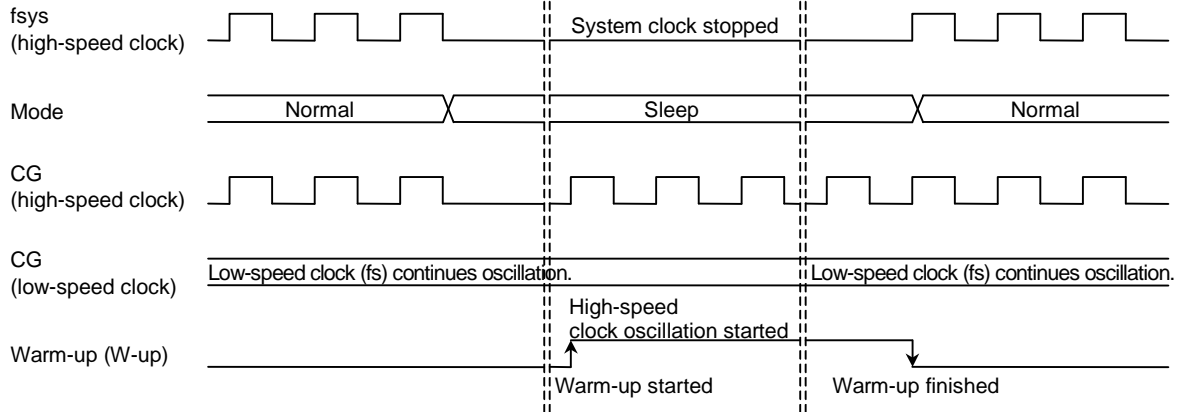
fsys
(high-speed clock) |‾|_|‾|_|‾|_   System clock stopped   _|‾|_|‾|_|‾|_

Mode   ___Normal___X_____Sleep_____X___Normal___

CG
(high-speed clock) |‾|_|‾|_|‾|_   _|‾|_|‾|_|‾|_   _|‾|_|‾|_|‾|_

CG
(low-speed clock)   Low-speed clock (fs) continues oscillation.   Low-speed clock (fs) continues oscillation.

Warm-up (W-up)   High-speed
                                            clock oscillation started
                                         ↑   ↓
                                         Warm-up started   Warm-up finished

When fosc = 8 MHz

W-up time selection SYSCR2<WUPT1:0>	W-up time (fc)
01(2^2/fosc)	Not allowed
01(2^8/fosc)	Not allowed
10(2^{14}/fosc)	2.048 ms
11(2^{16}/fosc)	8.192 ms

Note: <WUPT1:WUPT0> must not be set to 00 or 01
because those settings would not satisfy the
resuming time requirements for the internal system.

3) Operation mode transition from Slow through Stop to Slow

fsys
(low-speed clock) |‾|_|‾|_|‾|_   System clock stopped   _|‾|_|‾|_|‾|_

Mode   ___Slow___X_____Stop_____X___Slow___

CG
(low-speed clock) |‾|_|‾|_|‾|_   _|‾|_|‾|_|‾|_   _|‾|_|‾|_|‾|_

Warm-up (W-up)   Low-speed clock
                                   oscillation started
                                ↑   ↓
                                Warm-up started   Warm-up finished
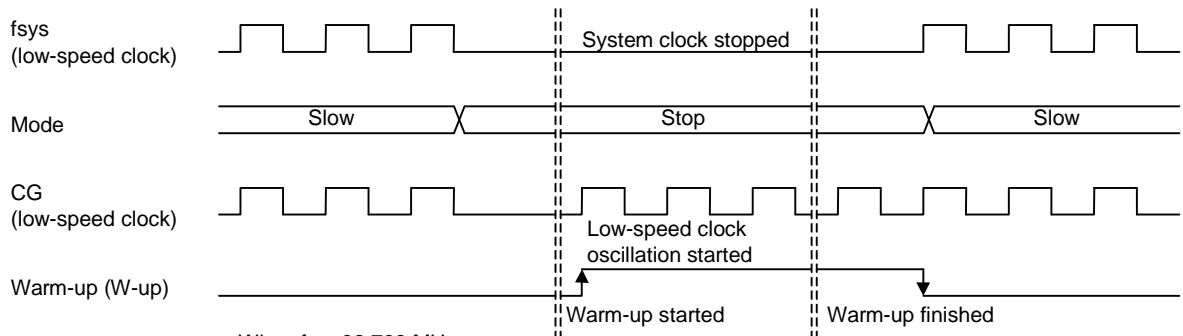
When fs = 32.768 MHz

W-up time selection SYSCR2<WUPT1:0>	W-up time (fc)
01(2^2/fosc)	Not allowed
01(2^8/fosc)	7.8 ms
10(2^{14}/fosc)	500 ms
11(2^{16}/fosc)	2000 ms

4) Operation mode transition from Slow through Sleep to Slow

fsys
(low-speed clock) |‾|_|‾|_|‾|_   System clock stopped   _|‾|_|‾|_|‾|_

Mode   ___Slow___X_____Sleep_____X___Slow___

CG
(low-speed clock) |‾|_|‾|_|‾|_   _|‾|_|‾|_|‾|_   _|‾|_|‾|_|‾|_

Warm-up (W-up)   Low-speed clock
                                   continues oscillation.
                                ↑   ↓
                                Warm-up started   Warm-up finished
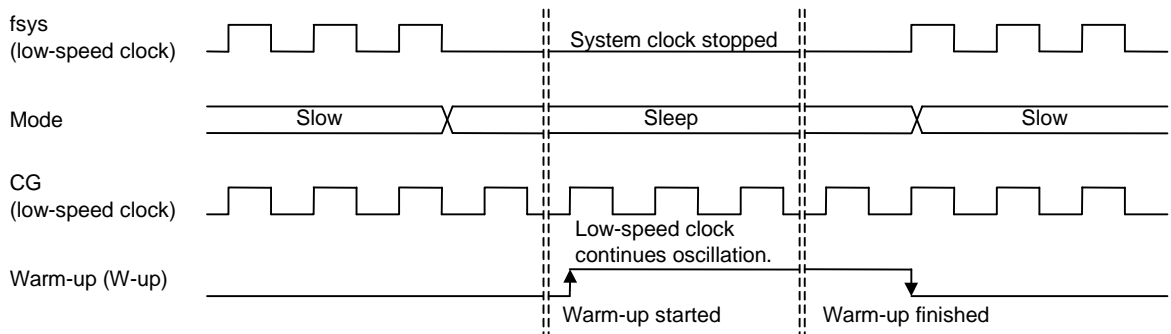
When fs = 32.768 MHz

W-up time selection SYSCR2<WUPT1:0>	W-up time (fc)
01(2^2/fs)	122 μs
01(2^8/fs)	7.8 ms
10(2^{14}/fs)	500 ms
11(2^{16}/fs)	2000 ms

Note: fs continues oscillation but the warm-up time need
be set. Set <WUPT1:WUPT0> to 00.

Table 3.3.8 Pin States in STOP Mode (1/2)

Pins	Input/Output	<DRVE> = 0	<DRVE> = 1
AD0~AD7	Input/Output	-	-
AD8~AD15	Input/Output	-	-
A0~A7/A16~A23	Output	-	Output
$\overline{RD}$ $\overline{WR}$	Output	-	Output
$\overline{WAIT}$, $\overline{BUSRQ}$	Input	PU*	Input
$\overline{HWR}$, $\overline{BUSAK}$,R/$\overline{W}$	Output	PU*	Output
P37	Output mode		
P40~43	Input mode	PU*	Input
	Output mode	PU*	Output
P44 (SCOUT)	Input mode	-	Input
	Output mode	-	Output
P50~57	Input pin	-	-
P60~67	Input pin	-	-
	Input mode(KEY0~KEY7)	Input	Input
P90~P91	Input mode	-	Input
	Output mode	-	Output
	Input mode(INT3,INT4)	Input	Input
P92~97	Input mode	-	Input
	Output mode	-	Output
PA0~PA1	Input mode	-	Input
	Output mode	-	Output
	Input mode(INT3,INT4)	Input	Input
PA2~PA7	Input mode	-	Input
	Output mode	-	Output
PA7	Input mode	-	Input
	Output mode	-	Output
	Input mode(KEYA)	Input	Input
PB1~PB4	Input mode	-	Input
	Output mode	-	Output
	Input mode(INTB~INTE)	Input	Input
PB0,PB5~PB6	Input mode	-	Input
	Output mode	-	Output
PB7	Input mode	-	Input
	Output mode	-	Output
	Input mode(KEYB)	Input	Input
PC0~PC5,PC7	Input mode	-	Input
	Output mode	-	Output
PC6	Input mode	-	Input
	Output mode	-	Output
	Input mode(KEYB)	Input	Input
PD0~PD5	Input mode	-	Input
	Output mode	-	Output
PD6 (XT1)~ PD7 (XT2)	Input mode	-	Input
	Output mode	-	Output
	XT1, XT2	-	-

Pins	Input/Output	<DRVE> = 0	<DRVE> = 1
PE0~PE5	Input mode	-	Input
	Output mode	-	Output
PE6~PE7	Input mode	-	Input
	Output mode	-	Output
	Input mode(INT1,INT2)	Input	Input
PF0,PF2~PF5	Input mode	-	Input
	Output mode	-	Output
PF1	Input mode	-	Input
	Output mode	-	Output
	Input mode(KEYD)	Input	Input
PF6	Input mode	-	Input
	Output mode	-	Output
	Input mode(INT0)	Input	Input
$\overline{\text{NMI}}$	Input pin	Input	Input
ALE	Output pin	Output Low	Output Low
$\overline{\text{RESET}}$	Input pin	Input	Input
BW0, BW1	Input pin	Input	Input
X1	Input pin	-	-
X2	Output pin	Output High	Output High

-:   Pins configured for input mode and input-only pins are disabled. Pins configured for output mode and output-only pins assume the high-Impedance state.

Input: The input gate is active; the input voltage must be held at either the high or low level to keep the input pin from floating.

Output: Pin direction is output.

PU*: Programmable pull-up. Because the input gate is always disabled, no overlap current flows while in high-impedance state.

3.4   Interrupts

Interrupts are controlled by the Status<CMask15:13> and Status<IEc> settings in the CP0 status register, as well as by the internal interrupt controller and the CG. For related information, refer to Section 5, "Exception Handling" in "TX19 Family Architecture".

Interrupts in the TMP1942 have the following features:

- Interrupts from the CPU itself (software interrupt instructions): 4 sources
- External interrupt pins ($\overline{\text{NMI}}$, INT0-INTE, KWUP0-KWUPD): 30 sources
- Interrupts from internal I/O: 46 sources
- Vector generation for each interrupt source
- 7 interrupt priority levels for each source
- Can be used to activate the DMAC

CG

INTnEN standby termination control

8

Detection circuit

Active High level

High or Low level/edge setting

INT0~4

5

8

INTC

8

High level

1

High level

1

Core

High level

7

Other interrupts

Active High level

KWUP

High or Low level/edge setting

KEY0~D

Disable/enable each key input

RTC

Active High level

Extended interrupts

High or Low level/edge setting

INTB~E

Disable/enable input for each interrupt source

Note 1: Standby termination is performed via the CG detection circuit. Since its output is a High-level active signal, the INTC must be set to accept a High-level active signal.

Note 2: The CG is bypassed for any processing other than standby termination. In that case, the active conditions for INT0 to INT4 must be set in the INTC.

Note 3: INTRTC requires CG settings for both standby termination and other processing. The INTC must be set to accept a High-level active signal.

Note 4: KWUP and INTB to INTD require settings in each circuit block for both standby termination and other processing. The INTC must be set to accept a High-level active signal.

Figure 3.44.1 Interrupt Connection Diagram

(1) External interrupts INT0 to INT4, INTB to INTE, KWUP0 to KWUPD, and INTRTC

1) INT0 to INT4

When used to terminate a standby mode, these interrupts must have their active state set (using IMCGxx<EMCGxx>) and must be enabled for input (using IMCGxx<INTxEN>) in the CG block. Then the active state of each of the interrupt source must be set to High (by setting IMCxx<EIMxx> to 01) in the INTC block. When these interrupts are not used to terminate a standby mode, set their active state in the INTC block.

2) INTB to INTE

When used to terminate a standby mode, these interrupts must have their active state set to High (by setting IMCGB2<21:20> to 10) and must be enabled for input (by setting IMCGB2<16> to 1) in the CG block. Then the active state of each of the interrupt source must be set to High (by setting IMCxx<EIMxx> to 01) in the INTC block. Use INTnST for each interrupt source to set the active state and enable or disable the interrupt. When these interrupts are not used to terminate a standby mode, make necessary settings in the INTC block and INTnST without having to make settings in the CG.

3) KWUP0 to KWUPD

When used to terminate a standby mode, these interrupts must have their active state set to High (by setting IMCGB1<21:20> to 10) and must be enabled for input (by setting IMCGB1<16> to 1) in the CG block. Then the active state of each of the interrupt source must be set to High (by setting IMCxx<EIMxx> to 01) in the INTC block. Use KWUPSTn for each interrupt source to set the active state and enable or disable the interrupt. When these interrupts are not used to terminate a standby mode, make necessary settings in the INTC block and KWUPSTn without having to make settings in the CG.

4) INTRTC

Regardless of whether INTRTC is used to terminate a standby mode, this interrupt must have its active state set to a rising edge (by setting IMCGB3<29:28> to 11) and must be enabled for input (by setting IMCGB3<24> to 1) in the CG block. Then the active state of each of the interrupt source must be set to High (by setting IMCxx<EIMxx> to 01) in the INTC block.

(2) External interrupts INT5 to INTA and internal interrupt signals (other than INTRTC)

All these interrupts must be set in the INTC block.

The INTC resolves priority conflicts between interrupt sources and notifies the TX19 processor core of the interrupt with the highest priority.

Interrupt		Register to be Set	Usable Interrupt Detection Level
INT0~INT4, INTRTC*		IMCGx reg.In CG IMCx reg.In INTC	When used to terminate a standby mode, the interrupt source active state must be set to High in the INTC block. The active state of these interrupts must be selected in the CG. However, when these interrupts are not used to terminate a standby mode, their active state must be selected in the INTC block. In both cases, Low level, High level, falling edge and rising edge are all acceptable.
INTB~INTE		IMCGx reg.In CG IMCx reg.In INTC INTnST	The interrupt source active state must always be set to High in the INTC block. When these interrupts are used to terminate a standby mode, the interrupt source active state must also be set to High in the CG. The active state of these interrupts must be selected in INTnST. However, when these interrupts are not used to terminate a standby mode, settings in the CG are not necessary. In both cases, Low level, High level, falling edge and rising edge are all acceptable.
KWUP0~D		IMCGx reg.In CG IMCx reg.In INTC KWUPSTn	The interrupt source active state must always be set to High in the INTC block. When these interrupts are used to terminate a standby mode, the interrupt source active state must also be set to High in the CG. The active state of these interrupts must be selected in KWUPSTn. However, when these interrupts are not used to terminate a standby mode, settings in the CG are not necessary. In both cases, Low level, High level, falling edge and rising edge are all acceptable.
INT5~INTA		IMCx reg.In INTC	Low level, High level, falling edge and rising edge are all acceptable in the INTC.
Internal I/O	INTDMAn	IMCx reg.In INTC	Falling edge
	Others	IMCx reg.In INTC	Rising edge

Note 1: Interrupt level 0 indicates that the corresponding interrupt is disabled.

Note 2: Only a rising edge can be used for INTRTC.

- Example interrupt settings

  When INT0 is used to request the termination of STOP/SLEEP mode (falling edge)

  a. Enabling the interrupt

  IMCGA0<EMCG01:00> = "10"   : Select falling edge for INT0 ⎫

  EICRCG<ICRCG2:0> = "000"   : Clear interrupt request for INT0 ⎬ CG block

  IMCGA0<INT0EN> = "1"   : Enable request input for INT0 ⎭

  IMC0L<EIM11:10> = "01"   : Select High level for INT0 ⎫

  INTCLR<EICLR5:0> = "000001"   : Clear interrupt request for INT0 ⎬ INTC block

  IMC0L<IL12:10> = "101"   : Set interrupt level to 5 ⎭

  Status<IEc> = "1", <CMask> = "xxx"   TX19 processor core

  b. Disabling the interrupt

  Status<IEc> = "0"   TX19 processor core

  IMC0L<IL12:10> = "000"   : Disable interrupt for INT0 ⎫ INTC block

  INTCLR<EICLR5:0> = "000001"   : Clear interrupt request for INT0 ⎭

  IMCGA0<INT0EN> = "0"   : Disable request input for INT0 ⎫ CG block

  EICRCG<ICRCG2:0> = "000"   : Clear interrupt request for INT0 ⎭

3.4.1   Interrupt sources

(1) Reset and non-maskable interrupts: $\overline{RESET}$, $\overline{NMI}$ and INTWDT (watchdog timer interrupt)

　　　Vector address: 0xBFC0_0000 (virtual address)

(2) Maskable interrupts: Software and hardware interrupts

　　　Vector addresses: 0xBFC0_0210 (virtual address) to 0xBFC0_0260 (virtual address)

Interrupt Source		Vector Address (virtual address)
Reset		0xBFC0_0000
Non-maskable		
Maskable	Software Swi0	0xBFC0_0210
	Swi1	0xBFC0_0220
	Swi2	0xBFC0_0230
	Swi3	0xBFC0_0240
	Hardware	0xBFC0_0260

Note 1: When vector addresses are located in the on-chip ROM, set the BEV bit in the system control coprocessor (CP0) Status register to 1.

Note 2: Maskable software interrupts are generated by setting <Sw3:Sw0> in CP0 Cause register. Do not confuse these software interrupts with Software Set, which is one of the hardware interrupt sources. The Software Set interrupt is generated by setting <IL02:IL00> in the interrupt controller (INTC) IMC0 register to any value other than 0.

Table 3.44.1 Hardware Interrupt Sources

Interrupt Number	IVR[9 : 0]	Interrupt Source	Interrupt control register	Address
0	000	Software Set	IMC0L	0xFFFF_E000
1	010	INT0 pin (standby termination)		
2	020	INT1 pin (standby termination)	IMC0H	0xFFFF_E002
3	030	INT2 pin (standby termination)		
4	040	INT3 pin (standby termination)	IMC1L	0xFFFF_E004
5	050	INT4 pin (standby termination)		
6	060	KWUP (standby termination)	IMC1H	0xFFFF_E006
7	070	INTB/C/D/E pin (standby termination)		
8	080	Reserved	IMC2L	0xFFFF_E008
9	090	Reserved		
10	0A0	INT5 pin	IMC2H	0xFFFF_E00A
11	0B0	INT6 pin		
12	0C0	INT7 pin	IMC3L	0xFFFF_E00C
13	0D0	INT8 pin		
14	0E0	INT9 pin	IMC3H	0xFFFF_E00E
15	0F0	INTA pin		
16	100	INTRX0: Serial reception (channel 0)	IMC4L	0xFFFF_E010
17	110	INTTX0: Serial transmission (channel 0)		
18	120	INTRX1: Serial reception (channel 1)	IMC4H	0xFFFF_E012
19	130	INTTX1: Serial transmission (channel 1)		
20	140	INTS2: Serial channel 2 interrupt	IMC5L	0xFFFF_E014
21	150	INTRX3: Serial reception (channel 3)		
22	160	INTTX3: Serial transmission (channel 3)	IMC5H	0xFFFF_E016
23	170	INTADHP: Highest-priority A/D conversion completed		
24	180	INTADM: A/D conversion monitor interrupt	IMC6L	0xFFFF_E018
25	190	INTTA0: 8-bit timer 0		
26	1A0	INTTA1: 8-bit timer 1	IMC6H	0xFFFF_E01A
27	1B0	INTTA2: 8-bit timer 2		
28	1C0	INTTA3: 8-bit timer 3	IMC7L	0xFFFF_E01C
29	1D0	INTTB0: 16-bit timer 0		
30	1E0	INTTB1: 16-bit timer 1	IMC7H	0xFFFF_E01E
31	1F0	INTRX4: Serial reception (channel 4)		
32	200	INTTX4: Serial transmission (channel 4)	IMC8L	0xFFFF_E020
33	210	INTRX5: Serial reception (channel 5)		
34	220	INTTX5: Serial transmission (channel 5)	IMC8H	0xFFFF_E022
35	230	Reserved		
36	240	Reserved	IMC9L	0xFFFF_E024
37	250	INTTA4: 8-bit timer 4		
38	260	INTTA5: 8-bit timer 5	IMC9H	0xFFFF_E026
39	270	INTTA6: 8-bit timer 6		
40	280	INTTA7: 8-bit timer 7	IMCAL	0xFFFF_E028
41	290	INTTA8: 8-bit timer 8		
42	2A0	INTTA9: 8-bit timer 9	IMCAH	0xFFFF_E02A
43	2B0	INTTAA: 8-bit timer A		
44	2C0	INTTAB: 8-bit timer B	IMCBL	0xFFFF_E02C
45	2D0	INTTBA: 16-bit timer A		
46	2E0	INTTBB: 16-bit timer B	IMCBH	0xFFFF_E02E
47	2F0	INTTBC: 16-bit timer C		
48	300	INTTBD: 16-bit timer D	IMCCL	0xFFFF_E030
49	310	INTTB2: 16-bit timer 2		
50	320	INTTB3: 16-bit timer 3	IMCCH	0xFFFF_E032
51	330	INTTB4: 16-bit timer 4		
52	340	INTTB5: 16-bit timer 5	IMCDL	0xFFFF_E034
53	350	INTTB6: 16-bit timer 6		
54	360	INTTB7: 16-bit timer 7	IMCDH	0xFFFF_E036
55	370	INTTB8: 16-bit timer 8		
56	380	INTTB9: 16-bit timer 9	IMCEL	0xFFFF_E038
57	390	Reserved		
58	3A0	INTRTC: Interrupt from timer for real-time clock	IMCEH	0xFFFF_E03A
59	3B0	INTAD: A/D conversion completed		
60	3C0	INTDMA0: DMA transfer completed (channel 0)	IMCFL	0xFFFF_E03C
61	3D0	INTDMA1: DMA transfer completed (channel 1)		
62	3E0	INTDMA2: DMA transfer completed (channel 2)	IMCFH	0xFFFF_E03E
63	3F0	INTDMA3: DMA transfer completed (channel 3)		

3.4.2 Interrupt detection

When using interrupts to terminate a standby mode, the following settings are necessary according to the interrupt type: Interrupts INT0 to INT4 have their active state set using the EMCGxx field in the CG's internal IMCGxx register, then the EIMxx field in the INTC's internal IMCx register is set to High. Extended interrupts INTB to INTE have their active state set to High using the EMCG field in the CG's internal IMCGB2 register, then the EIMxx field in the INTC's internal IMCx register is set to High. In addition, INTnST is used to set the active state for each interrupt source and enable/disable the interrupt source. KWUP0 to KWUPD have their active state set to High using the EMCG field in the CG's internal IMCGB1 register, then the EIMxx field in the INTC's internal IMCx register is set to High. In addition, KWUPSTn is used to set the active state for each interrupt source and enable/disable the interrupt source.

The RTC interrupt has its active state set to a rising edge using the EMCGxx field in the CG's internal IMCGxx register, then the EIMxx field in the INTC's internal IMCx register is set to High. Other interrupts have their active state set using only the EIMxx field in the INTC's internal IMCx register. The active state can be one of the following four: rising edge, falling edge, High level or Low level. When the TMP1942 detection circuit recognizes the active state of an interrupt request set in this way, it notifies the processor core or the INTC of the interrupt request. When the above interrupts are not used to terminate a standby mode, settings in the CG are not required: INT0 to INT4 require only settings in the INTC, INTB to INTE require the same settings in the INTC as for standby termination as well as setting in INTnST, and KWUP0 to KWUPD require the same settings in the INTC as for standby termination as well as setting in KWUPSTn.

Cancellation of interrupt signals is carried out by the interrupt handler after it has recognized the requested interrupt.

INTB to INTE are canceled by reading INTFLG.

Interrupt signals from INT0 to INT4 and INTRTC are cancelled by writing the appropriate value to the ICRCG field in the CG's internal EICRCG register and then writing the corresponding value to the EICLR field in the INTC's internal INTCLR register. KWUP0 to KWUPD are canceled by setting KWUPCLR. Other interrupt signals are canceled by writing the appropriate value to the EICLR field in the INTC's internal INTCLR register. These cancellation procedures apply regardless of whether the active state is an edge or level.

Figure 3.44.2 Flow for Setting External Interrupts
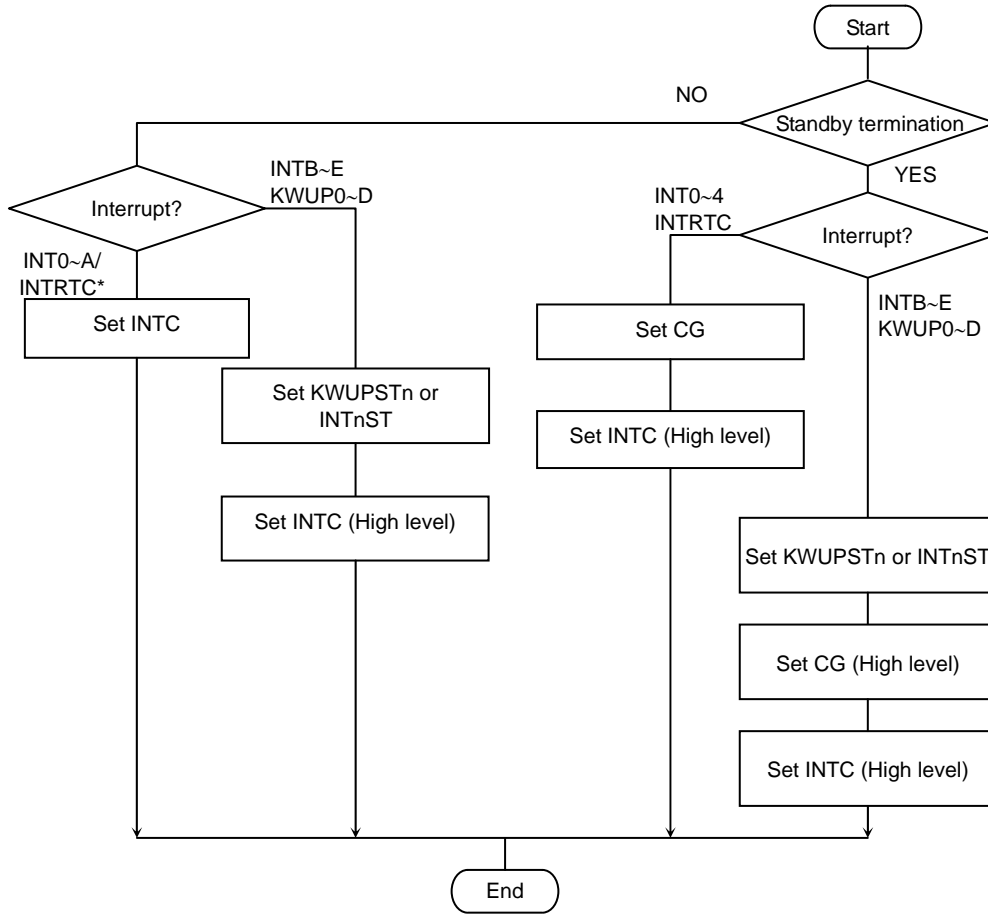
*   The INTRTC interrupt must have its active state set to a rising edge in the CG even when it is not used for standby termination.

Figure 3.44.2 Flow for Setting External Interrupts

Note: Each stage must be completed in the following sequence: set the active level, clear the interrupt request, and then enable the interrupt.

(Example of setting INT0 for standby termination)

IMCGA0<EMCG01:00> = "10"	: Select falling edge for INT0	} G block
EICRCG<ICRCG2:0> = "000"	: Clear interrupt request for INT0C	
IMCGA0<INT0EN> = "1"	: Enable request input for INT0	
IMC0L<EIM11:10> = "01"	: Select High level for INT0	} INTC block
INTCLR<EICLR5:0> = "000001"	: Clear interrupt request for INT0	
IMC0L<IL12:10> = "101"	: Set interrupt level to 5	
Status<IEc> = "1", <CMask> = "xxx"		TX19 processor core

3.4.3   Resolving interrupt priority

(1)   Seven interrupt priority levels

The TMP1942 has seven interrupt priority levels; thus for each interrupt source the priority can be set to one of seven levels.

The interrupt mode control register (IMCx) is used for setting interrupt levels. This register includes a 3-bit level-setting field (ILx). The greater the value (interrupt level) set in IMC<ILx2:ILx0>, the higher the interrupt priority. If the value set for an interrupt source in this field is 000 (i.e., the interrupt level is set to 0), no interrupt is generated for that interrupt source.

(2)   Notification of the interrupt level

When an interrupt occurs, the INTC notifies the TX19 processor core of the priority level of the interrupt. The TX19 processor core recognizes the interrupt level by reading the IL field in the Cause register. If multiple interrupts (with different priority levels) occur simultaneously, the TX19 processor core is notified of the interrupt with the highest priority.

(3)   Interrupt vector (notification of interrupt source)

When an interrupt occurs, the INTC also sets the vector for the source of the generated interrupt in the vector register (IVR). The TX19 processor core reads the vector register to determine the interrupt source. If multiple interrupts (with the same priority level) occur simultaneously, the TX19 processor core is notified of the vector for the interrupt source with the smallest request number. When there are no interrupt sources for which an interrupt has occurred, the IVR[9:4] field is 0.

When it is time for the TX19 processor core to read the vector register value, the INTC notifies the processor core. The processor core sets the Status<CMask> bit with the interrupt level which it reads.

3.4.4   INTC registers

Table 3.44.2 INTC Register Map

Address	Register Symbol	Register	Corresponding Interrupt Number
0xFFFF_E060	INTCLR	Interrupt request clear control	ALL (63 − 0)
0xFFFF_E040	IVR	Interrupt vector register	ALL (63 − 0)
0xFFFF_E03C	IMCF	Interrupt mode control register F	63 − 60
0xFFFF_E038	IMCE	Interrupt mode control register E	59 − 56
0xFFFF_E034	IMCD	Interrupt mode control register D	55 − 52
0xFFFF_E030	IMCC	Interrupt mode control register C	51 − 48
0xFFFF_E02C	IMCB	Interrupt mode control register B	47 − 44
0xFFFF_E028	IMCA	Interrupt mode control register A	43 − 40
0xFFFF_E024	IMC9	Interrupt mode control register 9	39 − 36
0xFFFF_E020	IMC8	Interrupt mode control register 8	35 − 32
0xFFFF_E01C	IMC7	Interrupt mode control register 7	31 − 28
0xFFFF_E018	IMC6	Interrupt mode control register 6	27 − 24
0xFFFF_E014	IMC5	Interrupt mode control register 5	23 − 20
0xFFFF_E010	IMC4	Interrupt mode control register 4	19 − 16
0xFFFF_E00C	IMC3	Interrupt mode control register 3	15 − 12
0xFFFF_E008	IMC2	Interrupt mode control register 2	11 − 8
0xFFFF_E004	IMC1	Interrupt mode control register 1	7 − 4
0xFFFF_E000	IMC0	Interrupt mode control register 0	3 − 0

Interrupt vector register (IVR): Indicates the vector for the source of each interrupt generated.

IVR (0xFFFF_E040)

	7	6	5	4	3	2	1	0
Bit Symbol	IVR7	IVR6	IVR5	IVR4				
Read/Write	R							
After reset	0	0	0	0	0	0	0	0
Function	Indicates the vectors for generated interrupt sources.							
	15	14	13	12	11	10	9	8
Bit Symbol							IVR9	IVR8
Read/Write	R/W						R	
After reset	0	0	0	0	0	0	0	0
Function							Indicates the vectors for generated interrupt sources.	
	23	22	21	20	19	18	17	16
Bit Symbol								
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								
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function								

Interrupt mode control registers: Set the priority level and active state for each interrupt source and set whether the interrupt is to be used to activate the DMAC.

IMC0
(0xFFFF_E000)

	7	6	5	4	3	2	1	0
Bit Symbol			EIM01	EIM00	DM0	IL02	IL01	IL00
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Sets the active state of the interrupt request. 00: Low level Other settings are not allowed.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 0 to activate the DMAC.	Sets the priority level for interrupt number 0 (Software Set) when DM0 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM0 = 1. 000-011: 0 to 3 100-111: Invalid settings		

	15	14	13	12	11	10	9	8
Bit Symbol			EIM11	EIM10	DM1	IL12	IL11	IL10
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Sets the active state of the interrupt request. 00: Low level 01: High level 10: Falling edge 11: Rising edge		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 1 to activate the DMAC.	Sets the priority level for interrupt number 1 (INT0) when DM1 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM1 = 1. 000-011: 0 to 3 100-111: Invalid settings		

	23	22	21	20	19	18	17	16
Bit Symbol			EIM21	EIM20	DM2	IL22	IL21	IL20
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Sets the active state of the interrupt request. 00: Low level 01: High level 10: Falling edge 11: Rising edge		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 2 to activate the DMAC.	Sets the priority level for interrupt number 2 (INT1) when DM2 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM2 = 1. 000-011: 0 to 3 100-111: Invalid settings		

	31	30	29	28	27	26	25	24
Bit Symbol			EIM31	EIM30	DM3	IL32	IL31	IL30
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Sets the active state of the interrupt request. 00: Low level 01: High level 10: Falling edge 11: Rising edge		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 3 to activate the DMAC.	Sets the priority level for interrupt number 3 (INT2) when DM3 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM3 = 1. 000-011: 0 to 3 100-111: Invalid settings		

Note : Before enabling the above interrupt requests, be sure to set their active state.

IMC1
(0xFFFF_E004)

	7	6	5	4	3	2	1	0
Bit Symbol			EIM41	EIM40	DM4	IL42	IL41	IL40
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Sets the active state of the interrupt request. 00: Low level 01: High level 10: Falling edge 11: Rising edge		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 4 to activate the DMAC.	Sets the priority level for interrupt number 4 (INT3) when DM4 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM4 = 1. 000-011: 0 to 3 100-111: Invalid settings		

	15	14	13	12	11	10	9	8
Bit Symbol			EIM51	EIM50	DM5	IL52	IL51	IL50
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Sets the active state of the interrupt request. 00: Low level 01: High level 10: Falling edge 11: Rising edge		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 5 to activate the DMAC.	Sets the priority level for interrupt number 5 (INT4) when DM5 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM5 = 1. 000-011: 0 to 3 100-111: Invalid settings		

	23	22	21	20	19	18	17	16
Bit Symbol			EIM61	EIM60	DM6	IL62	IL61	IL60
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Sets the active state of the interrupt request. 00: Low level 01: High level 10: Falling edge 11: Rising edge		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 6 to activate the DMAC.	Sets the priority level for interrupt number 6 (KWUP) when DM6 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM6 = 1. 000-011: 0 to 3 100-111: Invalid settings		

	31	30	29	28	27	26	25	24
Bit Symbol			EIM71	EIM70	DM7	IL72	IL71	IL70
Read/Write			R/W					
After reset		0	0	0	0	0	0	0
Function			Sets the active state of the interrupt request. 00: Low level 01: High level 10: Falling edge 11: Rising edge		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 7 to activate the DMAC.	Sets the priority level for interrupt number 7 (INTB/C/D/E) when DM7 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM7 = 1. 000-011: 0 to 3 100-111: Invalid settings		

Note : Before enabling the above interrupt requests, be sure to set their active state.

IMC2
(0xFFFF_E008)

	7	6	5	4	3	2	1	0
Bit Symbol			EIM81	EIM80	DM8	IL82	IL81	IL80
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 00.		Must be set to 0.	Must be set to 000.		

	15	14	13	12	11	10	9	8
Bit Symbol			EIM91	EIM90	DM9	IL92	IL91	IL90
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 00.		Must be set to 0.	Must be set to 000.		

	23	22	21	20	19	18	17	16
Bit Symbol			EIMA1	EIMA0	DMA	ILA2	ILA1	ILA0
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Sets the active state of the interrupt request. 00: Low level 01: High level 10: Falling edge 11: Rising edge		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 10 to activate the DMAC.	Sets the priority level for interrupt number 10 (INT5) when DMA = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DMA = 1. 000-011: 0 to 3 100-111: Invalid settings		

	31	30	29	28	27	26	25	24
Bit Symbol			EIMB1	EIMB0	DMB	ILB2	ILB1	ILB0
Read/Write			R/W					
After reset		0	0	0	0	0	0	0
Function			Sets the active state of the interrupt request. 00: Low level 01: High level 10: Falling edge 11: Rising edge		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 11 to activate the DMAC.	Sets the priority level for interrupt number 11 (INT6) when DMB = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DMB = 1. 000-011: 0 to 3 100-111: Invalid settings		

Note :　Before enabling the above interrupt requests, be sure to set their active state.

	7	6	5	4	3	2	1	0
Bit Symbol			EIMC1	EIMC0	DMC	ILC2	ILC1	ILC0
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Sets the active state of the interrupt request. 00: Low level 01: High level 10: Falling edge 11: Rising edge		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 12 to activate the DMAC.	Sets the priority level for interrupt number 12 (INT7) when DMC = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DMC = 1. 000-011: 0 to 3 100-111: Invalid settings		

IMC3 (0xFFFF_E00C)

	15	14	13	12	11	10	9	8
Bit Symbol			EIMD1	EIMD0	DMD	ILD2	ILD1	ILD0
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Sets the active state of the interrupt request. 00: Low level 01: High level 10: Falling edge 11: Rising edge		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 13 to activate the DMAC.	Sets the priority level for interrupt number 13 (INT8) when DMD = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DMD = 1. 000-011: 0 to 3 100-111: Invalid settings		

	23	22	21	20	19	18	17	16
Bit Symbol			EIME1	EIME0	DME	ILE2	ILE1	ILE0
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Sets the active state of the interrupt request. 00: Low level 01: High level 10: Falling edge 11: Rising edge		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 14 to activate the DMAC.	Sets the priority level for interrupt number 14 (INT9) when DME = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DME = 1. 000-011: 0 to 3 100-111: Invalid settings		

	31	30	29	28	27	26	25	24
Bit Symbol			EIMF1	EIMF0	DMF	ILF2	ILF1	ILF0
Read/Write			R/W					
After reset		0	0	0	0	0	0	0
Function			Sets the active state of the interrupt request. 00: Low level 01: High level 10: Falling edge 11: Rising edge		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 15 to activate the DMAC.	Sets the priority level for interrupt number 15 (INTA) when DMF = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DMF = 1. 000-011: 0 to 3 100-111: Invalid settings		

Note : Before enabling the above interrupt requests, be sure to set their active state.

IMC4
(0xFFFF_E010)

	7	6	5	4	3	2	1	0
Bit Symbol			EIM101	EIM100	DM10	IL102	IL101	IL100
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 16 to activate the DMAC.	Sets the priority level for interrupt number 16 (INTRX0) when DM10 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM10 = 1. 000-011: 0 to 3 100-111: Invalid settings		

	15	14	13	12	11	10	9	8
Bit Symbol			EIM111	EIM110	DM11	IL112	IL111	IL110
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 17 to activate the DMAC.	Sets the priority level for interrupt number 16 (INTTX0) when DM11 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM11 = 1. 000-011: 0 to 3 100-111: Invalid settings		

	23	22	21	20	19	18	17	16
Bit Symbol			EIM121	EIM120	DM12	IL122	IL121	IL120
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 18 to activate the DMAC.	Sets the priority level for interrupt number 18 (INTRX1) when DM12 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM12 = 1. 000-011: 0 to 3 100-111: Invalid settings		

	31	30	29	28	27	26	25	24
Bit Symbol			EIM131	EIM130	DM13	IL132	IL131	IL130
Read/Write			R/W					
After reset		0	0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 19 to activate the DMAC.	Sets the priority level for interrupt number 19 (INTTX1) when DM13 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM13 = 1. 000-011: 0 to 3 100-111: Invalid settings		

Note : Before enabling the above interrupt requests, be sure to set their active state.

		7	6	5	4	3	2	1	0
IMC5 (0xFFFF_E014)	Bit Symbol			EIM141	EIM140	DM14	IL142	IL141	IL140
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 20 to activate the DMAC.	Sets the priority level for interrupt number 20 (INTS2) when DM14 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM14 = 1. 000-011: 0 to 3 100-111: Invalid settings		

		15	14	13	12	11	10	9	8
	Bit Symbol			EIM151	EIM150	DM15	IL152	IL151	IL150
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 21 to activate the DMAC.	Sets the priority level for interrupt number 21 (INTRX3) when DM15 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM15 = 1. 000-011: 0 to 3 100-111: Invalid settings		

		23	22	21	20	19	18	17	16
	Bit Symbol			EIM161	EIM160	DM16	IL162	IL161	IL160
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 22 to activate the DMAC.	Sets the priority level for interrupt number 22 (INTTX3) when DM16 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM16 = 1. 000-011: 0 to 3 100-111: Invalid settings		

		31	30	29	28	27	26	25	24
	Bit Symbol			EIM171	EIM170	DM17	IL172	IL171	IL170
	Read/Write			R/W					
	After reset		0	0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 23 to activate the DMAC.	Sets the priority level for interrupt number 23 (INTADHP) when DM17 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM17 = 1. 000-011: 0 to 3 100-111: Invalid settings		

Note : Before enabling the above interrupt requests, be sure to set their active state.

		7	6	5	4	3	2	1	0
IMC6 (0xFFFF_E018)	Bit Symbol			EIM181	EIM180	DM18	IL182	IL181	IL180
	Read/Write					R/W			
	After reset			0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 24 to activate the DMAC.	Sets the priority level for interrupt number 24 (INTADM) when DM18 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM18 = 1. 000-011: 0 to 3 100-111: Invalid settings		

		15	14	13	12	11	10	9	8
	Bit Symbol			EIM191	EIM190	DM19	IL192	IL191	IL190
	Read/Write					R/W			
	After reset			0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 25 to activate the DMAC.	Sets the priority level for interrupt number 25 (INTTA0) when DM19 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM19 = 1. 000-011: 0 to 3 100-111: Invalid settings		

		23	22	21	20	19	18	17	16
	Bit Symbol			EIM1A1	EIM1A0	DM1A	IL1A2	IL1A1	IL1A0
	Read/Write					R/W			
	After reset			0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 26 to activate the DMAC.	Sets the priority level for interrupt number 26 (INTTA1) when DM1A = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM1A = 1. 000-011: 0 to 3 100-111: Invalid settings		

		31	30	29	28	27	26	25	24
	Bit Symbol			EIM1B1	EIM1B0	DM1B	IL1B2	IL1B1	IL1B0
	Read/Write					R/W			
	After reset		0	0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 27 to activate the DMAC.	Sets the priority level for interrupt number 27 (INTTA2) when DM1B = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM1B = 1. 000-011: 0 to 3 100-111: Invalid settings		

Note : Before enabling the above interrupt requests, be sure to set their active state.

IMC7
(0xFFFF_E01C)

	7	6	5	4	3	2	1	0
Bit Symbol			EIM1C1	EIM1C0	DM1C	IL1C2	IL1C1	IL1C0
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 28 to activate the DMAC.	Sets the priority level for interrupt number 28 (INTTA3) when DM1C = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM1C = 1. 000-011: 0 to 3 100-111: Invalid settings		

	15	14	13	12	11	10	9	8
Bit Symbol			EIM1D1	EIM1D0	DM1D	IL1D2	IL1D1	IL1D0
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 29 to activate the DMAC.	Sets the priority level for interrupt number 29 (INTTB0) when DM1D = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM1D = 1. 000-011: 0 to 3 100-111: Invalid settings		

	23	22	21	20	19	18	17	16
Bit Symbol			EIM1E1	EIM1E0	DM1E	IL1E2	IL1E1	IL1E0
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 30 to activate the DMAC.	Sets the priority level for interrupt number 30 (INTTB1) when DM1E = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM1E = 1. 000-011: 0 to 3 100-111: Invalid settings		

	31	30	29	28	27	26	25	24
Bit Symbol			EIM1F1	EIM1F0	DM1F	IL1F2	IL1F1	IL1F0
Read/Write			R/W					
After reset		0	0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 31 to activate the DMAC.	Sets the priority level for interrupt number 31 (INTRX4) when DM1F = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM1F = 1. 000-011: 0 to 3 100-111: Invalid settings		

Note : Before enabling the above interrupt requests, be sure to set their active state.

IMC8		7	6	5	4	3	2	1	0
(0xFFFF_E020)	Bit Symbol			EIM201	EIM200	DM20	IL202	IL201	IL200
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 32 to activate the DMAC.	Sets the priority level for interrupt number 32 (INTTX4) when DM20 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM20 = 1. 000-011: 0 to 3 100-111: Invalid settings		

		15	14	13	12	11	10	9	8
	Bit Symbol			EIM211	EIM210	DM21	IL212	IL211	IL210
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 33 to activate the DMAC.	Sets the priority level for interrupt number 33 (INTRX5) when DM21 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM21 = 1. 000-011: 0 to 3 100-111: Invalid settings		

		23	22	21	20	19	18	17	16
	Bit Symbol			EIM221	EIM220	DM22	IL222	IL221	IL220
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 34 to activate the DMAC.	Sets the priority level for interrupt number 34 (INTTX5) when DM22 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM22 = 1. 000-011: 0 to 3 100-111: Invalid settings		

		31	30	29	28	27	26	25	24
	Bit Symbol			EIM231	EIM230	DM23	IL232	IL231	IL230
	Read/Write			R/W					
	After reset		0	0	0	0	0	0	0
	Function			Must be set to 00.		Must be set to 0.	Must be set to 000.		

Note : Before enabling the above interrupt requests, be sure to set their active state.

		7	6	5	4	3	2	1	0
IMC9	Bit Symbol			EIM241	EIM240	DM24	IL242	IL241	IL240
(0xFFFF_E024)	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Must be set to 00.		Must be set to 0.	Must be set to 000.		

		15	14	13	12	11	10	9	8
	Bit Symbol			EIM251	EIM250	DM25	IL252	IL251	IL250
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 37 to activate the DMAC.	Sets the priority level for interrupt number 37 (INTTA4) when DM25 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM25 = 1. 000-011: 0 to 3 100-111: Invalid settings		

		23	22	21	20	19	18	17	16
	Bit Symbol			EIM261	EIM260	DM26	IL262	IL261	IL260
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 38 to activate the DMAC.	Sets the priority level for interrupt number 38 (INTTA5) when DM26 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM26 = 1. 000-011: 0 to 3 100-111: Invalid settings		

		31	30	29	28	27	26	25	24
	Bit Symbol			EIM271	EIM270	DM27	IL272	IL271	IL270
	Read/Write			R/W					
	After reset		0	0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 39 to activate the DMAC.	Sets the priority level for interrupt number 39 (INTTA6) when DM27 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM27 = 1. 000-011: 0 to 3 100-111: Invalid settings		

Note : Before enabling the above interrupt requests, be sure to set their active state.

	7	6	5	4	3	2	1	0
IMCA (0xFFFF_E028) Bit Symbol			EIM281	EIM280	DM28	IL282	IL281	IL280
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 40 to activate the DMAC.	Sets the priority level for interrupt number 40 (INTTA7) when DM28 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM28 = 1. 000-011: 0 to 3 100-111: Invalid settings		

	15	14	13	12	11	10	9	8
Bit Symbol			EIM291	EIM290	DM29	IL292	IL291	IL290
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 41 to activate the DMAC.	Sets the priority level for interrupt number 41 (INTTA8) when DM29 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM29 = 1. 000-011: 0 to 3 100-111: Invalid settings		

	23	22	21	20	19	18	17	16
Bit Symbol			EIM2A1	EIM2A0	DM2A	IL2A2	IL2A1	IL2A0
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 42 to activate the DMAC.	Sets the priority level for interrupt number 42 (INTTA9) when DM2A = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM2A = 1. 000-011: 0 to 3 100-111: Invalid settings		

	31	30	29	28	27	26	25	24
Bit Symbol			EIM2B1	EIM2B0	DM2B	IL2B2	IL2B1	IL2B0
Read/Write			R/W					
After reset		0	0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 43 to activate the DMAC.	Sets the priority level for interrupt number 43 (INTTAA) when DM2B = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM2B = 1. 000-011: 0 to 3 100-111: Invalid settings		

Note : Before enabling the above interrupt requests, be sure to set their active state.

IMCB
(0xFFFF_E02C)

	7	6	5	4	3	2	1	0
Bit Symbol			EIM2C1	EIM2C0	DM2C	IL2C2	IL2C1	IL2C0
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 44 to activate the DMAC.	Sets the priority level for interrupt number 44 (INTTAB) when DM2C = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM2C = 1. 000-011: 0 to 3 100-111: Invalid settings		

	15	14	13	12	11	10	9	8
Bit Symbol			EIM2D1	EIM2D0	DM2D	IL2D2	IL2D1	IL2D0
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 45 to activate the DMAC.	Sets the priority level for interrupt number 45 (INTTBA) when DM2D = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM2D = 1. 000-011: 0 to 3 100-111: Invalid settings		

	23	22	21	20	19	18	17	16
Bit Symbol			EIM2E1	EIM2E0	DM2E	IL2E2	IL2E1	IL2E0
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 46 to activate the DMAC.	Sets the priority level for interrupt number 46 (INTTBB) when DM2E = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM2E = 1. 000-011: 0 to 3 100-111: Invalid settings		

	31	30	29	28	27	26	25	24
Bit Symbol			EIM2F1	EIM2F0	DM2F	IL2F2	IL2F1	IL2F0
Read/Write			R/W					
After reset		0	0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 47 to activate the DMAC.	Sets the priority level for interrupt number 47 (INTTBC) when DM2F = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM2F = 1. 000-011: 0 to 3 100-111: Invalid settings		

Note : Before enabling the above interrupt requests, be sure to set their active state.

IMCC
(0xFFFF_E030)

	7	6	5	4	3	2	1	0
Bit Symbol			EIM301	EIM300	DM30	IL302	IL301	IL300
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 48 to activate the DMAC.	Sets the priority level for interrupt number 48 (INTTBD) when DM30 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM30 = 1. 000-011: 0 to 3 100-111: Invalid settings		

	15	14	13	12	11	10	9	8
Bit Symbol			EIM311	EIM310	DM31	IL312	IL311	IL310
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 49 to activate the DMAC.	Sets the priority level for interrupt number 49 (INTTB2) when DM31 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM31 = 1. 000-011: 0 to 3 100-111: Invalid settings		

	23	22	21	20	19	18	17	16
Bit Symbol			EIM321	EIM320	DM32	IL322	IL321	IL320
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 50 to activate the DMAC.	Sets the priority level for interrupt number 50 (INTTB3) when DM32 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM32 = 1. 000-011: 0 to 3 100-111: Invalid settings		

	31	30	29	28	27	26	25	24
Bit Symbol			EIM331	EIM330	DM33	IL332	IL331	IL330
Read/Write			R/W					
After reset		0	0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 51 to activate the DMAC.	Sets the priority level for interrupt number 51 (INTTB4) when DM33 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM33 = 1. 000-011: 0 to 3 100-111: Invalid settings		

Note : Before enabling the above interrupt requests, be sure to set their active state.

		7	6	5	4	3	2	1	0
IMCD	Bit Symbol			EIM341	EIM340	DM34	IL342	IL341	IL340
(0xFFFF_E034)	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 52 to activate the DMAC.	Sets the priority level for interrupt number 52 (INTTB5) when DM34 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM34 = 1. 000-011: 0 to 3 100-111: Invalid settings		

		15	14	13	12	11	10	9	8
	Bit Symbol			EIM351	EIM350	DM35	IL352	IL351	IL350
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 53 to activate the DMAC.	Sets the priority level for interrupt number 53 (INTTB6) when DM35 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM35 = 1. 000-011: 0 to 3 100-111: Invalid settings		

		23	22	21	20	19	18	17	16
	Bit Symbol			EIM361	EIM360	DM36	IL362	IL361	IL360
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 54 to activate the DMAC.	Sets the priority level for interrupt number 54 (INTTB7) when DM36 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM36 = 1. 000-011: 0 to 3 100-111: Invalid settings		

		31	30	29	28	27	26	25	24
	Bit Symbol			EIM371	EIM370	DM37	IL372	IL371	IL370
	Read/Write			R/W					
	After reset		0	0	0	0	0	0	0
	Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 55 to activate the DMAC.	Sets the priority level for interrupt number 55 (INTTB8) when DM37 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM37 = 1. 000-011: 0 to 3 100-111: Invalid settings		

Note : Before enabling the above interrupt requests, be sure to set their active state.

	7	6	5	4	3	2	1	0
Bit Symbol			EIM381	EIM380	DM38	IL382	IL381	IL380
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 56 to activate the DMAC.	Sets the priority level for interrupt number 56 (INTTB9) when DM38 = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM38 = 1. 000-011: 0 to 3 100-111: Invalid settings		

	15	14	13	12	11	10	9	8
Bit Symbol			EIM391	EIM390	DM39	IL392	IL391	IL390
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 00.		Must be set to 0.	Must be set to 000.		

	23	22	21	20	19	18	17	16
Bit Symbol			EIM3A1	EIM3A0	DM3A	IL3A2	IL3A1	IL3A0
Read/Write			R/W					
After reset			0	0	0	0	0	0
Function			Must be set to 01.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 58 to activate the DMAC.	Sets the priority level for interrupt number 58 (INTRTC) when DM3A = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM3A = 1. 000-011: 0 to 3 100-111: Invalid settings		

	31	30	29	28	27	26	25	24
Bit Symbol			EIM3B1	EIM3B0	DM3B	IL3B2	IL3B1	IL3B0
Read/Write			R/W					
After reset		0	0	0	0	0	0	0
Function			Must be set to 11.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 59 to activate the DMAC.	Sets the priority level for interrupt number 59 (INTAD) when DM3B = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM3B = 1. 000-011: 0 to 3 100-111: Invalid settings		

IMCE (0xFFFF_E038)

Note : Before enabling the above interrupt requests, be sure to set their active state.

		7	6	5	4	3	2	1	0
IMCF	Bit Symbol			EIM3C1	EIM3C0	DM3C	IL3C2	IL3C1	IL3C0
(0xFFFF_E03C)	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Must be set to 10.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 60 to activate the DMAC.	Sets the priority level for interrupt number 60 (INTDMA0) when DM3C = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM3C = 1. 000-011: 0 to 3 100-111: Invalid settings		
		15	14	13	12	11	10	9	8
	Bit Symbol			EIM3D1	EIM3D0	DM3D	IL3D2	IL3D1	IL3D0
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Must be set to 10.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 61 to activate the DMAC.	Sets the priority level for interrupt number 61 (INTDMA1) when DM3D = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM3D = 1. 000-011: 0 to 3 100-111: Invalid settings		
		23	22	21	20	19	18	17	16
	Bit Symbol			EIM3E1	EIM3E0	DM3E	IL3E2	IL3E1	IL3E0
	Read/Write			R/W					
	After reset			0	0	0	0	0	0
	Function			Must be set to 10.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 62 to activate the DMAC.	Sets the priority level for interrupt number 62 (INTDMA1) when DM3E = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM3E = 1. 000-011: 0 to 3 100-111: Invalid settings		
		31	30	29	28	27	26	25	24
	Bit Symbol			EIM3F1	EIM3F0	DM3F	IL3F2	IL3F1	IL3F0
	Read/Write			R/W					
	After reset		0	0	0	0	0	0	0
	Function			Must be set to 10.		Sets whether or not to activate the DMAC. 0: Not set. 1: Set interrupt number 63 to activate the DMAC.	Sets the priority level for interrupt number 63 (INTDMA2) when DM3F = 0. 000: Disable interrupt. 001-111: 1 to 7 Selects a DMAC channel when DM3F = 1. 000-011: 0 to 3 100-111: Invalid settings		

Note : Before enabling the above interrupt requests, be sure to set their active state.

Interrupt request clear register: Sets the value of IVR<LIVR9:LIVR4> for the interrupt whose request is to be cleared.

		7	6	5	4	3	2	1	0
INTCLR	Bit Symbol	—	—	EICLR5	EICLR4	EICLR3	EICLR2	EICLR1	EICLR0
(0xFFFF_E060)	Read/Write	—	—	W					
	After reset	—	—	—	—	—	—	—	—
	Function			Sets the value of IVR<9:4> for the interrupt whose request is to be cleared.					

Note1: Do not clear an interrupt request before reading the corresponding IVR value.

Note2: Follow the steps below to disable a particular interrupt with the Interrupt Controller (INTC).

    1. Globally disable the acceptance of interrupts by the core processor by clearing the IEc bit of the Status register.

    2. Disable the desired interrupt with the INTC by clearing the ILx[2:0] field of the IMCxx register.

    3. Execute the SYNC instruction.

    4. Enable the acceptance of interrupts by the core processor by setting the IEc bit of the Status register.

Example:

```
    mtc0    r0, r31          ; _DI () ;
    sb      r0, IMC**        ; IMC** = 0 ;
    sync                     ; _SYNC () ;
    mtc0    $sp, r31         ; _EI () ;
```

3.5 I/O Ports

The TMP1942 has 108 I/O port pins. All the port pins except a few share pins with alternate functions. They can be individually programmed as general-purpose I/O or dedicated I/O for the on-chip CPU or peripherals.

Table 3.5.1 Programmable I/O Ports(1/2)

Port	Pin Name	# of Pins	Direction	Pull Resistor	Direction Programmability	Alternate Functions
Port 0	P00~P07	8	Input/output		Bitwise	AD0~AD7
Port 1	P10~P17	8	Input/output		Bitwise	AD8~AD15 A8~A15
Port 2	P20~P27	8	Input/output		Bitwise	A0~A7 A16~A23
Port 3	P30	1	Output		Bitwise	$\overline{RD}$
	P31	1	Output		Bitwise	$\overline{WR}$
	P32	1	Input/output	Pull up	Bitwise	$\overline{HWR}$
	P33	1	Input/output	Pull up	Bitwise	$\overline{WAIT}$
	P34	1	Input/output	Pull up	Bitwise	$\overline{BUSRQ}$
	P35	1	Input/output	Pull up	Bitwise	$\overline{BUSAK}$
	P36	1	Input/output	Pull up	Bitwise	$R/\overline{W}$
	P37	1	Input/output	Pull up	Bitwise	$\overline{DSU}$
Port 4	P40	1	Input/output	Pull up	Bitwise	$\overline{CS0}$
	P41	1	Input/output	Pull up	Bitwise	$\overline{CS1}$
	P42	1	Input/output	Pull up	Bitwise	$\overline{CS2}$
	P43	1	Input/output	Pull up	Bitwise	$\overline{CS3}$
	P44	1	Input/output		Bitwise	SCOUT
Port 5	P50~P57	8	Input		Fixed	AN0~AN7 ADTRG
Port 6	P60~P67	8	Input		Fixed	AN8~AN15 KEY0-KEY7
Port 9	P90	1	Input/output		Bitwise	KEY8
	P91	1	Input/output		Bitwise	KEY9
	P92	1	Input/output		Bitwise	TB40UT
	P93	1	Input/output		Bitwise	TB5OUT
	P94	1	Input/output		Bitwise	TB6OUT
	P95	1	Input/output		Bitwise	TB7IN0
	P96	1	Input/output		Bitwise	TB7IN1
	P97	1	Input/output		Bitwise	TB7OUT

Table 3.5.1 Programmable I/O Ports(2/2)

Port	Pin Name	# of Pins	Direction	Pull Resistor	Direction Programmability	Alternate Functions		
Port A	PA0	1	Input/output		Bitwise	TB0IN0	INT3	
	PA1	1	Input/output		Bitwise	TB0IN1	INT4	
	PA2	1	Input/output		Bitwise	TB0OUT		
	PA3	1	Input/output		Bitwise	TB1IN0	INT5	
	PA4	1	Input/output		Bitwise	TB1IN1	INT6	
	PA5	1	Input/output		Bitwise	TB1OUT		
	PA6	1	Input/output		Bitwise	TA1OUT		
	PA7	1	Input/output		Bitwise	TA0IN	KEYA	
Port B	PB0	1	Input/output		Bitwise	TB2IN0	INTB	
	PB1	1	Input/output		Bitwise	TB2IN1	INTC	
	PB2	1	Input/output		Bitwise	TB2OUT	TB4IN0	
	PB3	1	Input/output		bit	TB3IN0	INTD	
	PB4	1	Input/output		bit	TB3IN1	INTE	
	PB5	1	Input/output		bit	TB3OUT	TB4IN1	
	PB6	1	Input/output		bit	TA3OUT		
	PB7	1	Input/output		bit	TA2IN	INT7	KEYB
Port C	PC0	1	Input/output		bit	TA4IN	INT8	
	PC1	1	Input/output		bit	TA6IN	INT9	
	PC2	1	Input/output		bit	TA8IN	INTA	
	PC3	1	Input/output		bit	TA5OUT		
	PC4	1	Input/output		bit	TAAIN		
	PC5	1	Input/output		bit	TA7OUT		
	PC6	1	Input/output		bit	TB8IN0	KEYC	
	PC7	1	Input/output		bit	TB8IN1	TA9OUT	
Port D	PD0	1	Input/output		bit	TXD0	TB9IN0	
	PD1	1	Input/output		bit	RXD0	TB9IN1	
	PD2	1	Input/output		bit	SCLK0	$\overline{CTS0}$	
	PD3	1	Input/output		bit	TXD1	TBAIN0	
	PD4	1	Input/output		bit	RXD1	TBAIN1	
	PD5	1	Input/output		bit	SCLK1	$\overline{CTS1}$	TABOUT
	PD6	1	Input/output		bit	XT1		
	PD7	1	Input/output		bit	XT2		
Port E	PE0	1	Input/output		bit	TXD3		
	PE1	1	Input/output		bit	RXD3		
	PE2	1	Input/output		bit	SCLK3	$\overline{CTS3}$	
	PE3	1	Input/output		bit	TXD4		
	PE4	1	Input/output		bit	RXD4		
	PE5	1	Input/output		bit	SCLK4	$\overline{CTS4}$	
	PE6	1	Input/output		bit	INT1	BOOT	
	PE7	1	Input/output		bit	INT2	INTLV	
Port F	PF0	1	Input/output		bit	TXD5		
	PF1	1	Input/output		bit	RXD5	KEYD	
	PF2	1	Input/output		bit	SCLK5	$\overline{CTS5}$	
	PF3	1	Input/output		bit	SCK		
	PF4	1	Input/output		bit	SO	SDA	
	PF5	1	Input/output		bit	SI	SCL	
	PF6	1	Input/output		bit	INT0		

Table 3.5.2 I/O Port Programmability (1/4)

Port	Pin Name	Direction / Function	I/O Register Settings			
			Pn	PnCR	PnFC	PnFC2
Port 0	P00~P07	Input	-	0		
		Output	-	1		
		AD0~AD7 Bus	-	-		
Port 1	P10~P17	Input	-	0	0	
		Output	-	1	0	
		AD8~AD15 Bus	-	0	1	
		A8~A15 Bus	-	1	1	
Port 2	P20~P27	Input	-	0	0	
		Output	-	1	0	
		A0~A7 Bus	-	0	1	
		A16~A23 Bus	-	1	1	
Port 3	P30	Output	-		0	
		$\overline{RD}$	-		1	
	P31	Output	-		0	
		$\overline{WR}$	-		1	
	P32	Input(RSTUP=1)	1	0	0	
		Input(RSTUP=0)	0	0	0	
		Output	-	1	0	
		$\overline{HWR}$	-	1	1	
	P33	Input(RSTUP=1)	1	0	0	
		Input(RSTUP=0)	0	0	0	
		Output	-	1	0	
		$\overline{WAIT}$	-	0	0	
	P34	Input(RSTUP=1)	1	0	0	
		Input(RSTUP=0)	0	0	0	
		Output	-	1	0	
		$\overline{BUSRQ}$	-	0	1	
	P35	Input(RSTUP=1)	1	0	0	
		Input(RSTUP=0)	0	0	0	
		Output	-	1	0	
		$\overline{BUSAK}$	-	1	1	
	P36	Input(RSTUP=1)	1	0	0	
		Input(RSTUP=0)	0	0	0	
		Output	-	1	0	
		R/$\overline{W}$	-	1	1	
	P37	Input	1	0		
		Output	1	1		
Port 4	P40	Input(RSTUP=1)	1	0	0	
		Input(RSTUP=0)	0	0	0	
		Output	-	1	0	
		$\overline{CS0}$	-	-	1	
	P41	Input(RSTUP=1)	1	0	0	
		Input(RSTUP=0)	0	0	0	
		Output	-	1	0	
		$\overline{CS1}$	-	-	1	

Table 3.5.2 I/O Port Programmability (2/4)

Port	Pin Name	Direction / Function	I/O Register Settings			
			Pn	PnCR	PnFC	PnFC2
Port 4	P42	Input(RSTUP=1)	1	0	0	
		Input(RSTUP=0)	0	0	0	
		Output	-	1	0	
		$\overline{CS2}$	-	-	1	
	P43	Input(RSTUP=1)	1	0	0	
		Input(RSTUP=0)	0	0	0	
		Output	-	1	0	
		$\overline{CS3}$	-	-	1	
	P44	Input	1	0	0	
		Output	1	1	0	
		SCOUT	-	-	1	
Port 5	P50~P57	Input	-		0	
		AN0~AN7	-		0	
		ADTRG	-		1	
Port 6	P60~P67	Input	-		0	
		AN8~AN15	-		0	
		KEY0~7	-		1	
Port 9	P90~P97	Input	-	0	0	
		Output	-	1	0	
	P90	KEY8	-	0	1	
	P91	KEY9	-	0	1	
	P92	TB40UT	-	1	1	
	P93	TB5OUT	-	1	1	
	P94	TB6OUT	-	1	1	
	P95	TB7IN0	-	0	1	
	P96	TB7IN1	-	0	1	
	P97	TB7OUT	-	1	1	
Port A	PA0~PA7	Input	-	0	0	
		Output	-	1	0	
	PA0	TB0IN0	-	0	1	
		INT3	-	0	1*	
	PA1	TB0IN1	-	0	1	
		INT4	-	0	1*	
	PA2	TB0OUT	-	1	1	
	PA3	TB1IN0	-	0	1	
		INT5	-	0	-	
	PA4	TB1IN1	-	0	1	
		INT6	-	0	-	
	PA5	TB1OUT	-	1	1	
	PA6	TA1OUT	-	1	1	
	PA7	TA0IN	-	0	1	
		KEYA	-	0	1	

Table 3.5.2 I/O Port Programmability (3/4)

Port	Pin Name	Direction / Function	I/O Register Settings			
			Pn	PnCR	PnFC	PnFC2
Port B	PB0~PB7	Input	-	0	0	
		Output	-	1	0	
	PB0	TB2IN0	-	0	1	
		INTB	-	0	1*	
	PB1	TB2IN1	-	0	1	
		INTC	-	0	1*	
	PB2	TB2OUT	-	1	1	
		TB4IN0	-	0	1	
	PB3	TB3IN0	-	0	1	
		INTD	-	0	1*	
	PB4	TB3IN1	-	0	1	
		INTE	-	0	1*	
	PB5	TB3OUT	-	1	1	
		TB4IN1	-	0	1	
	PB6	TA3OUT	-	1	1	
	PB7	TA2IN	-	0	1	
		INT7	-	0	-	
		KEYB	-	0	1	
Port C	PC0~PC7	Input	-	0	0	
		Output	-	1	0	
	PC0	TA4IN	-	0	1	
		INT8	-	0	-	
	PC1	TA6IN	-	0	1	
		INT9	-	0	-	
	PC2	TA8IN	-	0	1	
		INTA	-	0	-	
	PC3	TA5OUT	-	1	1	
	PC4	TAAIN	-	0	1	
	PC5	TA7OUT	-	1	1	
	PC6	TB8IN0	-	0	1	
		KEYC	-	0	1	
	PC7	TB8IN1	-	0	1	
		TA9OUT	-	1	1	
Port D	PD0~PD7	Input	-	0	0	-
		Output	-	1	0	-
	PD0	TXD0	-	1	1	-
		TB9IN0	-	0	1	-
	PD1	RXD0	-	0	1	-
		TB9IN1	-	0	1	-
	PD2	SCLK0(Input)	-	0	1	-
		SCLK0(Output)	-	1	1	-
		CTS0	-	0	1	-
	PD3	TXD1	-	1	1	-
		TBAIN0	-	0	1	-
	PD4	RXD1	-	0	1	-
		TBAIN1	-	0	1	-

Table 3.5.2 I/O Port Programmability (4/4)

Port	Pin Name	Direction / Function	\(Pn\)	PnCR	PnFC	PnFC2
			\(Pn\)	PnCR	PnFC	PnFC2
Port D	PD5	SCLK1(Input)	-	0	1	0
		SCLK1(Output)	-	1	1	0
		$\overline{CTS3}$	-	0	1	0
		TABOUT	-	1	0	1
	PD6	XT1	-	-	-	-
	PD7	XT2	-	-	-	-
Port E	PE0~PE7	Input	-	0	0	
		Output	-	1	0	
	PE0	TXD3	-	1	1	
	PE1	RXD3	-	0	1	
	PE2	SCLK3(Input)	-	0	1	
		SCLK3(Output)	-	1	1	
		$\overline{CTS3}$	-	0	1	
	PE3	TXD4	-	1	1	
	PE4	RXD4	-	0	1	
	PE5	SCLK4(Input)	-	0	1	
		SCLK4(Output)	-	1	1	
		$\overline{CTS4}$	-	0	1	
	PE6	INT1	-	0	1*	
	PE7	INT2	-	0	1*	
Port F	PF0~PF6	Input	-	0	0	
		Output	-	1	0	
	PF0	TXD5	-	1	1	
	PF1	RXD5	-	0	1	
		KEYD	-	0	1	
	PF2	SCLK5(Input)	-	0	1	
		SCLK5(Output)	-	1	1	
		$\overline{CTS}$	-	0	1	
	PF3	SCK(Input)	-	0	1	
		SCK(Output)	-	1	1	
	PF4	SO	-	1	1	
		SDA	-	1	1	
	PF5	SI	-	1	1	
		SCL	-	1	1	
	PF6	INT0	-	0	1*	

X: Don't care

Pn: Port n Register, PnCR: Port n Control Register, PnFC: Port n Function Register

*: Set this bit when using the pin for a STOP mode termination interrupt with SYSCR<DRVE> set to 0. Otherwise, the bit need not be set.

Note 1: $\overline{HWR}$, R/$\overline{W}$ and P40 to P43 have their internal pullup resistors enabled when the corresponding P4FC register bit is set and when the bus is released.

Note 2: When P50–P57 are configured as analog channels of the ADC, the ADCH[2:0] field in A/D Mode Control Register 1 (ADMOD1) is used to select a channel(s).

Note 3: When P57 is configured as $\overline{ADTRG}$, the ADTRGE bit in the ADMOD1 register is used to enable and disable the external trigger input to the ADC.

Note 4: When PD6–PD7 are configured as XT1–XT2, the SYSCR0 register must be programmed to enable oscillation, etc.

Note 5: When PortD and PortE and PortF are configured as SDA and SCL outputs for the SBI, the ODEA[7:6] field in the Open-Drain Enable (ODE) register can be used to configure them as either push-pull or open-drain ouptuts. Upon reset, the default is push-pull.

3.5.1   Port 0 (P00-P07)

Port 0 is an 8-bit general-purpose input/output port whose bits can each be set independently for input or output. Use the control register P0CR to set the port for input or output. A reset clears all bits of P0CR to 0 and puts port 0 in input mode.

In addition to functioning as a general-purpose input/output port, this port can also function as an address/data bus (AD0-AD7). When external memory is accessed, this port automatically functions as an address/data bus (AD0-AD7), with all bits of P0CR cleared to 0.
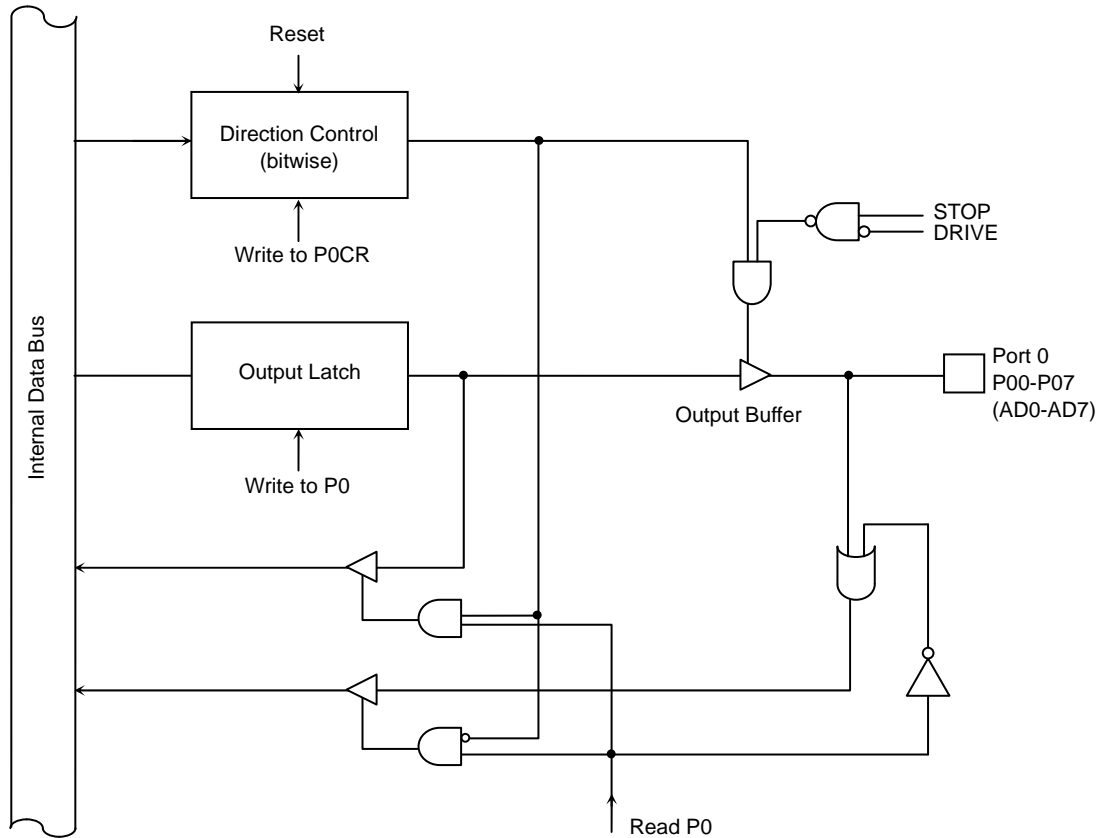
Figure 0.1 Port 0 (P00-P07)

Note: The above system diagram does not represent the address/data bus function.

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 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_F002)	Read/Write				W				
	After Reset	0	0	0	0	0	0	0	0
	Function	0: IN 1: OUT (Functions as AD7-AD0 when external area is accessed, with the register cleared to 0.)

Input/output setting for port 0

0	Input
1	Output

Figure 0.2 Registers Related to Port 0

3.5.2   Port 1 (P10-P17)

Port 1 is an 8-bit general-purpose input/output port whose bits can each be set independently for input or output. The control register P1CR and function register P1FC are used to set the port for input or output. A reset clears all bits of output latch P1 and all bits of P1CR and P1FC to 0, putting port 1 in input mode.

In addition to functioning as a general-purpose input/output port, this port can also function as an address/data bus (AD8-AD15) or an address bus (A8-A15). To access external memory, set this port to an address bus or address/data bus using P1CR and P1FC.
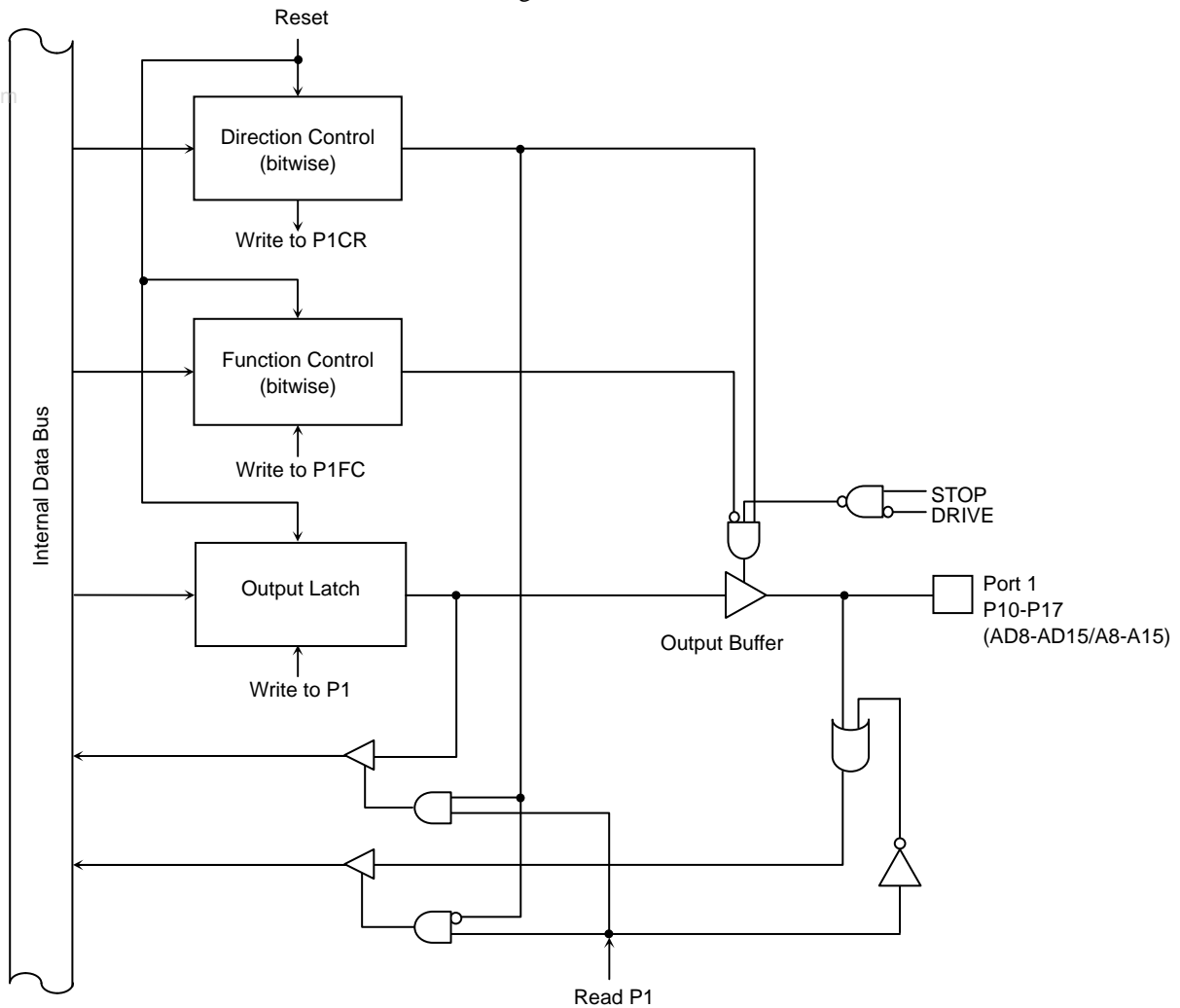
Figure 0.3 Port 1 (P10-P17)

Note: The above system diagram does not represent the address/data bus function.

Port 1 Register

P1		7	6	5	4	3	2	1	0
(0xFFFF_F001)	Bit Symbol	P17	P16	P15	P14	P13	P12	P11	P10
	Read/Write	R/W							
	After Reset	Input mode (output latch register cleared to 0)							

Port 1 Control Register

P1CR		7	6	5	4	3	2	1	0
(0xFFFF_F004)	Bit Symbol	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
	Read/Write	W							
	After Reset	0	0	0	0	0	0	0	0
	Function	<< Refer to P1FC. >>							

Port 1 Function Register

P1FC		7	6	5	4	3	2	1	0
(0xFFFF_F005)	Bit Symbol	P17F	P16F	P15F	P14F	P13F	P12F	P11F	P10F
	Read/Write	W							
	After Reset	0	0	0	0	0	0	0	0
	Function	P1FC/P1CR = 00: IN, 01: OUT, 10: AD15-8, 11: A15-8							

→ Function settings for port 1

P1CR <P1XC>	P1FC<P1XF>	
	0	1
0	Input port	Address/data bus (AD15-AD8)
1	Output port	Address bus (A15-A8)

Figure 0.4 Registers Related to Port 1

3.5.3   Port 2 (P20-P27)

Port 2 is an 8-bit general-purpose input/output port whose bits can each be set independently for input or output. The control register P2CR and function register P2FC are used to set the port for input or output. A reset sets all bits of output latch P2 to 1 and clears all bits of P2CR and P2FC to 0, putting port 2 in input mode.

In addition to functioning as a general-purpose input/output port, this port can function as an address bus (A0-A7 or A16-A23).
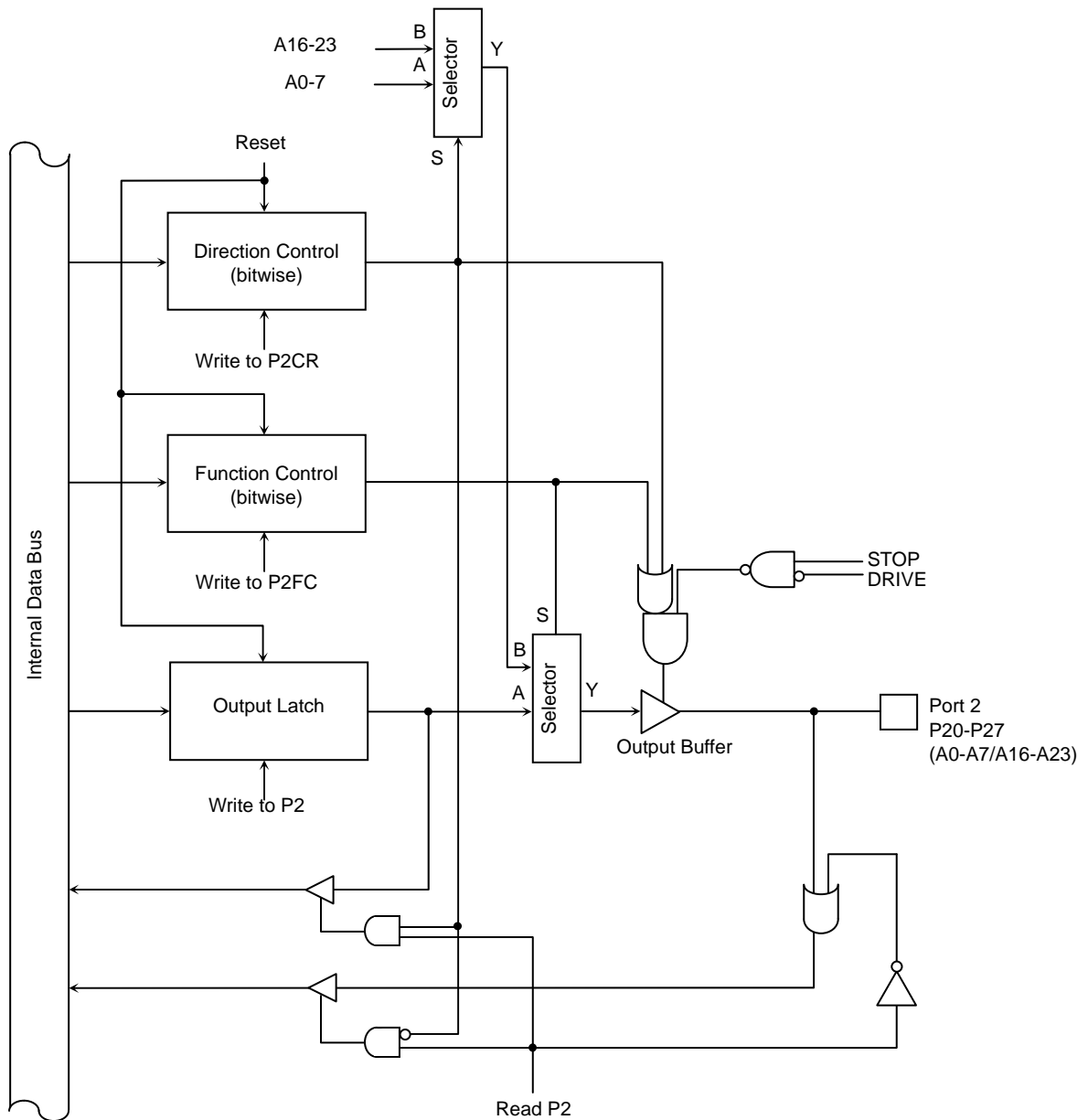
Figure 0.5 Port 2 (P20-P27)

Port 2 Control Register

		7	6	5	4	3	2	1	0
P2	Bit Symbol	P27	P26	P25	P24	P23	P22	P21	P20
(0xFFFF_F012)	Read/Write	R/W							
	After Reset	Input mode (output latch register set to 1)							

Port 2 Control Register

		7	6	5	4	3	2	1	0
P2CR	Bit Symbol	P27C	P26C	P25C	P24C	P23C	P22C	P21C	P20C
(0xFFFF_F014)	Read/Write	W							
	After Reset	0	0	0	0	0	0	0	0
	Function	<< Refer to P2FC.>>							

Port 2 Function Register

		7	6	5	4	3	2	1	0
P2FC	Bit Symbol	P27F	P26F	P25F	P24F	P23F	P22F	P21F	P20F
(0xFFFF_F015)	Read/Write	W							
	After Reset	0	0	0	0	0	0	0	0
	Function	P2FC/P2CR = 00: IN, 01: OUT, 10: A7-0, 11: A23-16							

Function settings for port 2

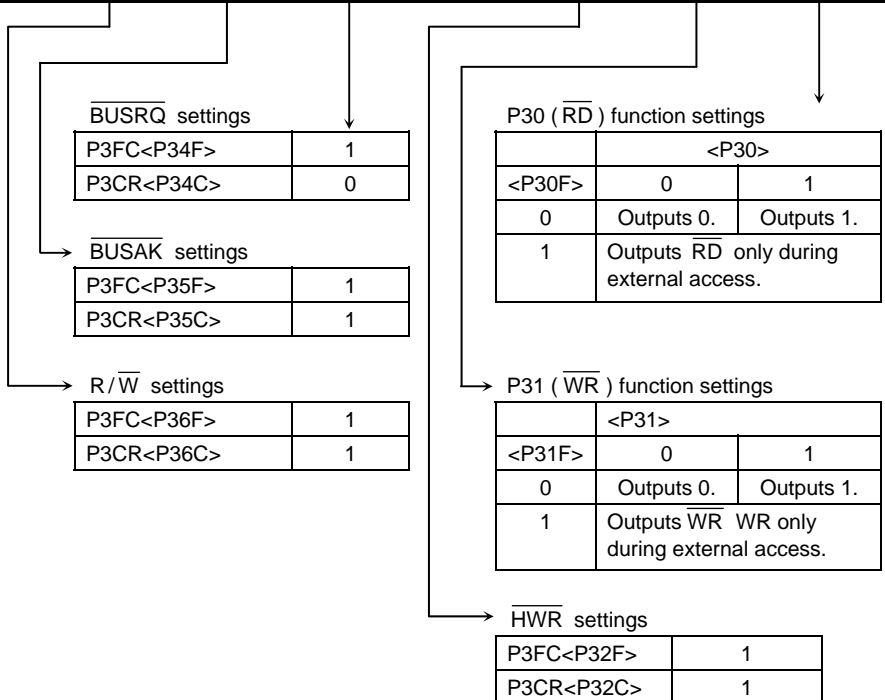| P2CR	P2FC<P2XF>	
<P2XC>	0	1
0	Input port	Address bus (A7-A0)
1	Output port	Address bus (A23-A16)

Figure 0.6 Registers Related to Port 2

3.5.4   Port 3 (P30-P37)

Port 3 is an 8-bit general-purpose input/output port whose bits can each be set independently for input or output, with the exception that P30 and P31 are output-only. The control register P3CR and function register P3FC are used to set the port for input or output. A reset sets bits P30, P31 and P37 of the output latch to 1. Bits P32 to P36 are set to 1 by a reset if RSTPUP is High or cleared to 0 if RSTPUP is Low. All bits of P3CR (bits 0 and 1 not used) and P3FC (bits 3 and 7 not used) are cleared to 0 by a reset, with P30 and P31 outputting a High signal and P32 to P36 placed in input mode with pull-up resistors enabled (if RSTPUP is High) or disabled (if RSTPUP is Low). P37 is placed in input mode with a pull-up resistor enabled regardless of the value of RSTPUP.

In addition to functioning as a general-purpose input/output port, this port can also input and output the CPU's control and status signals. The RD strobe is output only when an external address area is being accessed while the P30 pin is set for RD output (<P30F> = 1). Similarly, the WR strobe is output only when an external address area is being accessed while the P31 pin is set for WR output (<P31F> = 1).

P32 and P36 have their pull-up resistors enabled when BUSAK = 0 while <P3xFC> = 1.
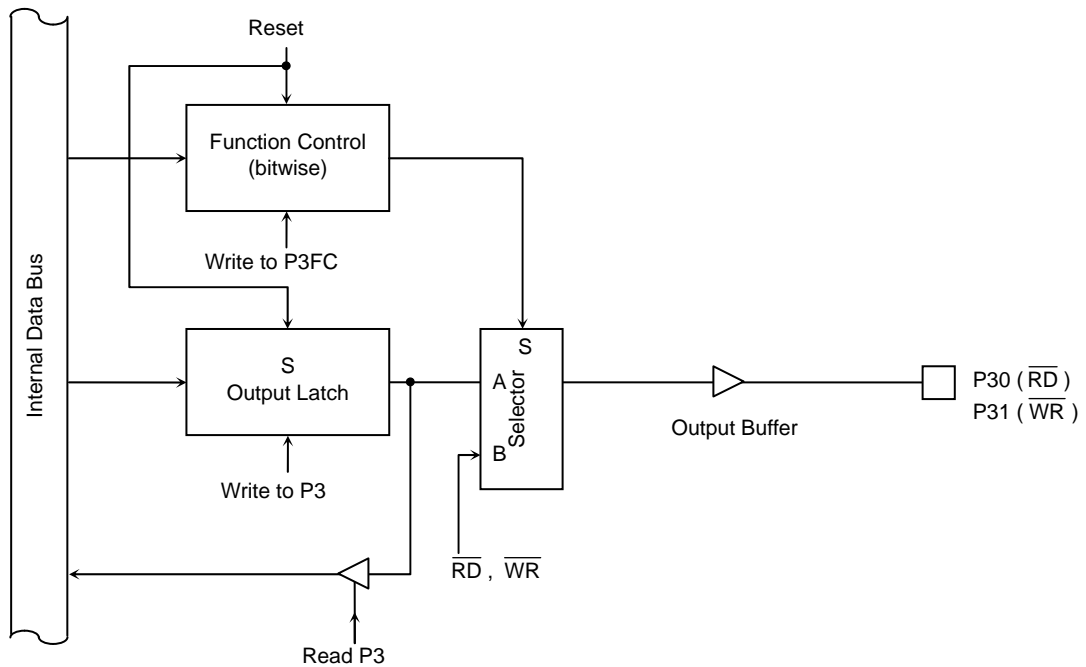
Figure 0.7 Port 3 (P30, P31)
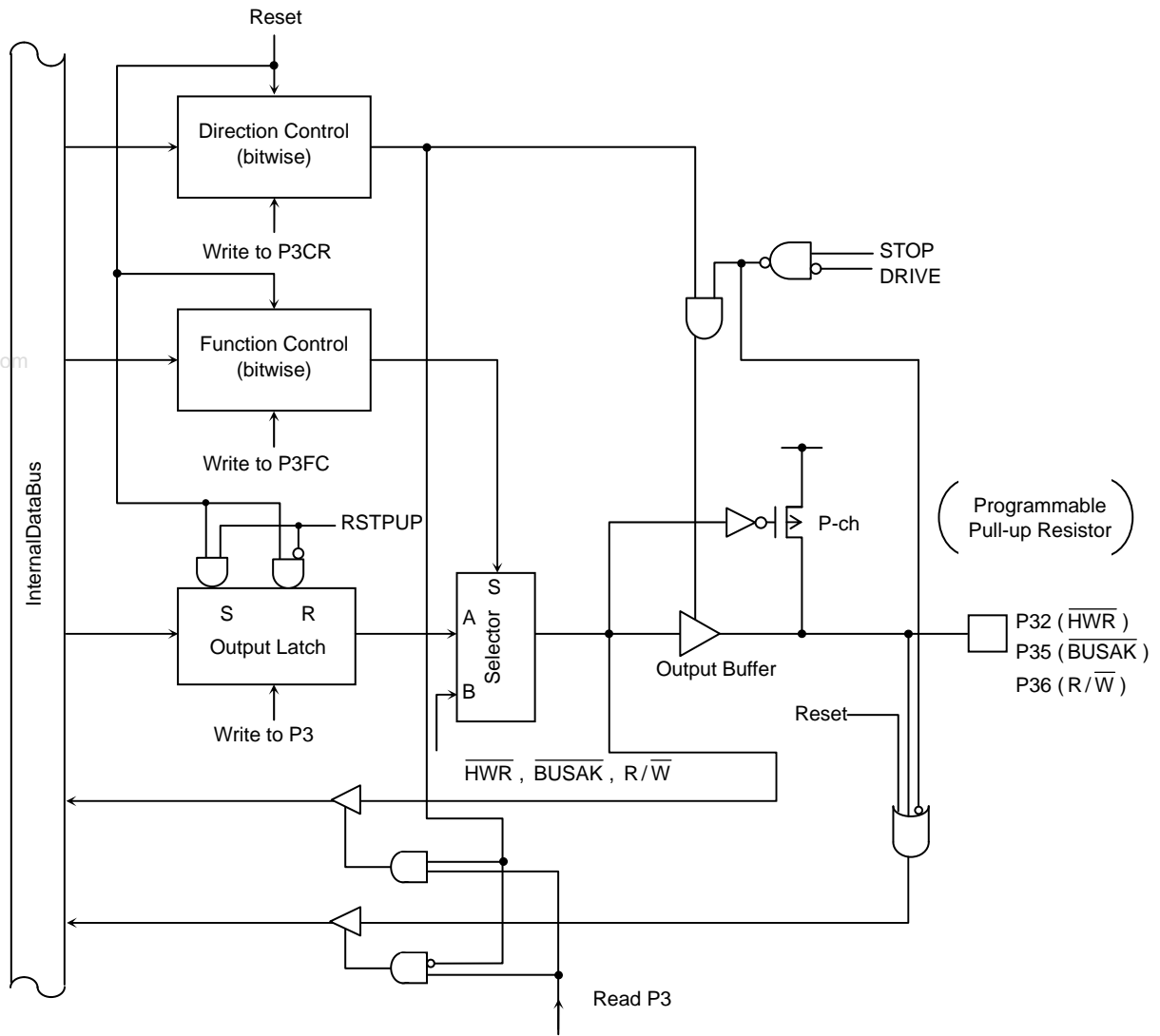
Figure 0.8 Port 3 (P32, P35, P36)
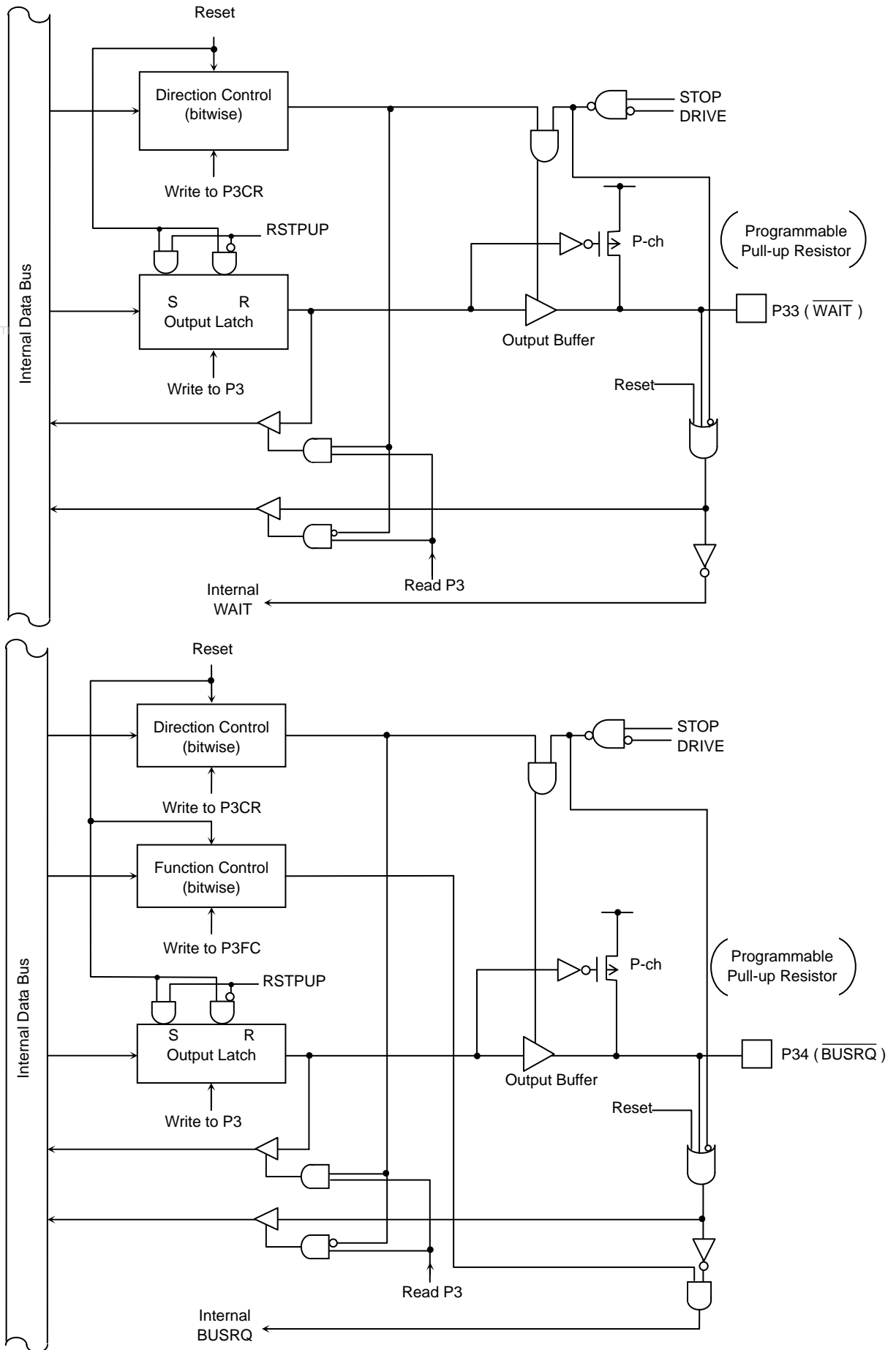
Figure 0.9 Port 3 (P33, P34)
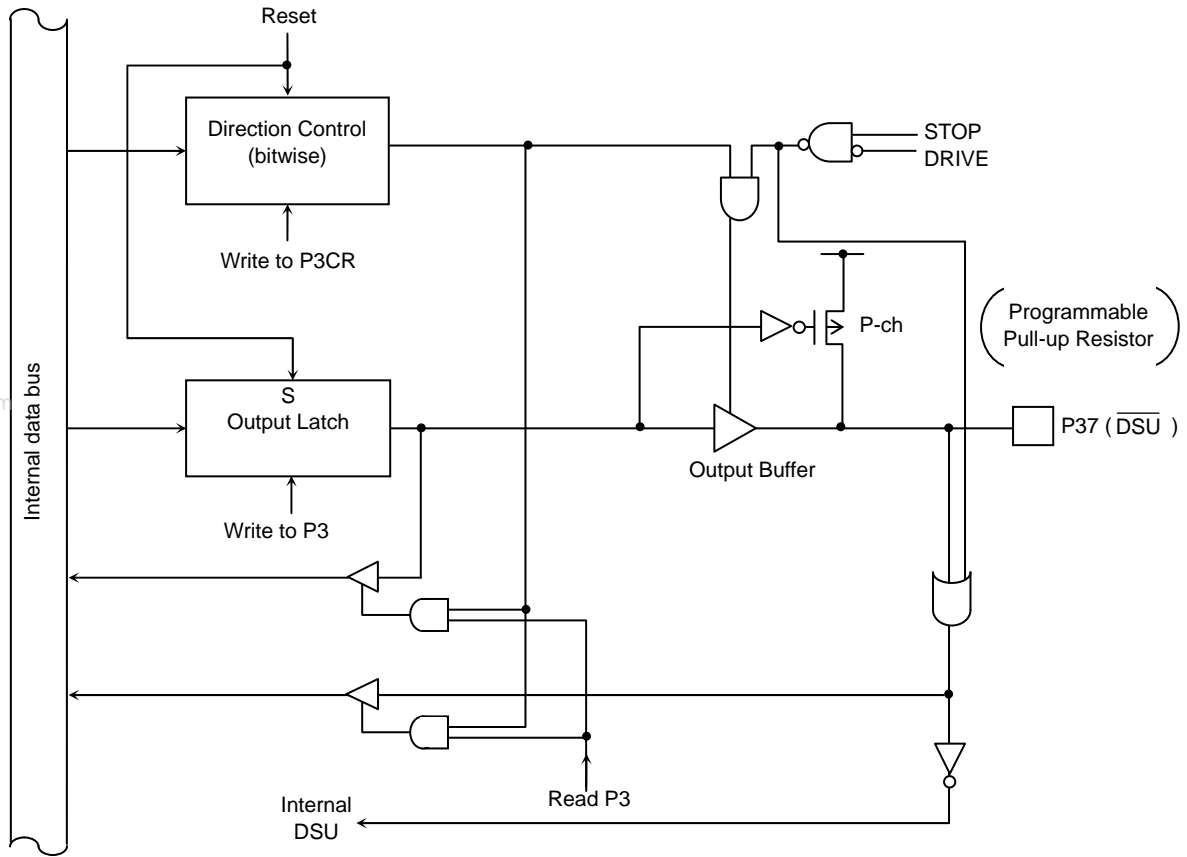
Figure 0.10 Port 3 (P37)

Port 3 Register

	7	6	5	4	3	2	1	0
Bit Symbol	P37	P36	P35	P34	P33	P32	P31	P30
Read/Write	R/W							
After Reset	Input Mode						Output Mode	
RSTPUP = 1	1 (Pull-UP)	1 (Pull-UP)	1 (Pull-UP)	1 (Pull-UP)	1 (Pull-UP)	1 (Pull-UP)	1	1
RSTPUP = 0	1 (Pull-UP)	0	0	0	0	0	1	1

P3
(0xFFFF_F018)

Port 3 Control Register

	7	6	5	4	3	2	1	0
Bit Symbol	P37C	P36C	P35C	P34C	P33C	P32C	—	—
Read/Write	W						—	—
After Reset	0	0	0	0	0	0	—	—
Function	0: IN 1: OUT							

P3CR
(0xFFFF_F01A)

Input/output settings for port 3	
0	Input
1	Output

Port 3 Function Register

	7	6	5	4	3	2	1	0
Bit Symbol	—	P36F	P35F	P34F	—	P32F	P31F	P30F
Read/Write	W							
After Reset	—	0	0	0	—	0	0	0
Function		0: PORT 1: R/$\overline{W}$	0: PORT 1: $\overline{BUSAK}$	0: PORT 1: $\overline{BUSRQ}$		0: PORT 1: $\overline{HWR}$	0: PORT 1: $\overline{WR}$	0: PORT 1: $\overline{RD}$

P3FC
(0xFFFF_F01B)

$\overline{BUSRQ}$ settings

P3FC<P34F>	1
P3CR<P34C>	0

$\overline{BUSAK}$ settings

P3FC<P35F>	1
P3CR<P35C>	1

R/$\overline{W}$ settings

P3FC<P36F>	1
P3CR<P36C>	1

P30 ($\overline{RD}$) function settings

	<P30>	
<P30F>	0	1
0	Outputs 0.	Outputs 1.
1	Outputs $\overline{RD}$ only during external access.	

P31 ($\overline{WR}$) function settings

	<P31>	
<P31F>	0	1
0	Outputs 0.	Outputs 1.
1	Outputs $\overline{WR}$ WR only during external access.	

$\overline{HWR}$ settings

P3FC<P32F>	1
P3CR<P32C>	1

Figure 0.11 Registers Related to Port 3

3.5.5   Port 4 (P40-P44)

Port 4 is a 5-bit general-purpose input/output port whose bits can each be set independently for input or output. The control register P4CR and function register P4FC are used to set the port for input or output. Bits P41 to P44 of the output latch register are set to 1 by a reset if RSTPUP is High or cleared to 0 if RSTPUP is Low. Bit P44 of the output latch register is set to 1 regardless of the value of RSTPUP. All bits of P4CR and P4FC are cleared to 0 by a reset, with P40 to P43 placed in input mode with pull-up resistors enabled (if RSTPUP is High) or disabled (if RSTPUP is Low). P44 is placed in input mode with a pull-up resistor disabled regardless of the value of RSTPUP.

In addition to functioning as a general-purpose input/output port, P40-P43 can also output the chip select signals (CS0-CS3), and P44 functions as the SCOUT pin, outputting the system clock.
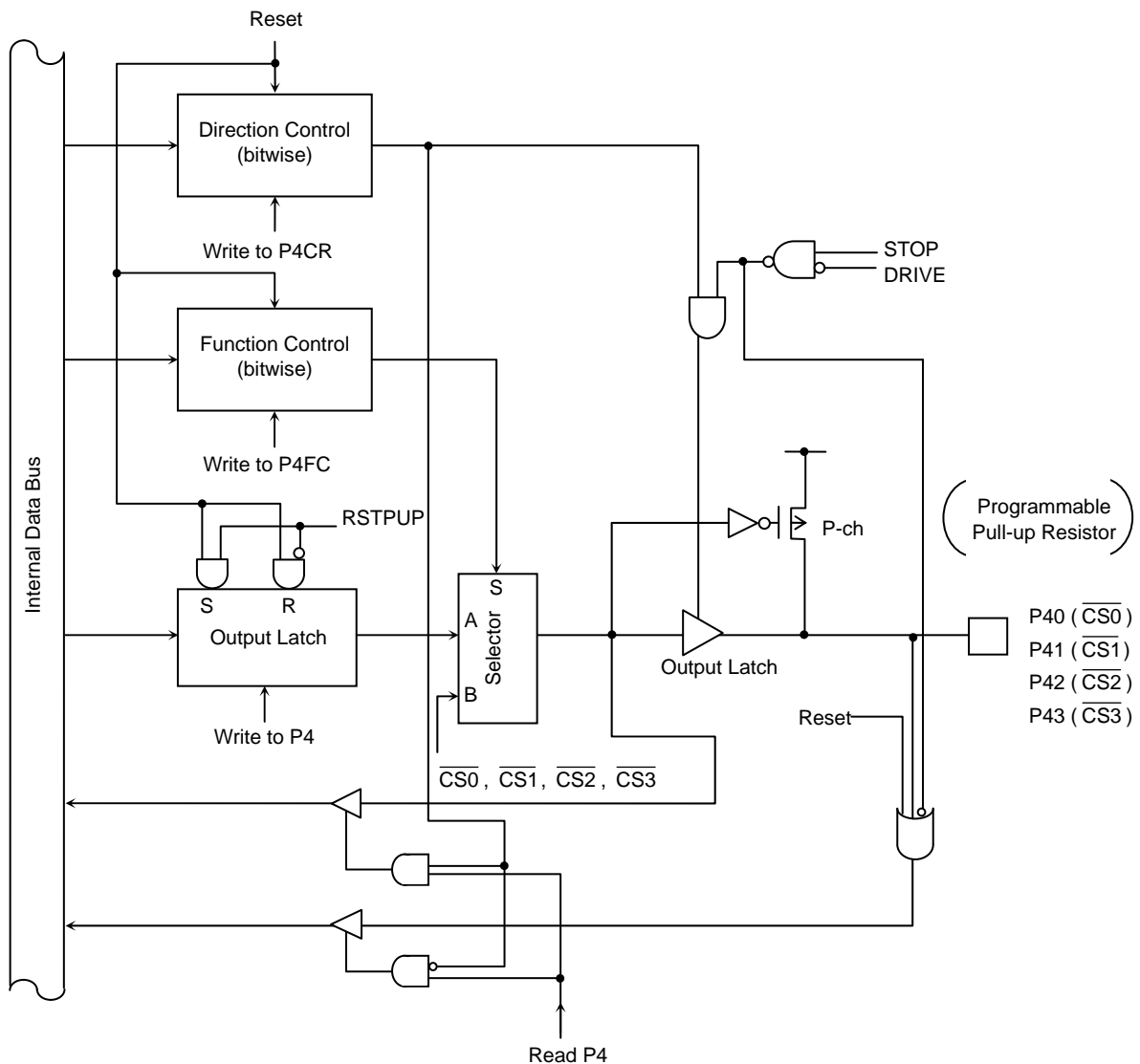
Figure 0.12   Port 4 (P40-P43)
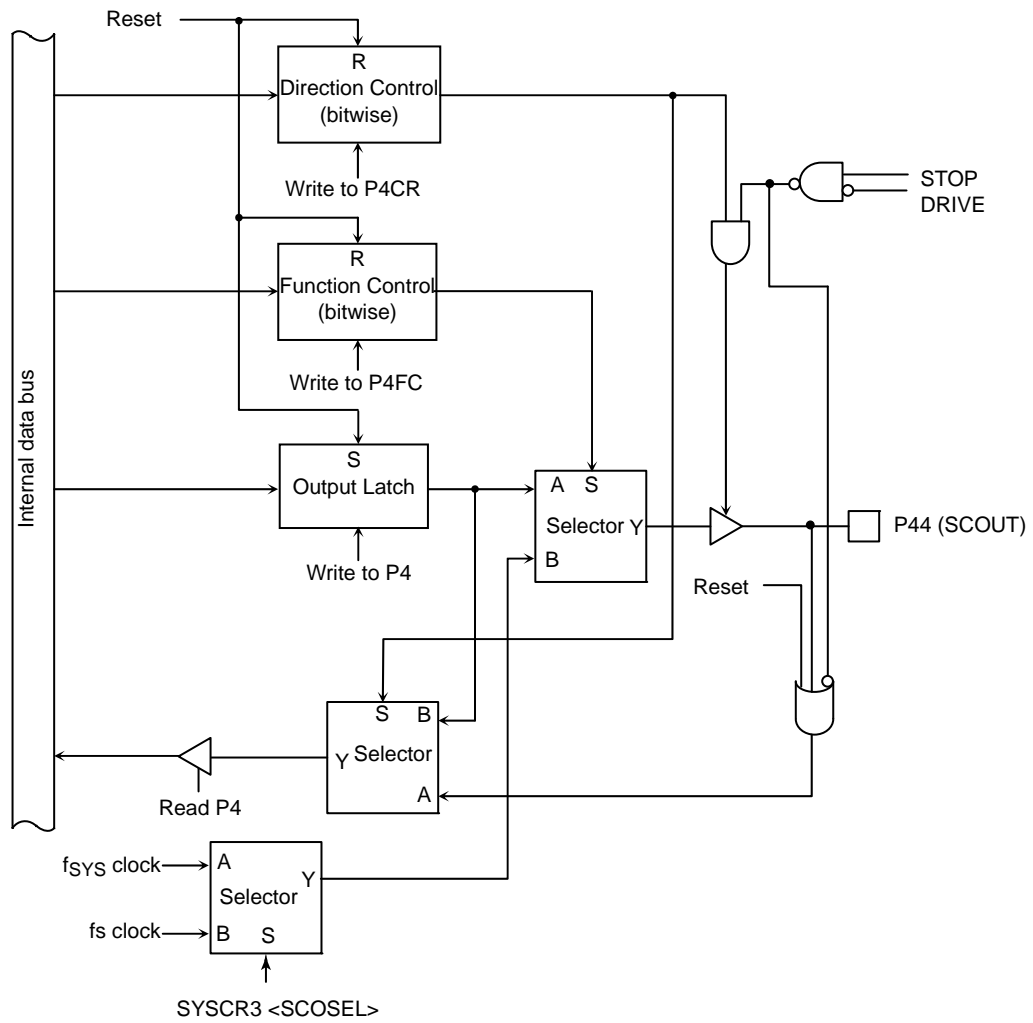
Figure 0.13 Port 4 (P44)

Port 4 Register

		7	6	5	4	3	2	1	0
P4	Bit Symbol	—	—	—	P44	P43	P42	P41	P40
(0xFFFF_F01E)	Read/Write	—	—	—			R/W		
	After Reset	—	—	—			Input mode		
	RSTPUP=1				1	1 (Pull-Up)	1 (Pull-Up)	1 (Pull-Up)	1 (Pull-Up)
	RSTPUP=0				1	0	0	0	0

RSTPUp = 1
RSTPUp = 0

Port 4 Control Register

		7	6	5	4	3	2	1	0
P4CR	Bit Symbol	—	—	—	P44C	P43C	P42C	P41C	P40C
(0xFFFF_F020)	Read/Write	—	—	—			W		
	After Reset	—	—	—	0	0	0	0	0
						0: IN		1: OUT	

Port 4 Function Register

		7	6	5	4	3	2	1	0
P4FC	Bit Symbol	—	—	—	P44F	P43F	P42F	P41F	P40F
(0xFFFF_F021)	Read/Write	—	—	—			W		
	After Reset	—	—	—	0	0	0	0	0
	Function				0: PORT 1: SCOUT		0: PORT 1: $\overline{CS}$		

0	PORT (P40)
1	$\overline{CS0}$

0	PORT (P41)
1	$\overline{CS1}$

0	PORT (P42)
1	$\overline{CS2}$

0	PORT (P43)
1	$\overline{CS3}$
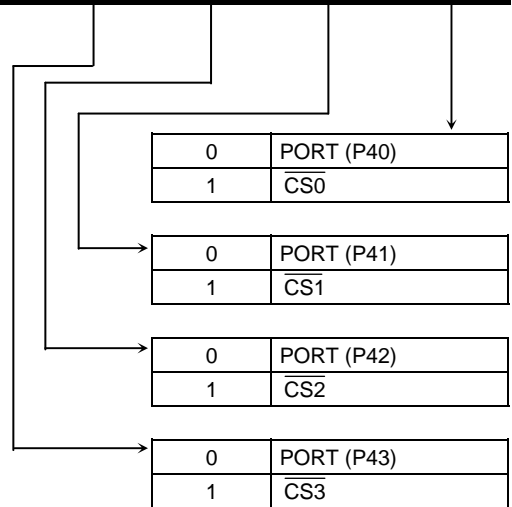
Figure 0.14 Registers Related to Port 4

3.5.6   Port 5 (P50-P57)

Port 5 is an 8-bit input-only port, and is shared with the A/D converter's analog input pins. P57 also functions as the A/D converter's A/D trigger input pin.
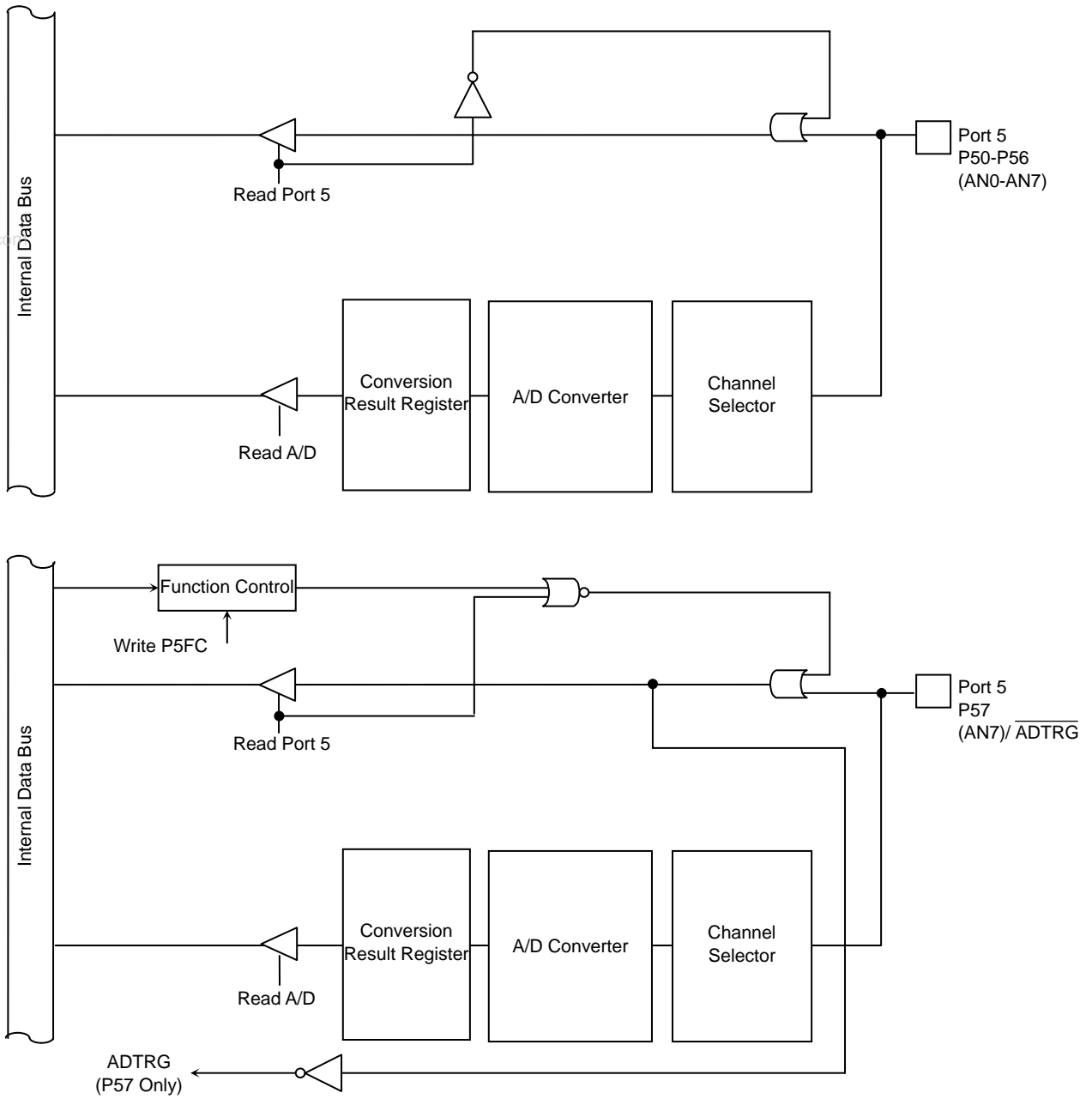
Figure 0.15 Port 5 (P50-P57)

Port 5 Register

	7	6	5	4	3	2	1	0
Bit Symbol	P57	P56	P55	P54	P53	P52	P51	P50
Read/Write	R							
After Reset	Input mode							

P5
(0xFFFF_F040)

Port 5 Function Register

	7	6	5	4	3	2	1	0
Bit Symbol	P57F	—	—	—	—	—	—	—
Read/Write	W	—						
After Reset	0	—	—	—	—	—	—	—
Function	0: Port or A/D input 1: $\overline{\text{ADTRG}}$							

P5FC
(0xFFFF_F043)

Figure 0.16 Port 5 (P50-P57)

Note 1: Use A/D converter mode register ADMOD4 to select A/D converter input channels and to enable A/D trigger input for P57.

Note 2: To use $\overline{\text{ADTRG}}$, first set <P57F> to 1 and then enable trigger input in A/D converter mode register ADMOD4. To stop using $\overline{\text{ADTRG}}$, first disable trigger input in ADMOD4 and then clear <P57F> to 0 (port).

3.5.7   Port 6 (P60-P67)

Port 6 is an 8-bit input-only port, and is shared with the A/D converter's analog input pins and key input pins. A reset clears P6FC to 0, placing port 6 in A/D or port input mode. Writing a 1 to a bit of P6FC enables the corresponding pin to be used as a key input pin. Port 6 has pull-up resistors, which are enabled only for those pins for which KWUPCNT<PE> is set to 1 in the key on wake-up circuit and key input is enabled in KWUPSTn. For details, refer to the description of the key on wake-up function.
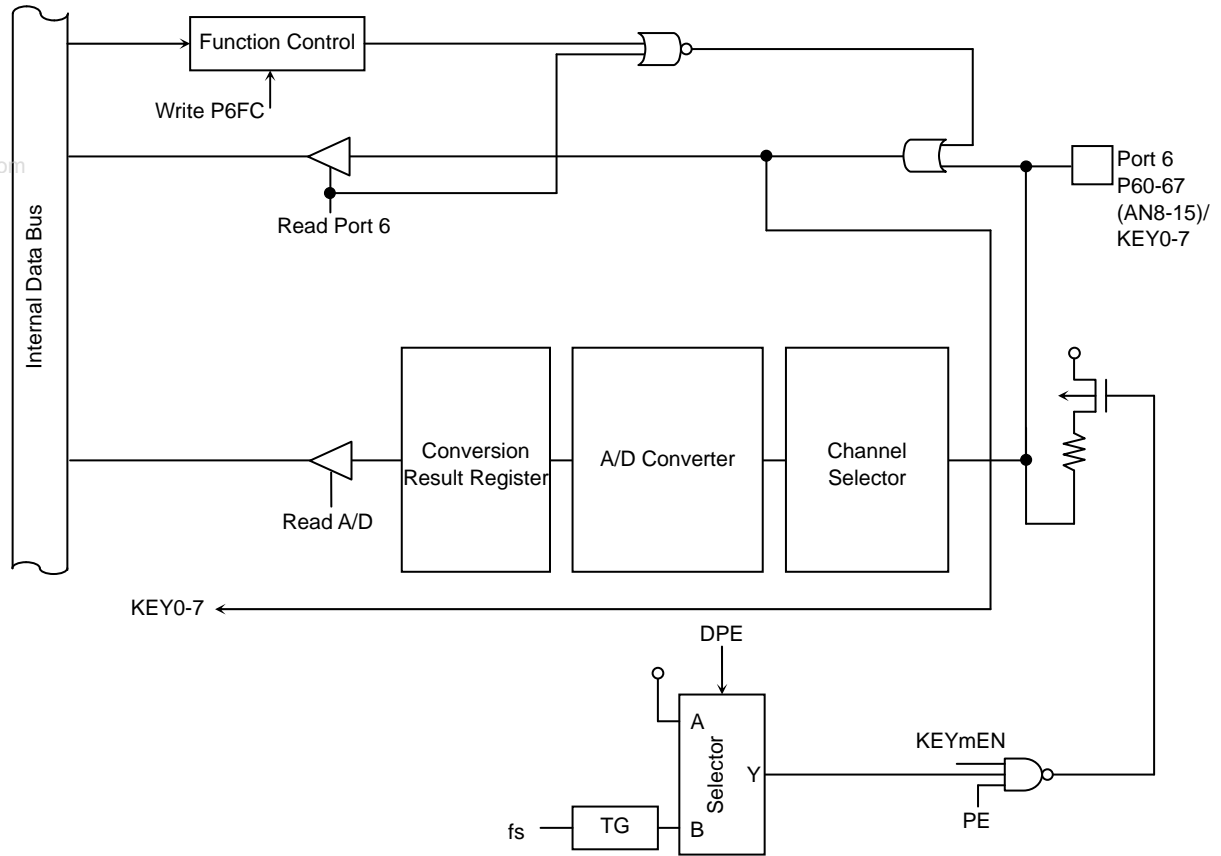
Figure 0.17 Port 6 (P60-P67)

Port 6 Register

		7	6	5	4	3	2	1	0
P6	Bit Symbol	P67	P66	P65	P64	P63	P62	P61	P60
(0xFFFF_F041)	Read/Write	R							
	After Reset	Input mode							

Port 6 Function Register

		7	6	5	4	3	2	1	0
P6FC	Bit Symbol	P67F	P66F	P65F	P64F	P63F	P62F	P61F	P60F
(0xFFFF_F045)	Read/Write	W							
	After Reset	0							
	Function	0: Port or A/D input 1: Key input							

Figure 0.18 Registers Related to Port 6

3.5.8   Port 9 (P90-P97)

Port 9 is an 8-bit general-purpose input/output port whose bits can each be set independently for input or output. The control register P9CR is used to set the port for input or output. A reset clears P9CR to 0, putting port 9 in input mode. In addition to functioning as an input/output port, the pins of this port can also function as various input/output pins: P90 and P91 function as key input, P92 to P94 and P97 as 16-bit timer output, and P95 and P96 as 16-bit timer input. These functions are enabled by setting the corresponding bits of P9FC to 1.

A reset clears P9CR and P9FC to 0, placing port 9 in input mode. Pins P90 and P91 have pull-up resistors, which are enabled only for those pins for which KWUPCNT<PE> is set to 1 in the key on wake-up circuit and key input is enabled in KWUPSTn. For details, refer to the description of the key on wake-up function. When a pin is functioning as a port pin, its pull-up resistor is disabled.

When the DSU is enabled, port 9 functions as a DSU interface regardless of the settings in P9CR and P9FC, so that the pins cannot be used as general-purpose port pins or peripheral function pins as described above.
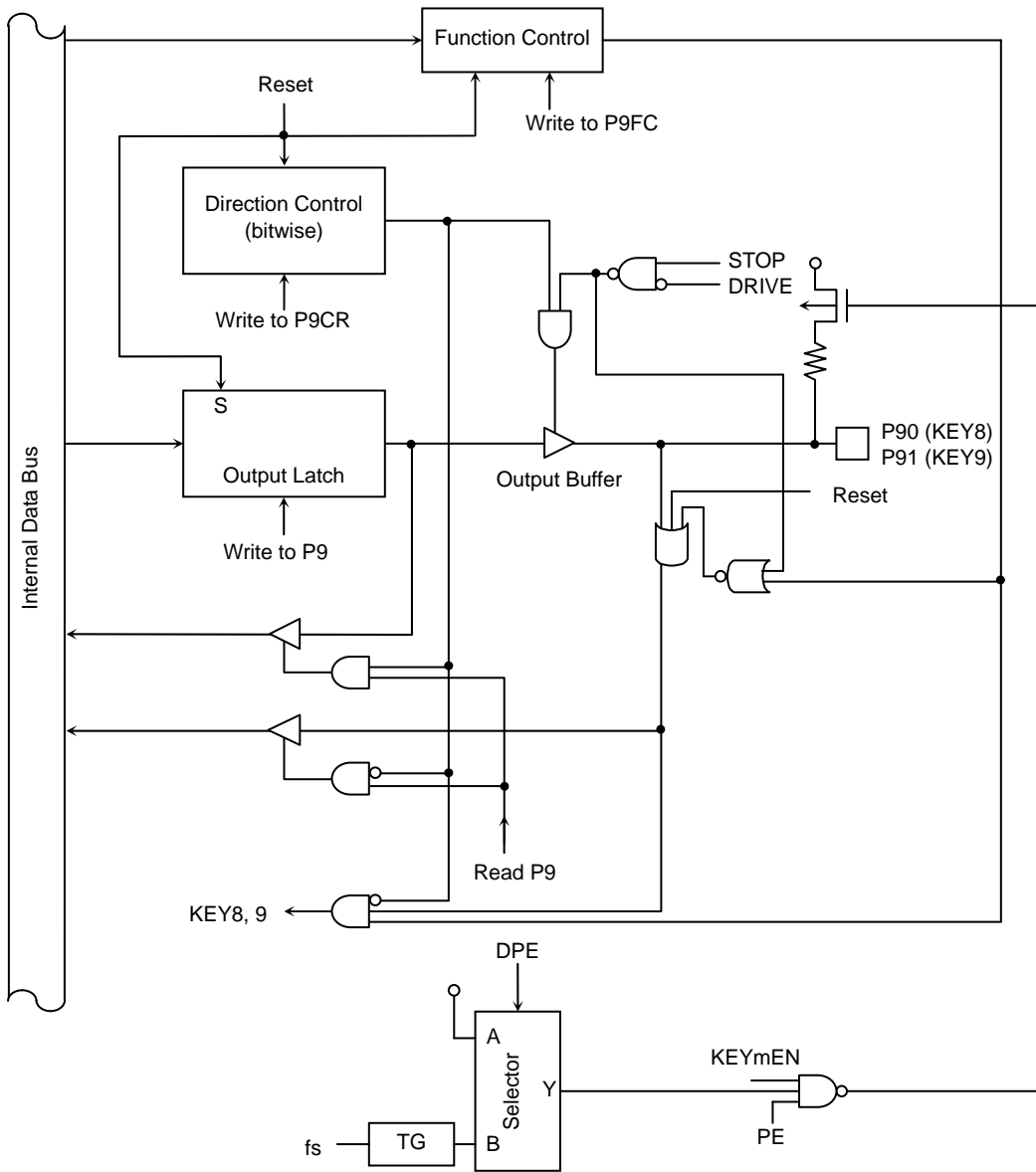
Figure 0.19 Port 9 (P90, P91)
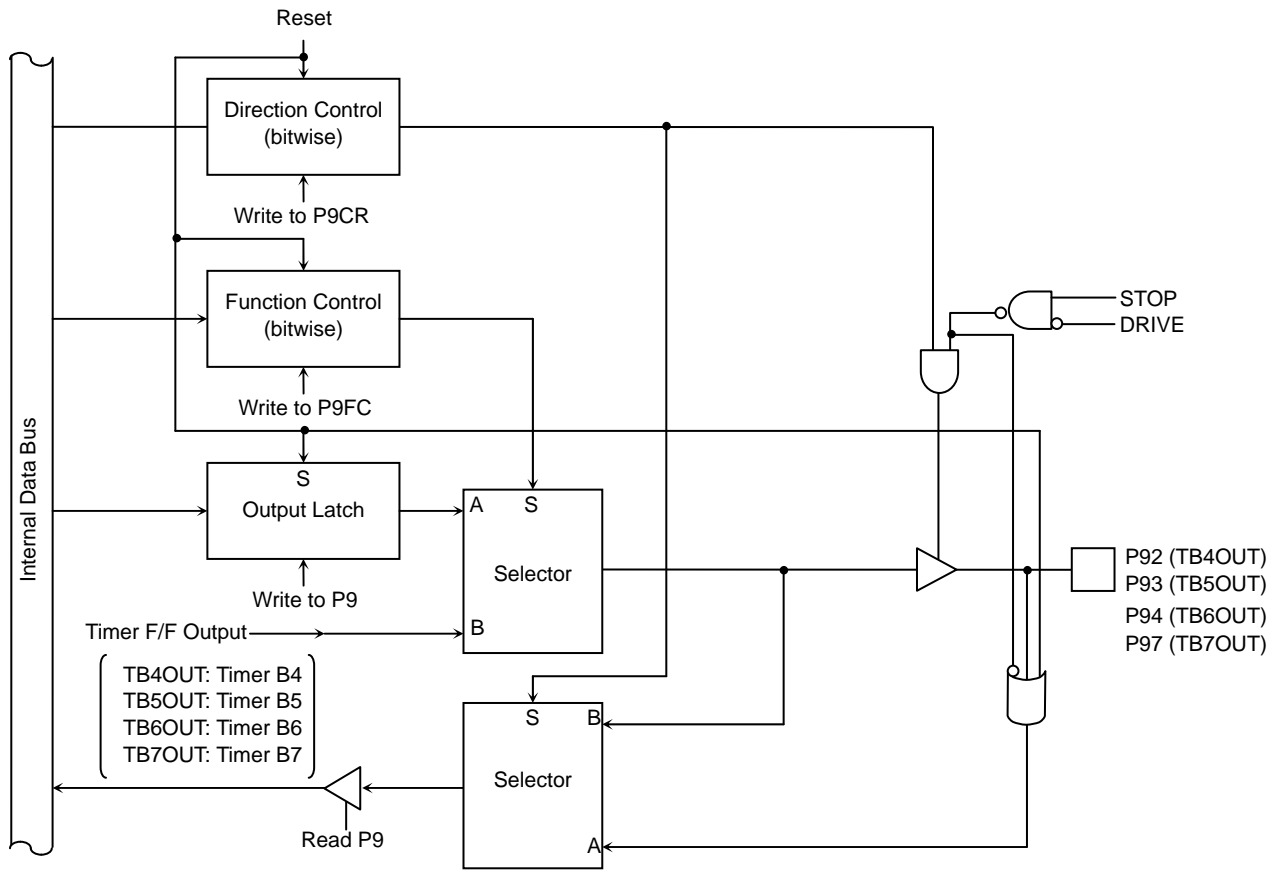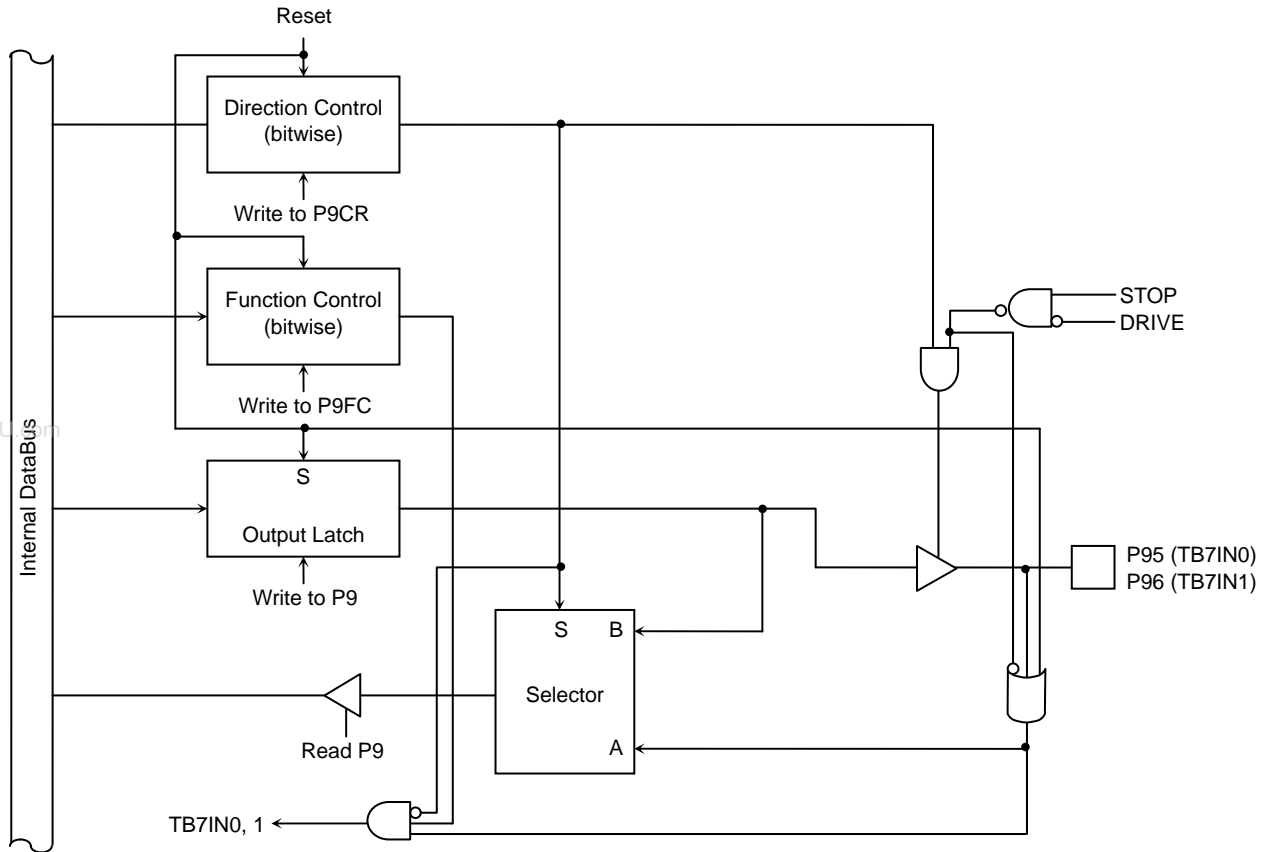
Figure 0.20 Port 9 (P92-P97)

Port 9 Register

		7	6	5	4	3	2	1	0
P9	Bit Symbol	P97	P96	P95	P94	P93	P92	P91	P90
(0xFFFF_F04C)	Read/Write				R/W				
	After Reset				Input mode (output latch register 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_F04E)	Read/Write				W				
	After Reset	0	0	0	0	0	0	0	0
	Function			0: IN		1: OUT			

Input/output settings for port 9

0	Input
1	Output

Port 9 Function Register

		7	6	5	4	3	2	1	0
P9FC	Bit Symbol	P97F	P96F	P95F	P94F	P93F	P92F	P91F	P90F
(0xFFFF_F04F)	Read/Write				W				
	After Reset	0	0	0	0	0	0	0	0
	Function	0: PORT 1: TB7OUT	0: PORT 1: TB7IN1	0: PORT 1: TB7IN0	0: PORT 1: TB6OUT	0: PORT 1: TB5OUT	0: PORT 1: TB4OUT	0: PORT 1: KEY9	0: PORT 1: KEY8

Function	Corresponding P9FC Bit	Corresponding P9CR Bit	Port Used
Select KEY8 input	1	0	P90
Select KEY9 input	1	0	P91
Select TB4OUT output	1	1	P92
Select TB5OUT output	1	1	P93
Select TB6OUT output	1	1	P94
Select TB7IN0 input	1	0	P95
Select TB7IN1 input	1	0	P96
Select TB7OUT output	1	1	P97

Figure 0.21 Registers Related to Port 9

3.5.9 Port A (PA0-PA7)

Port A is an 8-bit general-purpose input/output port whose bits can each be set independently for input or output. The control register PACR is used to set the port for input or output. A reset clears PACR to 0, putting port A in input mode. In addition to functioning as an input/output port, the pins of this port can also function as various input/output pins: PA0, PA1, PA3 and PA4 function as 16-bit timer input or external interrupt input, PA2 and PA5 as 16-bit timer output, PA6 as 8-bit timer output, and PA7 as 8-bit timer input or key input. These functions are enabled by setting the corresponding bits of PAFC to 1.

A reset clears PACR and PAFC to 0, placing port A in input mode. PA7 has a pull-up resistor, which is enabled only when KWUPCNT<PE> is set to 1 in the key on wake-up circuit and key input is enabled by setting 1 in PAFC. When the pin is functioning as a port pin, its pull-up resistor is disabled.
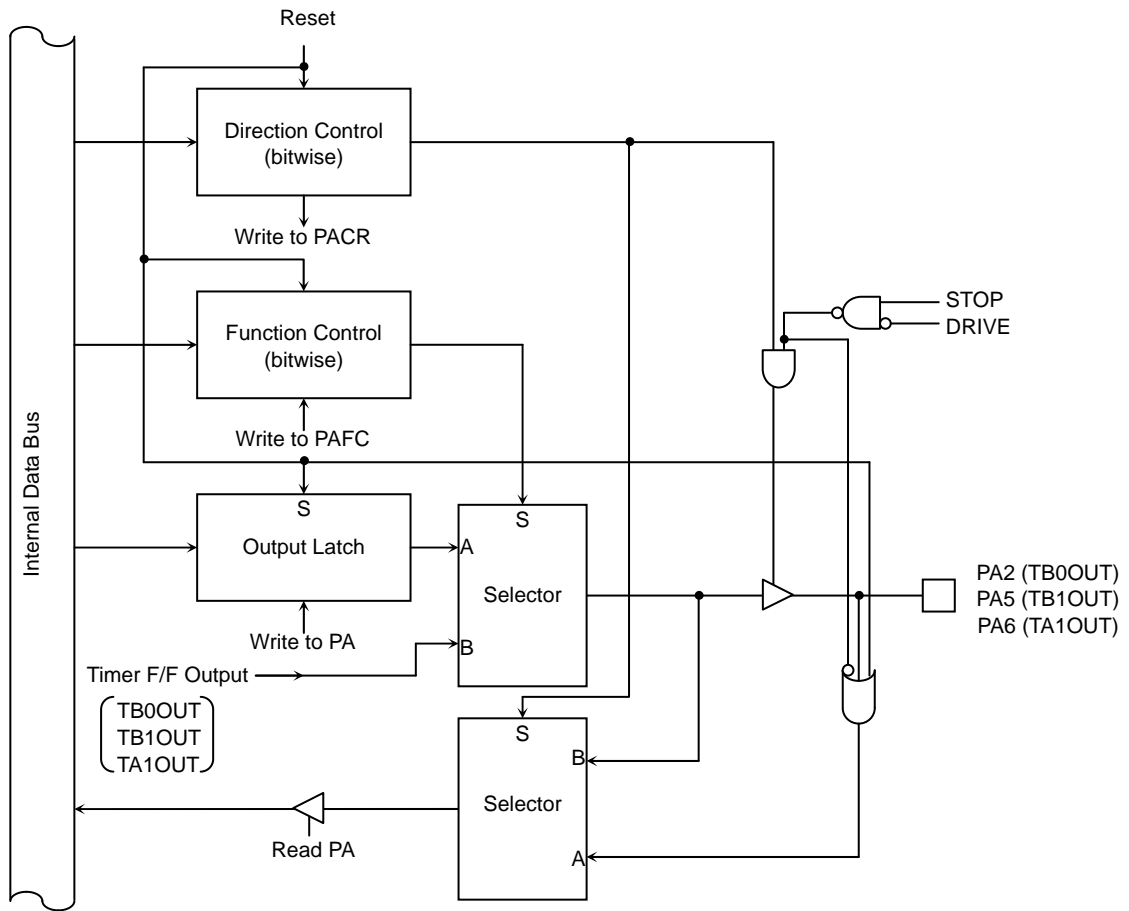
Figure 3.5.21 Port A (PA2, PA5, PA6)
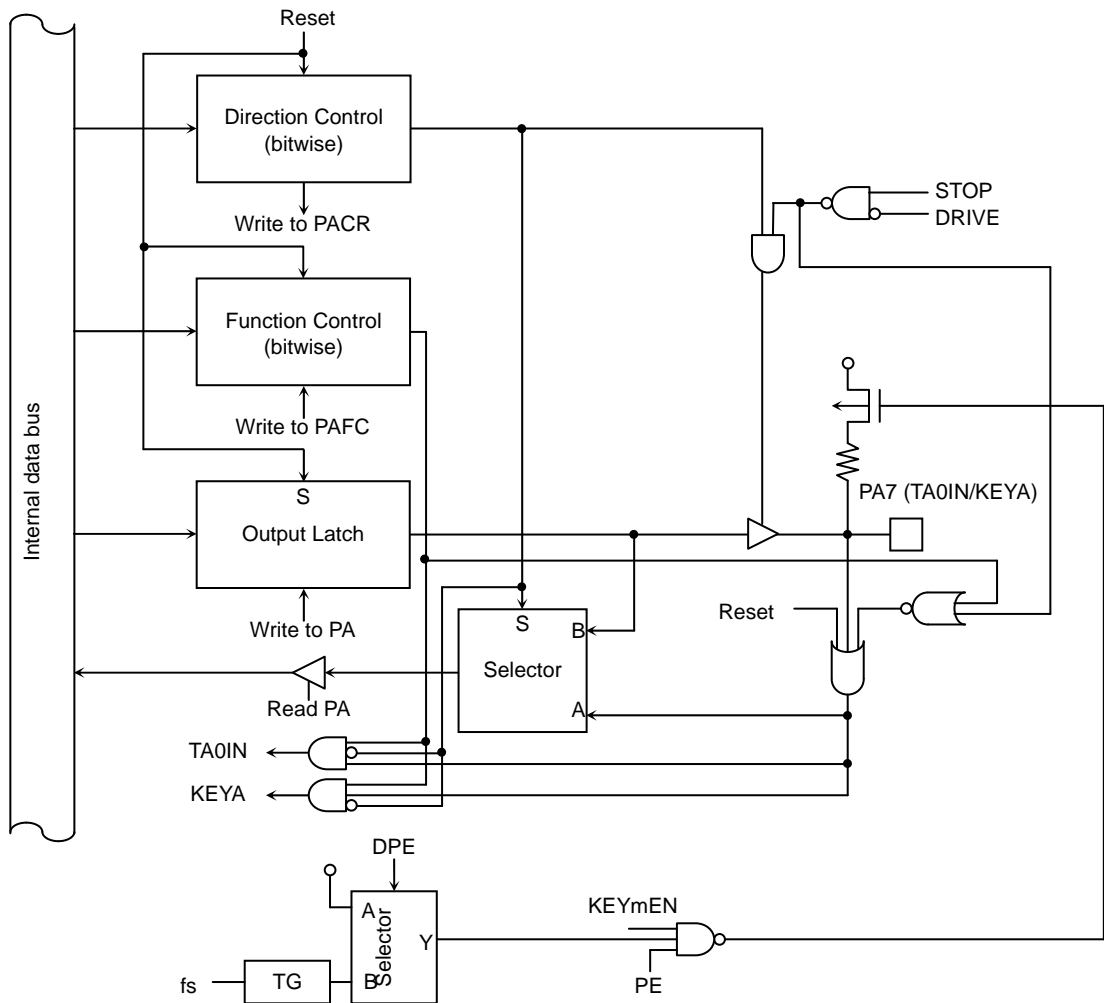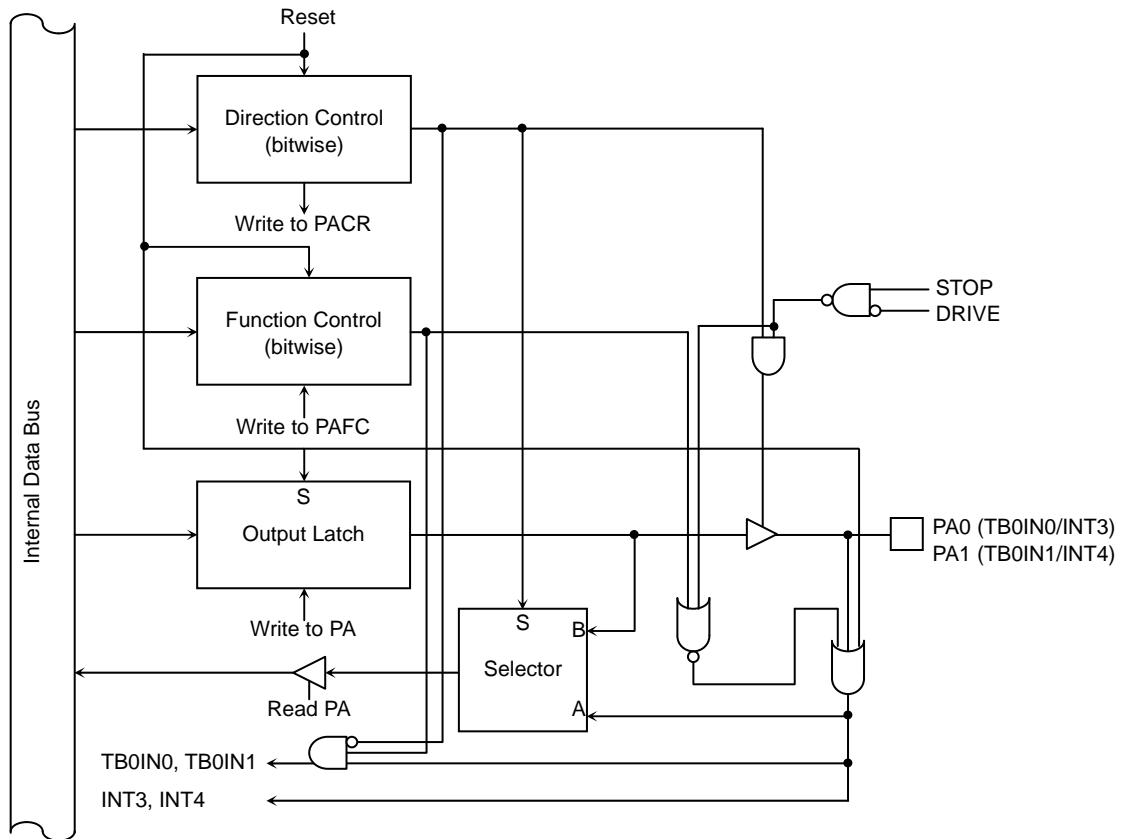
Figure 3.5.22 Port A (PA0, PA1, PA7)
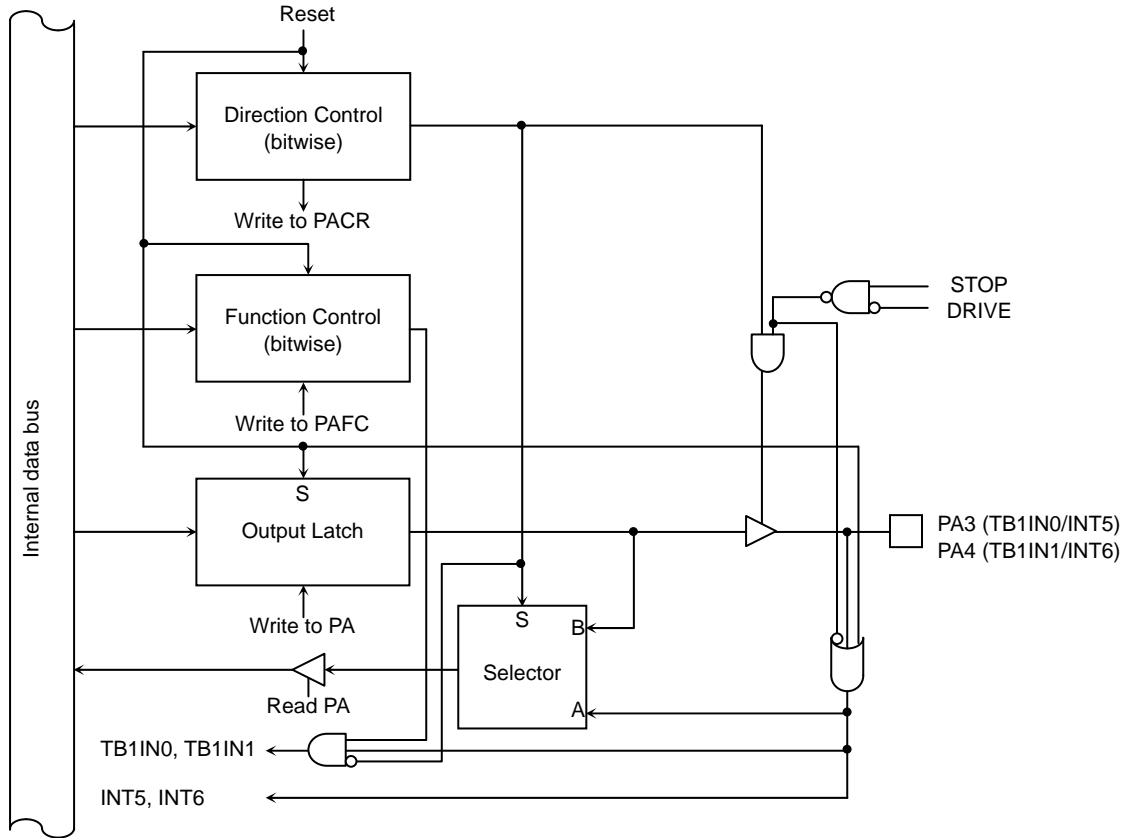
Figure 3.5.23 Port A (PA3, PA4)

Port A Register

	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 set to 1)							
	1	1	1	1	1	1	1	1

PA
(0xFFFF_F050)

Port A Control Register

	7	6	5	4	3	2	1	0
Bit Symbol	PA7C	PA6C	PA5C	PA4C	PA3C	PA2C	PA1C	PA0C
Read/Write	W							
After Reset	0	0	0	0	0	0	0	0
Function	0: IN 1: OUT							

PACR
(0xFFFF_F052)

Input/output settings for port A

0	Input
1	Output

Port A Function Register

	7	6	5	4	3	2	1	0
Bit Symbol	PA7F	PA6F	PA5F	PA4F	PA3F	PA2F	PA1F	PA0F
Read/Write	W							
After Reset	0	0	0	0	0	0	0	0
Function	0: PORT 1: TA0IN KEYA	0: PORT 1: TA1OUT	0: PORT 1: TB1OUT	0: PORT 1: TB1IN1 INT6	0: PORT 1: TB1INT0 INT5	0: PORT 1: TB0OUT	0: PORT 1: TB0IN1 INT4	0: PORT 1: TB0IN0 INT3

PAFC
(0xFFFF_F053)

Function	Corresponding PAFC Bit	Corresponding PACR Bit	Port Used
Select TB0IN0 input	1	0	PA0
Select INT3 input	1 (*1)	0	
Select TB0IN1 input	1	0	PA1
Select INT4 input	1 (*1)	0	
Select TB0OUT output	1	1	PA2
Select TB1IN0 input	1	0	PA3
Select INT5 input	Need not be set	0	
Select TB1IN1 input	1	0	PA4
Select INT6 input	Need not be set	0	
Select TB1OUT output	1	1	PA5
Select TA1OUT output	1	1	PA6
Select TA0IN input	1	0	PA7
Select KEYA input	1	0	

(*1)   Set this bit when using the pin for a STOP mode termination interrupt with
        SYSCR<DRVE> set to 0. Otherwise, the bit need not be set.

Note: For a pin to which two input functions are assigned in addition to the port function, use
        the control register for each function module to specify which function is used.

Figure 3.5.24 Registers Related to Port A

3.5.10   Port B (PB0-PB7)

Port B is an 8-bit general-purpose input/output port whose bits can each be set independently for input or output. The control register PBCR is used to set the port for input or output. A reset clears PBCR to 0, putting port B in input mode. In addition to functioning as an input/output port, the pins of this port can also function as various input/output pins: PB0, PB1, PB3 and PB4 function as 16-bit timer input or external interrupt input, PB2 and PB5 as 16-bit timer input or output, PB7 as 8-bit timer input, interrupt input or key input. These functions are enabled by setting the corresponding bits of PBFC to 1.

A reset clears PBCR and PBFC to 0, placing port B in input mode. PB7 has a pull-up resistor, which is enabled only when KWUPCNT<PE> is set to 1 in the key on wake-up circuit and key input is enabled in KWUPSTn. For details, refer to the description of the key on wake-up function. When the pin is functioning as a port pin, its pull-up resistor is disabled.
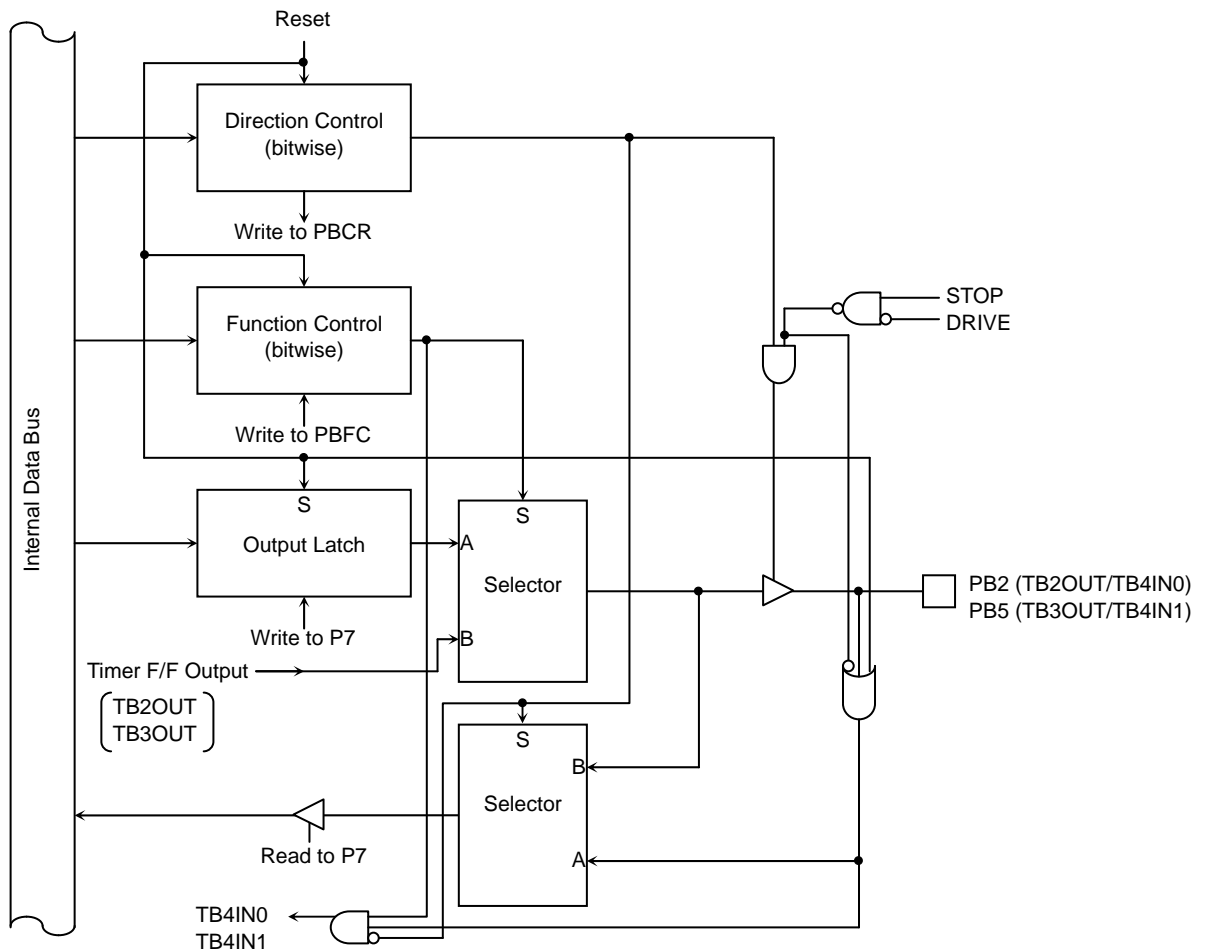
Figure 3.5.25 Port B (PB2, PB5)
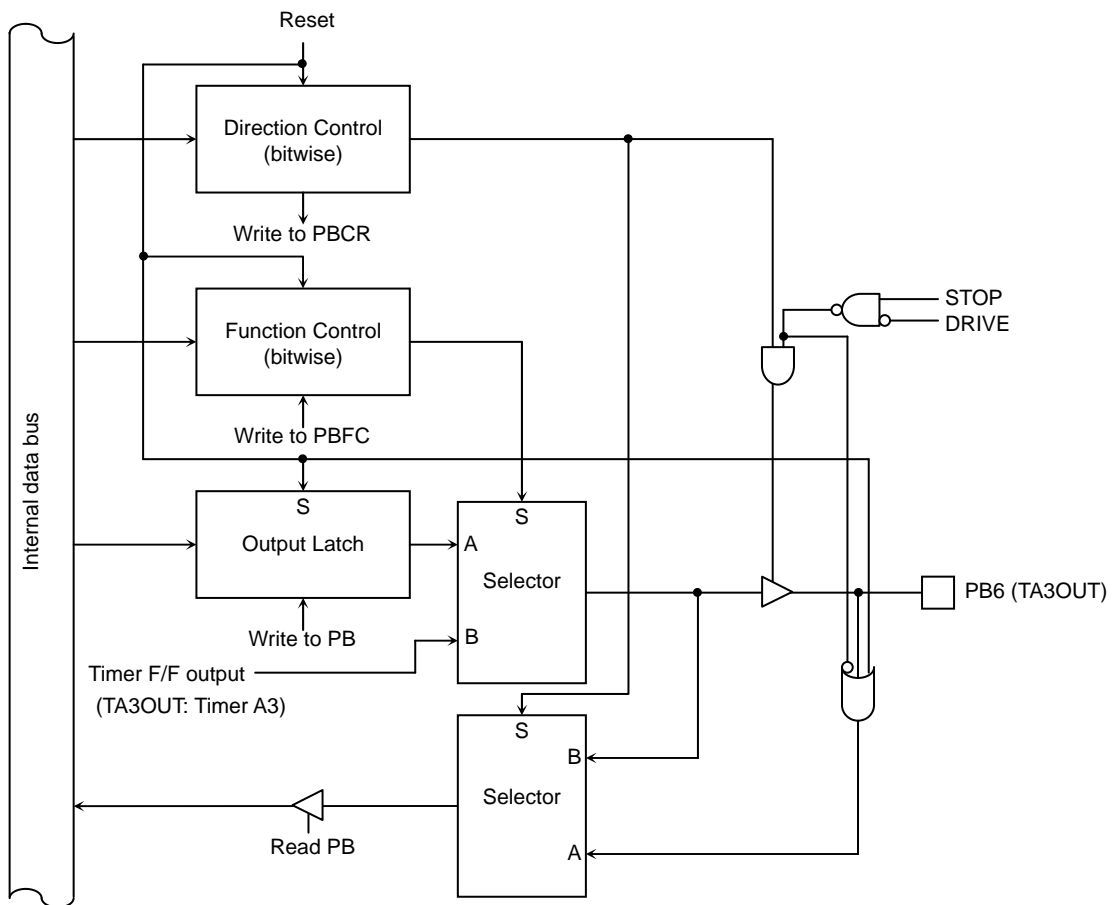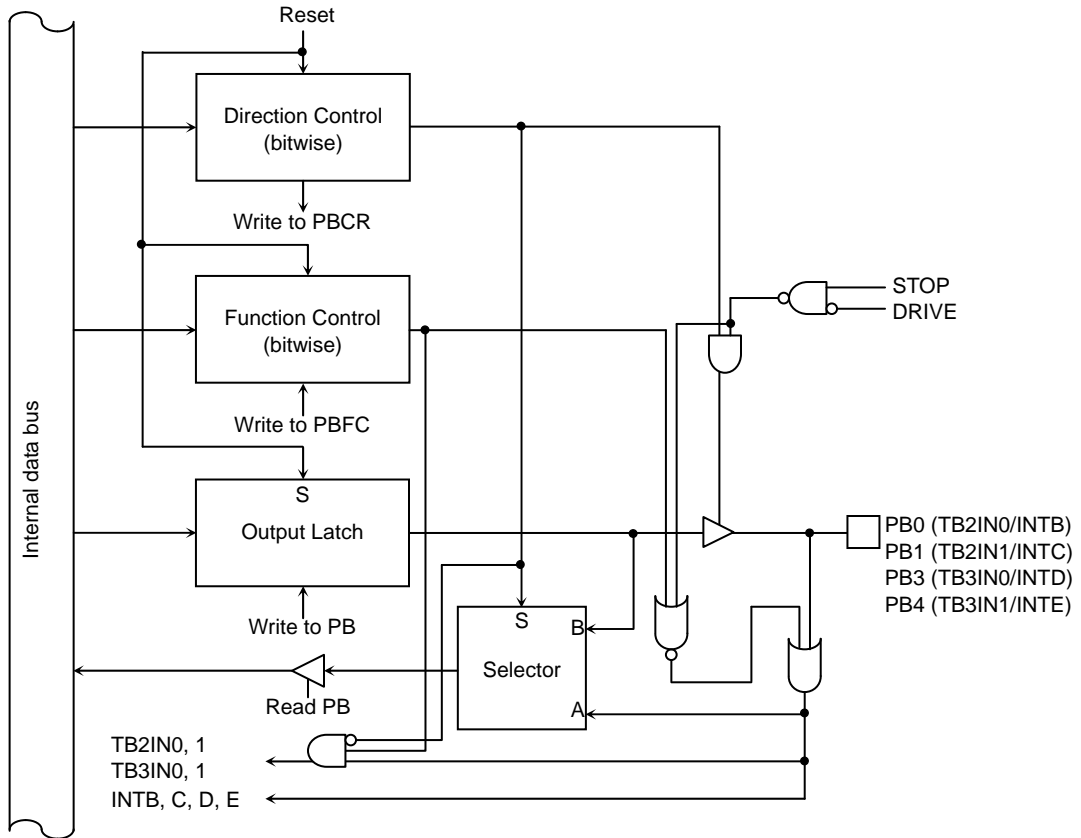
Figure 3.5.26 Port B (PB0, PB1, PB3, PB4, PB6)
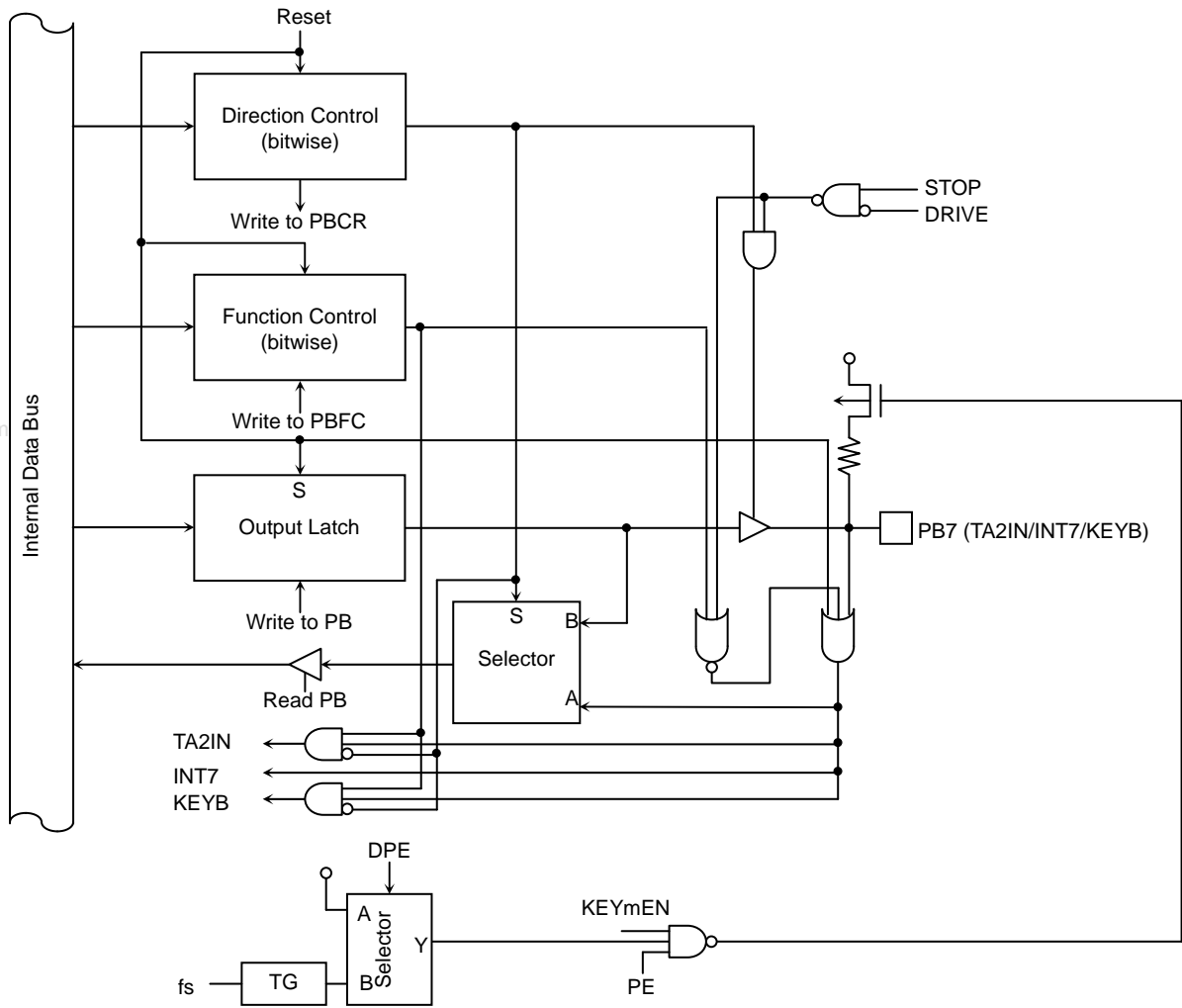
Figure 3.5.27 Port B (PB7)

Port B Register

		7	6	5	4	3	2	1	0
PB (0xFFFF_F051)	Bit Symbol	PB7	PB6	PB5	PB4	PB3	PB2	PB1	PB0
	Read/Write	R/W							
	After Reset	Input mode (output latch register set to 1)							

Port B Control Register

		7	6	5	4	3	2	1	0
PBCR (0xFFFF_F054)	Bit Symbol	PB7C	PB6C	PB5C	PB4C	PB3C	PB2C	PB1C	PB0C
	Read/Write	W							
	After Reset	0	0	0	0	0	0	0	0
	Function	0: IN 1: OUT							

Input/output settings for port B

0	Input
1	Output

Port B Function Register

		7	6	5	4	3	2	1	0
PBFC (0xFFFF_F055)	Bit Symbol	PB7F	PB6F	PB5F	PB4F	PB3F	PB2F	PB1F	PB0F
	Read/Write	W							
	After Reset	0	0	0	0	0	0	0	0
	Function	0:PORT 1:TA2IN INT7 KEYB	0: PORT 1: TA3OUT	0: PORT 1: TB3OUT TB4IN1	0: PORT 1: INTE TB3IN1	0: PORT 1: INTD TB3IN0	0: PORT 1: TB2OUT TB4IN0	0: PORT 1: INTC TB2IN1	0: PORT 1: INTB TB2IN0

Function	Corresponding PBFC Bit	Corresponding PBCR Bit	Port Used
Select TB2IN0 input	1	0	PB0
Select INTB input	1 (*1)	0	
Select TB2IN1 input	1	0	PB1
Select INTC input	1 (*1)	0	
Select TB2OUT output	1	1	PB2
Select TB4IN0 input	0	1	
Select TB3IN0 input	1	0	PB3
Select INTD input	1 (*1)	0	
Select TB3IN1 input	1	0	PB4
Select INTE input	1 (*1)	0	
Select TB3OUT output	1	1	PB5
Select TB4IN1 input	0	1	
Select TA3OUT output	1	1	PB6
Select TA2IN input	1	0	PB7
Select INT7 input	1	0	
Select KEYB input	1	0	

(*1)   Set this bit when using the pin for a STOP mode termination interrupt with
SYSCR<DRVE> set to 0. Otherwise, the bit need not be set.

Note: For a pin to which two or three input functions are assigned in addition to the port
function, use the control register for each function module to specify which function is
used.

Figure 3.5.28 Registers Related to Port B

3.5.11 Port C (PC0-PC7)

Port C is an 8-bit general-purpose input/output port whose bits can each be set independently for input or output. The control register PCCR is used to set the port for input or output. A reset clears PCCR to 0, putting port C in input mode. In addition to functioning as an input/output port, the pins of this port can also function as various input/output pins: PC0, PC1 and PC2 function as 8-bit timer input or external interrupt input, PC3 and PC5 as 8-bit timer output, PC6 as 16-bit timer input or key input, PC4 as 8-bit timer input, and PC7 as 16-bit timer input or 8-bit timer output. These functions are enabled by setting the corresponding bits of PCFC to 1.

A reset clears PCCR and PCFC to 0, placing port C in input mode. PC6 has a pull-up resistor, which is enabled only when KWUPCNT<PE> is set to 1 in the key on wake-up circuit and key input is enabled in KWUPSTn. For details, refer to the description of the key on wake-up function. When the pin is functioning as a port pin, its pull-up resistor is disabled.

Port C becomes a 5 V input/output port when 5 V is supplied to its dedicated power supply pin DVCC52. It becomes a VCC-based (3 V) port when VCC is supplied to DVCC52.
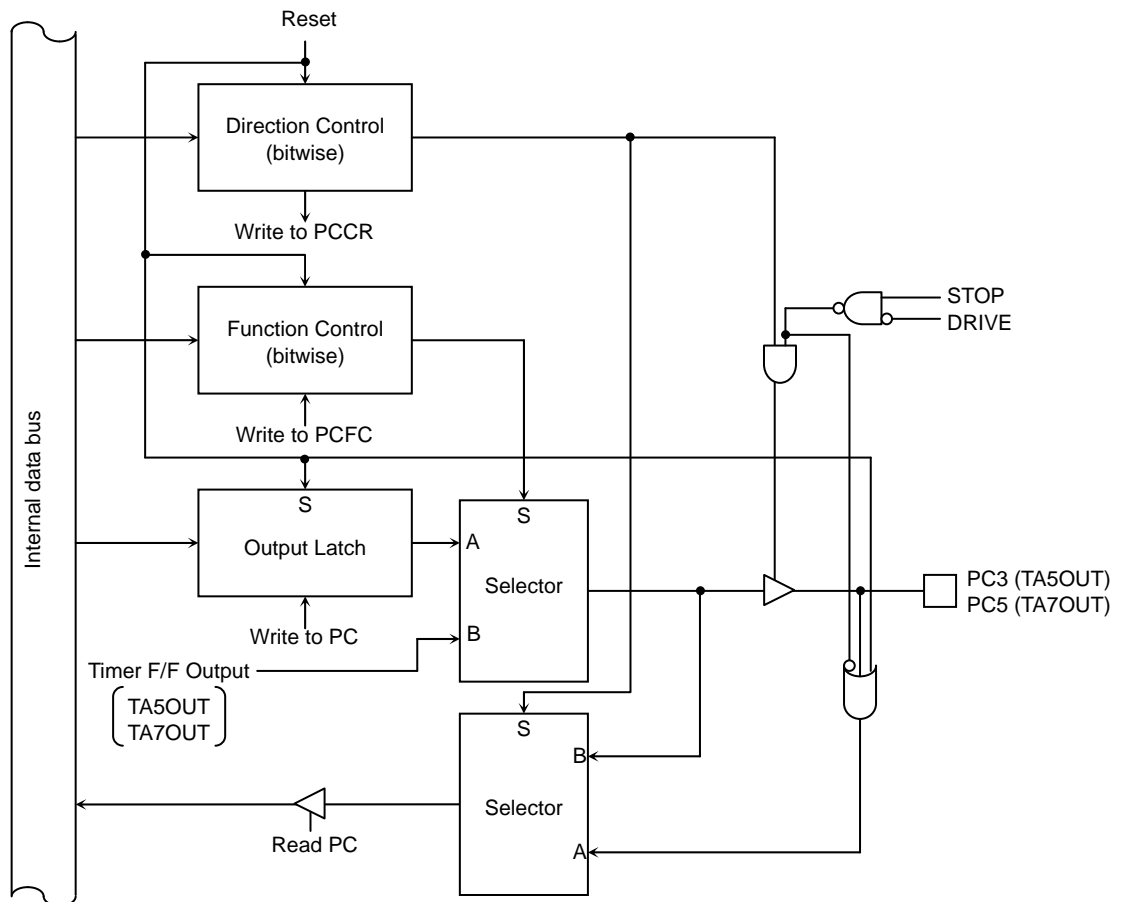
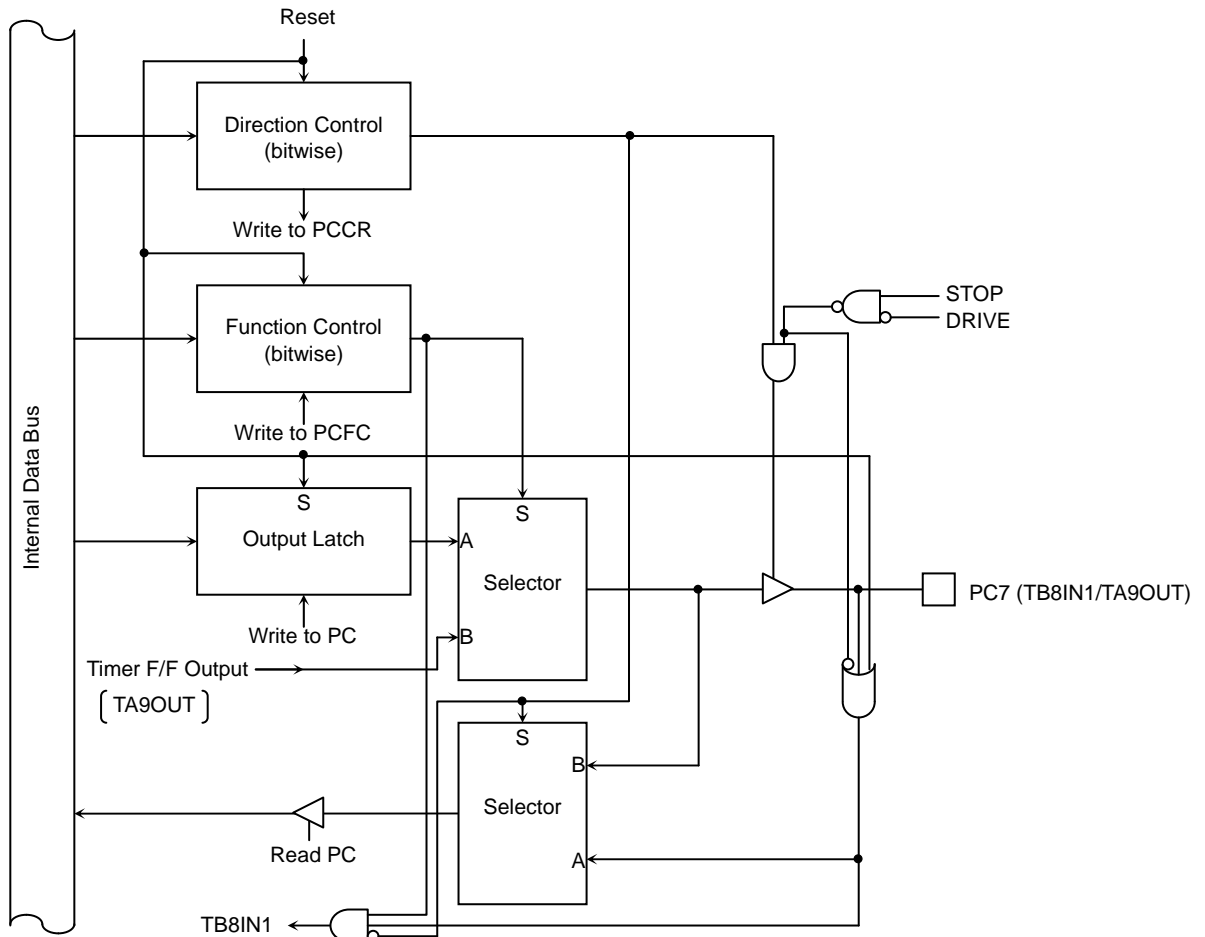Figure 3.5.29 Port C (PC3, PC5)
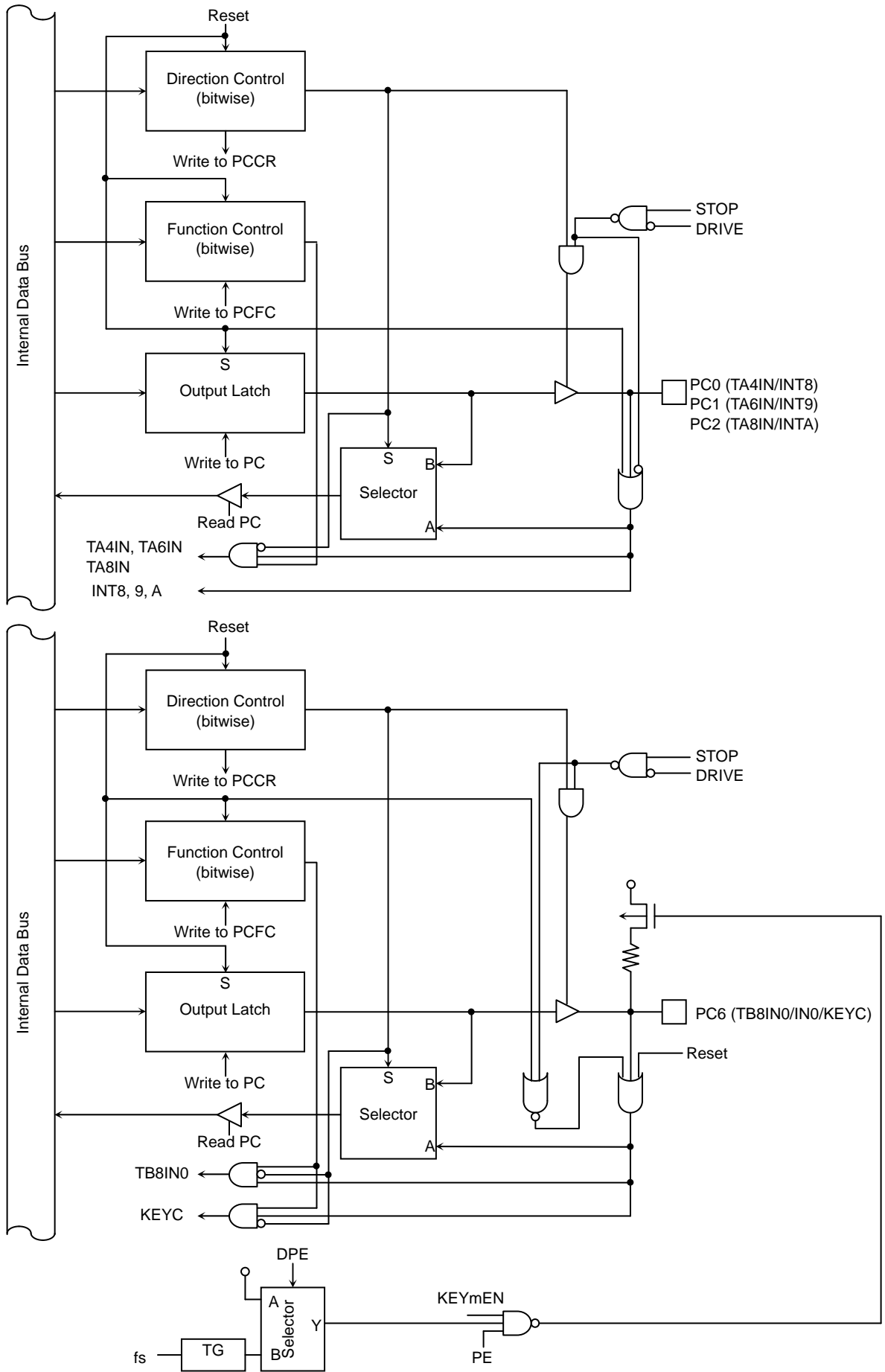
Figure 3.5.30 Port C (PC0, PC1, PC2, PC6)
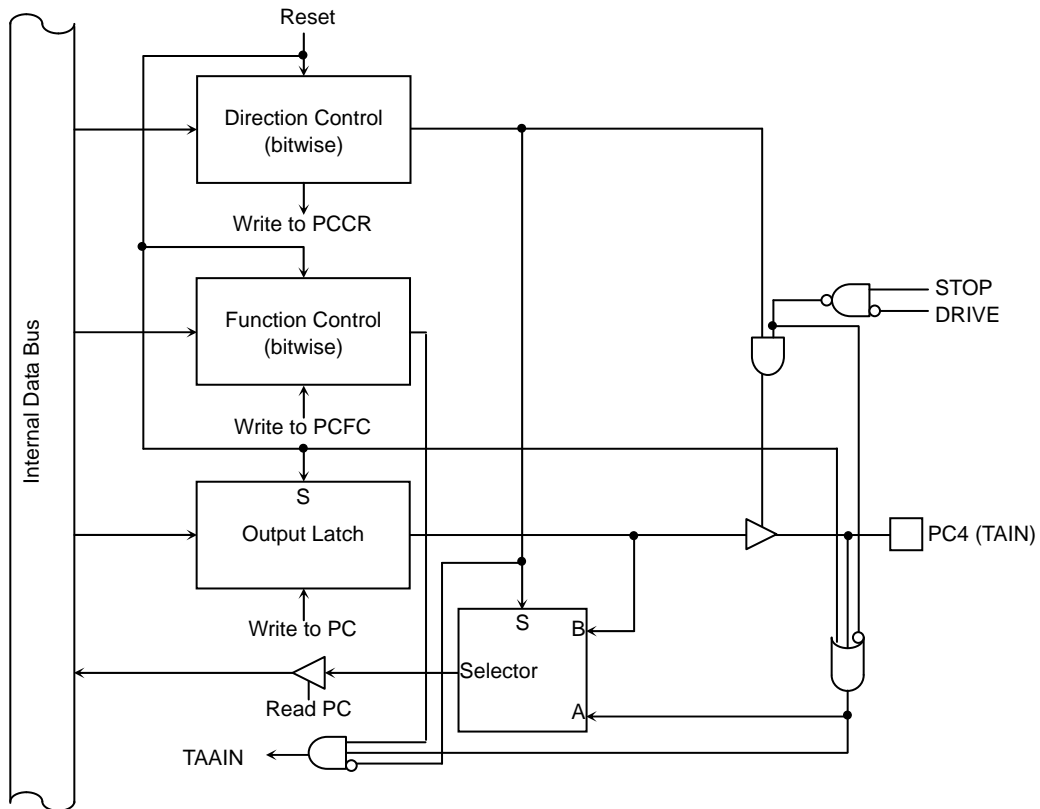
Figure 3.5.31 Port C (PC7, PC4)

Port C Register

	7	6	5	4	3	2	1	0
Bit Symbol	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
Read/Write	R/W							
After Reset	Input mode (output latch register set to 1)							

PC
(0xFFFF_F058)

Port C Control Register

	7	6	5	4	3	2	1	0
Bit Symbol	PC7C	PC6C	PC5C	PC4C	PC3C	PC2C	PC1C	PC0C
Read/Write	W							
After Reset	0	0	0	0	0	0	0	0
Function	0: IN 1: OUT							

PCCR
(0xFFFF_F05A)

Port C Control Register	
0	Input
1	Output

Port C Function Register

	7	6	5	4	3	2	1	0
Bit Symbol	PC7F	PC6F	PC5F	PC4F	PC3F	PC2F	PC1F	PC0F
Read/Write	W							
After Reset	0	0	0	0	0	0	0	0
Function	0:PORT 1:TB8IN TA9OUT	0: PORT 1: KEYC TB8IN0	0: PORT 1: TA7OUT	0: PORT 1: TAAIN	0: PORT 1: TA5OUT	0: PORT 1: TA8IN INTA	0: PORT 1: TA6IN INT9	0: PORT 1: TA4IN INT8

PCFC
(0xFFFF_F05B)

Function	Corresponding PCFC Bit	Corresponding PCCR Bit	Port Used
Select TA4IN input	1	0	PC0
Select INT8 input	Need not be set	0	
Select TA6IN input	1	0	PC1
Select INT9 input	Need not be set	0	
Select TA8IN input	1	0	PC2
Select INTA input	Need not be set	0	
Select TA5OUT output	1	1	PC3
Select TAAIN input	1	0	PC4
Select TA7OUT output	1	1	PC5
Select TB8IN0 input	1	0	PC6
Select KEYC input	1	0	
Select TB8IN1 input	1	0	PC7
Select TA9OUT output	1	1	

Note: For a pin to which two input functions are assigned in addition to the port function, use the control register for each function module to specify which function is used.

Figure 3.5.32 Registers Related to Port C

3.5.12   Port D (PD0-PD7)

Port D is an 8-bit general-purpose input/output port whose bits can each be set independently for input or output. The control register PDCR is used to set the port for input or output. A reset clears PDCR to 0, putting port D in input mode. In addition to functioning as an input/output port, the pins of this port can also function as various input/output pins: PD0 and PD3 function as 16-bit timer input or SIO data output, PD1 and PD4 function as 16-bit timer input or SIO data input, PD2 as SIO serial clock input/output or CTS*input, and PD5 as SIO serial clock input/output, CTS*input, or 16-bit timer output. PD6 and PD7 can be connected to a low-frequency oscillator. These functions are enabled by setting the corresponding bits of PDFC1 to 1. For PD5, however, a combination of PDFC1 and PDFC2 determines whether it is used for a port, SIO, or timer. The output open-drain control register (PDODE) can be used to set PD0, PD2, PD3 and PD5 to open-drain output when they are used for output. PD6 and PD7 are always open-drain output when they are used for output.

A reset clears PDCR, PDFC1 and PDFC2 to 0, placing port D in input mode.

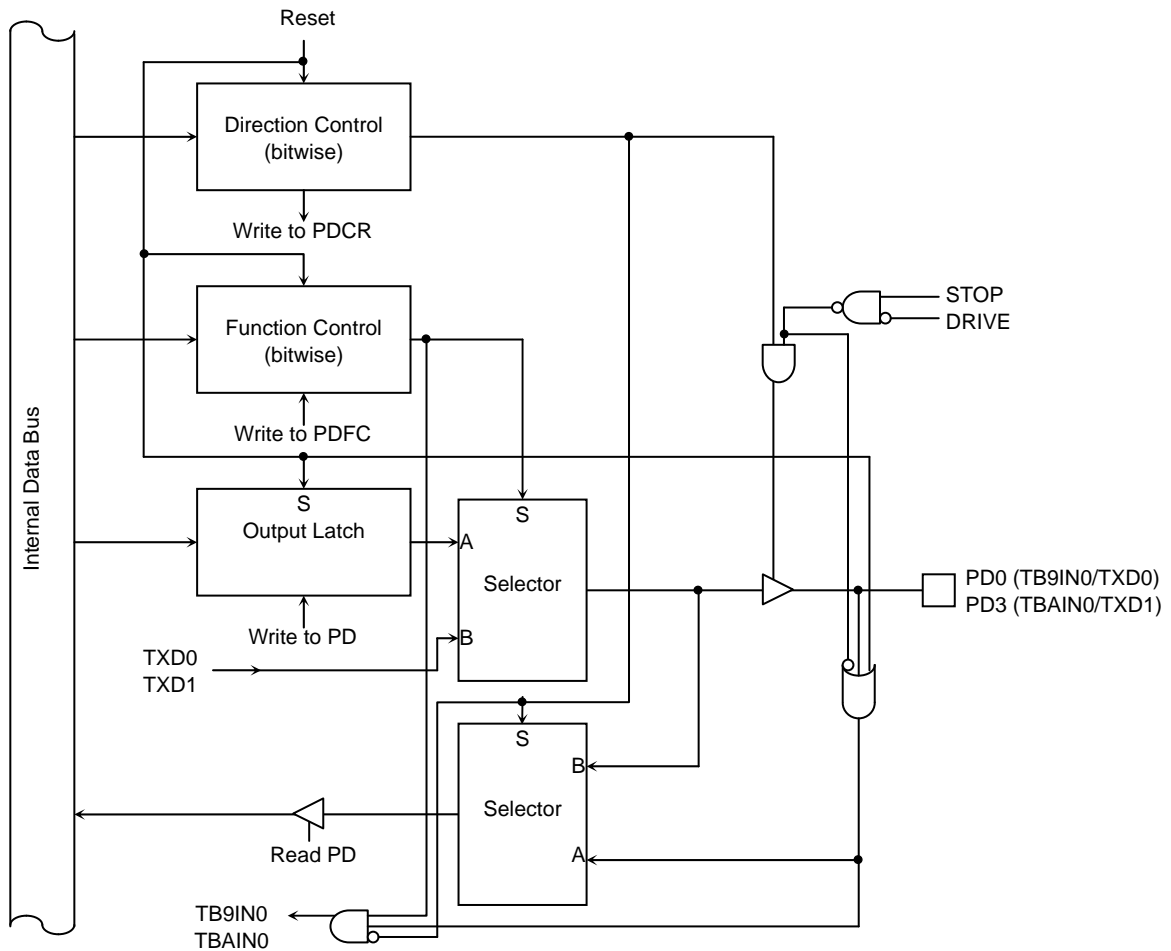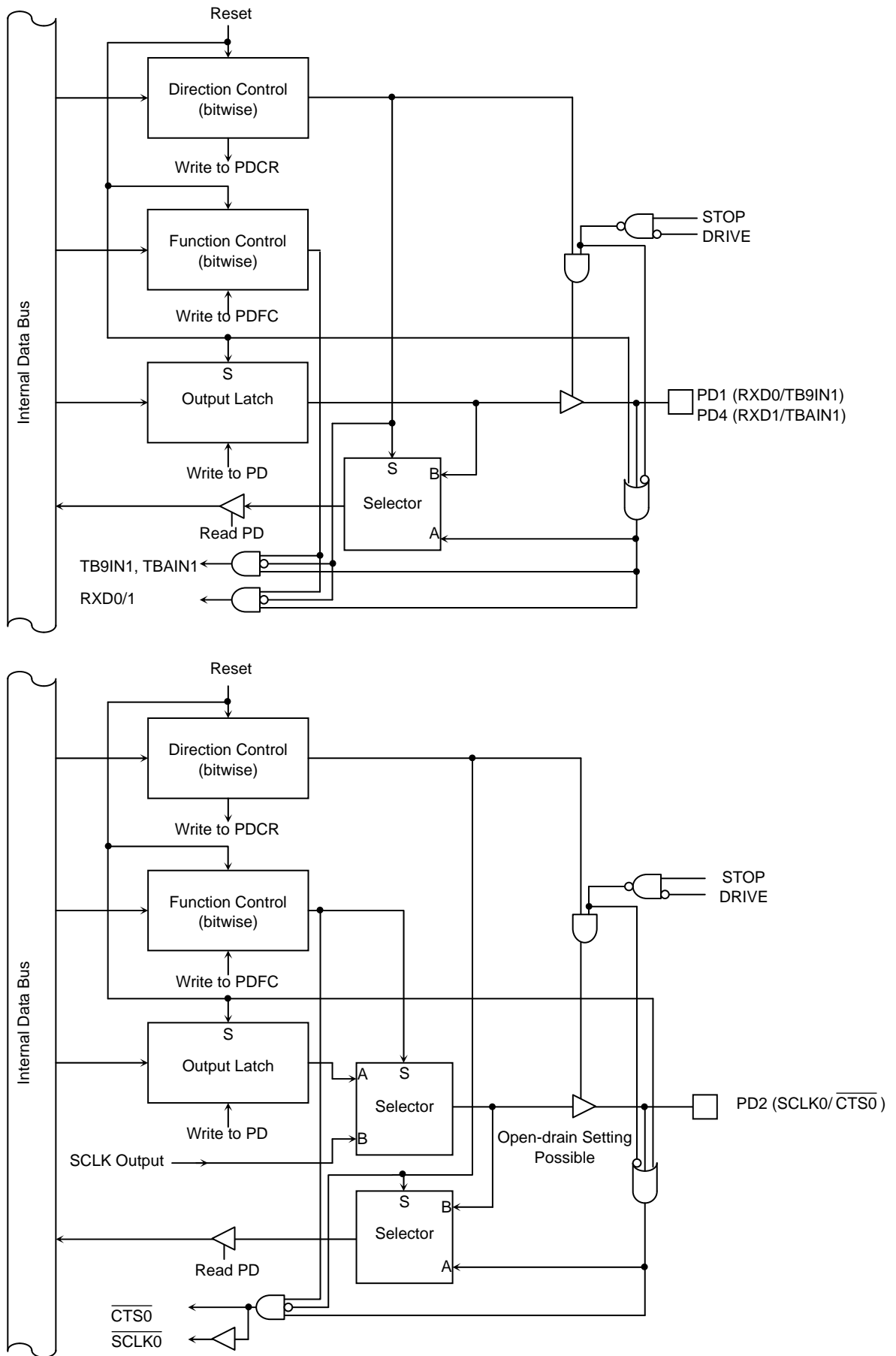Figure 3.5.33 Port D (PD0, PD3)

www.DataSheet4U.com

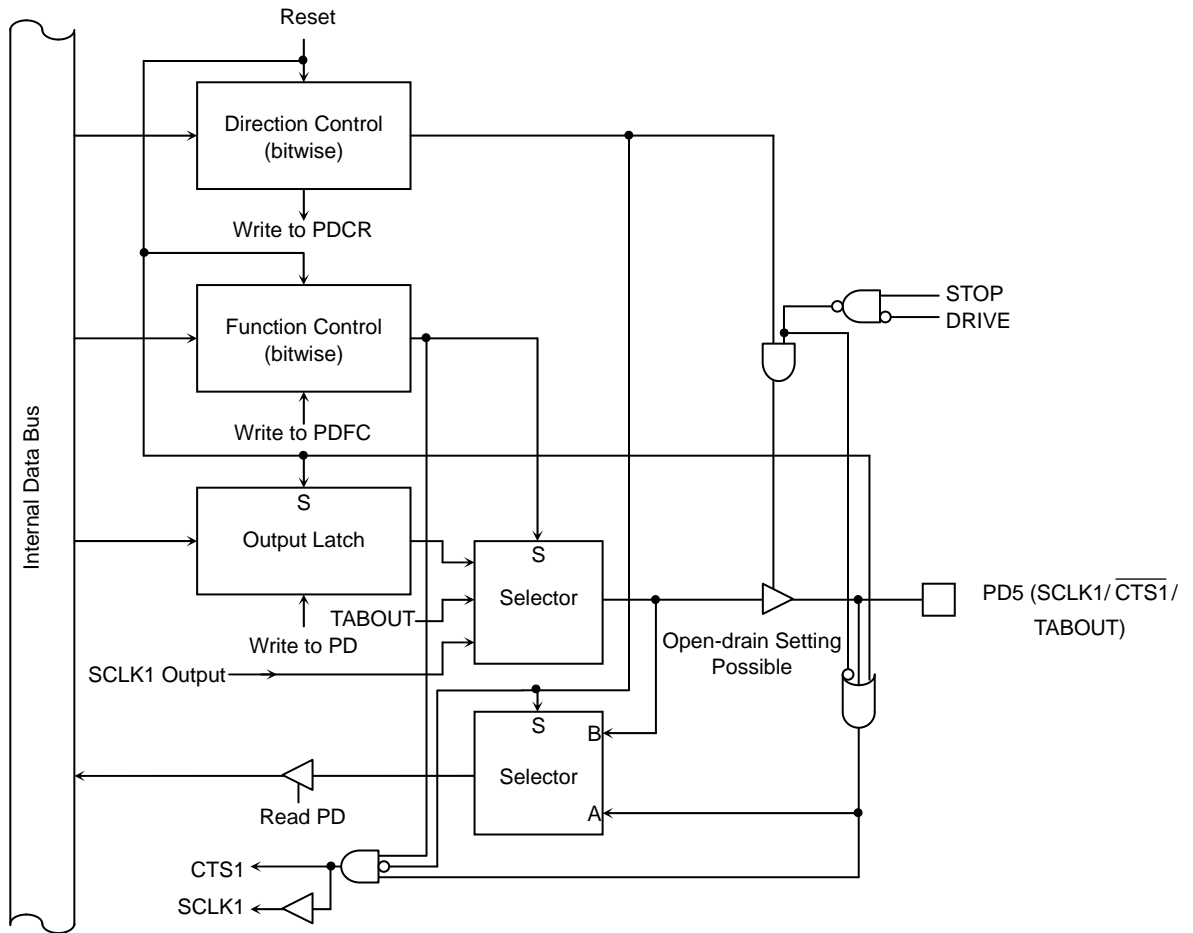Figure 3.5.34 Port D (PD1, PD4, PD2)

www.DataSheet4U.com

Figure 3.5.35 Port D (PD5)

Note: The output mode is selected by a combination of PDFC1 and PDFC2. When PDFC1<bit 5> = 0 and PDFC2<bit 5> = 0, port output is selected. When PDFC1<bit 5> = 1 and PDFC2<bit 5> = 0, SCLK output is selected. When PDFC1<bit 5> = 0 and PDFC2<bit 5> = 1, TABOUT output is selected. Setting both PDFC1<bit 5> and PDFC2<bit 5> to 1 is not allowed.
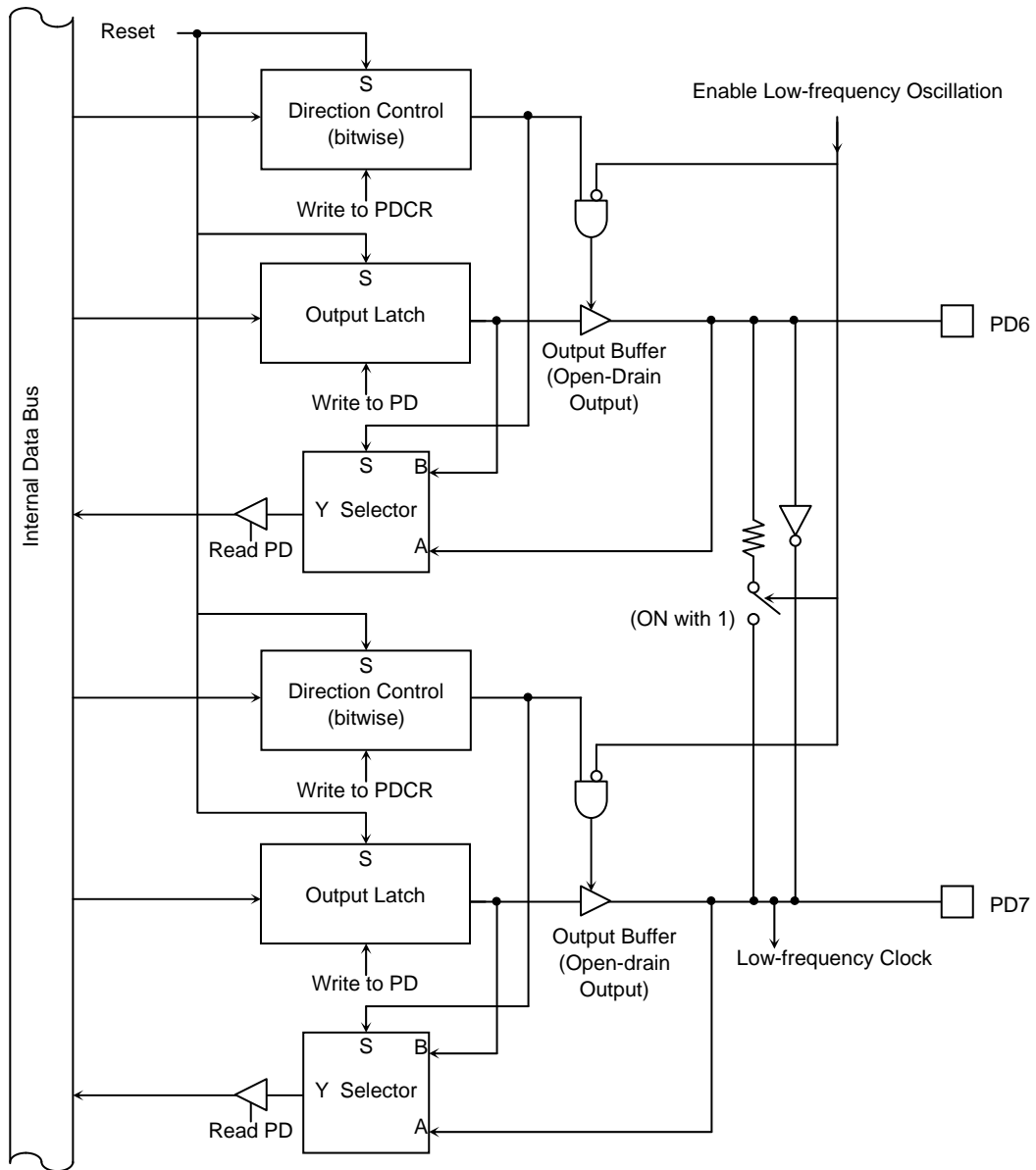
Figure 3.5.36 Port D (PD6, PD7)

Port D Register

		7	6	5	4	3	2	1	0
PD	Bit Symbol	PD7	PD6	PD5	PD4	PD3	PD2	PD1	PD
(0xFFFF_F059)	Read/Write	R/W							
	After Reset	Input mode (output latch register set to 1)							

Port D Control Register

		7	6	5	4	3	2	1	0
PDCR	Bit Symbol	PD7C	PD6C	PD5C	PD4C	PD3C	PD2C	PD1C	PD0C
(0xFFFF_F05C)	Read/Write	W							
	After Reset	1	1	0	0	0	0	0	0
	Function			0: IN	1: OUT				

Input/output settings for port D

0	Input
1	Output

Port D Function Register 1

		7	6	5	4	3	2	1	0
PDFC1	Bit Symbol	—	—	PD5F	PD4F	PD3F	PD2F	PD1F	PD0F
(0xFFFF_F05D)	Read/Write	W							
	After Reset	—	—	0	0	0	0	0	0
	Function			0: PORT 1: SCLK1/ CTS1*	0: PORT 1: TBAIN1 RXD1	0: PORT 1: TBAIN0 TXD1	0: PORT 1: SCLK0/ CTS0*	0: PORT 1: TB9IN1 RXD0	0: PORT 1: TB9IN0 TXD0

Port D Function Register 2

		7	6	5	4	3	2	1	0
PDFC2	Bit Symbol	—	—	PD5F2	—	—	—	—	—
(0xFFFF_F05E)	Read/Write	—	—	W	—	—	—	—	—
	After Reset	—	—	0	—	—	—	—	—
	Function			0: PORT 1: TABOUT					

Port D Open-drain Control Register

		7	6	5	4	3	2	1	0
PDODE	Bit Symbol	—	—	PDODE5	—	PDODE3	PDODE2	—	PDODE0
(0xFFFF_F05F)	Read/Write	W							
	After Reset	—	—	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

Function	Corresponding PDFC1 Bit	Corresponding PDFC2 Bit	Corresponding PDCR Bit	Port Used
Select TB9IN0 input	1	Need not be set (no bit provided)	0	PD0
Select TXD0 output	1	Need not be set (no bit provided)	1	
Select TB9IN1 input	1	Need not be set (no bit provided)	0	PD1
Select RXD0 input	1	Need not be set (no bit provided)	0	
Select SCLK0 input	1	Need not be set (no bit provided)	0	PD2
Select SCLK0 output	1	Need not be set (no bit provided)	1	
Select CTS0* input	1	Need not be set (no bit provided)	0	
Select TBAIN0 input	1	Need not be set (no bit provided)	0	PD3
Select TXD1 output	1	Need not be set (no bit provided)	1	
Select TBAIN1 input	1	Need not be set (no bit provided)	0	PD4
Select RXD1 input	1	Need not be set (no bit provided)	0	
Select SCLK1 input	1	0	0	PD5
Select SCLK1 output	1	0	1	
Select $\overline{\text{CTS1}}$ input	1	0	0	
Select TABOUT output	0	1	1	

Note: For a pin to which two input functions are assigned in addition to the port function, use the control register for each function module to specify which function is used.

Figure 3.5.37 Registers Related to Port D

3.5.13   Port E (PE0-PE7)

Port E is an 8-bit general-purpose input/output port whose bits can each be set independently for input or output. The control register PECR is used to set the port for input or output. A reset clears PECR to 0, putting port E in input mode. In addition to functioning as an input/output port, the pins of this port can also function as various input/output pins: PE0 and PE3 function as SIO data output, PE1 and PE4 as SIO data input, PE2 and PE5 as SIO CLK input/output or CTS* input, and PE6 and PE7 as external interrupt input. These functions are enabled by setting the corresponding bits of PEFC to 1.

A reset clears PECR and PEFC to 0, placing port E in input mode. The output open-drain control register (PEODE) can be used to set PE0, PE2, PE3 and PE5 to open-drain output when they are used for output.

Figure 3.5.38 Port E (PE0, PE1)
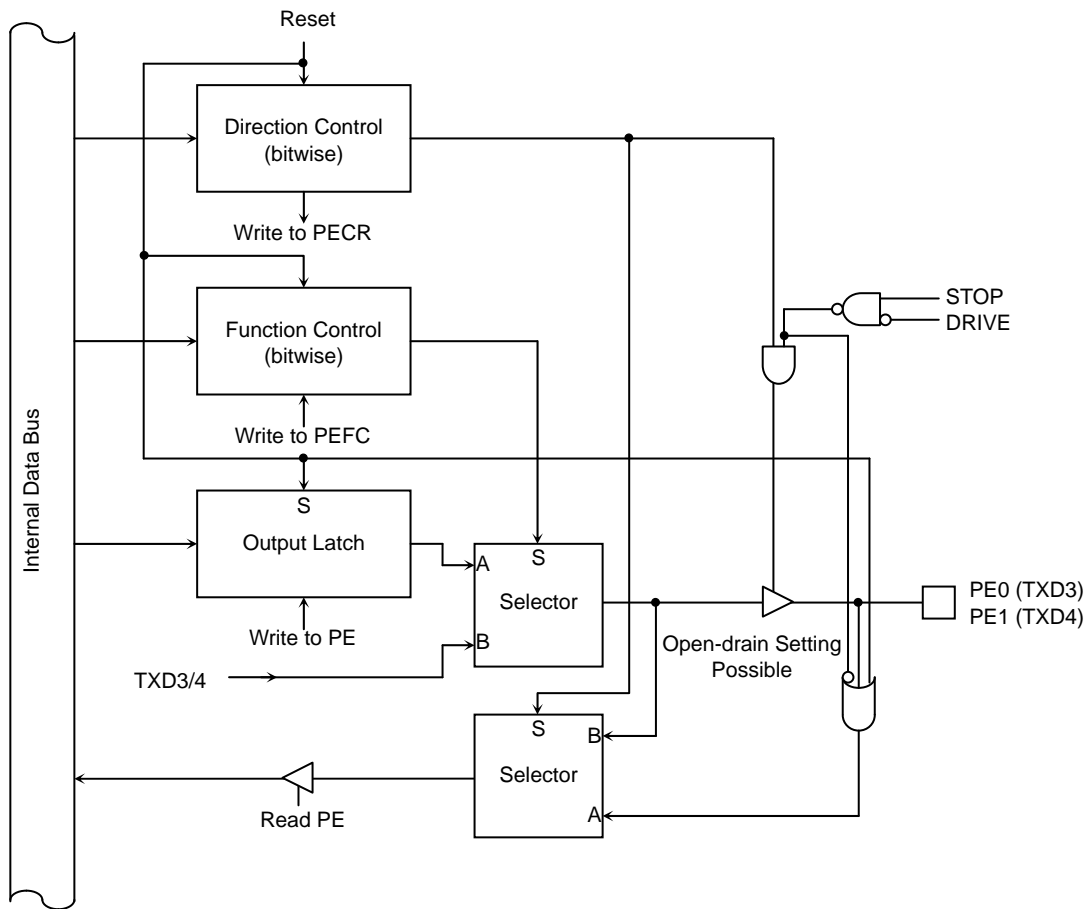
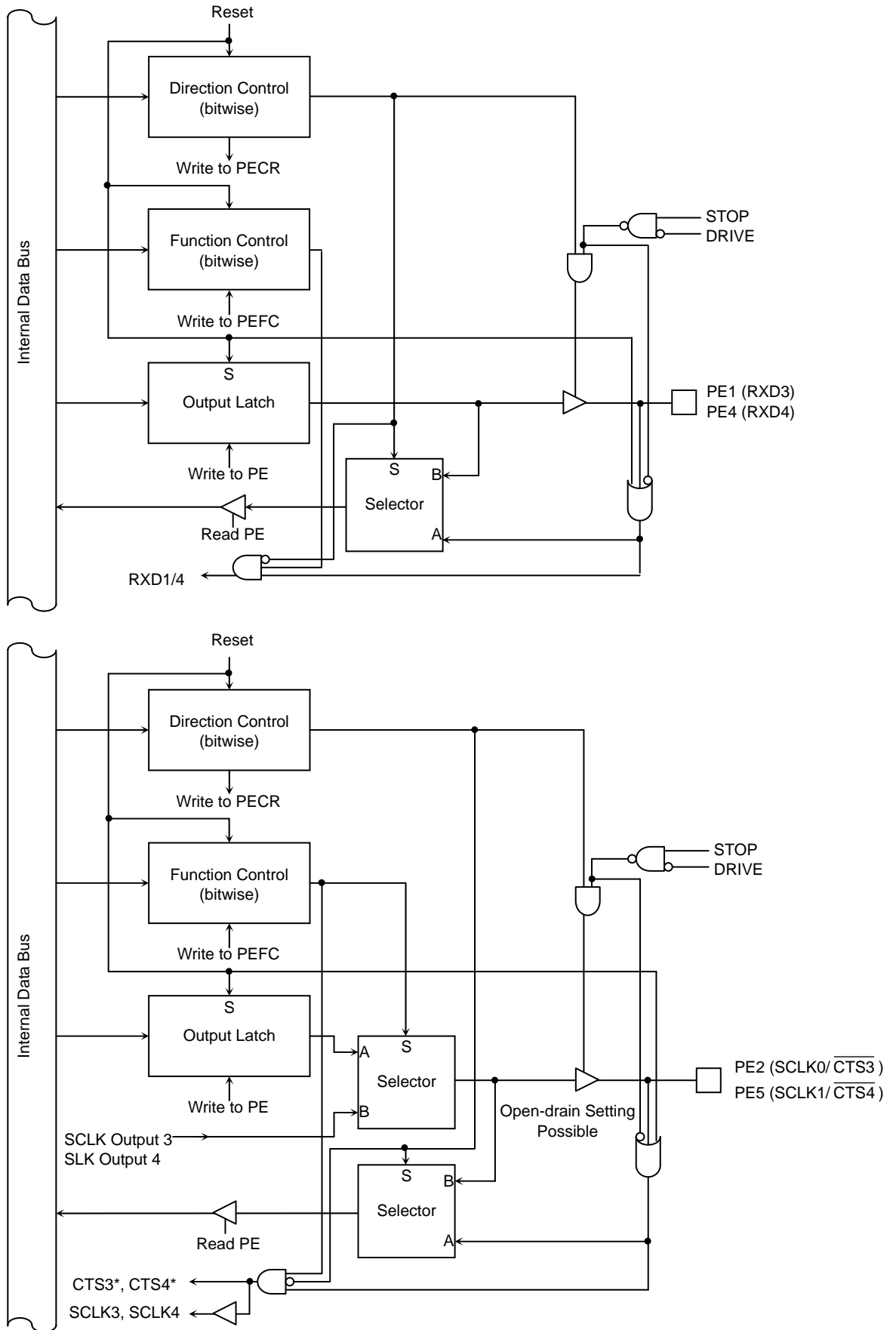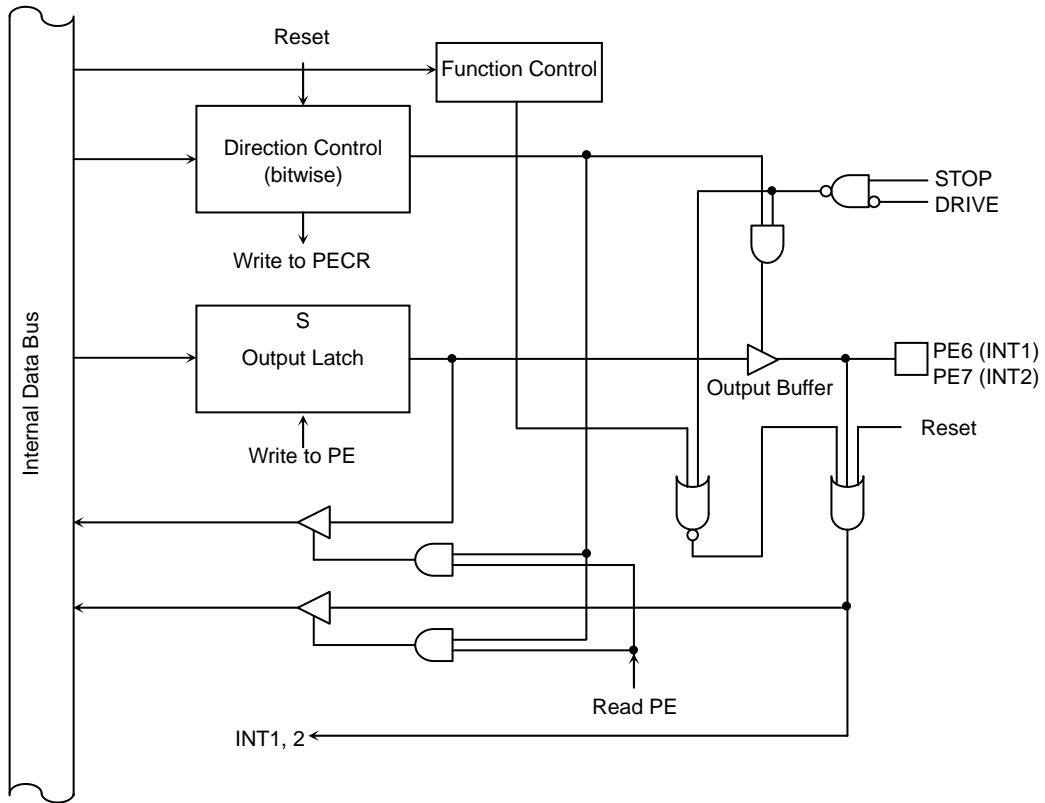Figure 3.5.39 Port E (PE1, PE2, PE4, PE5)

Figure 3.5.40 Port E (PE6, PE7)

Port E Register

	7	6	5	4	3	2	1	0
Bit Symbol	PE7	PE6	PE5	PE4	PE3	PE2	PE1	PE0
Read/Write	R/W							
After Reset	Input mode (output latch register set to 1)							

PE
(0xFFFF_F060)

Port E Control Register

	7	6	5	4	3	2	1	0
Bit Symbol	PE7C	PE6C	PE5C	PE4C	PE3C	PE2C	PE1C	PE0C
Read/Write	W							
After Reset	0	0	0	0	0	0	0	0
Function	0: IN 1: OUT							

PECR
(0xFFFF_F062)

Input/output settings for port E

0	Input
1	Output

Port E Function Register

	7	6	5	4	3	2	1	0
Bit Symbol	PE7F	PE6F	PE5F	PE4F	PE3F	PE2F	PE1F	PE0F
Read/Write	W							
After Reset	0	0	0	0	0	0	0	0
Function	0: PORT 1: INT2	0: PORT 1: INT1	0: PORT 1: SCLK4/ CTS4*	0: PORT 1: RXD4	0: PORT 1:ITXD4	0: PORT 1: SCLK3/ CTS3*	0: PORT 1: RXD3	0: PORT 1: TXD3

PEFC
(0xFFFF_F063)

Port E Open-drain Control Register

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	PEODE5	—	PEODE3	PEODE2	—	PEODE0
Read/Write	—		W	—	W	W	—	W
After Reset	—	—	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

PEODE
(0xFFFF_F066)

Function	Corresponding PEFC Bit	Corresponding PECR Bit	Port Used
Select TXD3 output	1	1	PE0
Select RXD3 input	1	0	PE1
Select SCLK3 input	1	0	PE2
Select SCLK3 output	1	1	
Select CTS3 input	1	0	
Select TXD4 output	1	1	PE3
Select RXD4 input	1	0	PE4
Select SCLK4 input	1	0	PE5
Select SCLK4 output	1	1	
Select CTS4 input	1	0	
Select INT1 input	1 (*1)	0	PE6
Select INT2 input	1 (*1)	0	PE7

*1   Set this bit when using the pin for a STOP mode termination interrupt with
        SYSCR<DRVE> set to 0. Otherwise, the bit need not be set.

Note: For a pin to which two input functions are assigned in addition to the port function, use
        the control register for each function module to specify which function is used.

Figure 3.5.41 Registers Related to Port E

3.5.14   Port F (PF0-PF6)

Port F is a 7-bit general-purpose input/output port whose bits can each be set independently for input or output. The control register PFCR is used to set the port for input or output. A reset clears PFCR to 0, putting port F in input mode. In addition to functioning as an input/output port, the pins of this port can also function as various input/output pins: PF0 functions as SIO data output, PF1 as SIO data input or key input, PF2 as SIO CLK input/output or CTS* input, PF3, PF4 and PF5 as SBI input/output, and PF6 as external interrupt input. These functions are enabled by setting the corresponding bits of PFFC to 1.

A reset clears PFCR and PFFC to 0, placing port F in input mode. The output open-drain control register (PFODE) can be used to set PF0, PF2, PF4 and PF5 to open-drain output when they are used for output. Port F becomes a 5 V input/output port when 5 V is supplied to its dedicated power supply pin DVCC51. It becomes a VCC-based (3 V) port when VCC is supplied to DVCC51.
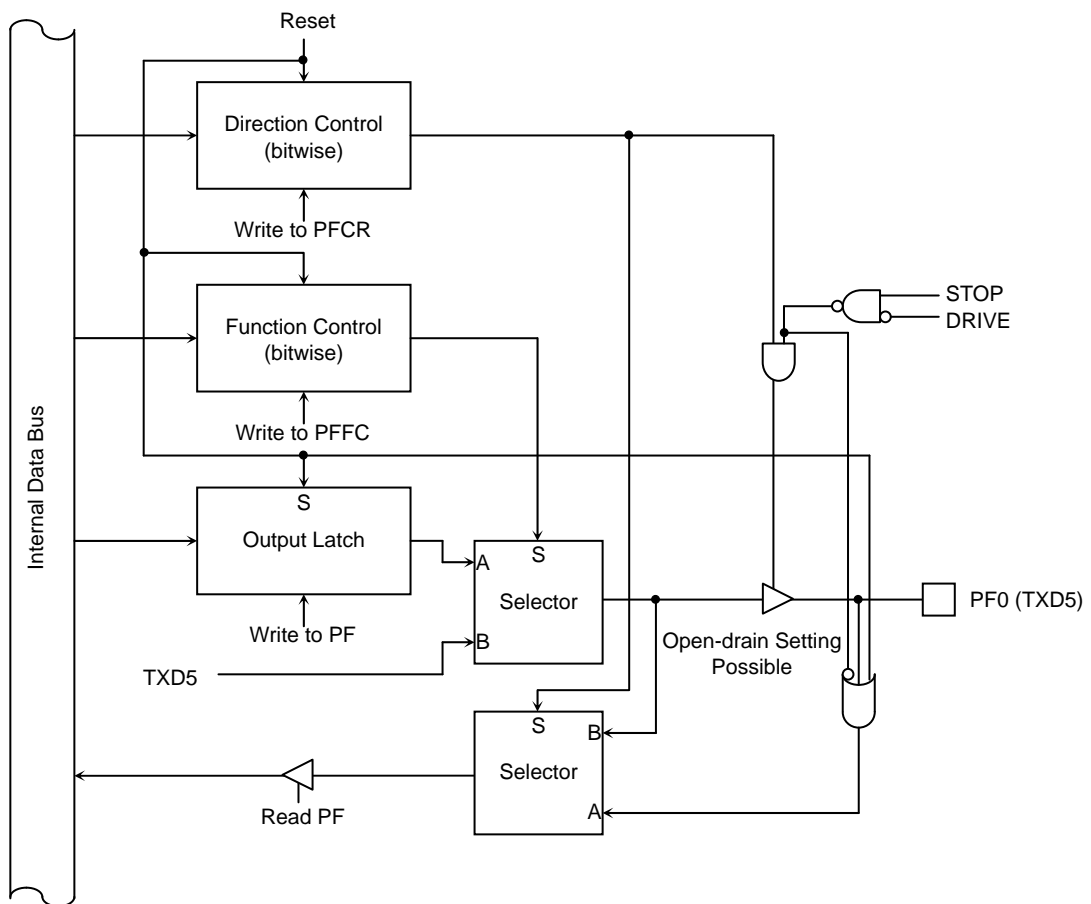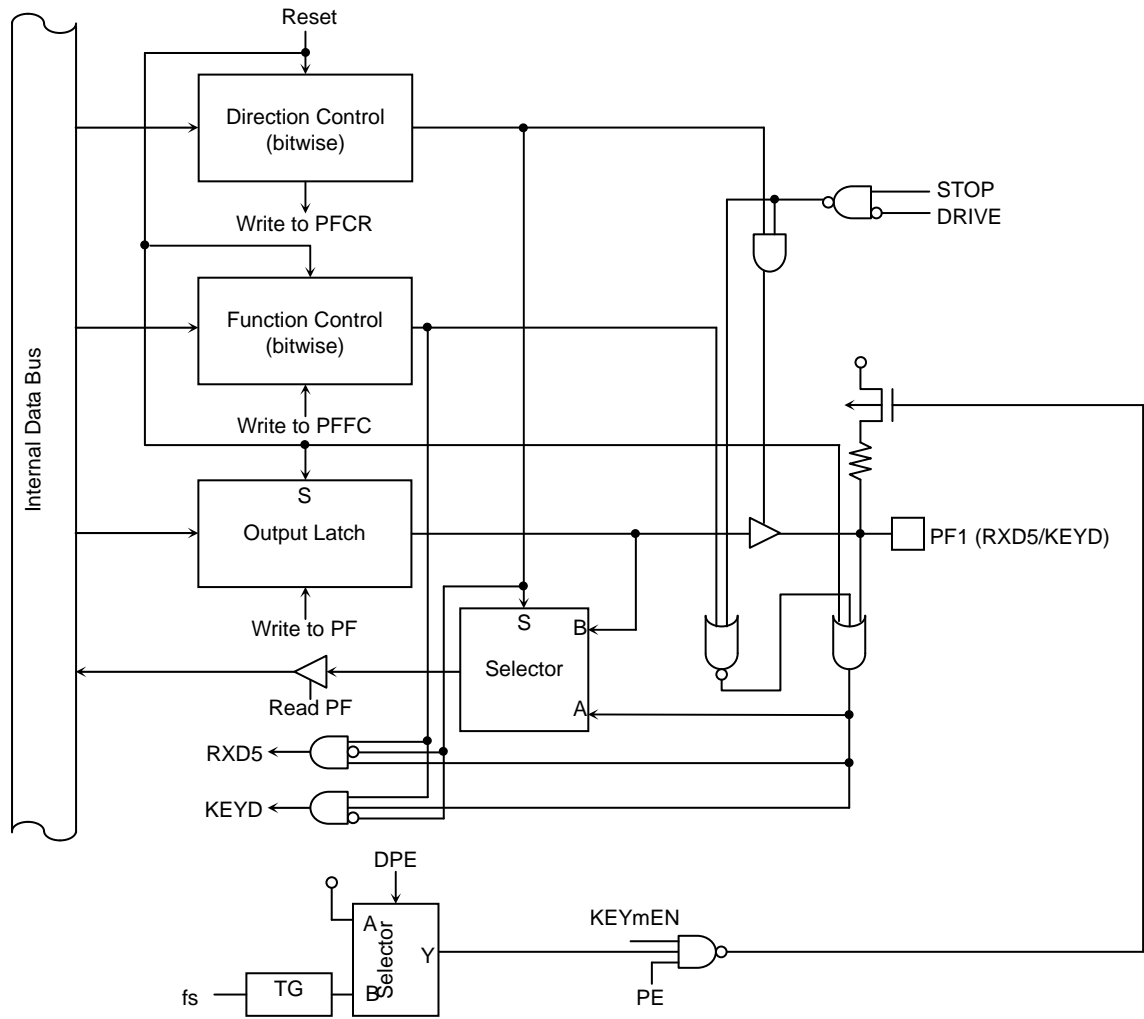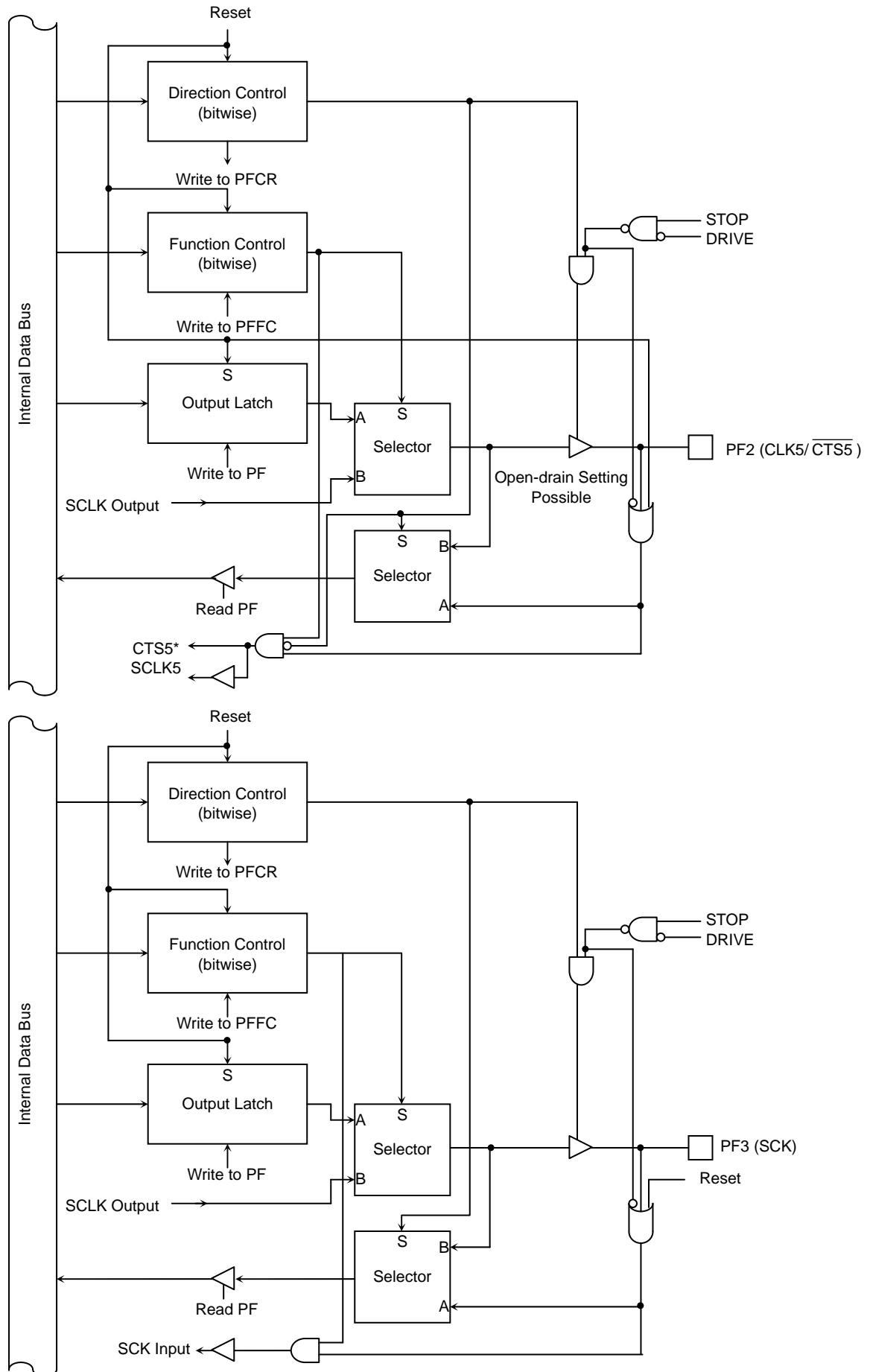

Figure 3.5.42 Port F (PF0)

Figure 3.5.43 Port F (PF1)

Figure 3.5.44 Port F (PF2, PF3)

www.DataSheet4U.com
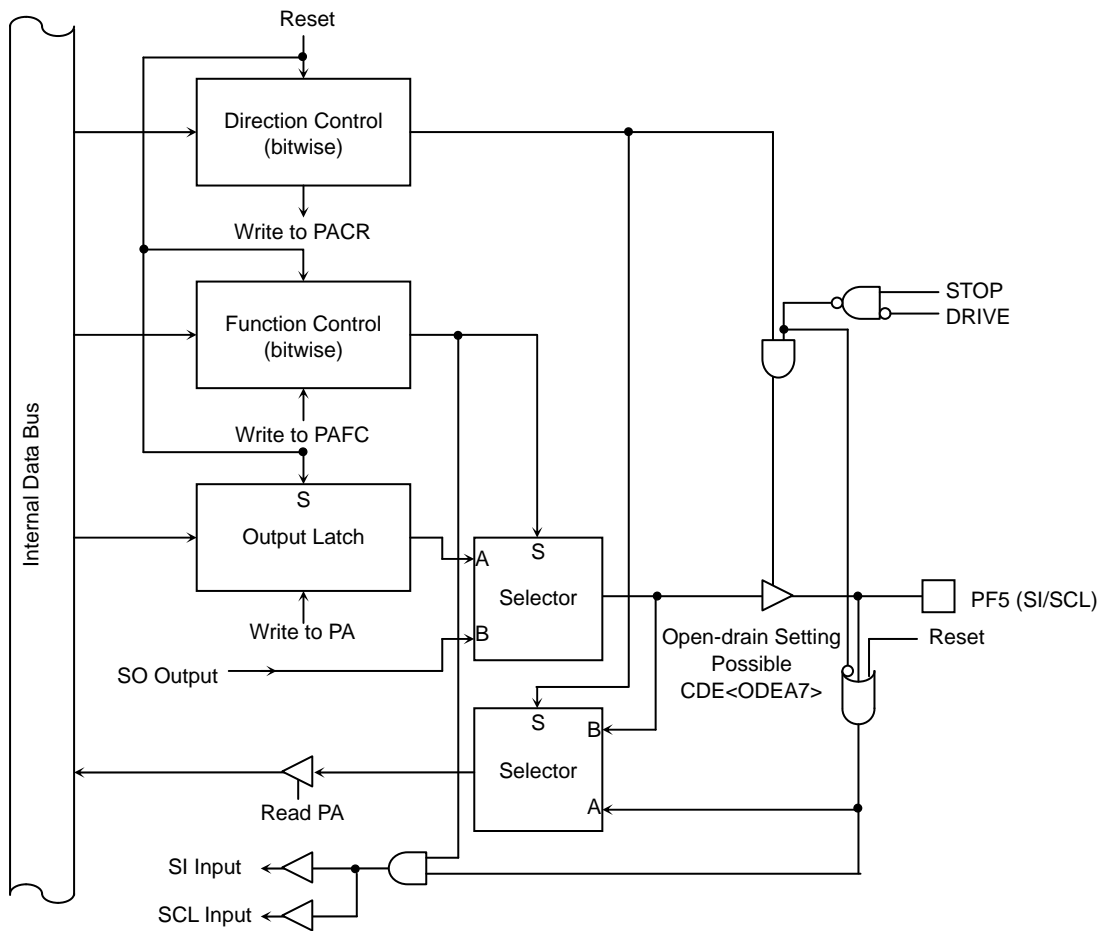
Figure 3.5.45 Port F (PF4, PF5)
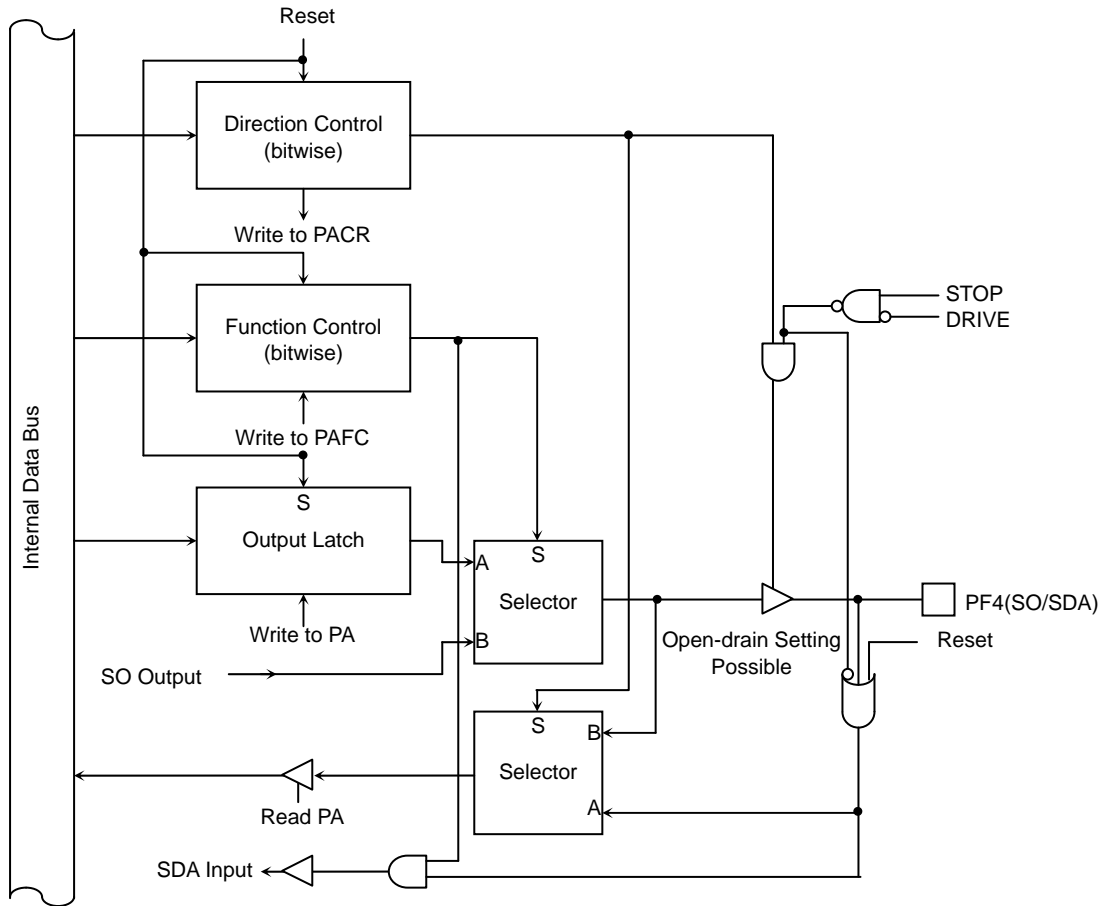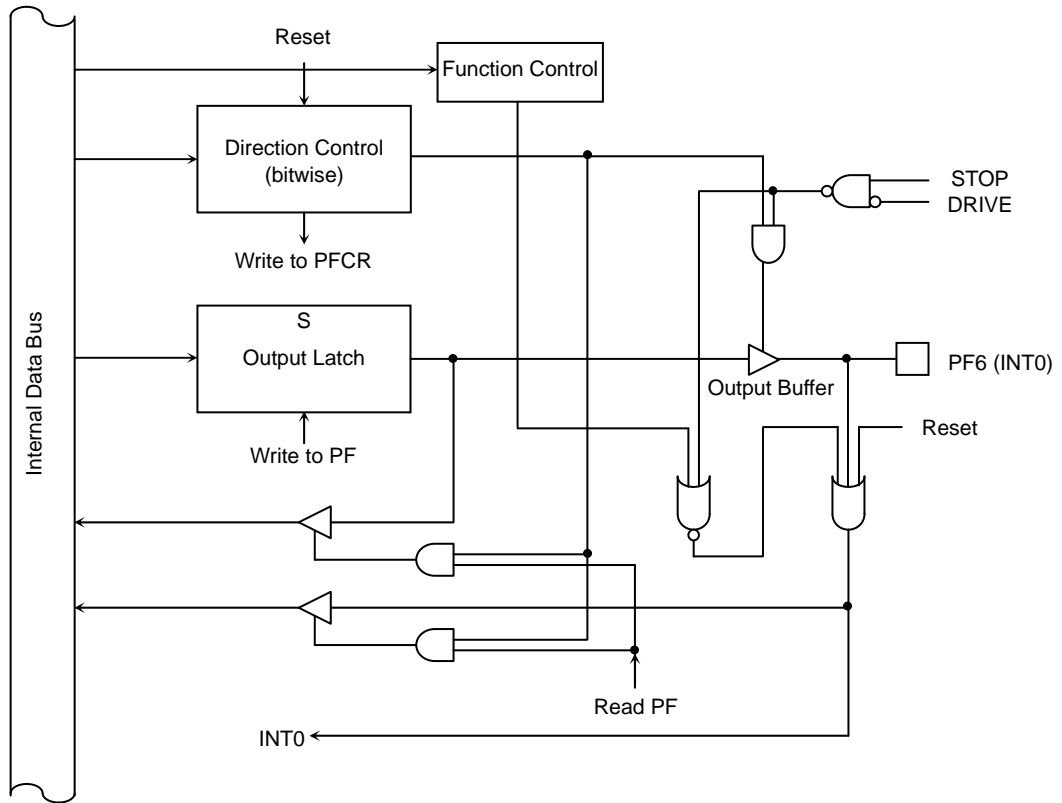
Figure 3.5.46 Port F (PF6)

Port F Register

	7	6	5	4	3	2	1	0
Bit Symbol	—	PF6	PF5	PF4	PF3	PF2	PF1	PF0
Read/Write	R/W							
After Reset	Input mode (output latch register set to 1)							

PF (0xFFFF_F061)

Port F Control Register

	7	6	5	4	3	2	1	0
Bit Symbol	—	PF6C	PF5C	PF4C	PF3C	PF2C	PF1C	PF0C
Read/Write	W							
After Reset	0	0	0	0	0	0	0	0
Function	0: IN 1: OUT							

PFCR (0xFFFF_F064)

Input/output settings for port F

0	Input
1	Output

Port F Function Register

	7	6	5	4	3	2	1	0
Bit Symbol	—	PF6F	PF5F	PF4F	PF3F	PF2F	PF1F	PF0F
Read/Write	W			—				
After Reset	—	0	0	0	0	0	0	0
Function		0: PORT 1: INT0	0: PORT 1: SI/SCIA	0: PORT 1: SO/SDA	0: PORT 1: SOK	0: PORT 1: SOLK4 COTS5	0: PORT 1: KEYD RXD5	0: PORT 1: TXD5

PEFC (0xFFFF_F065)

Port F Open-drain Control Register

	7	6	5	4	3	2	1	0
Bit Symbol	—	—	PFODE5	PFODE4	—	PFODE2	—	PFODE0
Read/Write	—		W	—	W	W	—	W
After Reset	—	—	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

PEODE (0xFFFF_F067)

Function	Corresponding PFFC Bit	Corresponding PFCR Bit	Port Used
Select TXD5 output	1	1	PF0
Select RXD5 input	1	0	PF1
Select KEYD input	1	0	
Select SCLK4 input	1	0	PF2
Select SCLK4 output	1	1	
Select CTS5 input	1	0	
Select SCK output	1	1	PF3
Select SCK input	1	0	
Select SO/SDA	1	1	PF4
Select SI/SCL	1	1	PF5
Select INT0 input	1(*1)	0	PF6

*1   Set this bit when using the pin for a STOP mode termination interrupt with SYSCR<DRVE> set to 0. Otherwise, the bit need not be set.

Note: For a pin to which two input functions are assigned in addition to the port function, use the Control Register for each function module to specify which function is used.

Figure 3.5.47 Registers Related to Port F

3.6   External Bus Interface

The TMP1942 contains an external bus interface function which is necessary for connecting memory or I/Os which are external to the chip. This function is implemented by the external bus interface circuit (EBIF) and the CS (chip select)/wait controller.

The CS/wait controller specifies mapping addresses for any four address spaces, and controls a wait state and data bus width (8 bits or 16 bits) for these four address spaces and other external address spaces.

The external bus interface circuit (EBIF) controls timing for the external bus based on settings made with the CS/wait controller. The EBIF also controls dynamic bus sizing and the arbitration of bus contention with external bus masters.
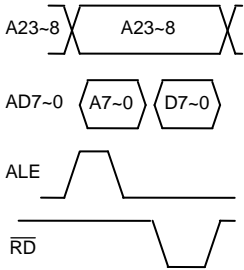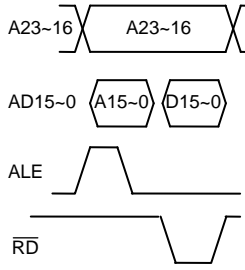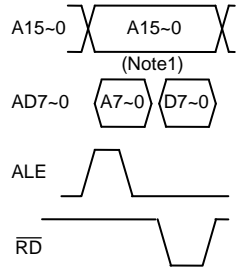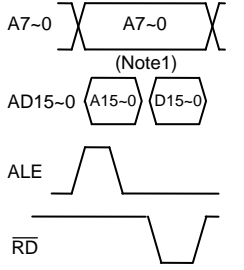
- Wait function

  Can be set individually for each block.

  - A wait state of up to 7 clock cycles can be automatically inserted.

  - Wait states can be inserted from the $\overline{\text{WAIT}}$ pin.

- Data bus width

  The bus width can be independently selected as 8 bits or 16 bits for each block.

- Read recovery cycle

  When a external bus cycle is immediately followed by a next external bus cycle, up to two dummy clock cycles can be inserted.

  Insertion of the dummy cycle(s) can be set individually for each block.

- Control of ALE width

  The ALE width can be set to 0.5 or 1.5 clock cycles.

  The set ALE width applies to all blocks in common.

- Arbitration of bus contention

3.6.1 Address and data pins

(1) Setting address and data pins

For external memory connections, port 0 (AD0-AD7), port 1 (AD8-AD15/A8-A15) and port 2 (A16-A23/A0-A7) pins can be used as the address bus and the data bus. One of the following four bus configurations can be selected by setting up the port registers.

		(1)	(2)	(3)	(4)
Number of address bus lines		max.24 (~16 MB)	max.24 (~16 MB)	max.16 (~64 KB)	max.8 (~256 B)
Number of data bus lines		8	16	8	16
Number of multiplexed address/data bus lines		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		A23~8 / AD7~0 (A7~0)(D7~0) / ALE / $\overline{RD}$	A23~16 / AD15~0 (A15~0)(D15~0) / ALE / $\overline{RD}$	A15~0 (Note1) / AD7~0 (A7~0)(D7~0) / ALE / $\overline{RD}$	A7~0 (Note1) / AD15~0 (A15~0)(D15~0) / ALE / $\overline{RD}$

Note 1: Even for cases (3) and (4), addresses are output because the data bus pins are shared with the address bus.

Note 2: Ports 0 to 2 are set for input after a reset, and do not function as address or data bus pins.

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

---

(2) Address hold when an internal area is accessed

When an internal area is accessed, the address bus retains the previous address which was output by the external area device; thus the address does not change. In addition, the address/data bus is placed in high-impedance state.

3.6.2 External bus operation

This section explains various bus timings. In the following timing diagrams, the address bus is chosen to be A23-A16 and the address/data bus is chosen to be AD15-AD0.

(1) Basic bus operation

External bus cycles in the TMP1942 essentially consist of three clock cycles. A wait state can be inserted, as will be explained later. The basic clock for external bus cycles is the same as the internal system clock.

Figure 3.6.1 shows a read bus timing. Figure 3.6.2 shows a write bus timing. During internal access, the address bus does not change, as shown in the diagram, nor does ALE output a latch pulse. The address/data bus is placed in high-impedance state, and neither $\overline{RD}$ and $\overline{WR}$ nor other control signals are asserted.
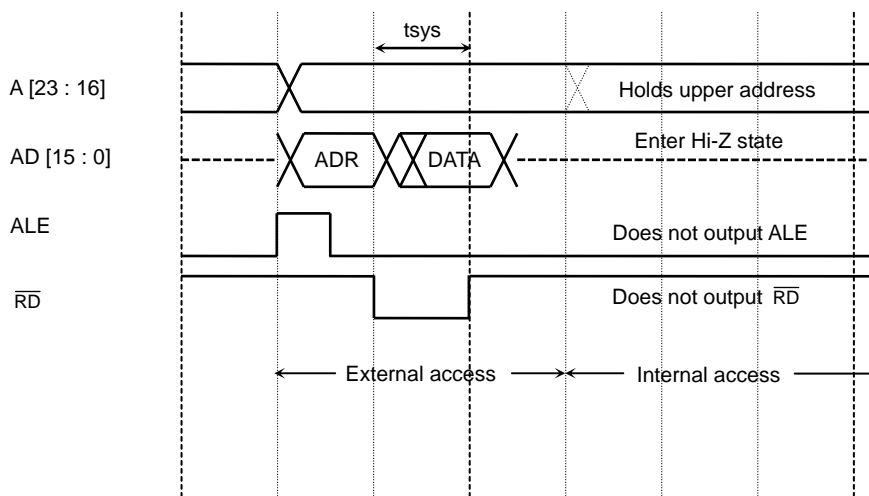
Figure 3.6.1 Read Operation Timing Diagram
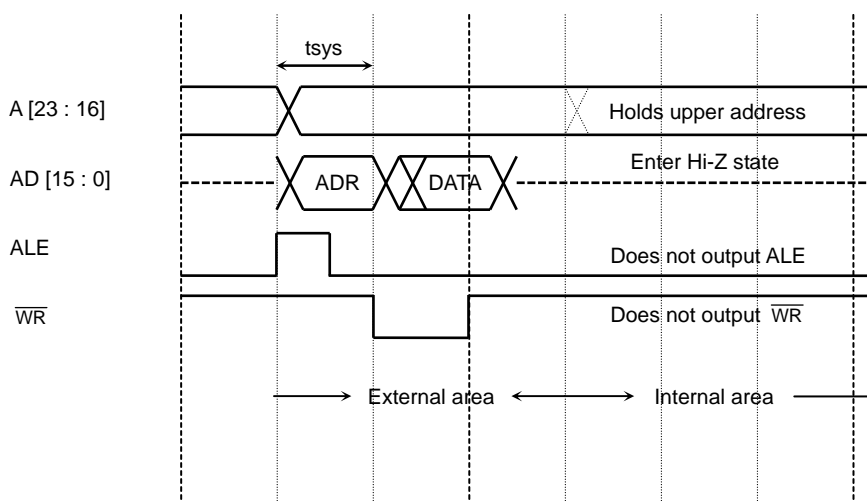
Figure 3.6.2 Write Operation Timing Diagram

Note: fsys expresses one period of share of system clock.

(2) Wait timing

Wait cycles can be inserted individually for each block by using the CS/wait controller. The following two types of wait insertion can be used:

a. Automatic wait insertion of up to 7 clock cycles

b. Wait insertion from $\overline{WAIT}$ pin

Note: "Please set the number of wait as "+1" when you use = long and BUSRQ the ALE width."

Timing diagrams with a wait state inserted are shown below.
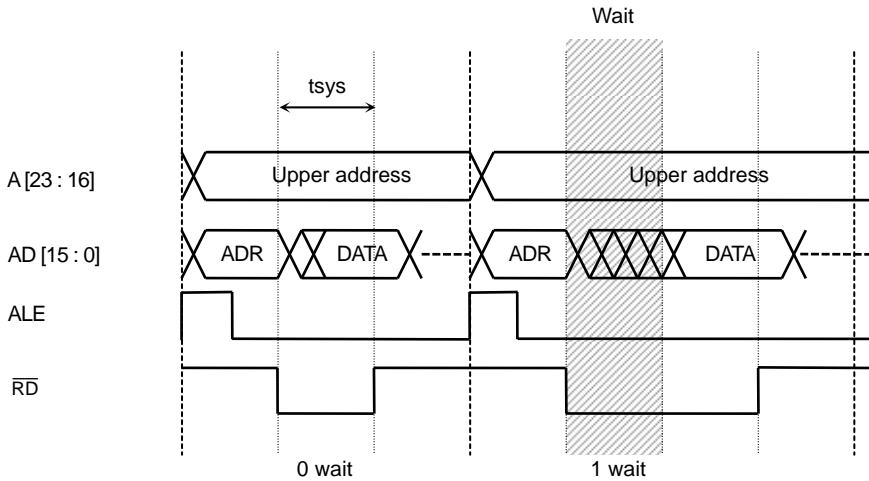
Figure 3.6.3 Read Operation Timing Diagram (with 0 Wait Cycles and 1 Wait Cycle)
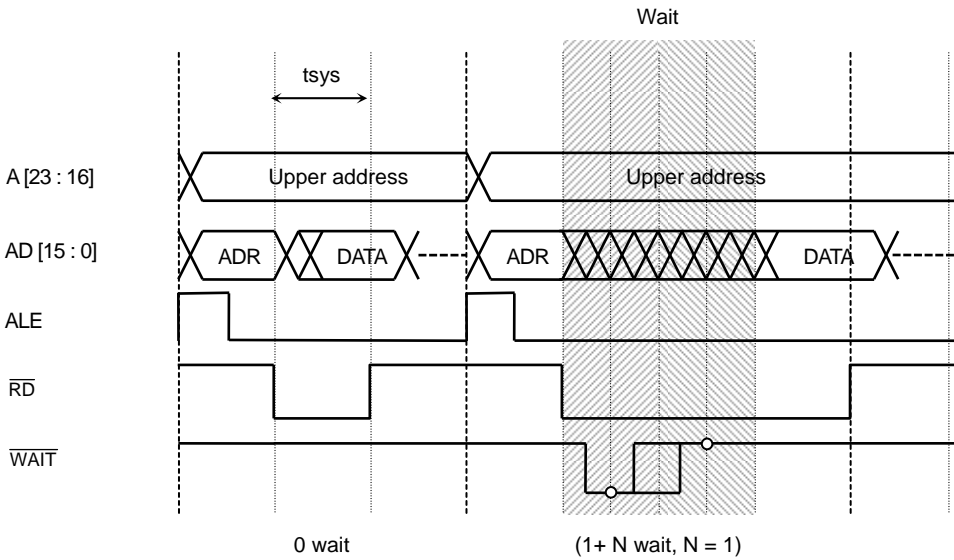
Figure 3.6.4 Read Operation Timing Diagram (1+N Wait Cycles, N = 1)
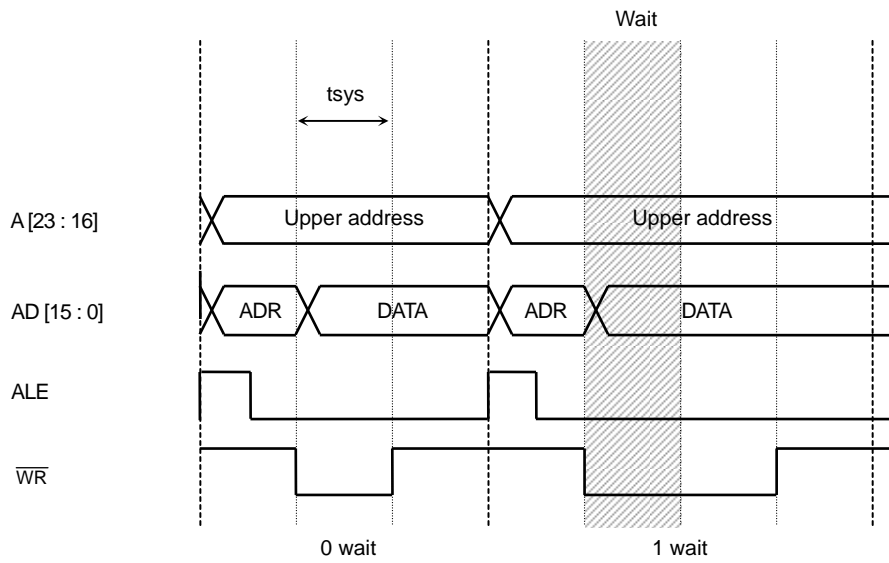
Figure 3.6.5 Write Operation Timing Diagram (with 0 Wait Cycles and 1 Wait Cycle)
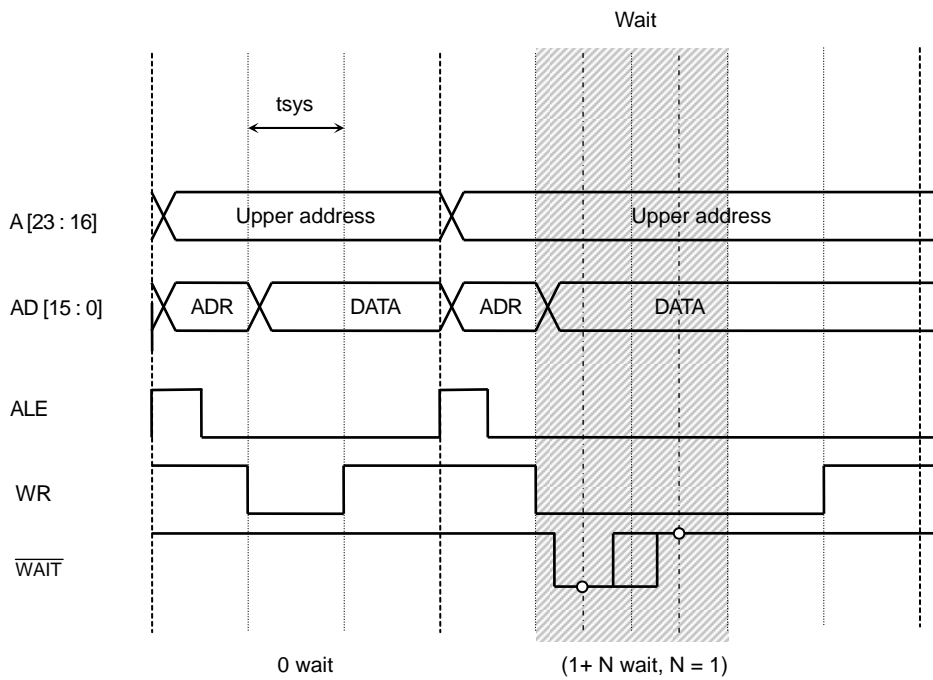
Figure 3.6.6 Write Operation Timing Diagram (1+N Wait Cycles, N = 1)

(3) ALE assertion time

The ALE assertion time can be selected as either 0.5 or 1.5 clock cycles. The bit for setting this assertion time is provided in the system clock control register. The default assertion time is 1.5 clock cycles. The assertion time cannot be set individually for blocks in the external area; it applies universally to the entire external address space.

Note: "Please set the number of wait as "+1" when you use = long and BUSRQ the ALE width."
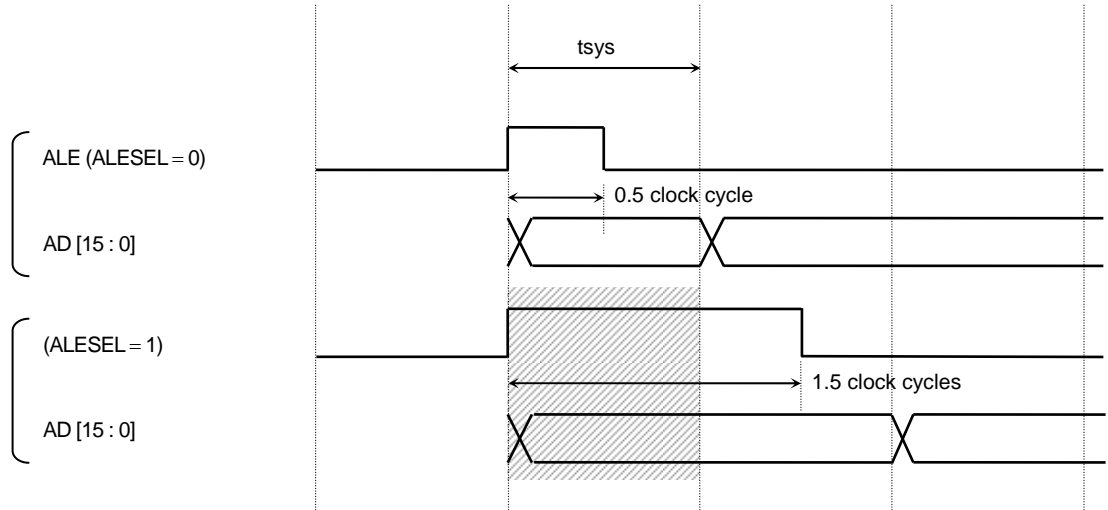
Figure 3.6.7 ALE Assertion Time

Figure 3.6.8 shows read operation timing with an ALE assertion time of 0.5 clock cycles and that with an ALE assertion time of 1.5 clock cycles.
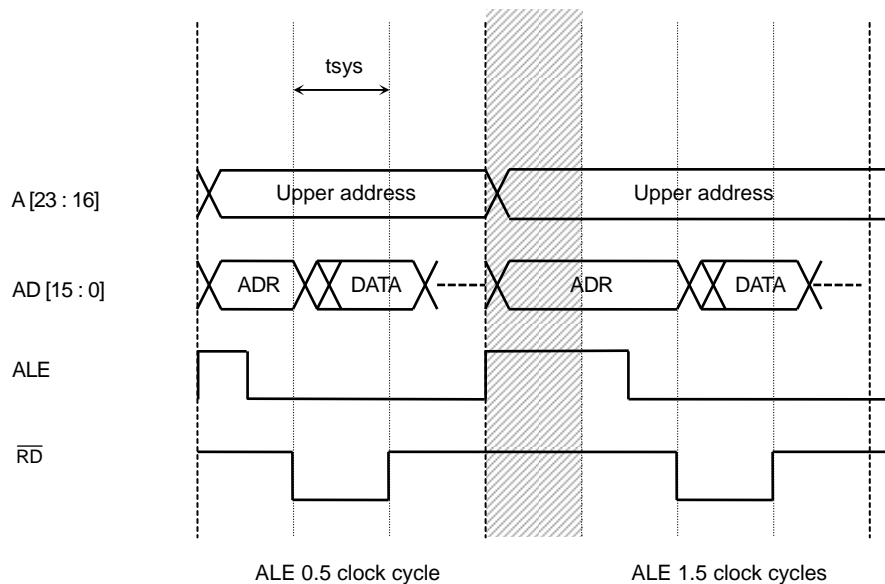
Figure 3.6.8 Read Operation Timing Diagram (with ALE Asserted for 0.5 and 1.5 Clock Cycles)

(4) Read recovery time

When an external access occurs after reading from an external area, a dummy cycle can be inserted to create a recovery time. Dummy cycles can only be inserted when the immediately preceding cycle is a read cycle.

External read followed by external read: Can be inserted

External read followed by external write: Can be inserted

External write followed by external access: Cannot be inserted

The number of dummy cycles can be specified independently for each block as one clock cycle or two clock cycles. Use the CS/wait controller to set the number of clock cycles.
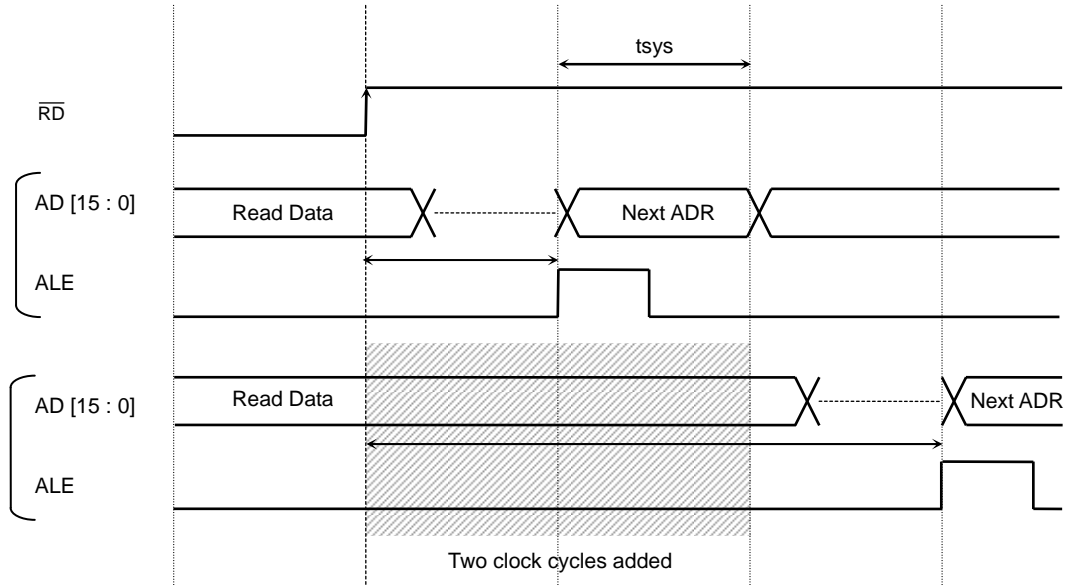
Figure 3.6.9 Read Recovery Time

As shown above, by adding two dummy clock cycles, a sufficient time from the rise of $\overline{RD}$ to the output of the next address can be secured even when the device is operating at a fast clock speed. Figure 3.6.10 shows a bus timing diagram where one and two dummy clock cycles are inserted.
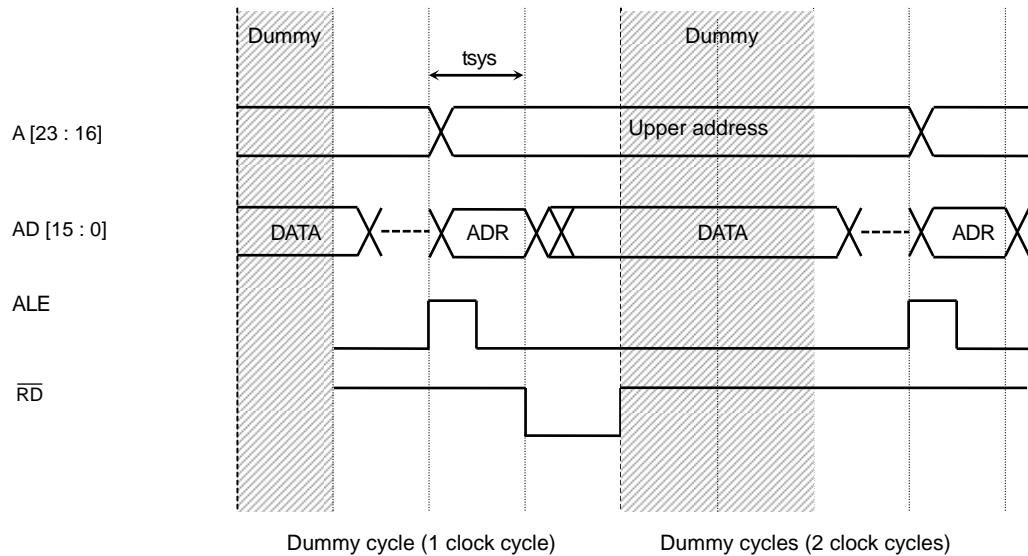
Figure 3.6.10 Read Operation Timing Diagram (with Dummy Cycles Inserted)

3.6.3   Bus arbitration

The TMP1942 allows external bus masters to be connected to the chip. Two signals $\overline{\text{BUSRQ}}$ and $\overline{\text{BUSAK}}$ are used to arbitrate contention for bus control between the processor and external bus masters. External bus masters can only gain control of buses external to the TMP1942. External bus masters cannot gain control of the device's internal bus.

(1)   Access range for external bus masters

External bus masters can only gain control of buses external to the TMP1942. External bus masters cannot gain control of the device's internal bus (G-Bus). Therefore, external bus masters cannot access the device's internal memory and internal I/O blocks. Contention for control of the external bus is arbitrated by the external bus interface circuit (EBIF); hence the CPU and the internal DMAC are not involved in bus arbitration. Even when an external bus master has control of the external bus, the CPU and the internal DMAC can access the internal ROM and RAM and the internal registers. On the other hand, when the CPU or the internal DMAC attempts to access external memory while an external bus master has control of the external bus, the CPU or the internal DMAC is kept waiting until the external bus master finishes control of the external bus. Therefore, if $\overline{\text{BUSRQ}}$ remains asserted for an excessive period of time, the TMP1942 may get locked.

(2)   Gaining control of the bus

An external bus master requests control of the bus from the TMP1942 by asserting the $\overline{\text{BUSRQ}}$ signal. The TMP1942 samples the $\overline{\text{BUSRQ}}$ signal during a break in the external bus cycles on the internal bus (G-Bus) to determine whether or not to grant control of the bus. To give control of the bus to the external bus master, it asserts the $\overline{\text{BUSAK}}$ signal. At the same time, it places the address bus, data bus and bus control signals in high-impedance state.

If the data size to be loaded or stored is larger than the width of the bus for the external memory, multiple bus cycles may occur for a single data transfer (bus sizing). In such a case, a break in the external bus cycles will occur when the last bus cycle has finished.

The TMP1942 allows the insertion of dummy cycles when external access continues for successive bus cycles. Even in this case it is only when a break in the external bus cycles occurs on the internal bus (G-Bus) that a request for bus control is accepted. During a dummy cycle the next external bus cycle is already activated on the internal bus, so that if the $\overline{\text{BUSRQ}}$ signal is asserted during a dummy cycle, the bus will only be released after the next bus cycle has been completed.

Make sure the $\overline{\text{BUSRQ}}$ signal remains asserted until control of the bus has been finished. Figure 3.6.11 shows a timing sequence in which control of the bus is gained by an external bus master.

Note: "Please set the number of wait as "+1" when you use = long and BUSRQ the ALE width."
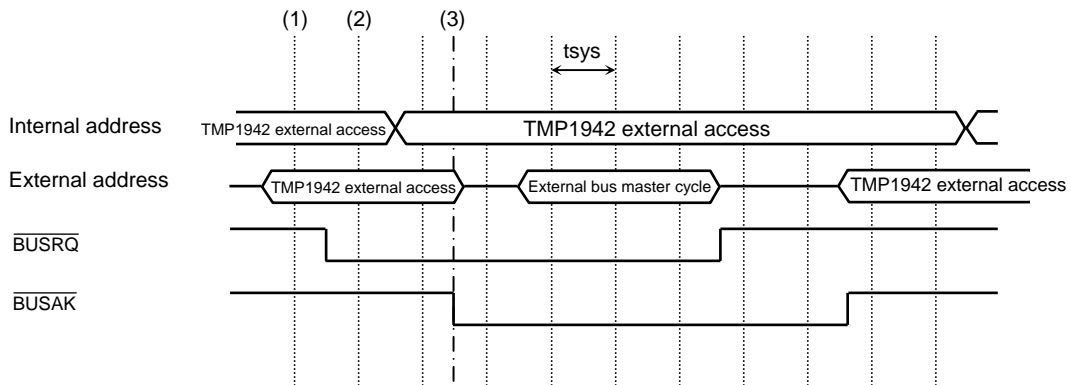
(1) $\overline{\text{BUSRQ}}$ is High.

(2) The TMP1942 recognizes that $\overline{\text{BUSRQ}}$ has been pulled Low and releases the bus when the bus cycle has been completed.

(3) The TMP1942 asserts $\overline{\text{BUSAK}}$ upon completion of the bus cycle. The external bus master recognizes that $\overline{\text{BUSAK}}$ has been asserted Low and gains control of the bus, thereby initiating its bus operation.

Figure 3.6.11 Timing at Which Control of the Bus is Gained

(3) Relinquishing control of the bus

An external bus master relinquishes control of the bus in the following case:

- When it no longer requires control of the bus

1) Relinquishing control of the bus when an external bus master no longer requires control of the bus.

When the external bus master no longer needs the control of the bus which it gained, it deasserts the $\overline{\text{BUSRQ}}$ signal to return control of the bus to the TMP1942. Figure 3.6.12 shows a timing sequence in which the bus is released because the external bus master no longer requires control of it.
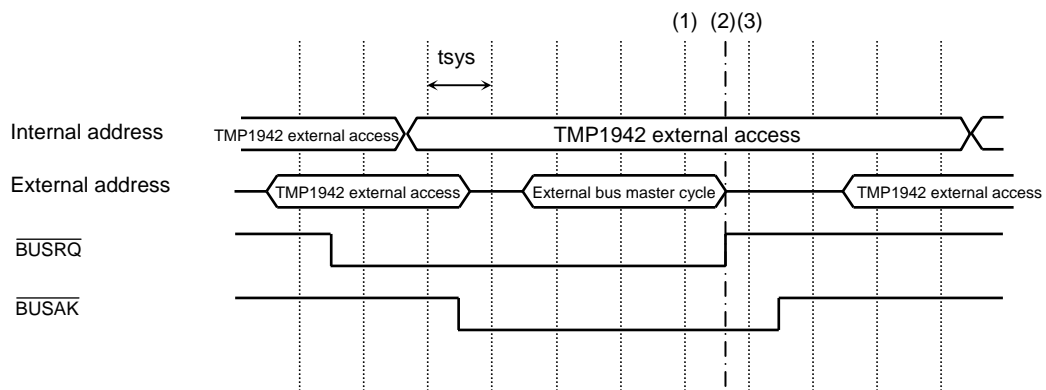
(1) The external bus master has control of the bus.

(2) Because the external bus master no longer requires control of the bus, it deasserts $\overline{\text{BUSRQ}}$.

(3) The TMP1942 recognizes that $\overline{\text{BUSRQ}}$ has reverted to High and responds by deasserting $\overline{\text{BUSAK}}$.

Figure 3.6.12 Timing at Which Control of the Bus is Relinquished

(4) Bus Release Timings

BUSRQ asserted during internal access (no external wait)

BUSRQ asserted during internal access (no external wait)

BUSRQ asserted during internal access (no external wait)
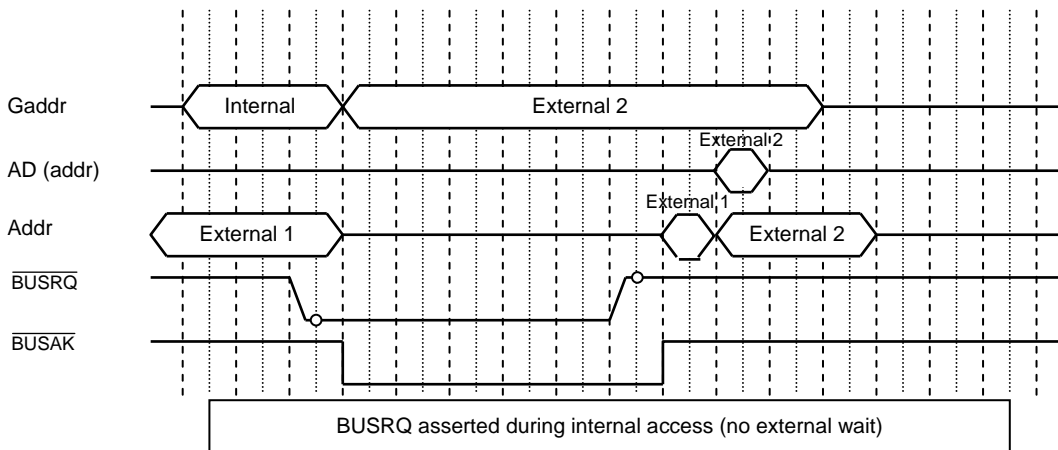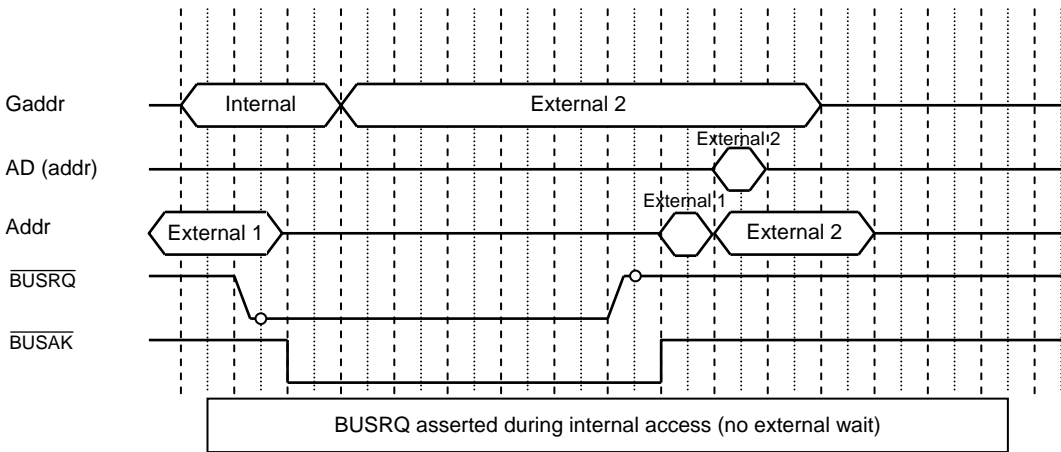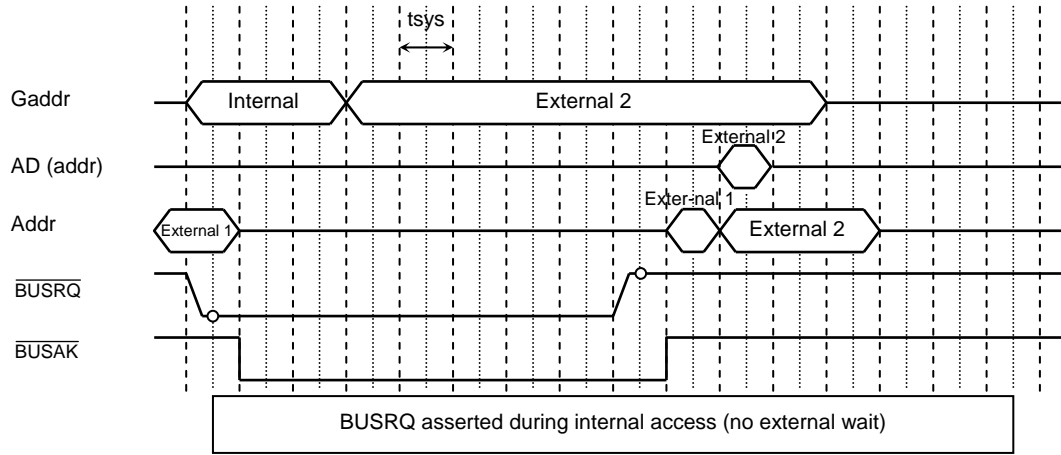
Note: Gaddr indicates the address on the G-Bus. AD (addr) indicates the address on the address/data bus. Addr indicates the address on the address bus.

BUSRQ asserted during external access, followed by internal access (no external wait)

BUSRQ asserted during external access, followed by internal access (no external wait)

BUSRQ asserted during external access, followed by external access (no external wait)

Bus sizing applied (no external wait)

TMP1942-128

Idle

Gaddr: External 1 | Internal | External 2

AD (addr): External 1 | External 2

Addr: External 1 | External 1 | External 2

$\overline{BUSRQ}$

$\overline{BUSAK}$

BUSRQ asserted during external access, followed by internal access (no external wait, 1 idle cycle)

Idle

Gaddr: External 1 | Internal | External 2

AD (addr): External 1 | External 2

Addr: External 1 | External 1 | External 2 | External 3

$\overline{BUSRQ}$

$\overline{BUSAK}$

BUSRQ asserted during external access, followed by internal access (no external wait, 1 idle cycle)

Idle

Gaddr: External 1 | External 2 | External 3

AD (addr): External 1 | External 2 | External 3

Addr: External 1 | External 2 | External 1 | External 3

$\overline{BUSRQ}$

$\overline{BUSAK}$

BUSRQ asserted during external access, followed by external access (no external wait, 1 idle cycle)

3.7   Chip Select/Wait Controller

The TMP1942 supports direct connections to external devices (I/O devices, ROM and SRAM).

The TMP1942 provides four programmable chip select signals. Programmable features include variable block sizes, data bus width, wait state insertion, and dummy cycle insertion for back-to-back bus cycles.

$\overline{CS0}$ - $\overline{CS3}$ (multiplexed with P40-P43) are the chip select output pins for the CS0-CS3 address ranges. These chip select signals are generated when the CPU or on-chip DMAC issues an address within the programmed ranges. The P40-P43 pins must be configured as CS0-CS3 by programming the Port 4 Control (P4CR) register and the Port 4 Function (P4FC) register.

Chip select address ranges are defined in terms of a base address and an address mask. There is a Base/Mask Address (BMAn) register for each of the four chip select signals, where n is a number from 0 to 3.

There is also a set of three Chip Select/Wait Control registers, B01CS, B23CS and BEXCS, each of which consists of a master enable bit, a data bus width bit, a wait state field and a dummy cycle field.

External memory devices can also use the $\overline{WAIT}$ pin to insert wait states and consequently prolong read and write bus cycles.

3.7.1   Programming Chip Select Ranges

Each of the four chip select address ranges is defined in the BMAn register. The basic chip select model allows one of the chip select output signals ($\overline{CS0}$ - $\overline{CS3}$) to assert when an address on the address bus falls within a particular programmed range. The B01CS register defines specific operations for CS0 and CS1, and the B23CS register defines specific operations for CS2 and CS3 (see Section 3.7.2).

(1) Base/Mask Address Registers

The organizations of the BMAn registers are shown in Fig.3.7.1 and Fig. 3.7.2. The base address (BAn) field specifies the starting address for a chip select. Any set bit in the address mask field (MAn) masks the corresponding base address bit. The address mask field determines the block size of a particular chip select line. The address is compared on every bus cycle.

/Base address

The base address (BAn) field specifies the upper 16 bits (A31-A16) of the starting address for a chip select. The lower 16 bits (A15-A0) are assumed to be zero. Thus, the base address is any multiple of 64 Kbytes starting at 0x0000_0000. Figure 3.7.3 shows the relationships between starting addresses and the BMAn values.

/Address mask

The address mask (MAn) field defines whether any particular bits of the address should be compared or masked. Any set bit masks the corresponding base address bit. The address compare logic uses only the address bits that are not masked (i.e., mask bit cleared to 0) to detect an address match.

Address bits that can be masked (i.e., supported block sizes) differ for the four chip select spaces as follows:

    CS0 and CS1 spaces: A29-A14

    CS2 and CS3 spaces: A30-A15

Note: Use physical addresses in the BMAn registers.

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

BMA0 (0xFFFF_E400)		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	Sets the size of the CS0 space. 0: Used for comparing addresses							
		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	Must always be set to 0.							
		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	Sets A23-A16 for the start address.							
		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	Sets A31-A24 for the start address.							

BMA1 (0xFFFF_E404)		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	Sets the size of the CS1 space. 0: Used for comparing addresses							
		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	Must always be set to 0.							
		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	Sets A23-A16 for the start address.							
		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	Sets A31-A24 for the start address.							

Note: Bits 10-15 in BMA0 and BMA1 must always be set to 0.
This is because, although the CS0 and CS1 spaces can have a size of 16 KB to 1 GB, the TMP1942's external address space is limited to 16 MB, which requires setting bits 10-15 to 0 so as not to mask the A24-A29 address bits.

Figure 3.7.1 Base/Mask Address Registers (BMA0 and BMA1)

BMA2 (0xFFFF_E408)		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	Sets the size of the CS2 space. 0: Used for comparing addresses							
		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	Must always be set to 0.							
		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	Sets A23-A16 for the start address.							
		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	Sets A31-A24 for the start address.							

BMA3 (0xFFFF_E40C)		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	Sets the size of the CS2 space. 0: Used for comparing addresses							
		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	Must always be set to 0.							
		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	Sets A23-A16 for the start address.							
		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	Sets A31-A24 for the start address.							

Note: Bits 9-15 in BMA2 and BMA3 must always be set to 0.
This is because, although the CS2 and CS3 spaces can have a size of 32 KB to 2 GB, the TMP1942's external address space is limited to 16 MB, which requires setting bits 9-15 to 0 so as not to mask the A24-A30 address bits.

Figure 3.7.2 Base/Mask Address Registers (BMA2 and BMA3)
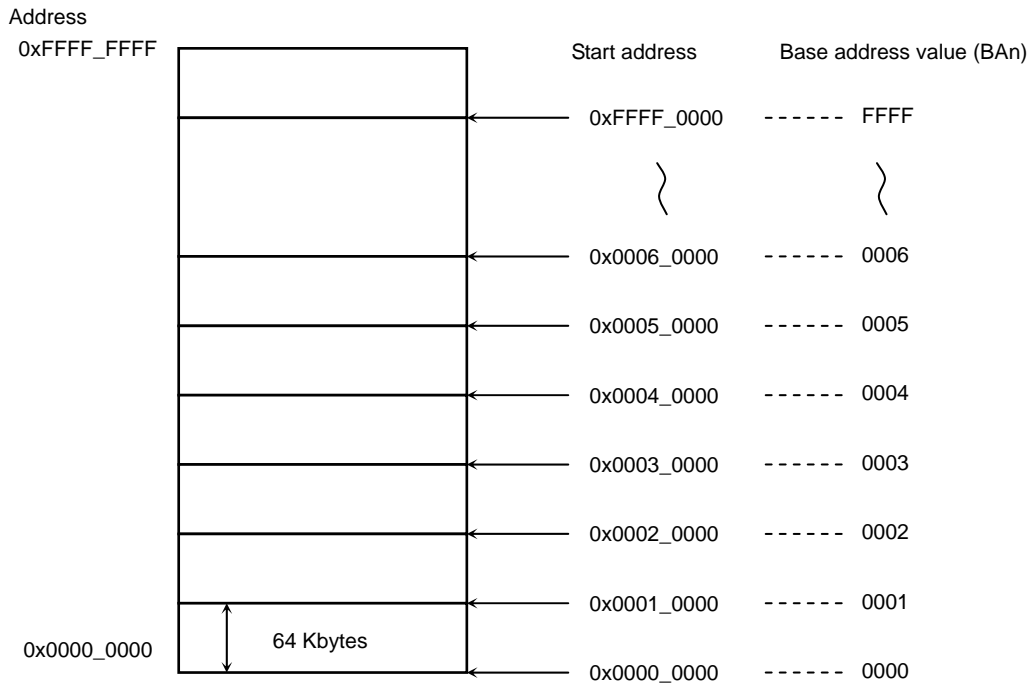
Figure 3.7.3 Relationship Between Start Address and Base Address Register Values

(2) Setting the start address and address space size

- Program the BMA0 register as follows to cause CS0 to be asserted in the 64 Kbytes of address space starting at 0xC000_0000.

BMA0 Register Value
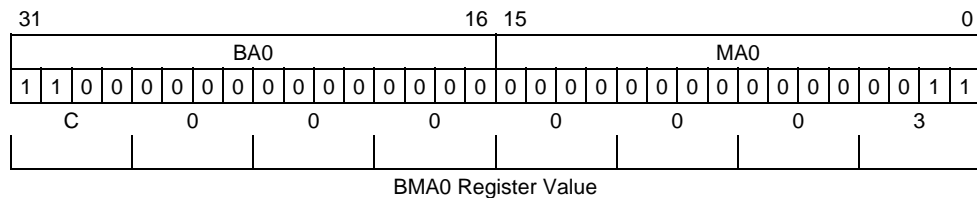
The BA0 field specifies the upper 16 bits of the starting address, or 0xC000. The MA0 field determines whether the A29-A14 bits of the address should be compared or masked. The A31 and A30 bits are always compared. Bits 15-10 of the MA0 field must be cleared so that the A29-A24 bits are always compared.

When the BMA0 register is programmed as shown above, the A31-A16 bits of the address are compared to the value of the BA0 field. Consequently, the 64-Kbyte address range between 0xC000_0000 and 0xC000_FFFF is defined as the $\overline{CS0}$ space.

- Program the BMA2 register as follows to cause CS2 to be asserted in the 512 Kbyte of address space starting at 0x1FC8_0000.

```
31                              16  15                              0
┌─────────────────────────────────┬──────────────────────────────────┐
│              BA2                 │                                  │
│              MA2                 │                                  │
├─┬─┬─┬─┬─┬─┬─┬─┬─┬─┬─┬─┬─┬─┬─┬─┬─┼─┬─┬─┬─┬─┬─┬─┬─┬─┬─┬─┬─┬─┬─┬─┬─┤
│0│0│0│1│1│1│1│1│1│1│0│1│0│0│0│0│0│0│0│0│0│0│0│0│0│0│0│1│1│1│1│1│
└─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┴─┘
     1       F       D       0       0       0       1       F
```

BMA2 Register Value
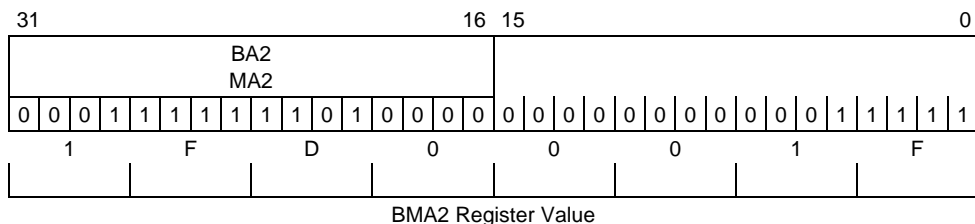
The BA2 field specifies the upper 16 bits of the starting address, or 0x1FC8. The MA2 field determines whether the A30-A15 bits of the address should be compared or masked. The A31 bit is always compared. Bits 15-9 of the MA0 field must be cleared so that the A30-A24 bits are always compared.

When the BMA2 register is programmed as shown above, the A31-A19 bits of the address are compared to the value of the BA2 field. Consequently, the 1-Mbyte address range between 0x1FC8_0000 and 0x1FCF_FFFF is defined as the CS2 space.

> Note: The TMP1942 does not assert any $\overline{CSn}$ signal in the following address ranges:
>   0xFFFF_8000 through 0x1FFF_BFFF

Upon reset, the CS0, CS1 and CS3 spaces are disabled while the CS2 space is enabled and spans the entire 4-GB address space.

(3) Specifying the size of an address space

Table 3.7.1 shows the possible sizes of each CS space. If two or more address spaces are specified which overlap one another, the address space with the lowest CS space number will be selected since it has priority.

Example: The start address of the CS0 space is 0xC000_0000 and the space size is 16 Kbytes.
The start address of the CS1 space is 0xC000_0000 and the space size is 64 Kbytes.
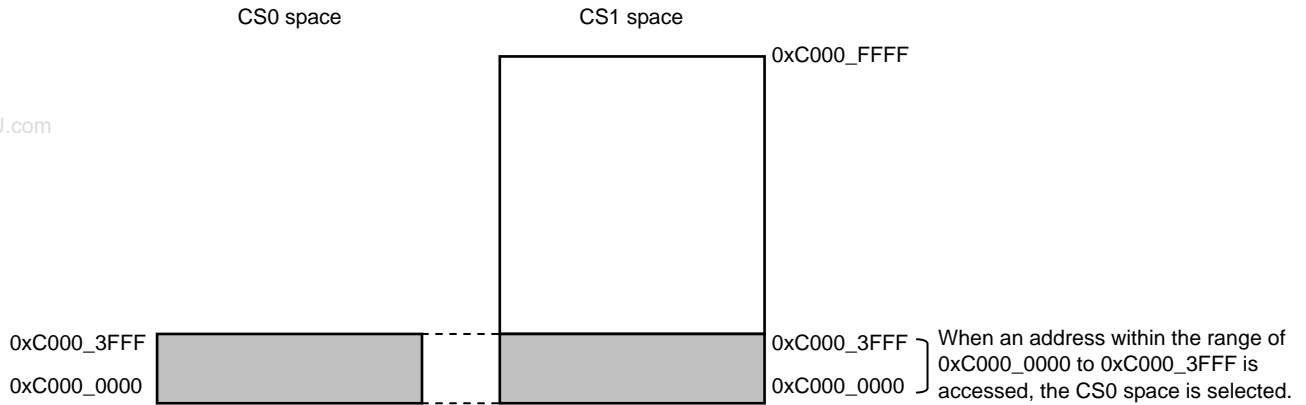
Table 3.7.1 CS Spaces and Their Possible Sizes

Size (Bytes) / CS Space	16 K	32 k	64 K	128 K	256 K	512 K	1 M	2 M	4 M	8 M	16 M
CS0	O	O	O	O	O	O	O	O	O	O	O
CS1	O	O	O	O	O	O	O	O	O	O	O
CS2		O	O	O	O	O	O	O	O	O	O
CS3		O	O	O	O	O	O	O	O	O	O

3.7.2 Chip select/wait control registers

The chip select/wait control registers are shown in Figure 3.7.4 to Figure 3.7.6. For each address space (i.e., the CS0-CS3 spaces and any other address space), the corresponding chip select/wait control register (B01CS-B23CS or BEXCS) can be used to enable/disable the master, select a chip select output waveform and data bus width, set the number of wait cycles and insert dummy cycles. If two or more address spaces are specified which overlap one another, the address space with the lowest CS space number will be selected since it has priority. (The priority order is CS0 > CS1 > CS2 > CS3 > EXCS.)

B01CS (0xFFFF_E480), B23CS (0xFFFF_E484), BEXCS (0xFFFF_E488)

B01CS
(0xFFFF_E480)

	7	6	5	4	3	2	1	0
Bit symbol	B0OM		—	B0BUS	B0W			
Read/Write	W		—	W				
After reset	0	0	—	0	0	1	0	1
Function	Selects chip select output waveform. 00: ROM/RAM Other settings are not allowed.			Selects data bus width. 0: 16 bits 1: 8 bits	Sets the number of wait cycles 0000: 0 cycles 0001: 1 cycle 0010: 2 cycles 0011: 3 cycles 0100: 4 cycles 0101: 5 cycles 0110: 6 cycles 0111: 7 cycles 1111: (1+N) cycles Other settings are not allowed.			

	15	14	13	12	11	10	9	8
Bit symbol	—	—	—	—	B0E	—	B0RCV	
Read/Write	—	—	—	—	W	—	—	
After reset	—	—	—	—	0	—	0	0
Function					CS0 enable 0: Disable 1: Enable		Sets the number of dummy cycles to be inserted. (Read recovery time) 00: 2 cycles 01: 1 cycle 10: None 11: Setting not allowed	

	23	22	21	20	19	18	17	16
Bit symbol	B1OM		—	B1BUS	B1W			
Read/Write	W		—	W				
After reset	0	0	—	0	0	1	0	1
Function	Selects chip select output waveform. 00: ROM/RAM Other settings are not allowed.			Selects data bus width. 0: 16 bits 1: 8 bits	Sets the number of wait cycles 0000: 0 cycles 0001: 1 cycle 0010: 2 cycles 0011: 3 cycles 0100: 4 cycles 0101: 5 cycles 0110: 6 cycles 0111: 7 cycles 1111: (1+N) cycles Other settings are not allowed.			

	31	30	29	28	27	26	25	24
Bit symbol	—	—	—	—	B1E	—	B1RCV	
Read/Write	—	—	—	—	W	—	W	
After reset	—	—	—	—	0	—	0	0
Function					CS1 enable 0: Disable 1: Enable		Sets the number of dummy cycles to be inserted. (Read recovery time) 00: 2 cycles 01: 1 cycle 10: None 11: Setting not allowed	

Figure 3.7.4 Chip select/wait control registers

Note: "Please set the number of wait as "+1" when you use = long and BUSRQ the ALE width."

B23CS		7	6	5	4	3	2	1	0
(0xFFFF_E484)	Bit symbol	B2OM		—	B2BUS	B2W			
	Read/Write	W		—	W				
	After reset	0	0	—	0	0	1	0	1
	Function	Selects chip select output waveform. 00: ROM/RAM Other settings are not allowed.			Selects data bus width. 0: 16 bits 1: 8 bits	Sets the number of wait cycles 0000: 0 cycles 0001: 1 cycle 0010: 2 cycles 0011: 3 cycles 0100: 4 cycles 0101: 5 cycles 0110: 6 cycles 0111: 7 cycles 1111: (1+N) cycles Other settings are not allowed.			

		15	14	13	12	11	10	9	8
	Bit symbol	—	—	—	—	B2E	B2M	B2RCV	
	Read/Write	—	—	—	—	W			
	After reset	—	—	—	—	1	0	0	0
	Function					CS2 enable 0: Disable 1: Enable	Selects CS2 space. 0: 4-Gbyte space 1: CS space	Sets the number of dummy cycles to be inserted. (Read recovery time) 00: 2 cycles 01: 1 cycle 10: None 11: Setting not allowed	

		23	22	21	20	19	18	17	16
	Bit symbol	B3OM		—	B3BUS	B3W			
	Read/Write	W		—	W				
	After reset	0	0	—	0	0	1	0	1
	Function	Selects chip select output waveform. 00: ROM/RAM Other settings are not allowed.			Selects data bus width. 0: 16 bits 1: 8 bits	Sets the number of wait cycles 0000: 0 cycles 0001: 1 cycle 0010: 2 cycles 0011: 3 cycles 0100: 4 cycles 0101: 5 cycles 0110: 6 cycles 0111: 7 cycles 1111: (1+N) cycles Other settings are not allowed.			

		31	30	29	28	27	26	25	24
	Bit symbol	—	—	—	—	B3E	—	B3RCV	
	Read/Write	—	—	—	—	W	—	W	
	After reset	—	—	—	—	0	—	0	0
	Function					CS3 enable 0: Disable 1: Enable		Sets the number of dummy cycles to be inserted. (Read recovery time) 00: 2 cycles 01: 1 cycle 10: None 11: Setting not allowed	

Note: The initial value of B23CS<B2BUS> is 1 when AM = High and 0 when AM = Low.

Figure 3.7.5 Chip select/wait control registers

Note: "Please set the number of wait as "+1" when you use = long and BUSRQ the ALE width."

		7	6	5	4	3	2	1	0
BEXCS	Bit symbol	BEXOM		—	BEXBUS	BEXW			
(0xFFFF_E488)	Read/Write	W		—	W				
	After reset	0	0	—	0	0	1	0	1
	Function	Selects chip select output waveform. 00: ROM/RAM Other settings are not allowed.			Selects data bus width. 0: 16 bits 1: 8 bits	Sets the number of wait cycles 0000-0111: 0 cycles to 7 cycles 1111: (1+N) cycles Other settings are not allowed.			
		15	14	13	12	11	10	9	8
	Bit symbol	—	—	—	—	—	—	BEXRCV	
	Read/Write	—	—	—	—	—	—	W	
	After reset	—	—	—	—	—	—	0	0
	Function							Sets the number of dummy cycles to be inserted. (Read recovery time) 00: 2 cycles 01: 1 cycle 10: None 11: Setting not allowed	

Figure 3.7.6 Chip select/wait control registers

Note: "Please set the number of wait as "+1" when you use = long and BUSRQ the ALE width."

3.7.3   Example of Use

Figure 3.7.7 shows an example of a TMP1942 system configuration with external memory connected. In this example a 128-Kbyte ROM is connected with a data width of 16 bits and 256-Kbyte RAM is connected with a data width of 16 bits.
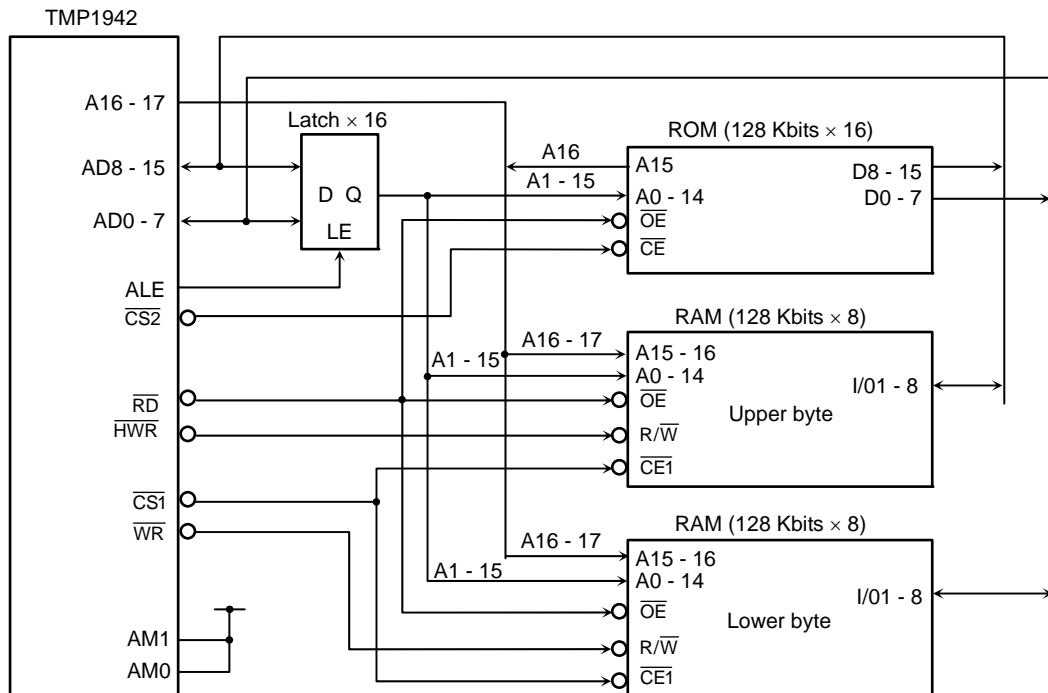
Figure 3.7.7 Example of External Memory Connection (ROM width = 16 bits, RAM width = 16 bits)

When the TMP1942 is reset, the port 4 control register (P4CR) and port 4 function register (P4FC) are both cleared to 0, so that the CS signal output is disabled. To output a CS signal from this port, set the corresponding bits in these registers to 1, first in P4FC and then in P4CR.

3.8 DMA Controller (DMAC)

The TMP1942 incorporates a four-channel DMA controller.

3.8.1 Features

The DMAC included in the TMP1942 has the following features:

(1) Independent 4-channel DMA

(2) Two types of request for control of the bus: with snoop request or without snoop request

(3) Transfer request: Internal transfer request: Start by software
                     External transfer request: Request by interrupt

(4) Transfer mode: Dual-address mode

(5) Transfer devices: Memory-to-memory, memory-to-I/O, I/O-to-memory

(6) Device size: 32 bits for memory (16 or 8 bits can also be specified using the CS/wait controller); 8, 16 or 32 bits for I/O

(7) Address change: Increment, decrement, fixed, irregular increment or irregular decrement

(8) Channel priority: Fixed

3.8.2   Configuration

3.8.2.1   Internal connections in the TMP1942

Figure 3.8.1 shows how the DMAC is connected internally within the TMP1942.

Note * : internal signal
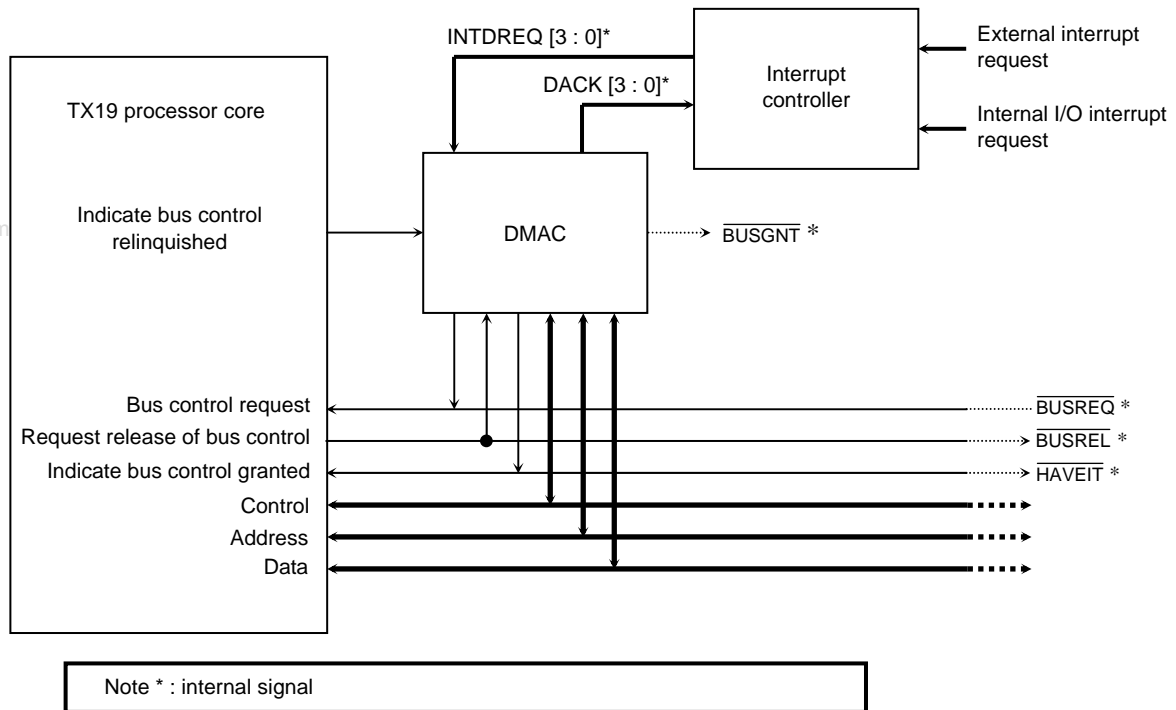
Figure 3.8.1 Internal Connection of DMAC Within the TMP1942

The DMAC has four DMA channels. These channels each receive a data transfer request signal (INTDREQn) from the interrupt controller and return an acknowledge signal ($\overline{\text{DACKn}}$) in response to INTDREQn. The letter 'n' denotes the channel number: 0 to 3. Channel 0 has priority over channel 1, channel 1 has priority over channel 2 and channel 2 has priority over channel 3.

The TX19 processor core has a snoop function. The snoop function entails the TX19 processor core releasing the core data bus to the DMAC so that the DMAC can access the internal ROM or internal RAM connected to the TX19 processor core. The DMAC can choose whether or not to use the snoop function. For details of the snoop function, refer to Section 3.8.2.3, "Snoop function".

There are two types of request for bus control: SREQ and GREQ. The type which is selected depends on whether or not the DMAC is using the snoop function. GREQ is used to request control of the bus when the snoop function is not in use and SREQ is used to request control of the bus when the snoop function is in use. An SREQ bus request has higher priority than a GREQ bus request.

Note : DMA channel priority exists only among those using the same type of bus request signal(SREQ or GREQ).For example, once a given DMA channel has acquired bus mastership using SREQ, no other DMA channel can assume bus mastership using GREQ until the ongoing DMA transaction is completed.

3.8.2.2   Internal blocks of the DMAC

Figure 3.8.2 shows the internal blocks of the DMAC.
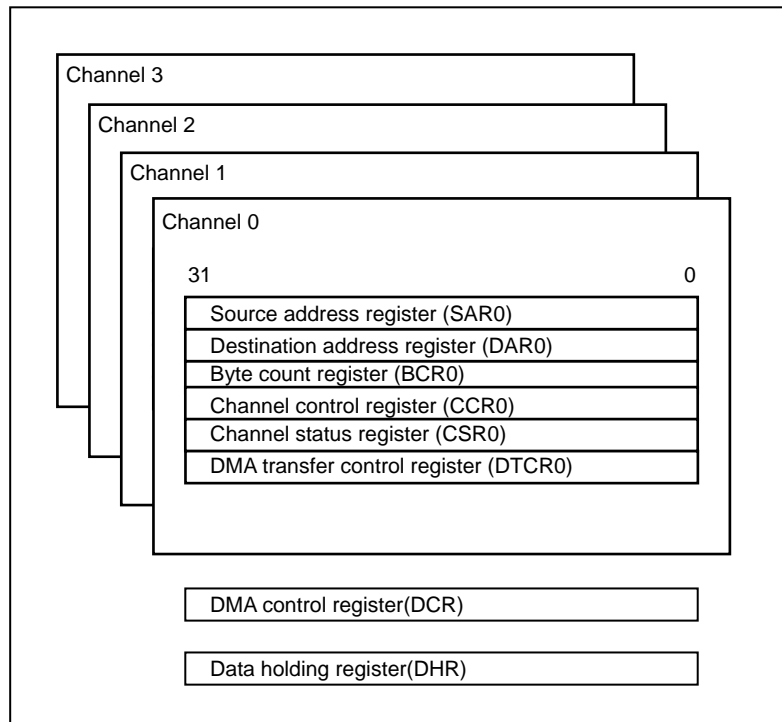
Figure 3.8.2 Internal Blocks of the DMAC

3.8.2.3   Snoop function

The TX19 processor core has a snoop function. This function is used to release the TX19 processor core's data bus to the DMAC.

When the snoop function is activated, the TX19 processor core releases its data bus to the DMAC. At the same time the TX19 processor core stops operating and remains idle until control of the data bus is returned to it by the DMAC. Since the DMAC can access the processor's internal RAM or internal ROM while the snoop function is active, the RAM or ROM can be specified as the source or destination of a transfer.

The TMP1942's internal DMAC can select whether or not to use the TX19 processor core's snoop function. If the DMAC chooses to use the snoop function, it can then access the processor's internal RAM and internal ROM. The CPU in the TX19 processor core will then be stalled until the DMAC cancels the bus request.

If the DMAC chooses not to use the snoop function, it cannot access the processor's internal RAM or internal ROM. However, since in this case too the G-Bus is released to the DMAC, if the TX19 processor core attempts to access memory or I/O via the G-Bus and the DMAC does not respond to the request for release of bus control, the TX19 processor core will not be able to execute bus operation, and as a result the pipeline will stall.

Note: When the snoop function is not used, the TX19 processor core does not release the data bus to the DMAC. Therefore, if the processor's internal RAM or internal ROM is specified as the source or destination of a DMA transfer, no acknowledge signal will be returned for the DMAC's transfer cycle, resulting in the bus being locked.

3.8.3 Registers

The DMAC incorporates twenty-six 32-bit registers. Table 3.8.1 shows the DMAC register map.

Table 3.8.1 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_E280	DCR	DMA control register (DMAC)
0xFFFF_E28C	DHR	Data-holding register (DMAC)

3.8.3.1 DMA control register (DCR)
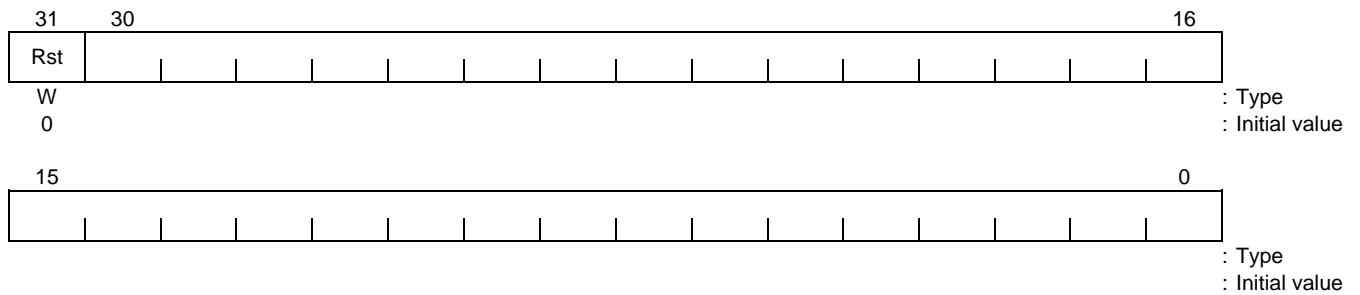
Bit	Mnemonic	Field Name	Description
31	Rst	Reset	Reset (initial value: —)
			Resets the DMAC by software. When the Rst bit is set to 1, all of the DMAC's internal registers are reset to their initial values. Also, all transfer requests are canceled and the four DMA channels are turned off.
			0: Don't care
			1: Initialize the DMAC.

Figure 3.8.3 DMA Control Register (DCR)

Note1: When the snoop request is disabled (CCRn.SReq=0), a software reset of the DMAC must be performed in the following sequence:
  1. Disable interrupts.
  2. Execute NOP four times.
  3. Perform a software reset.
  4. Perform a software reset again.
  5. Re-enable interrupts.
Execute steps 3 and 4 consecutively.

Note 2: If the software reset command is written to the DCR register immediately after the completion of the last transfer cycle of a DMA transaction, the DMA-done interrupt will not be cleared. In this case, the software reset only initializes channel registers, etc.

Note 3: Don't issue a software reset command to the DCR register via a DMA transfer.

3.8.3.2 Channel control registers (CCRn)

31	30				25	24	23	22	21	20	19	18	17	16	
Str	0					—	NIEn	AbIEn	—	—	—	—	Big	—	
W						W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W	: Type
0							1	1	1	0	0	0	1	0	: Initial value

15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
—	ExR	PosE	Lev	SReq	RelEn	SIO	SAC		DIO	DAC		TrSiz		DPS		
R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W		R/W	R/W		R/W		R/W		: Type
0	0	0	0	0	0	0	00		0	00		00		00		: Initial value

Bit	Mnemonic	Field Name	Description
31	Str	Channel Start	Start (initial value: —) Starts channel operation. When this bit is set to 1, the channel enters ready state. Data transfer can now commence as soon as a transfer request is received. 1 is the only valid value which can be written to this bit; if a 0 is written, it is ignored. When read, this bit always appears to be 0. 1: Start channel operation.
24	—	(Reserved)	This bit is reserved. Make sure that it is always set to 0.
23	NIEn	Normal Completion Interrupt Enable	Normal Completion Interrupt Enable (initial value: 1) 1: Enable normal completion interrupts. 0: Disable normal completion interrupts.
22	AbIEn	Abnormal Completion Interrupt Enable	Abnormal Completion Interrupt Enable (initial value 1) 1: Enable abnormal completion interrupts. 0: Disable abnormal completion interrupts.
21	—	(Reserved)	This bit is reserved. Although this bit is initially set to 1, make sure that it is always set to 0.
20	—	(Reserved)	This bit is reserved. Make sure that it is always set to 0.
19	—	(Reserved)	This bit is reserved. Make sure that it is always set to 0.
18	—	(Reserved)	This bit is reserved. Make sure that it is always set to 0.
17	Big	Big-Endian	Big-Endian (initial value: 1) 1: The channel operates in big endian mode. 0: The channel operates in little endian mode. On the TMP1942, set this bit to 0.
16	—	(Reserved)	This bit is reserved. Make sure that it is always set to 0.
15	—	(Reserved)	This bit is reserved. Make sure that it is always set to 0.
14	ExR	External Request Mode	External Request Mode (initial value: 0) Specifies the transfer request mode. 1: External transfer request (interrupt-driven start) 0: Internal transfer request (soft start)
13	PosE	Positive Edge	Positive Edge (initial value: 0) Specifies the valid level for the transfer request signal INTDREQn. This specification is effective only when the transfer request is an external transfer request (i.e., when the ExR bit = 1). In the case of internal transfer requests (i.e., when the ExR bit = 0) the value of PosE is ignored. Be sure to set the PosE bit to 0 and the adjacent Lev bit to 1 .

Figure 3.8.4 Channel Control Registers (CCRn) (1/2)

Bit	Mnemonic	Field Name	Description
12	Lev	Level Mode	Level Mode (initial value: 0) Specifies the method for requesting external transfer. This specification is effective only when the transfer request is an external transfer request (i.e., when the ExR bit = 1). In the case of internal transfer requests (i.e., when the ExR bit = 0), the value of Lev is ignored. Be sure to set the Lev bit to 1.
11	SReq	Snoop Request	Snoop Request (initial value: 0) Specifies whether or not the snoop function is to be used as the bus control request mode. When the function is selected for use, the TX19 processor core's snoop function is activated with the result that the DMAC can use the processor core's data bus. When the function is not selected for use, the TX19 processor core's snoop function remains inactive. 1: The snoop function is used (i.e., the device is in SREQ mode). 0: The snoop function is not used (i.e., the device is in GREQ mode).
10	RelEn	Bus Control Release Request Enable	Release Request Enable (initial value: 0) Specifies whether the DMAC will respond to requests for release of bus control issued from the TX19 processor core. This function is only effective in GREQ mode. In SREQ mode, this function would have no effect since the TX19 processor core cannot generate a request for release of bus control. 1: After the DMAC has taken over bus control, it will respond to requests for release of bus control. When the TX19 processor core issues a request for release of bus control, the DMAC will return control of the bus to the TX19 processor core when a break in bus operation occurs. 0: The DMAC will not respond to requests for release of bus control.
9	SIO	Source I/O	Source Type: I/O (initial value: 0) Specifies the source device from which to perform transfer. 1: I/O device 0: Memory
8 : 7	SAC	Source Address Count	Source Address Count (initial value: 00) Specifies the way in which the source address changes. 1x: The address is fixed. 01: The address is decremented. 00: The address is incremented.
6	DIO	Destination I/O	Destination Type: I/O (initial value: 0) Specifies the destination device to which to perform transfer. 1: I/O device 0: Memory
5 : 4	DAC	Destination Address Count	Destination Address Count (initial value: 00) Specifies the way in which the destination address changes. 1x: The address is fixed. 01: The address is decremented. 00: The address is incremented.
3 : 2	TrSiz	Transfer Size	Transfer Size (initial value: 00) Indicates 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 (4 bytes)
1 : 0	DPS	Device Port Size	Device Port Size (initial value: 00) Specifies the bus width for the I/O device which has been specified as the source or destination device. 11: 8 bits (1 byte) 10: 16 bits (2 bytes) 0x: 32 bits (4 bytes)

Figure 3.8.4 Channel Control Registers (CCRn) (2/2)

3.8.3.3 Channel status registers (CSRn)

31										23	22	21	20	19	18		16
Act				0						NC	AbC	—	BES	BED	Conf		00

| R | | | | | | | | | | R/W | R/W | R/W | R | R | R | | : Type |
| 0 | | | | | | | | | | 0 | 0 | 0 | 0 | 0 | 0 | | : Initial value |

15												3	2		0
					0								—	—	—

R/W : Type
000 : Initial value

Bit	Mnemonic	Field Name	Description
31	Act	Channel Active	Channel Active (initial value: 0) Indicates whether the channel is in ready state. 1: Channel is in ready state. 0: Channel is not in ready state.
23	NC	Normal Completion	Normal Completion (initial value: 0) Indicates whether channel operation has terminated normally. If normal completion interrupts have been enabled by the CCR register, the DMAC generates an interrupt request when this bit is set to 1. The NC bit can be cleared by writing a 0 to it. If a normal completion interrupt has been requested, the interrupt request is dropped when the NC bit is set to 0. If an attempt is made to set the Str bit to 1 while the NC bit = 1, an error results. Be sure to clear the NC bit to 0 before starting the next transfer. Writing a 1 to this bit has no effect. 1: Channel operation has terminated normally. 0: Channel operation has not terminated normally.
22	AbC	Abnormal Completion	Abnormal Completion (initial value: 0) Indicates whether channel operation has terminated abnormally. If abnormal completion interrupts have been enabled by the CCR register, the DMAC generates an interrupt request when this AbC bit is set to 1. The AbC bit can be cleared by writing a 0 to it. If an abnormal completion interrupt has been requested, the interrupt request is cancelled when the AbC bit is set to 0. When the AbC bit is cleared, the BES, BED and Conf bits are also cleared to 0. If an attempt is made to set the Str bit to 1 while the AbC bit = 1, an error results. Be sure to clear the AbC bit to 0 before starting the next transfer. Writing a 1 to this bit has no effect. 1: Channel operation has terminated abnormally. 0: Channel operation has not terminated abnormally.
21	—	(Reserved)	This bit is reserved. Make sure that it is always set to 0.
20	BES	Source Bus Error	Source Bus Error (initial value: 0) 1: A bus error has occurred while the source was being accessed. 0: No bus error has occurred while the source was being accessed.
19	BED	Destination Bus Error	Destination Bus Error (initial value: 0) 1: A bus error has occurred while destination was being accessed. 0: No bus error has occurred while destination was being accessed.
18	Conf	Configuration Error	Configuration Error (initial value: 0) 1: A configuration error has occurred. 0: No configuration error has occurred.
2 : 0	—	(Reserved)	These three bits are all reserved. Always set all of these bits to 0.

Figure 3.8.5 Channel Status Registers (CSRn)

3.8.3.4　Source address registers (SARn)

```
31                                    SAddr                                    16
┌──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┐
│  │  │  │  │  │  │  │  │  │  │  │  │  │  │  │  │
└──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┘
                       R/W                         : Type
                        —                          : Initial value

15                                    SAddr                                     0
┌──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┐
│  │  │  │  │  │  │  │  │  │  │  │  │  │  │  │  │
└──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┘
                       R/W                         : Type
                        —                          : Initial value
```
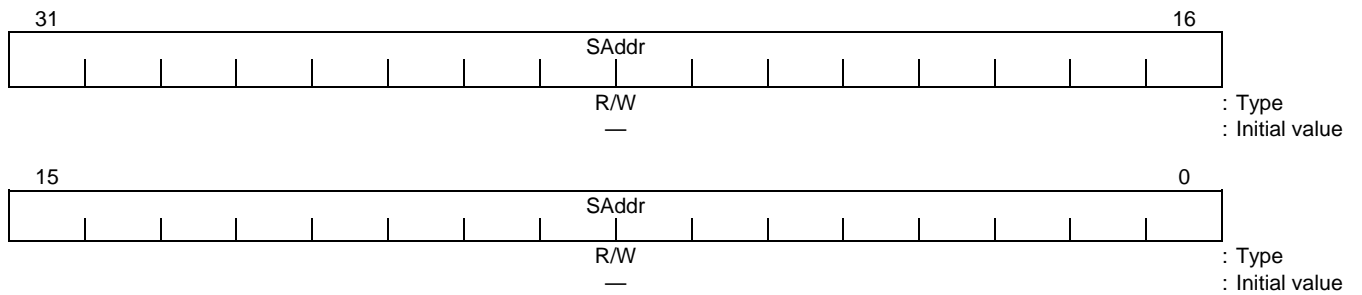
Bit	Mnemonic	Field Name	Description
31:0	SAddr	Source Address	Source Address (initial value: —) Sets the physical source address from which data will be transferred. After each transfer the address will change by the value specified in the DPS bits of the CCRn register.

Figure 3.8.6 Source Address Registers (SARn)

3.8.3.5　Destination address registers (DARn)

```
31                                    DAddr                                    16
┌──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┐
│  │  │  │  │  │  │  │  │  │  │  │  │  │  │  │  │
└──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┘
                       R/W                         : Type
                        —                          : Initial value

15                                    DAddr                                     0
┌──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┬──┐
│  │  │  │  │  │  │  │  │  │  │  │  │  │  │  │  │
└──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┴──┘
                       R/W                         : Type
                        —                          : Initial value
```
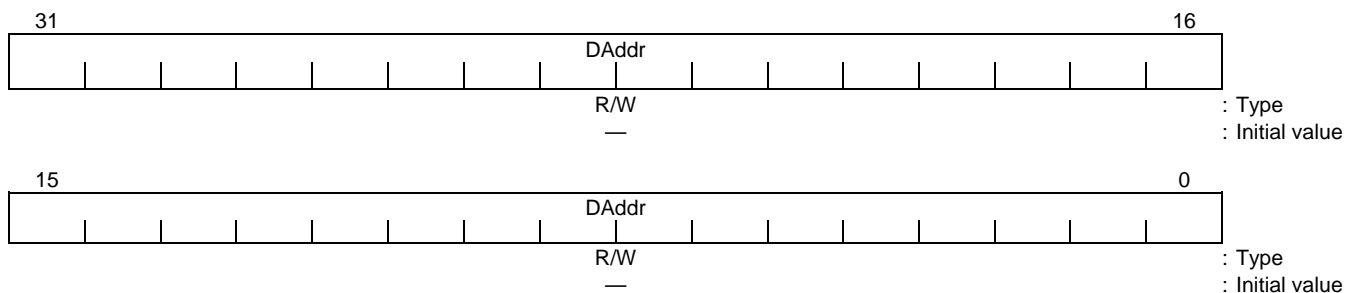
Bit	Mnemonic	Field Name	Description
31:0	DAddr	Destination Address	Destination Address (initial value: —) Sets the physical destination address to which data will be transferred. After each transfer the address will change by the value specified in the DPS bits of the CCRn register.

Figure 3.8.7 Destination Address Registers (DARn)

3.8.3.6 Byte count registers (BCRn)
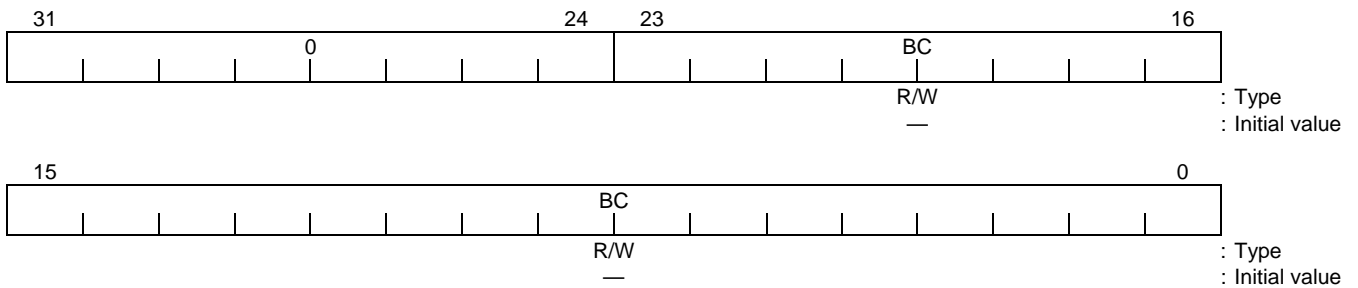
Bit	Mnemonic	Field Name	Description
23:0	BC	Byte count	Byte Count (initial value: —) Sets the number of bytes of data to be transferred. The amount by which the byte count is decremented after each transfer depends on the value specified in the TrSiz bits of the CCRn register.

Figure 3.8.8 Byte Count Registers (BCRn)

3.8.3.7 DMA transfer control registers (DTCRn)

Bit	Mnemonic	Field Name	Description
5 : 3	DACM	Destination Address Count Mode	Destination Address Count Mode Specifies the mode used for counting the destination address. 000: Count the address beginning at bit 0 of the address counter. 001: Count the address beginning at bit 4 of the address counter. 010: Count the address beginning at bit 8 of the address counter. 011: Count the address beginning at bit 12 of the address counter. 100: Count the address beginning at bit 16 of the address counter. 101: Reserved 110: Reserved 111: Reserved
2 : 0	SACM	Source Address Count Mode	Source Address Count Mode Specifies the mode used for counting the source address. 000: Count the address beginning at bit 0 of the address counter. 001: Count the address beginning at bit 4 of the address counter. 010: Count the address beginning at bit 8 of the address counter. 011: Count the address beginning at bit 12 of the address counter. 100: Count the address beginning at bit 16 of the address counter. 101: Reserved 110: Reserved 111: Reserved

Figure 3.8.9 DMA Transfer Control Registers (DTCRn)

3.8.3.8   Data-holding register (DHR)

31   16

| | | | | | | | | DOT | | | | | | | | |

R/W   : Type
—   : Initial value

15   0

| | | | | | | | | DOT | | | | | | | | |

R/W   : Type
—   : Initial value
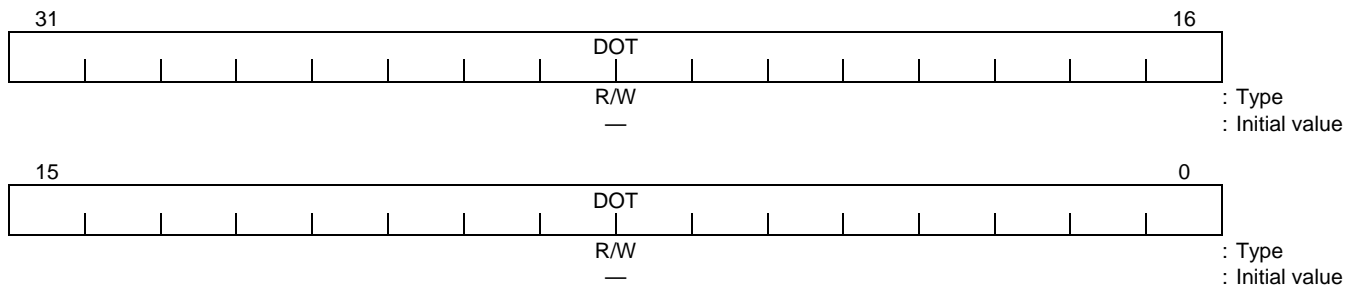
Bit	Mnemonic	Field Name	Description
31 : 0	DOT	Data on Transfer	Data on Transfer (initial value: —) This is the data read from the source during a transfer in dual-address mode.

Figure 3.8.10 Data-Holding Register (DHR)

3.8.4   Functions

This section describes the functions of the DMAC.

3.8.4.1   Outline

The DMAC is a 32-bit DMA controller capable of performing high-speed data transfers in a system incorporating the TX19 processor core without the need for any intervention by the TX19 processor core itself.

(1)   Source and destination

The DMAC performs data transfers between one memory device and another or between a memory device and an I/O device. The device from which data is transferred is referred to as the source device and the device to which data is transferred is referred to as the destination device. Both memory devices and I/O devices can be specified as the source and destination devices. However, the DMAC can only transfer data from a memory device to an I/O device, from an I/O device to memory, or from memory to memory; it cannot transfer data between two I/O devices.

The difference between memory devices and I/O devices resides in the methods by which the devices are accessed. When the DMAC accesses an I/O device, it asserts the DACKn signal. Because only one DACKn signal line is available for each channel, the DMAC can only perform one data transfer involving an I/O device at a time; hence the DMAC cannot transfer data from one I/O device to another.

An interrupt source can be specified for transfer requests to the DMAC. When an interrupt occurs, the interrupt controller generates a request to the DMAC. (In this case, no interrupt request to the TX19 processor core is generated. For details, refer to Section 3.4, "Interrupts".) This interrupt request from the interrupt controller is canceled by the DACKn signal. Therefore, when an I/O device has been set as a transfer device, a request to the DMAC is cancelled for each transfer performed (i.e., each time the amount of data specified by the TrSiz bits is transferred). On the other hand, in memory-to-memory transfers, DACKn is asserted only when the number of bytes to be transferred (as specified by the value of the BCRn register) falls to 0; hence several data transfers can be performed successively by a single transfer request.

For example, when the DMAC is transferring data between the TMP1942's internal I/O and internal (or external) memory, although a transfer request from the internal I/O to the DMAC is cancelled for each transfer performed, the DMAC is kept waiting for the next transfer request unless the number of bytes to be transferred (as specified by the value of the BCRn register) falls to 0. Consequently, DMA transfer is performed successively until the BCRn register value is reduced to 0.

(2)   Switching control of the bus (bus arbitration)

When a transfer request is issued by the DMAC's internal circuitry, the DMAC requests control of the bus from the TX19 processor core. If an acknowledge signal is returned by the TX19 processor core, the DMAC gains control of the bus and can perform data transfer bus cycles.

The DMAC can request two types of bus control: either bus control plus the use of the TX19 processor core's data bus (i.e., the snoop function), or bus control without the snoop function. This can be set independently for each channel in the corresponding register.

The TX19 processor core may request release of bus control from the DMAC. Whether the DMAC should respond to this request is set using independent register settings for each channel. However, this response function is effective only when the DMAC does not request the snoop function (i.e., in GREQ mode). When the snoop function is requested (i.e., in SREQ mode), the response function will have no effect because the TX19 processor core cannot generate requests for release of bus control in this mode.

When there are no more transfer requests, the DMAC will finish control of the bus.

Note1: The NMI interrupt is left pending while the DMAC has control of the bus.
Note2: Do not place the TMP1962 in Halt power-down mode while the DMAC is operating.

---

(3)   Transfer request modes

The DMAC has two transfer request modes: internal transfer request mode and external transfer request mode.

In internal transfer request mode, transfer requests are generated internally in the DMAC. A transfer request is generated by setting the start bit in one of the DMAC's internal registers (the channel control register's Str bit) to 1, upon which the DMAC will start a transfer operation.

In external transfer request mode, transfer requests are generated by assertion of the transfer request signal (INTDREQn), which is output by the interrupt controller after the start bit has been set to 1. The DMAC can select level mode, in which a transfer request is generated on detection of a High- or Low-level INTDREQn signal, or edge mode, in which a transfer request is generated on detection of the rising or falling edge of the INTDREQn signal. However, because the INTDREQn signal in the TMP1942 is low-active, always make sure that the transfer request signal is set to be detected at Low level.

(4)   Address modes

Dual-address mode is the only address mode available for the DMAC in the TMP1942. There is no single-address mode for the DMAC.

In dual-address mode, data transfers are performed between two memory devices or between memory and an I/O device. The addresses of the source and destination devices are output by the DMAC. When accessing an I/O device, the DMAC asserts the DACKn signal. In dual-address mode, the DMAC executes two bus operations, one for reading and one for writing. The transfer data read from the source device is temporarily stored in the DMAC's internal data-holding register (DHR) before being written to the destination device.

(5) Channel operation

The DMAC has four channels (channels 0 to 3). Each channel is activated by setting the start bit (Str) in the channel control register (CCRn) to 1, so that the device enters ready state.

When a transfer request occurs while a channel is in ready state, the DMAC gains control of the bus and performs a data transfer. When there are no more transfer requests, the DMAC finishes control of the bus, thereby entering ready state. When transfer for a channel is completed, the channel is placed in idle state. Transfers may be terminated either normally or abnormally (for example, when an error occurs during transfer). An interrupt signal can be generated on completion of transfer.

Figure 3.8.11 is a state transition diagram for channel operations.
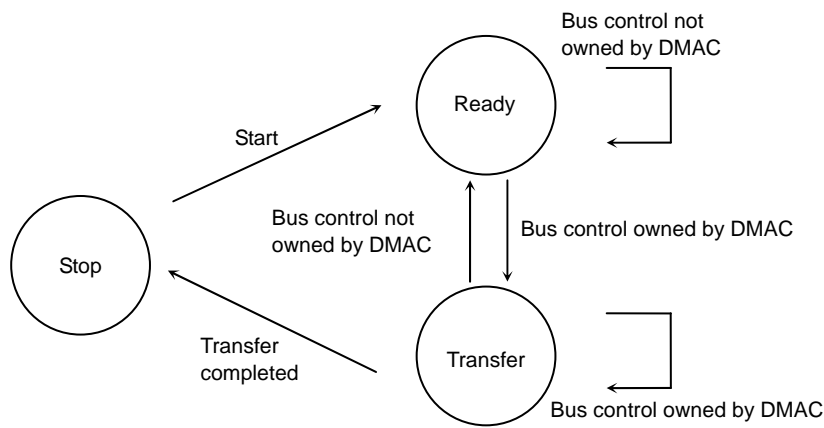
Figure 3.8.11 State Transitions for Channel Operations

(6) Summary of transfer mode combinations

The DMAC can perform data transfers as follows according to the combination of mode settings.

Transfer Request	Edge/Level	Address Mode	Transfer Devices
Internal	—	Dual	Memory-to-memory
External	Low-level	Dual	Memory-to-memory
			Memory-to-I/O
			I/O-to-memory

(7) Address change

There are essentially three methods for changing the transfer address: increment, decrement or fixed. The method can be set independently for the source and destination addresses using the SAC and DAC bits in the CCRn register. If the transfer device is a memory device, increment, decrement or fixed may be specified. If the transfer device is an I/O device, only fixed may be specified. When an I/O device is selected as the source or destination device, be sure to set the SAC and DAC bits in the CCRn register to "fixed".

If "increment" or "decrement" is selected as the address change method, the bit position at which counting begins can be set using the SACM and DACM bits in the DTCRn register. SACM corresponds to the source address and DACM the destination address. The bit position at which counting the address begins can be specified as bit 0, 4, 8, 12 or 16. Selecting bit 0 results in normal increment or decrement, increment or irregular decrement.

Examples of how the address changes are shown below.

Example 1: When regular increment is selected for the source device and irregular increment is selected for the destination device

SAC:	Increment the address
DAC:	Increment the address
TrSiz:	Transfer in units of 32 bits
Source address:	0xA000_1000
Destination address:	0xB000_0000
SACM = 000:	Count the address beginning at bit 0 of the address counter
DACM = 001:	Count the address beginning at bit 4 of the address counter

	Source	Destination
First time	0xA000_1000	0xB000_0000
Second time	0xA000_1004	0xB000_0010
Third time	0xA000_1008	0xB000_0020
Fourth time	0xA000_100C	0xB000_0030

Example 2: When irregular decrement is selected for the source device and regular decrement is selected for the destination device

SAC:	Decrement the address
DAC:	Decrement the address
TrSiz:	Transfer in units of 16 bits
Source address:	Initial value 0xA000_1000
Destination address:	0xB000_0000
SACM = 010:	Count the address beginning at bit 8 of the address counter
DACM = 000:	Count the address beginning at bit 0 of the address counter

	Source	Destination
First time	0xA000_1000	0xB000_0000
Second time	0x9FFF_FF00	0xAFFF_FFFE
Third time	0x9FFF_FE00	0xAFFF_FFFC
Fourth time	0x9FFF_FD00	0xAFFF_FFFA

3.8.4.2   Transfer requests

For data to be transferred by the DMAC, a transfer request must be generated and transmitted to the DMAC. There are two types of DMAC transfer requests: internal transfer requests and external transfer requests. The transfer request type can be set individually for each channel.

For either type of transfer request, when a transfer request occurs after channel operation has been activated, the DMAC will gain control of the bus and perform data transfer.

- Internal transfer requests

    A transfer request can be generated immediately by setting the Str bit in the CCRn register to 1 while the ExR bit in the same register = 0. This transfer request is referred to as an internal transfer request.

    In the case of an internal transfer request, because the transfer request remains active until channel operation has been completed, data transfers will be performed successively unless transition to a higher priority channel occurs or until bus control is transferred to a higher priority bus master.

    Internal transfer requests can only be used for transfers between memory and memory.

- External transfer requests

    A transfer request is generated when the interrupt controller is notified of a transfer request by the assertion of the INTDREQn signal for a channel after the channel has been placed in ready state by setting the Str bit of the CCRn register to 1 while the ExR bit in the CCRn register = 0. This transfer request is referred to as an external transfer request. External transfer requests can be used for transfers between two memory devices and between memory and an I/O device.

    Assertion of the INTDREQn signal is recognized by detecting an edge or a level. The active edge or level is specified using the PosE bit in the CCRn register. However, because the INTDREQn signal in the TMP1942 is low-active, always make sure that the signal is set to be detected at Low level.

    The amount of data to be transferred for one transfer request is specified using the TrSiz field in the CCRn register. This can be specified as 32 bits, 16 bits or 8 bits.

    Transfer requests from the interrupt controller are cleared by assertion of the DACKn signal. The DACKn signal is asserted only when the number of bytes to be transferred during an I/O device bus cycle or a memory-to-memory transfer (as specified by the value of the BCRn register) falls to 0. Consequently, for data transfer between memory and an I/O device INTDREQn is cancelled every transfer request with the result that only one transfer is performed for the amount of data specified by TrSiz. On the other hand, in memory-to-memory transfers, INTDREQn is not cancelled until the number of bytes to be transferred (as specified by the value of the BCRn register) falls to 0; hence several data transfers can be performed successively by a single transfer request.

    Note that if an interrupt of the type specified for INTDREQn is acknowledged by the DMAC, but the interrupt is cleared by the interrupt controller or by another device before the DMAC starts the DMA transfer, one DMA transfer may be performed after the interrupt has been cleared.

3.8.4.3   Address modes

The TMP1942 only supports dual-address mode in which both the source and destination devices are explicitly addressed..

In dual-address mode the DMAC first executes a read from the source device. The data read from the source device is temporarily stored in the DMAC's internal register DHR. Next, the DMAC executes a write to the destination device to write this data to the destination device, thus performing a data transfer from the source to the destination device.

Although bit 15 of the CCRn register in the TMP1942 can be used to specify the address mode, this bit must always be set to 0 because the TMP1942 only supports dual-address mode.
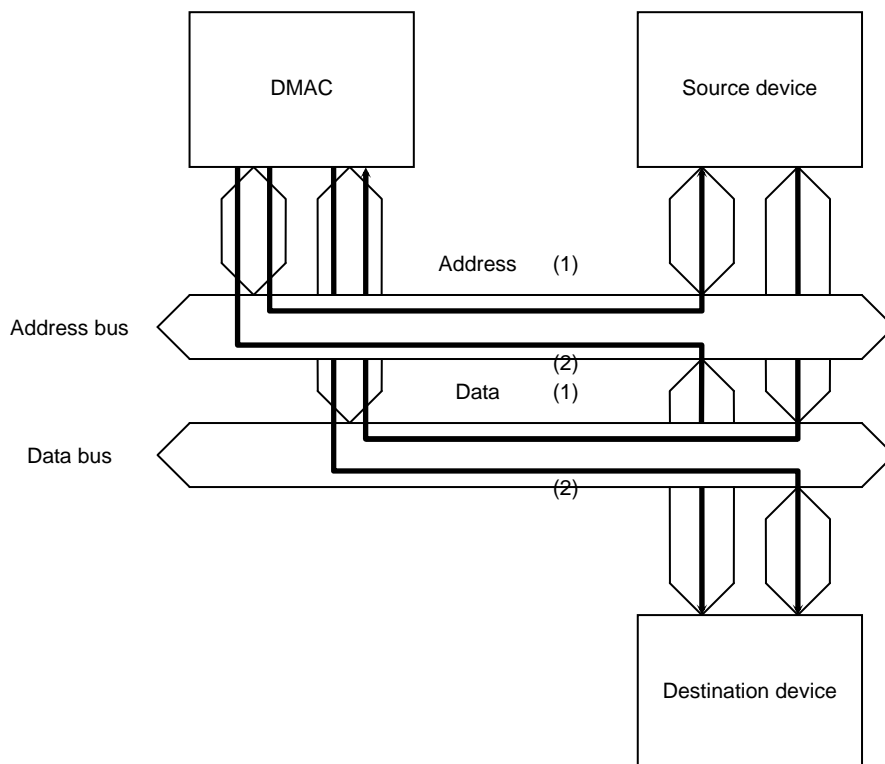
Figure 3.8.12 Diagram of Data Transfer in Dual-Address Mode
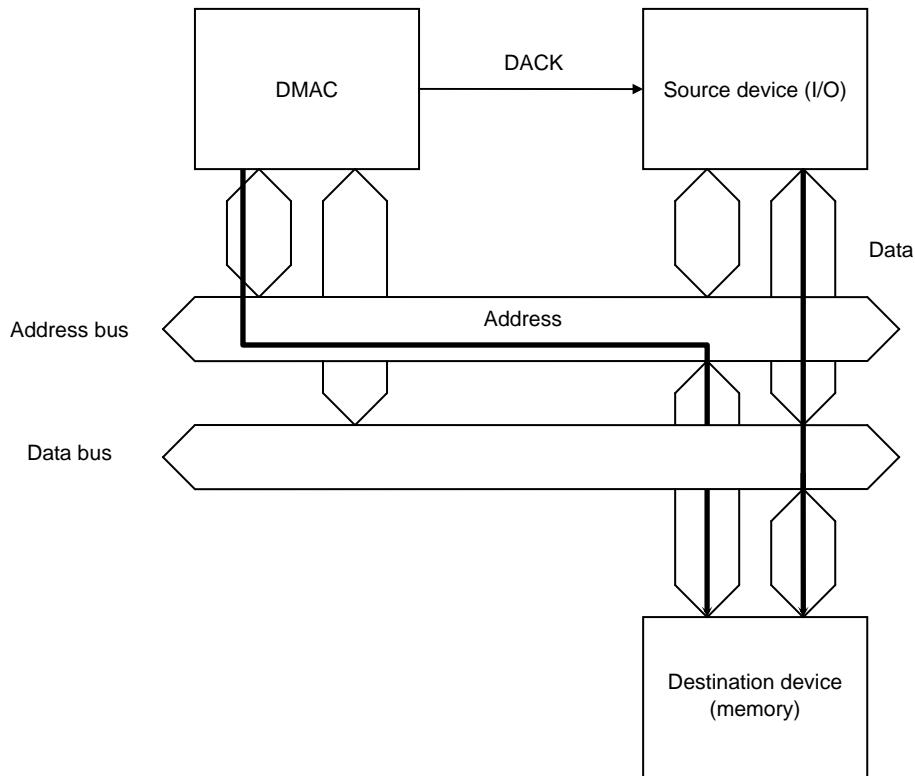
Figure 3.8.13 Diagram of Data Transfer in Single-Address Mode

- Dual-address mode

    In dual-address mode, a data transfer is executed using two bus operations:

  -Read operation, in which the DMAC outputs the address of the source device, reads data from the source device and stores the data in its internal register DHR

  -Write operation, in which the DMAC outputs the address of the destination device and writes the stored data from DHR to the destination device

    In dual-address mode, three types of transfers can be performed:

  -Memory-to-memory

  -Memory-to-I/O device

  -I/O device-to-memory

    The units of data transfer performed by the DMAC are equal to the amount of data (32 bits, 16 bits or 8 bits) specified in the TrSiz field of the CCRn register. This amount of data is transferred each time a transfer request is recognized.

    In dual-address mode, an amount of data equal to the transfer unit is read from the source device into the DHR register, then the data is written from the DHR register to the destination device.

    Memory accesses occur at intervals equal to the unit of data transfer which has been set. When external memory is accessed, if the transfer unit is 32 bits and the bus width set by the CS/wait controller is 16 bits, then two 16-bit accesses will occur. Similarly, if the transfer unit is 32 bits and the bus width set by the CS/wait controller is 8 bits, then four 8-bit accesses will occur.

For memory-to-I/O device or I/O device-to-memory data transfers, the bus width of the I/O device (the device port size) needs to be set (to 32 bits, 16 bits or 8 bits) using the DPS field in the CCRn register, in addition to the unit of data transfer.

If the unit of data transfer and the device port size are equal, the DMAC will perform one read or write operation for the I/O device.

If the device port size is smaller than the unit of data transfer, the DMAC will perform multiple read or write operations for the I/O device. For example, when performing a transfer to memory from an I/O device whose device port size is 8 bits when the unit of data transfer is 32 bits, the DMAC will read data from the I/O device and store it in the DHR register four times, 8 bits at a time, and then write 32 bits of data from the DHR register to memory in one operation (or in two operations if the external memory's data bus is 16 bits wide).

The source and destination addresses change at intervals equal to the unit of data transfer. The value of the BCRn register also changes by an amount equal to the unit of data transfer. The device port size cannot be set to a value greater than the unit of data transfer. Table 3.8.2 summarizes the above information:

Table 3.8.2 Unit of Data Transfer and Device Port Size (Dual-Address Mode)

TrSiz	DPS	Number of Bus Operations Performed on I/O Device
0x (32 bits)	0x (32 bits)	Once
0x (32 bits)	10 (16 bits)	Twice
0x (32 bits)	11 (8 bits)	Four times
10 (16 bits)	0x (32 bits)	Setting prohibited
10 (16 bits)	10 (16 bits)	Once
10 (16 bits)	11 (8 bits)	Twice
11 (8 bits)	0x (32 bits)	Setting prohibited
11 (8 bits)	10 (16 bits)	Setting prohibited
11 (8 bits)	11 (8 bits)	Once

Note: The DMAC does not incremnt or decrement the address for I/O peripherals. Therefore, if, for example, TrSiz is programmed to 16 bits and DPS is programmed to 8 bits, both the first and second bus cycles access the lower eight bits of the I/O data bus.

3.8.4.4　Channel operations

A channel is activated when the Str bit in the CCRn register for the channel is set to 1. When a channel is activated, it is checked for errors; if no error is found, it is placed in ready state.

If a transfer request occurs while a channel is in ready state, the DMAC gains control of the bus and starts a transfer operation.

Channel operation may terminate normally or abnormally, for example, when operation is forcibly terminated or terminated by an error. This status is indicated by the CSRn register.

(1)　Starting channel operation

A channel is activated when the Str bit in the CCRn register for the channel is set to 1.

When a channel is activated, it is checked for a configuration error; if no error is found, it is placed in ready state. If an error is detected, the channel operation terminates abnormally. When a channel is placed in ready state, the Act bit in the CSRn register for the channel is set to 1.

If internal transfer requests have been set for the channel, a transfer request will be generated immediately, upon which the DMAC will gain control of the bus and start a data transfer. If external transfer requests have been set for the channel, a transfer request will be generated by assertion of INTDREQn, upon which the DMAC will gain control of the bus and start a data transfer.

(2)　Terminating channel operation

Channel operation may terminate either normally or abnormally. This status is indicated in the CSRn register.

If an attempt is made to set the Str bit in the CCRn register to 1 while the NC bit or AbC bit of the CSRn register = 1, channel operation will not start and will terminate abnormally.

Normal termination

Channel operation terminates normally in the following case. Note that, in this case, transfer will always terminate after the DMAC has finished transferring an amount of data equal to the unit of data transfer (the value set in the TrSiz field of the CCRn register).

- When data transfer has been completed after the value of the BCRn register has fallen to 0

Abnormal termination

Data transfers by the DMAC may terminate abnormally in the following cases:

- Termination due to configuration errors

    A configuration error is an error in the DMA transfer settings. Since a configuration error occurs before the DMAC starts data transfer operation, the SARn, DARn and BCRn register values will remain as set. When operation for a channel terminates abnormally due to a configuration error, the Conf bit in the CSRn register is set to 1 at the same time that the AbC bit is set to 1. Causes of configuration errors are shown below.

-Both SIO and DIO are set to 1.

-The CCRn Str bit is set to 1 when the NC bit or AbC bit in the CSRn register = 1.

-A value which cannot be divided by the unit of data transfer is set in the BCRn register.

-Values which cannot be divided by the unit of data transfer are set in the SARn and DARn registers.

-An illegal combination of the device port size and data transfer unit has been set.

-The Str bit in the CCRn register is set to 1 while the BCRn register = 0.

- Termination due to bus errors

    When transfer terminates abnormally due to a bus error, the BES or BED bit in the CSRn register is set to 1 at the same time that the AbC bit in the CSRn register is set to 1.

    -The CPU is notified that a bus error has occurred during data transfer.

3.8.4.5 Channel priority

    The DMAC has four channels. A channel with a lower channel number always has higher priority. Therefore, if transfer requests occur for channels 0 and 1 simultaneously, the DMAC will perform the transfer operation for channel 0's transfer request first. When there are no more transfer requests for channel 0, if the transfer request for channel 1 is still in effect, the DMAC will perform the transfer operation on channel 1. (For internal transfer requests, the transfer request is held unless it is cleared. For external transfer requests, this depends on the active state which has been set for the interrupt request assigned to DMA requests by the interrupt controller. If the active state is set to edge mode, the transfer request will be held by the interrupt controller. However, if the active state is set to level mode, the interrupt controller will not hold the transfer request. Therefore, if level mode is set, the interrupt request signal must be kept asserted until it is recognized by the DMAC.)

    If a transfer request for channel 0 occurs while data transfer on channel 1 is under way, a channel transition will occur. The data transfer on channel 1 will be suspended and the DMAC will start transfer on channel 0. When there are no more transfer request for channel 0, the DMAC will resumes the transfer operation on channel 1.

    Channel transition occurs when the DMAC has finished transferring an amount of data equal to the unit of data transfer. In dual-address mode, this is when the DMAC has finished writing all the stored data from the DHR register to the destination device.

3.8.4.6 Interrupts

    The DMAC can generate an interrupt request to the TX19 processor core on completion of channel operation. There are two types of interrupts which can be requested in this case: normal completion interrupt and abnormal completion interrupt.

- Normal completion interrupt

    When channel operation terminates normally, the NC bit in the CSRn register is set to 1. At this time, if normal completion interrupts have been enabled using the NIEn bit in the CCRn register, an interrupt request to the TX19 processor core is generated.

- Abnormal completion interrupt

    When channel operation terminates abnormally, the AbC bit in the CSRn register is set to 1. At this time, if abnormal completion interrupts have been enabled by the AbIEn bit in the CCRn register, an interrupt request to the TX19 processor core is generated.

3.8.4.7   Endian mode

If the unit of data transfer and the device port size are not equal in dual-address mode, the DMAC will assemble or disassemble data in the DHR register.

For example, if the source device is an I/O device whose port size is 8 bits while the destination device is a memory device, and the unit of data transfer is 32 bits, the DMAC reads data from the I/O device four times and assembles it into 32 bits of data in the DHR register before writing it to memory.

For example, the diagram below shows the relationship between an 8-bit I/O device and a 32-bit DHR register.

The TMP1942 supports only little-endian data alignment.
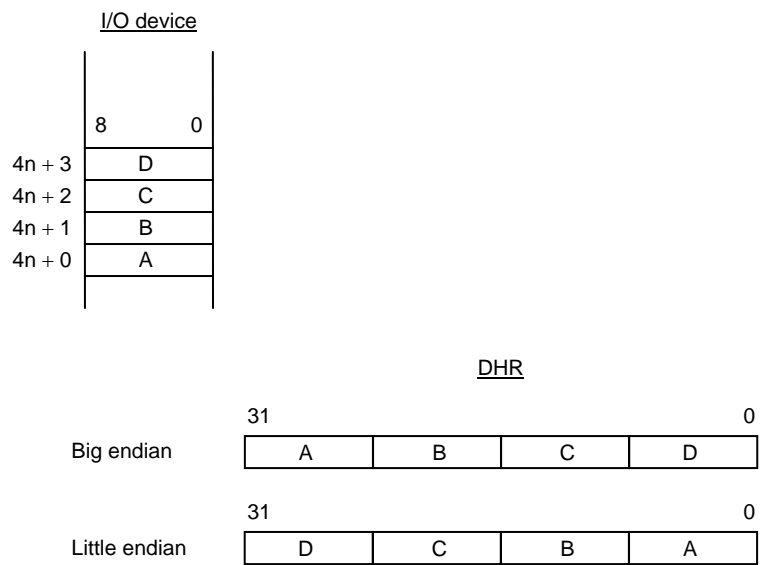
Figure 3.8.14 Data Packing and Unpacking

3.8.5 Operation

DMAC operations are synchronized to the rising edges of SYSCLK.

3.8.5.1 Dual-address mode

- Memory-to-memory transfer

    Figure 3.8.15 shows a timing example for one transfer session when 16-bit data is being transferred from external memory (which is 16 bits wide) to external memory (which is also 16 bits wide). Although it is not shown here, data is transferred successively until the value of the BCRn register falls to 0.
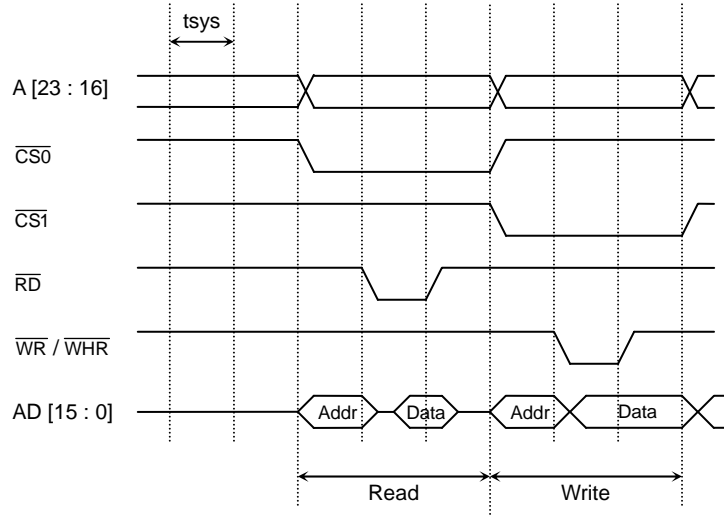
Figure 3.8.15 Dual-Address Mode (Memory to Memory)

- Memory-to-I/O device transfer

    Figure 3.8.16 shows a timing example for memory-to-I/O device transfer for cases where the unit of data transfer and the device port size are set to 16 bits and 8 bits, respectively.
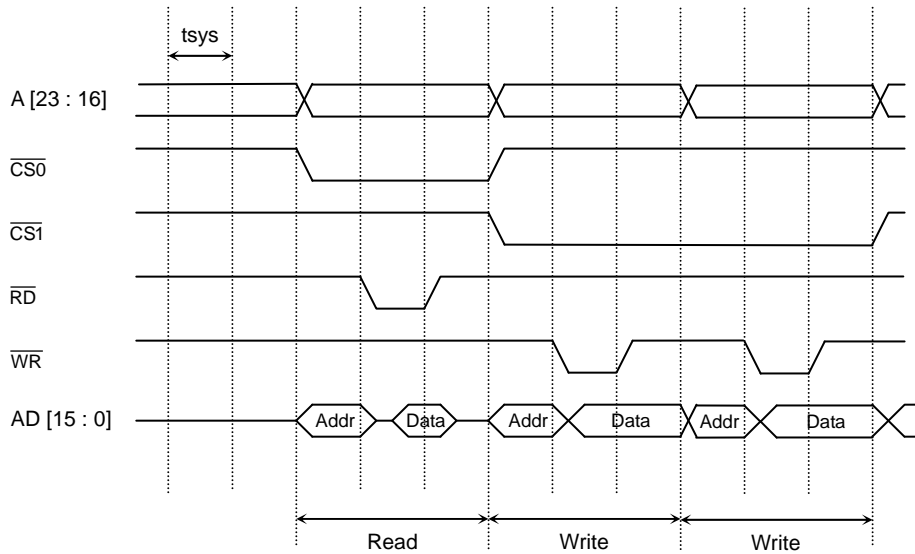
Figure 3.8.16 Dual-Address Mode (Memory to I/O Device)

- I/O device-to-memory transfer

    Figure 3.8.17 shows a timing example for I/O device-to-memory transfer for cases where the unit of data transfer and the device port size are set to 16 bits and 8 bits, respectively.
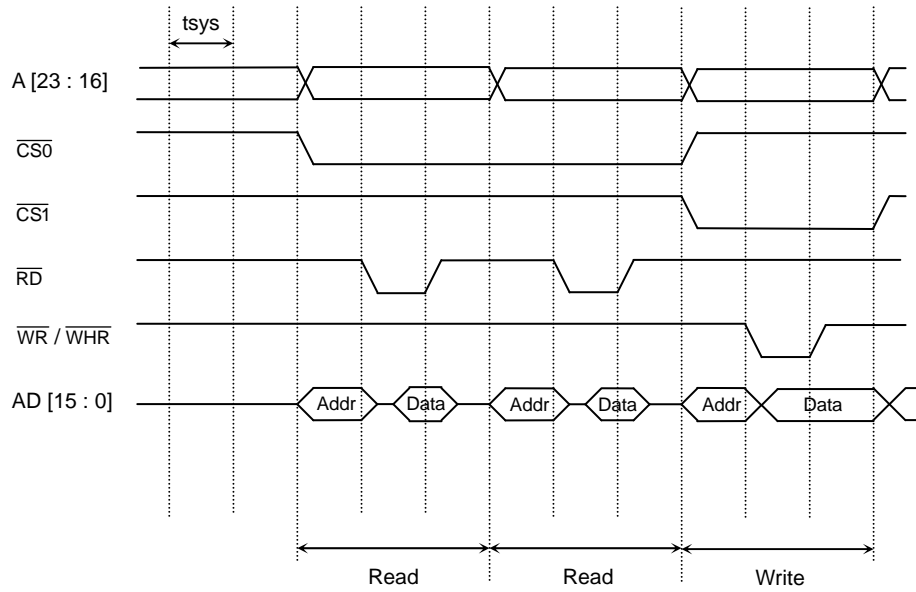
Figure 3.8.17 Dual-Address Mode (I/O Device to Memory)

Example: DMA transfer of serially received data (SCnBUF) to internal RAM

Example DMA settings

- Channel used: 0

- Source address: SC1BUF

- Destination: 0xFFFF_9800 (physical address)

- Number of bytes transferred: 256

Example serial channel settings

- Data length: 8 bits, UART

- Serial channel: Channel 1

- Transfer rate: 9600 bps

DMA (channel 0) is used for transfer. DMA0 is activated by an interrupt received on SIO1.

DMA0 settings

```
DCR     ←   0x8000_0000              /* Reset DMA */
IMCFL   ←   15  7  0
        ←   xxxx, xxxx, xx10, x100   /* Level = 4 (arbitrary value) */
INTCLR  ←   0x3c                     /* Value of IVR [9:4] */
DTCR0   ←   0x0000_0000              /* DACM = 000 */
        ←                            /* SACM = 000 */
SAR0    ←   0xFFFF_F208              /* Physical address of SC1BUF */
DAR0    ←   0xFFFF_9800              /* Physical address of destination */
BCR0    ←   0x0000_00FF              /* 256 (number of bytes to be transferred) */
CCR0    ←   0x80C0_5B0F
```

(Contents)

31				27				23				19			
1	0	0	0	0	0	0	0	1	1	0	0	0	0	0	0

15				11				7				3			
0	1	0	1	1	x	1	1	x	0	0	0	1	1	1	1
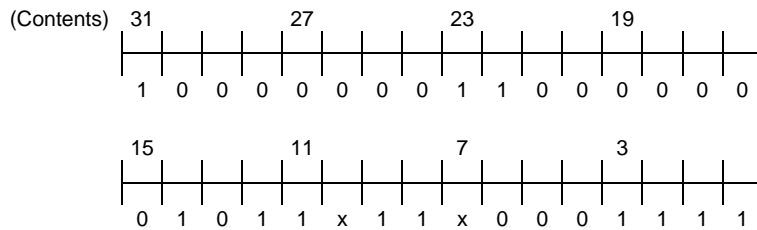
SIO channel 1 settings

```
IMCCH   ←   31  15
            xxxx, xxxx, xx11, 1000   /* Assign to DMC0 activation source */
INTCLR  ←   0x32                     /* IVR [9:4], INTRX1 interrupt source */
SC1MOD0 ←   0x29                     /* UART mode, 8-bit length, baud rate generator */
SC1CR   ←   0x00
BR1CR   ←   0x1d                     /* @fc = 32 MHz (approx. 9615 bps) */
```

3.9   8-Bit Timers (TMRA)

The TMP1942 contains twelve 8-bit timer channels (TMRA0-TMRAB).

There are six TMRA modules, referred to as TMRA01, TMRA23, TMRA45, TMRA67, TMRA89 and TMRAAB, each of which is comprised of two channels. Each module can operate in the following four modes:

- 8-bit interval timer mode

- 16-bit interval timer mode

- 8-bit programmable square wave output mode (PPG: variable duty with variable cycle)

- 8-bit pulse width modulation output mode (PWM: variable duty with constant cycle)

Figure 3.9.1 shows a block diagram of TMRA01. Each channel consists primarily of an 8-bit up-counter, an 8-bit comparator and an 8-bit timer register. Each pair of channels also incorporates one prescaler and one timer flip-flop. Timer operation modes and flip-flops are controlled by five registers.

The six modules (TMRA01, TMRA23, TMRA45, TMRA67, TMRA89 and TMRAAB) operate independently of each other. Because each module functions the same way except for a few differences as shown in Tables Table 3.9.1 and Table 3.9.2, operation of the TMRA01 only is described here.

Table 3.9.1   Specification Differences Among the TMRA Modules

Specification		Module		
		TMRA01	TMRA23	TMRA45
External pins	External clock input pin	TA0IN (Shared with PA7)	TA2IN (Shared with PB7)	TA4IN (Shared with PC0)
	Timer flip-flop output pin	TA1OUT (Shared with PA6)	TA3OUT (Shared with PB6)	TA5OUT (Shared with PC3)
SFR Name (address)	Timer run register	TA01RUN (0xFFFF_F100)	TA23RUN (0xFFFF_F108)	TA45RUN (0xFFFF_F110)
	Timer registers	TA0REG (0xFFFF_F102) TA1REG (0xFFFF_F103)	TA2REG (0xFFFF_F10A) TA3REG (0xFFFF_F10B)	TA4REG (0xFFFF_F112) TA5REG (0xFFFF_F113)
	Timer mode register	TA01MOD (0xFFFF_F104)	TA23MOD (0xFFFF_F10C)	TA45MOD (0xFFFF_F114)
	Timer flip-flop control register	TA1FFCR (0xFFFF_F105)	TA3FFCR (0xFFFF_F10D)	TA5FFCR (0xFFFF_F115)

Table 3.9.2   Specification Differences Among the TMRA Modules

Specification		Module		
		TMRA67	TMRA89	TMRAAB
External pins	External clock input pin	TA6IN (Shared with PC1)	TA8IN (Shared with PC2)	TAAIN (Shared with PC4)
	Timer flip-flop output pin	TA7OUT (Shared with PC5)	TA9OUT (Shared with PC7)	TABOUT (Shared with PD5)
SFR Name (address)	Timer run register	TA67RUN (0xFFFF_F118)	TA89RUN (0xFFFF_F120)	TAABRUN (0xFFFF_F128)
	Timer registers	TA6REG (0xFFFF_F11A) TA7REG (0xFFFF_F11B)	TA8REG (0xFFFF_F122) TA9REG (0xFFFF_F123)	TAAREG (0xFFFF_F12A) TABREG (0xFFFF_F12B)
	Timer mode register	TA67MOD (0xFFFF_F11C)	TA89MOD (0xFFFF_F124)	TAABMOD (0xFFFF_F12C)
	Timer flip-flop control register	TA7FFCR (0xFFFF_F11D)	TA9FFCR (0xFFFF_F125)	TABFFCR (0xFFFF_F12D)

3.9.1 Block diagram of each module

Only a block diagram of TMRA01 is described here. It applies to all other modules with the exception of differences in register, signal and other element names.
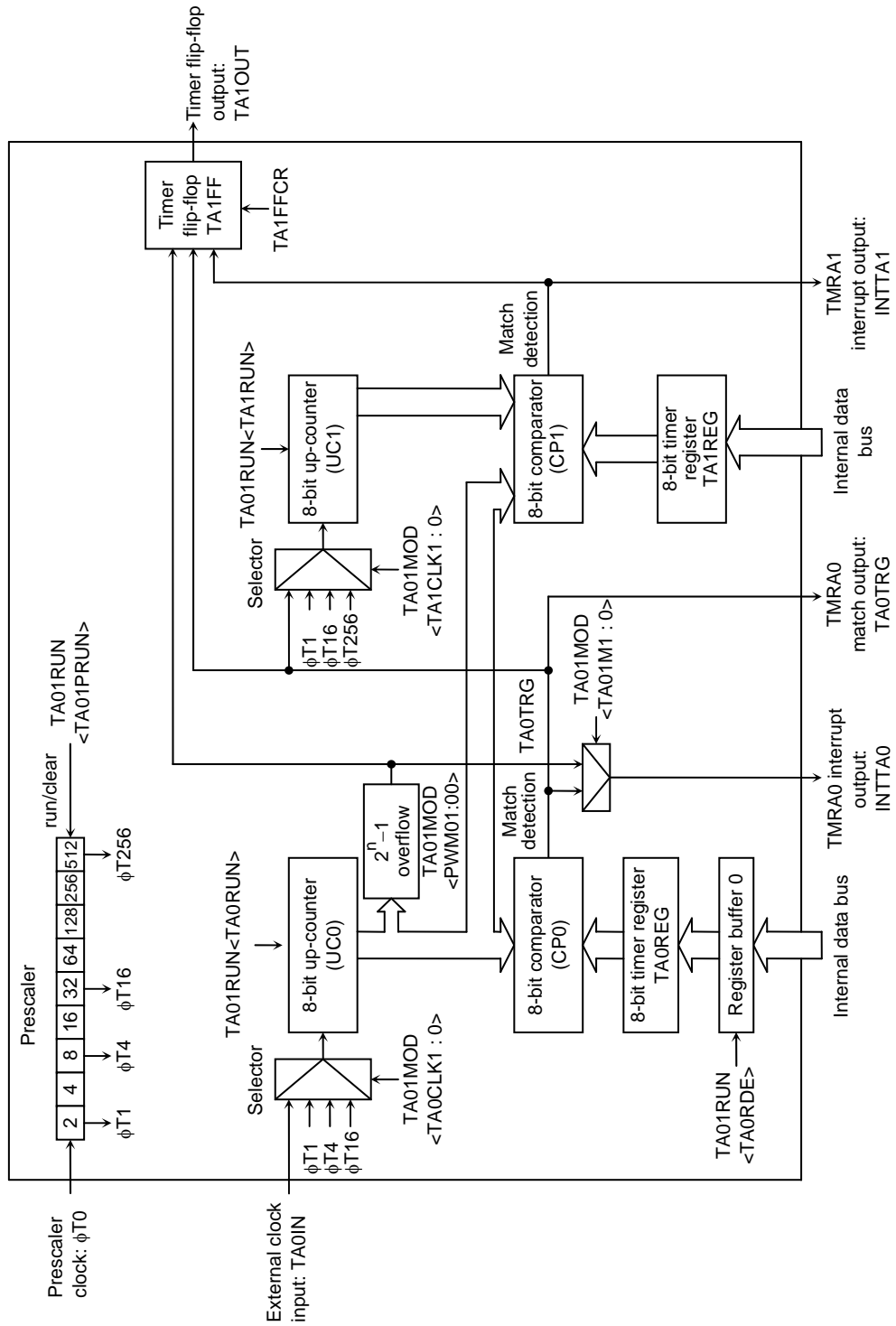
Figure 3.9.1 TMRA01 Block D iagram

3.9.2 Functional description of each circuit

(1) Prescaler

The TMP1942 has a 9-bit prescaler to supply a clock to TMRA01. The prescaler's input clock ϕT0 has a frequency of fperiph, fperiph/2 or fperiph/4 as selected by SYSCR0<PRCK1:PRCK0> in the CG block.

fperiph is either the clock fgear as selected by SYSCR1<FPSEL> in the CG block or the clock fc before division by the clock gear.

The prescaler is set to either run or stop by TA01RUN<TA0PRUN>. Writing a 1 to this bit causes the prescaler to start counting and writing 0 causes it to clear itself and stop counting. Table 3.9.3 shows the resolutions of the prescaler output clocks.

Table 3.9.3 Prescaler Output Clock Resolutions

@fc = 32 MHz

Peripheral Clock Selection <FPSEL>	Clock Gear Value <GEAR1:0>	Selected Prescaler Clock <PRCK1:0>	Prescaler Output Clock Resolution			
			ϕT1	ϕT4	ϕT16	ϕT256
0 (fgear)	00 (fc)	00 (fperiph/4)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^{11}$ (64 μs)
		01 (fperiph/2)	$fc/2^2$ (0.125 μs)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^{10}$ (32 μs)
		10 (fperiph)	—	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^9$ (16 μs)
	01 (fc/2)	00 (fperiph/4)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{12}$ (128 μs)
		01 (fperiph/2)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^{11}$ (64 μs)
		10 (fperiph)	—	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^{10}$ (32 μs)
	10 (fc/4)	00 (fperiph/4)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16 μs)	$fc/2^{13}$ (256 μs)
		01 (fperiph/2)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{12}$ (128 μs)
		10 (fperiph)	—	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^{11}$ (64 μs)
	11 (fc/8)	00 (fperiph/4)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32 μs)	$fc/2{14}$ (512 μs)
		01 (fperiph/2)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16 μs)	$fc/2^{13}$ (256 μs)
		10 (fperiph)	—	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{12}$ (128 μs)
1 (fc)	00 (fc)	00 (fperiph/4)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^{11}$ (64 μs)
		01 (fperiph/2)	$fc/2^2$ (0.125 μs)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^{10}$ (32 μs)
		10 (fperiph)	—	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^9$ (16 μs)
	01 (fc/2)	00 (fperiph/4)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^{11}$ (64 μs)
		01 (fperiph/2)	—	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^{10}$ (32 μs)
		10 (fperiph)	—	$fc/2^3$ (0.25 μs)	$fc/25$ (1.0 μs)	$fc/2^9$ (16 μs)
	10 (fc/4)	00 (fperiph/4)	—	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^{11}$ (64 μs)
		01 (fperiph/2)	—	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^{10}$ (32 μs)
		10 (fperiph)	—	—	$fc/2^5$ (1.0 μs)	$fc/2^9$ (16 μs)
	11 (fc/8)	00 (fperiph/4)	—	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^{11}$ (64 μs)
		01 (fperiph/2)	—	—	$fc/2^6$ (2.0 μs)	$fc/2^{10}$ (32 μs)
		10 (fperiph)	—	—	$fc/2^5$ (1.0 μs)	$fc/2^9$ (16 μs)

Note 1: The prescaler's output clock ϕTn must be selected such that the relationship ϕTn < fsys/2 is satisfied (i.e., ϕTn must be slower than fsys/2).
Note 2: Do not change the clock gear value while the timer is operating.
Note 3: The - character meams "Don't use".

(2) Up-counters (UC0 and UC1)

UC0 and UC1 are 8-bit binary counters which count up synchronously with the input clock selected in timer mode register TA01MOD.

The input clock for UC0 is either the external clock entered via the TA0IN pin or one of the three prescaler output clocks, ϕT1, ϕT4 or ϕT16, according to the value set in TA01MOD <TA0CLK1: TA0CLK0>.

The input clock for UC1 varies with the operating mode. In 16-bit timer mode, up-counter UC0's overflow output is the input clock for UC1; in any other mode, the input clock for UC1 is either one of the three prescaler output clocks, ϕT1, ϕT16 or ϕT256, or the comparator output (match detection) from TMRA0, as determined by the value set in TA01MOD<TA1CLK1:TA1CLK0>.

The TA01RUN<TA0RUN> and <TA1RUN> bits set the up-counters to run or stop. When reset, the up-counters are cleared with the timers stopped.

(3) Timer registers (TA0REG and TA1REG)

TA0REG and TA1REG are 8-bit registers used to set interval times. When the value set in one of these timer registers matches the corresponding up-counter value, the comparator's match detection signal becomes active. If the value set is 00H, this signal will become active when the up-counter overflows.

TA0REG is paired with a register buffer to form a dual-buffer structure. The double-buffer is controlled by the setting of TA01RUN<TA0RDE>. The double-buffer is disabled when <TA0RDE> = 0 and enabled when <TA0RDE> = 1.

When the double-buffer is enabled, data transfer from the register buffer to the timer register is initiated by a 2^n-1 overflow in PWM mode or by cycle match detection in PPG mode. The double-buffer cannot be used in timer mode.

When reset, <TA0RDE> is initialized to 0, disabling the double-buffer. To use the double-buffer, write data in the timer register and set <TA0RDE> to 1, then write the following data in the register buffer.

Figure 3.9.2 shows the structure of TA0REG.
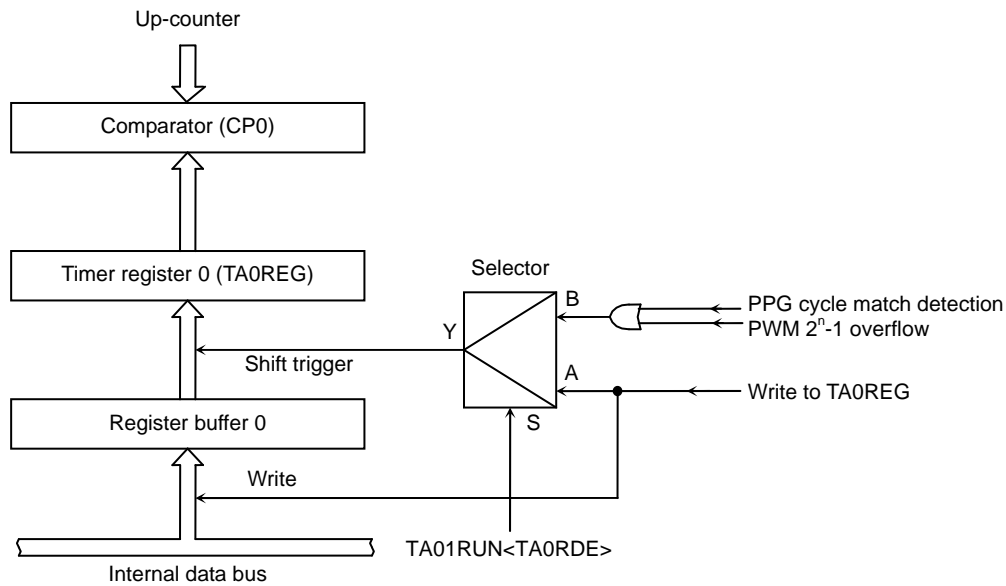
Figure 3.9.2 Structure of Timer Register 0 (TA0REG)

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

The addresses of the individual timer registers are as follows:

TA0REG: 0xFFFF_F102H   TA1REG: 0xFFFF_F103H

TA2REG: 0xFFFF_F10AH   TA3REG: 0xFFFF_F10BH

Each register is a write-only register and cannot be read.

(4) Comparator (CP0)

This circuit compares the up-counter value with the timer register value. When the values match, it clears the up-counter to 0 and at the same time generates an INTTA0 or INTTA1 interrupt. Also, if timer flip-flop inversion is enabled, it inverts the timer flip-flop value.

(5) Timer flip-flop (TA1FF)

The timer flip-flop TA1FF is designed to be inverted by a match detection signal from the comparator. Inversion can be disabled or enabled by setting TA1FFCR<TAFF1IE>.

TA1FF is cleared to 0 by a reset. The TA1FF value can be set to 1 or 0 by writing 01 or 10 to TA1FFCR<TAFF1C1:TAFFCR0>. Also, the TA1FF value can be inverted by writing 00 to these bits (this is known as a soft inversion).

The TA1FF value can be forwarded to the timer flip-flop output pin, TA1OUT (shared with PA6). When timer output is needed, this pin must be set for that purpose by using the port A registers PACR and PAFC.

3.9.3   Register description

TMRA01 run register

TA01RUN
(0xFFFF_F100)

	7	6	5	4	3	2	1	0
Bit symbol	TA0RDE	—	—	—	I2TA01	TA01PRUN	TA1RUN	TA0RUN
Read/Write	R/W	—	—	—	R/W			
After reset	0	—	—	—	0	0	0	0
Function	Double Buffer 0: Disable 1: Enable				IDLE 0: Idle 1: Operate	Timer Run/Stop Control 0: Stop and cleared 1: Count		

This bit controls the TA0REG double-buffer.

0	Disable
1	Enable

I2TA01:   Operation in IDLE mode
TA01PRUN: Operation of the prescaler
TA1RUN: Operation of timer 1
TA0RUN: Operation of timer 0

Note: TA01RUN bits 4, 5 and 6 are undefined when read.

TMRA23 run register

TA23RUN
(0xFFFF_F108)

	7	6	5	4	3	2	1	0
Bit symbol	TA2RDE	—	—	—	I2TA23	TA23PRUN	TA3RUN	TA2RUN
Read/Write	R/W	—	—	—	R/W			
After reset	0	—	—	—	0	0	0	0
Function	Double Buffer 0: Disable 1: Enable				IDLE 0: Idle 1: Operate	Timer Run/Stop Control 0: Stop and cleared 1: Count		

This bit controls the TA2REG double-buffer.

0	Disable
1	Enable

I2TA23:   Operation in IDLE mode
TA23PRUN: Operation of the prescaler
TA3RUN: Operation of timer 3
TA2RUN: Operation of timer 2

Note: TA23RUN bits 4, 5 and 6 are undefined when read.

Figure 3.9.3 TMRA Registers

TMRA45 run register

	7	6	5	4	3	2	1	0
Bit symbol	TA4RDE	—	—	—	I2TA45	TA45PRUN	TA5RUN	TA4RUN
Read/Write	R/W	—	—	—	R/W			
After reset	0	—	—	—	0	0	0	0
Function	Double Buffer 0: Disable 1: Enable				IDLE 0: Idle 1: Operate	Timer Run/Stop Control 0: Stop and cleared 1: Count		

TA45RUN
(0xFFFF_F110`

This bit controls the TA4REG double-buffer.

0	Disable
1	Enable

I2TA45:   Operation in IDLE mode
TA45PRUN: Operation of the prescaler
TA5RUN: Operation of timer 5
TA4RUN: Operation of timer 4

Note: TA45RUN bits 4, 5 and 6 are undefined when read.

TMRA67 run register

	7	6	5	4	3	2	1	0
Bit symbol	TA6RDE	—	—	—	I2TA67	TA67PRUN	TA7RUN	TA6RUN
Read/Write	R/W	—	—	—	R/W			
After reset	0	—	—	—	0	0	0	0
Function	Double Buffer 0: Disable 1: Enable				IDLE 0: Idle 1: Operate	Timer Run/Stop Control 0: Stop and cleared 1: Count		

TA67RUN
(0xFFFF_F118)

This bit controls the TA6REG double-buffer.

0	Disable
1	Enable

I2TA67:   Operation in IDLE mode
TA67PRUN: Operation of the prescaler
TA7RUN: Operation of timer 7
TA6RUN: Operation of timer 6

Note: TA67RUN bits 4, 5 and 6 are undefined when read.

Figure 3.9.4 TMRA Registers

TMRA89 run register

	7	6	5	4	3	2	1	0
Bit symbol	TA8RDE	—	—	—	I2TA89	TA89PRUN	TA9RUN	TA8RUN
Read/Write	R/W	—	—	—	R/W			
After reset	0	—	—	—	0	0	0	0
Function	Double Buffer 0: Disable 1: Enable				IDLE 0: Idle 1: Operate	Timer Run/Stop Control 0: Stop and cleared 1: Count		

TA89RUN
(0xFFFF_F120
`

This bit controls the TA8REG double-buffer.

0	Disable
1	Enable

I2TA89:   Operation in IDLE mode
TA89PRUN: Operation of the prescaler
TA9RUN: Operation of timer 9
TA8RUN: Operation of timer 8

Note: TA89RUN bits 4, 5 and 6 are undefined when read.

TMRAAB run register

	7	6	5	4	3	2	1	0
Bit symbol	TAARDE	—	—	—	I2TAAB	TAABPRUN	TABRUN	TAARUN
Read/Write	R/W	—	—	—	R/W			
After reset	0	—	—	—	0	0	0	0
Function	Double Buffer 0: Disable 1: Enable				IDLE 0: Idle 1: Operate	Timer Run/Stop Control 0: Stop and cleared 1: Count		

TAABRUN
(0xFFFF_F128)

This bit controls the TAAREG double-buffer.

0	Disable
1	Enable

I2TAAB: Operation in IDLE mode
TAABPRUN: Operation of the prescaler
TABRUN: Operation of timer B
TAARUN: Operation of timer A

Note: TAABRUN bits 4, 5 and 6 are undefined when read.

Figure 3.9.5 TMRA Registers

TMRA01 mode register

TA01MOD (0xFFFF_F104)		7	6	5	4	3	2	1	0
	Bit symbol	TA01M1	TA01M0	PWM01	PWM00	TA1CLK1	TA1CLK0	TA0CLK1	TA0CLK0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Operating mode 00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		PWM cycle 00: Reserved 01: 2^6-1 10: 2^7-1 11: 2^8-1		TMRA1 source clock 00: TA0TRG 01: ϕT1 10: ϕT16 11: ϕT256		TMRA0 source clock 00: TA0IN pin input 01: ϕT1 10: ϕT4 11: ϕT16	

→ TMRA0 input clock

00	External input (TA0IN pin input)
01	ϕT1 (prescaler)
10	ϕT4 (prescaler)
11	ϕT16 (prescaler)

→ TMRA1 input clock

	TA01MOD<TA01M1:0>≠01	TA01MOD<TA01M1:0>=01
00	TMRA0 match detection	TMRA0 overflow output
01	ϕT1	
10	ϕT16	16-bit timer mode
11	ϕT256	

→ Selects cycle in 8-bit PWM mode

00	Reserved
01	(2^6-1) × clock source
10	(2^7-1) × clock source
11	(2^8-1) × clock source

→ Selects operating mode for TMRA01

00	8-bit timer × 2
01	16-bit timer
10	8-bit programmable square wave output
11	8-bit PWM (TMRA0), 8-bit timer (TMRA1)

Figure 3.9.6 TMRA Registers

TMRA23 mode register

TA23MOD (0xFFFF_F10C)		7	6	5	4	3	2	1	0
	Bit symbol	TA01M1	TA01M0	PWM01	PWM00	TA1CLK1	TA1CLK0	TA0CLK1	TA0CLK0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Operating mode 00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		PWM cycle 00: Reserved 01: 2^6-1 10: 2^7-1 11: 2^8		TMRA3 source clock 00: TA2TRG 01: ϕT1 10: ϕT16 11: ϕT256		TMRA2 source clock 00: TA2IN pin input 01: ϕT1 10: ϕT4 11: ϕT16	

TMRA2 input clock

00	External input (TA2IN pin input)
01	ϕT1 (prescaler)
10	ϕT4 (prescaler)
11	ϕT16 (prescaler)

TMRA3 input clock

	TA23MOD<TA23M1:0>≠01	TA23MOD<TA23M1:0>=01
00	TMRA2 match detection	TMRA2 overflow output
01	ϕT1	16-bit timer mode
10	ϕT16	
11	ϕT256	

Selects cycle in 8-bit PWM mode

00	Reserved
01	$(2^6$-1$) \times$ clock source
10	$(2^7$-1$) \times$ clock source
11	$(2^8$-1$) \times$ clock source

Selects operating mode for TMRA23

00	8-bit timer $\times$ 2
01	16-bit timer
10	8-bit programmable square wave output
11	8-bit PWM (TMRA2), 8-bit timer (TMRA3)

Figure 3.9.7 TMRA Registers

TMRA45 mode register

TA45MOD (0xFFFF_F114)		7	6	5	4	3	2	1	0
	Bit symbol	TA45M1	TA45M0	PWM41	PWM40	TA5CLK1	TA5CLK0	TA4CLK1	TA4CLK0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Operating mode 00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		PWM cycle 00: Reserved 01: 2^6-1 10: 2^7-1 11: 2^8-1		TMRA5 source clock 00: TA4TRG 01: ϕT1 10: ϕT16 11: ϕT256		TMRA4 source clock 00: TA4IN 01: ϕT1 10: ϕT4 11: ϕT16	

→ TMRA4 input clock

00	TA4IN	
01	ϕT1	(prescaler)
10	ϕT4	(prescaler)
11	ϕT16	(prescaler)

→ TMRA5 input clock

	TA45MOD<TA45M1:0>≠01	TA45MOD<TA45M1:0>=01
00	TMRA4 match detection	TMRA4 overflow output
01	ϕT1	
10	ϕT16	16-bit timer mode
11	ϕT256	

→ Selects cycle in 8-bit PWM mode

00	Reserved
01	(2^6-1) × clock source
10	(2^7-1) × clock source
11	(2^8-1) × clock source

→ Selects operating mode for TMRA45

00	8-bit timer × 2
01	16-bit timer
10	8-bit programmable square wave output
11	8-bit PWM (TMRA4), 8-bit timer (TMRA5)

Figure 3.9.8 TMRA Registers

TMRA67 mode register

TA67MOD (0xFFFF_F11C)		7	6	5	4	3	2	1	0
	Bit symbol	TA67M1	TA67M0	PWM61	PWM60	TA7CLK1	TA7CLK0	TA6CLK1	TA6CLK0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Operating mode 00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		PWM cycle 00: Reserved 01: 2^6-1 10: 2^7-1 11: 2^8-1		TMRA7 source clock 00: TA6TRG 01: ϕT1 10: ϕT16 11: ϕT256		TMRA6 source clock 00: TA6IN 01: ϕT1 10: ϕT4 11: ϕT16	

TMRA6 input clock

00	TA6IN
01	ϕT1 (prescaler)
10	ϕT4 (prescaler)
11	ϕT16 (prescaler)

TMRA7 input clock

	TA67MOD<TA67M1:0>≠01	TA67MOD<TA67M1:0>=01
00	TMRA6 match detection	TMRA6 overflow output
01	ϕT1	16-bit timer mode
10	ϕT16	
11	ϕT256	

Selects cycle in 8-bit PWM mode

00	Reserved
01	$(2^6$-1$) \times$ clock source
10	$(2^7$-1$) \times$ clock source
11	$(2^8$-1$) \times$ clock source

Selects operating mode for TMRA67

00	8-bit timer $\times$ 2
01	16-bit timer
10	8-bit programmable square wave output
11	8-bit PWM (TMRA6), 8-bit timer (TMRA7)

Figure 3.9.9 TMRA Registers

TMRA89 mode register

TA89MOD (0xFFFF_F124)		7	6	5	4	3	2	1	0
	Bit symbol	TA89M1	TA89M0	PWM81	PWM80	TA9CLK1	TA9CLK0	TA8CLK1	TA8CLK0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Operating mode 00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		PWM cycle 00: Reserved 01: 2^6-1 10: 2^7-1 11: 2^8-1		TMRA9 source clock 00: TA8TRG 01: ϕT1 10: ϕT16 11: ϕT256		TMRA8 source clock 00: TA8IN 01: ϕT1 10: ϕT4 11: ϕT16	

→ TMRA8 input clock

00	TA8IN
01	ϕT1 (prescaler)
10	ϕT4 (prescaler)
11	ϕT16 (prescaler)

→ TMRA9 input clock

	TA89MOD<TA89M1:0>≠01	TA89MOD<TA89M1:0>=01
00	TMRA8 match detection	TMRA8 overflow output
01	ϕT1	16-bit timer mode
10	ϕT16	
11	ϕT256	

→ Selects cycle in 8-bit PWM mode

00	Reserved
01	$(2^6$-1) × clock source
10	$(2^7$-1) × clock source
11	$(2^8$-1) × clock source

→ Selects operating mode for TMRA89

00	8-bit timer × 2
01	16-bit timer
10	8-bit programmable square wave output
11	8-bit PWM (TMRA8), 8-bit timer (TMRA9)

Figure 3.9.10 TMRA Registers

TMRAAB mode register

TAABMOD (0xFFFF_F12C)		7	6	5	4	3	2	1	0
	Bit symbol	TAABM1	TAABM0	PWMA1	PWMA0	TABCLK1	TABCLK0	TAACLK1	TAACLK0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Operating mode 00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		PWM cycle 00: Reserved 01: 2^6-1 10: 2^7-1 11: 2^8-1		TMRAB source clock 00: TAATRG 01: ϕT1 10: ϕT16 11: ϕT256		TMRAA source clock 00: TAAIN 01: ϕT1 10: ϕT4 11: ϕT16	

→ TMRAA input clock

00	TAAIN	
01	ϕT1	(prescaler)
10	ϕT4	(prescaler)
11	ϕT16	(prescaler)

→ TMRAB input clock

	TAABMOD<TAABM1:0>≠01	TAABMOD<TAABM1:0>=01
00	TMRAA match detection	TMRAA overflow output
01	ϕT1	
10	ϕT16	16-bit timer mode
11	ϕT256	

→ Selects cycle in 8-bit PWM mode

00	Reserved
01	$(2^6$-1$) \times$ clock source
10	$(2^7$-1$) \times$ clock source
11	$(2^8$-1$) \times$ clock source

→ Selects operating mode for TMRAAB

00	8-bit timer $\times$ 2
01	16-bit timer
10	8-bit programmable square wave output
11	8-bit PWM (TMRAA), 8-bit timer (TMRAB)

Figure 3.9.11 TMRA Registers

TMRA1 flip-flop control register

		7	6	5	4	3	2	1	0
TA1FFCR (0xFFFF_F105)	Bit symbol	—	—	—	—	TAFF1C1	TAFF1C0	TAFF1IE	TAFF1IS
	Read/Write	—	—	—	—	R/W			
	After reset	—	—	—	—	1	1	0	0
	Function					00: Invert TA1FF value (soft inversion). 01: Set TA1FF to 1. 10: Clear TA1FF to 0. 11: Don't care (These bits are always 11 when read.)		Controls TA1FF inversion. 0: Disable inversion. 1: Enable inversion.	Selects TA1FF inversion signal. 0: TMRA0 1: TMRA1

Selects the signal which inverts timer flip-flop 1 (TA1FF).
(Don't care unless in 8-bit timer mode)

0	Inverted by TMRA0
1	Inverted by TMRA1

Note: TA1FFCR bits 4, 5, 6 and 7 are undefined when read.

TMRA3 flip-flop control register

		7	6	5	4	3	2	1	0
TA3FFCR (0xFFFF_F10D)	Bit symbol	—	—	—	—	TAFF3C1	TAFF3C0	TAFF3IE	TAFF3IS
	Read/Write	—	—	—	—	R/W			
	After reset	—	—	—	—	1	1	0	0
	Function					00: Invert TA3FF value (soft inversion). 01: Set TA3FF to 1. 10: Clear TA3FF to 0. 11: Don't care (These bits are always 11 when read.)		Controls TA3FF inversion. 0: Disable inversion. 1: Enable inversion.	Selects TA3FF inversion signal. 0: TMRA2 1: TMRA3

Selects the signal which inverts timer flip-flop 3 (TA3FF).
(Don't care unless in 8-bit timer mode)

0	Inverted by TMRA2
1	Inverted by TMRA3

Note: TA3FFCR bits 4, 5, 6 and 7 are undefined when read.

Figure 3.9.12 TMRA Registers

TMRA5 flip-flop control register

		7	6	5	4	3	2	1	0
TA5FFCR (0xFFFF_F115)	Bit symbol	—	—	—	—	TAFF5C1	TAFF5C0	TAFF5IE	TAFF5IS
	Read/Write	—	—	—	—	R/W			
	After reset	—	—	—	—	1	1	0	0
	Function					00: Invert TA5FF value (soft inversion). 01: Set TA5FF to 1. 10: Clear TA5FF to 0. 11: Don't care (These bits are always 11 when read.)		Controls TA5FF inversion. 0: Disable inversion. 1: Enable inversion.	Selects TA5FF inversion signal. 0: TMRA4 1: TMRA5

Selects the signal which inverts timer flip-flop 5 (TA5FF).
(Don't care unless in 8-bit timer mode)

0	Inverted by TMRA4
1	Inverted by TMRA5

Note: TA5FFCR bits 4, 5, 6 and 7 are undefined when read.

TMRA7 flip-flop control register

		7	6	5	4	3	2	1	0
TA7FFCR (0xFFFF_F11D)	Bit symbol	—	—	—	—	TAFF7C1	TAFF7C0	TAFF7IE	TAFF7IS
	Read/Write	—	—	—	—	R/W			
	After reset	—	—	—	—	1	1	0	0
	Function					00: Invert TA7FF value (soft inversion). 01: Set TA7FF to 1. 10: Clear TA7FF to 0. 11: Don't care (These bits are always 11 when read.)		Controls TA7FF inversion. 0: Disable inversion. 1: Enable inversion.	Selects TA7FF inversion signal. 0: TMRA6 1: TMRA7

Selects the signal which inverts timer flip-flop 7 (TA7FF).
(Don't care unless in 8-bit timer mode)

0	Inverted by TMRA6
1	Inverted by TMRA7

Note: TA7FFCR bits 4, 5, 6 and 7 are undefined when read.

Figure 3.9.13 TMRA Registers

TMRA9 flip-flop control register

	7	6	5	4	3	2	1	0
Bit symbol	—	—	—	—	TAFF9C1	TAFF9C0	TAFF9IE	TAFF9IS
Read/Write	—	—	—	—	R/W			
After reset	—	—	—	—	1	1	0	0
Function					00: Invert TA9FF value (soft inversion). 01: Set TA9FF to 1. 10: Clear TA9FF to 0. 11: Don't care (These bits are always 11 when read.)		Controls TA9FF inversion. 0: Disable inversion. 1: Enable inversion.	Selects TA9FF inversion signal. 0: TMRA8 1: TMRA9

TA9FFCR
(0xFFFF_F125)

Selects the signal which inverts timer flip-flop 9 (TA9FF).
(Don't care unless in 8-bit timer mode)

0	Inverted by TMRA8
1	Inverted by TMRA9

Note: TA9FFCR bits 4, 5, 6 and 7 are undefined when read.

TMRAB flip-flop control register

	7	6	5	4	3	2	1	0
Bit symbol	—	—	—	—	TAFFBC1	TAFFBC0	TAFFBIE	TAFFBIS
Read/Write	—	—	—	—	R/W			
After reset	—	—	—	—	1	1	0	0
Function					00: Invert TABFF value (soft inversion). 01: Set TABFF to 1. 10: Clear TABFF to 0. 11: Don't care (These bits are always 11 when read.)		Controls TABFF inversion. 0: Disable inversion. 1: Enable inversion.	Selects TABFF inversion signal. 0: TMRAA 1: TMRAB

TABFFCR
(0xFFFF_F12D)

Selects the signal which inverts timer flip-flop B (TABFF).
(Don't care unless in 8-bit timer mode)

0	Inverted by TMRAA
1	Inverted by TMRAB

Note: TABFFCR bits 4, 5, 6 and 7 are undefined when read.

Figure 3.9.14 TMRA Registers

3.9.4   Functional description for each mode

(1) 8-bit timer mode

TMRA0 and TMRA1 can be used as 8-bit interval timers independently of each other. You must stop TMRA0 and TMRA1 before attempting to set their functions or count data.

a. Generating interrupts periodically

The following description uses TMRA1 as an e×ample. To generate a TRAM1 interrupt, INTTA1, at certain intervals, first stop timer 1 and set the operating mode, input clock and cycle in the TA01MOD and TA1REG registers. Ne×t, enable the INTTA1 interrupt and start timer 1.

E×ample: To generate INTTA1 interrupts every 20 μs with fc = 32 MHz, set the registers in the following sequence:

$\ast$Clock conditions ⎰ System clock: High-speed (fc)
                      ⎱ Prescaler clock: fperiph/4 (fperiph = fsys)

```
          MSB           LSB
          7  6  5  4  3  2  1  0
TA01RUN  ←  −  −  X  X  −  −  0  −      Stop TMRA1 and clear it to 0.
TA01MOD  ←  0  0  X  X  1  0  X  X      Select 8-bit timer mode and set input clock to φT1
                                       (0.25 μs resolution, fc = 32 MHz).
TA1REG   ←  0  1  0  1  0  0  0  0      Write 20 μs/φT1 = 80 (50H) to TA1REG.
IMC5LH   ←  X  X  1  1  0  1  0  1      Enable INTTA1 and set interrupt level = 5 and rising edge
                                       detection.
TA01RUN  ←  −  X  X  X  −  1  1  −      Start TMRA1.
```

Note: X = Don't care; "—" = No change

For a description of input clock selection, refer to Table 3.9.3.

Note:	The input clocks for TMRA0 and TMRA1 differ as shown below.
	TMRA0: TA0IN pin input, ϕT1, ϕT4 or ϕT16
	TMRA1: TMRA0 match detection signal, ϕT1, ϕT16 or ϕT256

b. Outputting a 50% duty cycle square wave

Invert the value of timer flip-flop TA1FF at certain intervals and forward the inverted value to the timer flip-flop output pin, TA1OUT.

E×ample: To output a 1.5-µs cycle square wave with fc = 32 MHz on the TA1OUT pin, set each register in the following sequence. In this e×ample TMRA1 is used to show how to set the registers, although either TMRA0 or TMRA1 may be used.

∗Clock conditions ⎧ System clock: High-speed (fc)
                  ⎨ High-speed clock gear: × 1 (fc)
                  ⎩ Prescaler clock: fperiph/4 (fperiph = fsys)

```
                7  6  5  4  3  2  1  0
TA01RUN   ←  −  X  X  X  −  −  0  −      Stop TMRA1 and clear it to 0.
TA01MOD   ←  0  0  X  X  0  1  −  −      Select 8-bit timer mode and set input clock to φT1 (0.25 µs
                                        resolution, fc = 32 MHz).
TA1REG    ←  0  0  0  0  0  0  1  1      Write (1.5 µs/φT1)/2 = 3 to TA1REG.
TA1FFCR   ←  X  X  X  X  1  0  1  1      Clear TA1FF to 0 and set it to be cleared by match
                                        detection signal from TMRA1.
P7CR      ←  −  −  −  −  −  −  1  −   ⎫  Set PA6 to TA1OUT output pin.
P7FC      ←  −  −  −  −  −  −  1  −   ⎬
TA01RUN   ←  −  X  X  X  −  1  1  −      Start TMRA1.
```

Note: X = Don't care; "—" = No change
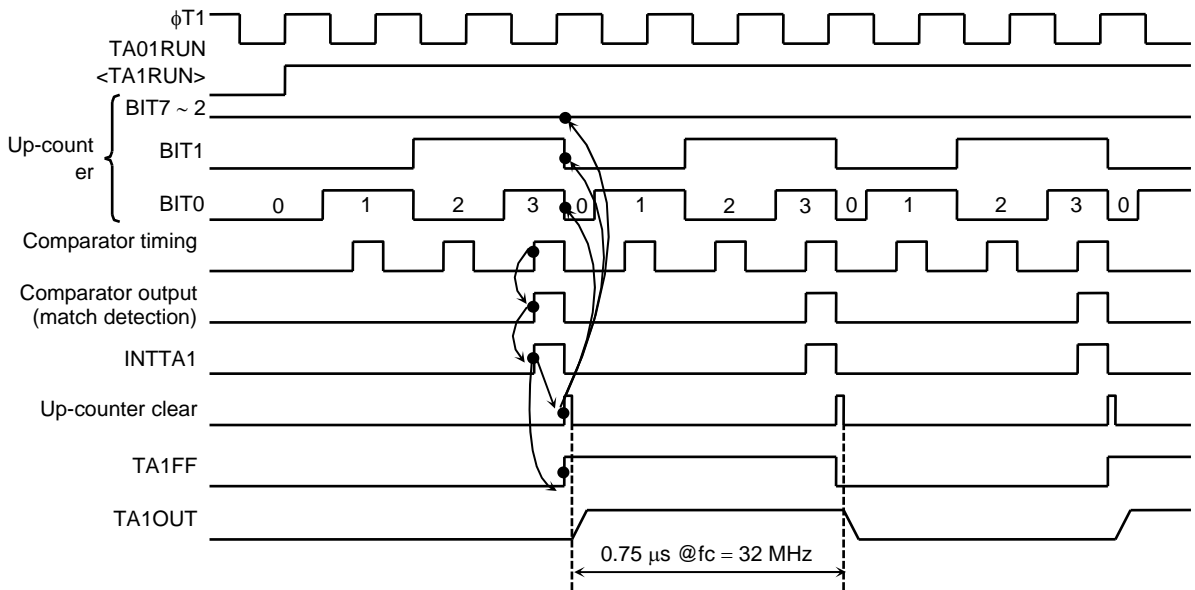
Figure 3.9.15 Square Wave Output Timing (50% Duty Cycle)

c. Using a match signal from TMRA0 to make TMRA1 count

Select 8-bit timer mode and set the TMRA1 input clock to the TMRA0 comparator output.
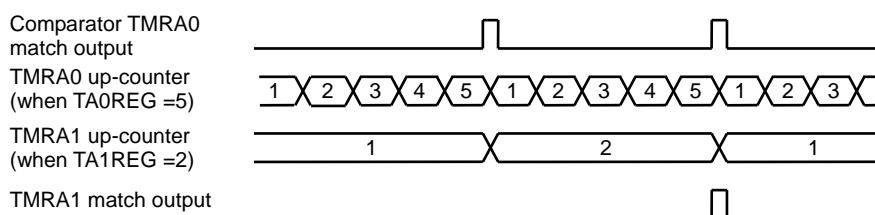
Figure 3.9.16 TMRA1 Counting Based on TMRA0

TMP1942CY/CZ-182

(2) 16-bit timer mode

TMRA0 and TMRA1 can be used together as a 16-bit interval timer. To select 16-bit timer mode, set TA01MOD<TA01M1:TA01M0> to 01.

In 16-bit timer mode, the TMRA1 input clock is derived from the TMRA0 overflow output regardless of the TA01MOD<TA1CLK1:TA1CLK0> settings. For a description of TMRA1 input clock selection, refer to Table 3.9.3.

Set the timer interrupt cycle in the timer registers TA0REG and TA1REG by writing the eight low-order bits to TA0REG and the eight high-order bits to TA1REG. Always set TA0REG first. This is because compare operation is temporarily disabled when data is written to TA0REG; it is re-enabled when data is subsequently written to TA1REG.

E×ample: To generate INTTA1 interrupts every 0.2 second with fc = 32 MHz, set the values shown below in the timer registers TA0REG and TA1REG.

*Clock conditions   System clock: High-speed (fc)
                     High-speed clock gear: × 1 (fc)
                     Prescaler clock: fperiph/4 (fperiph = fsys)

With ϕT16 (= 4.0 μs at 32 MHz) used as the input clock,

    0.2 s/4.0 μs = 50000 = C350H

Therefore, TA1REG must be set to 03H and TA0REG to 50H.

A TMRA0 comparator output is generated each time the up-counter UC0 and timer register TA0REG match (the up-counter UC0 is not cleared). In this case INTTA0 is not generated.

The TMRA1 comparator outputs a match detection signal at each comparator timing when the up-counter UC1 and timer register TA1REG match. If the comparators in both TMRA0 and TMRA1 output a match detection signal at the same time, the up-counters UC0 and UC1 will be cleared to 0 and an INTTA1 interrupt is generated. Also, if inversion is enabled, the value of timer flip-flop TA1FF will be inverted.

E×ample: TA1REG = 04H and TA0REG = 80H

Up-counter values
(UC1 and UC0)   0000H   0080H   0180H   0280H   0380H   0480H
Match detection signal from
TMRA0 comparator

Interrupt INTTA1

Timer output TA1OUT   Invert
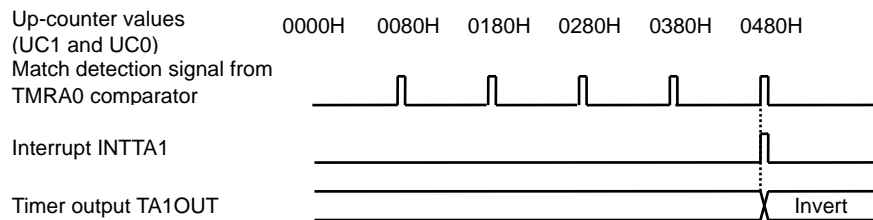
Figure 3.9.17 Timer Output in 16-Bit Timer Mode

(3) 8-bit PPG (programmable square wave) output mode

A square wave of any frequency with any duty cycle can be output using TMRA0. Either Low-active or High-active output pulses can be selected.

TMRA1 cannot be used in this mode. The square wave is forwarded to TA1OUT (shared with PA6).
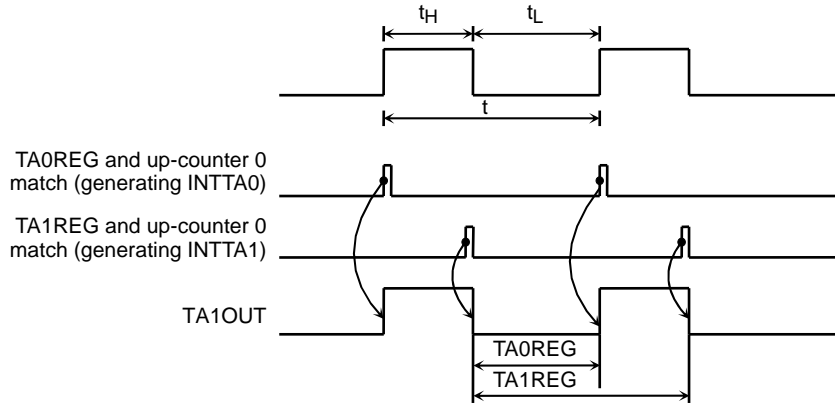
Figure 3.9.18 8-Bit PPG Output Waveform

This mode is used to output a programmable square wave by inverting the timer output every time the 8-bit up-counter UC0 matches the timer registers TA0REG and TA1REG.

However, the condition (TA0REG set value) < (TA1REG set value) must be satisfied. Although the up-counter UC1 of TMRA1 cannot be used in this mode, TA01RUN<TA1RUN> must be set to 1 to enable TMRA1 counting.

Figure 3.9.19 shows a block diagram of 8-bit PPG output mode.
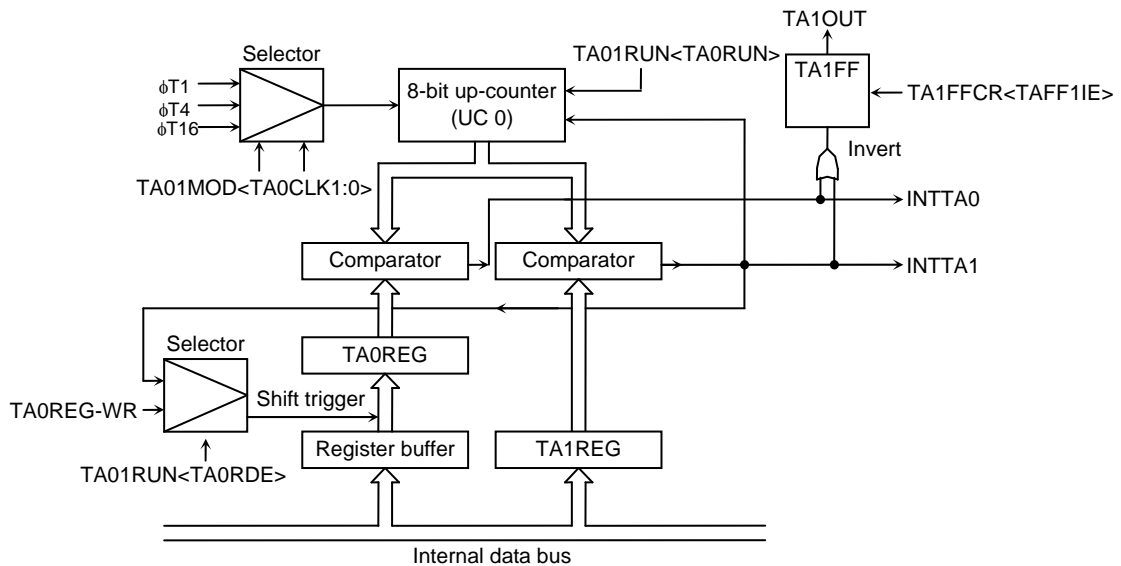
Figure 3.9.19 8-Bit PPG Output Mode Block Diagram

If TA0REG has its double-buffer enabled in this mode, the value in the register buffer is shifted into TA0REG when TA1REG and UC0 match.

If it is necessary to change the duty cycle, using the double-buffer facilitates satisfying the requirements for small duty cycle waveforms.
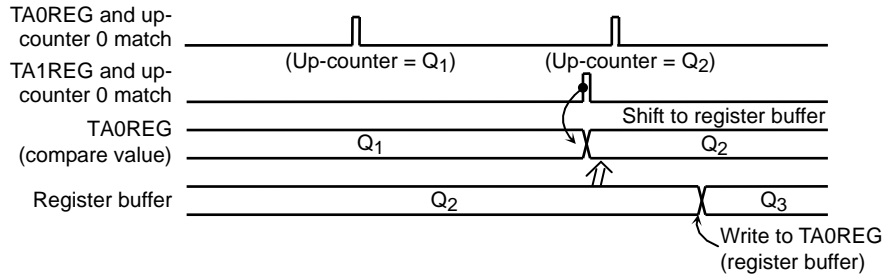
Figure 3.9.20 Register Buffer Operation

E×ample: To output a 1/4 duty cycle 50-kHz pulse (fc = 32 MHz)
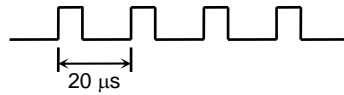
20 μs

*Clock conditions

System clock:	High-speed (fc)
High-speed clock gear:	× 1 (fc)
Prescaler clock:	fperiph/4 (fperiph = fsys)

Calculate the values to be set in the timer registers as follows:

To obtain a frequency of 50 kHz, generate a waveform with a period t = 1/50 kHz = 20 μs. When $\phi T1 = 0.25$ μs (at fc = 32 MHz),

20 μs/0.25 μs = 80

Therefore, TA1REG must be set to 80 (= 50H).

Ne×t, to obtain a 1/4 duty cycle, using the formula $t \times 1/4 = 20$ μs × 1/4 = 5 μs,

5 μs/0.25 μs = 20

Therefore, TA0REG must be set to 20 (= 14H).

```
            7  6  5  4  3  2  1  0
TA01RUN  ←  0  X  X  X  −  0  0  0    Stop TMRA0 and TMRA1 and clear them to 0.
TA01MOD  ←  1  0  X  X  X  X  0  1    Select 8-bit PPG mode and set input clock to φT1.
TA0REG   ←  0  0  0  1  0  1  0  0    Write 14H.
TA1REG   ←  0  1  0  1  0  0  0  0    Write 50H.
TA1FFCR  ←  X  X  X  X  0  1  1  X    Set TA1FF and enable inversion.
                                 If these bits are set to 10, Low-active output waveform will
                                 be obtained.
P7CR     ←  −  −  −  −  −  −  1  −  ⎫
P7FC     ←  −  −  −  −  −  −  1  −  ⎬ Set PA6 to TA1OUT output pin.
TA01RUN  ←  1  X  X  X  −  1  1  1  ⎭ Start TMRA0 and TMRA1.
```

Note: X = Don't care; "—" = No change

(4) 8-bit PWM output mode

This mode, only available for TMRA0, can output PWM pulses with up to 8-bit resolution. PWM output is forwarded to the TA1OUT pin (shared with PA6).

In this mode TMRA1 can be used as an 8-bit timer.

Timer output is inverted when the up-counter UC0 and the value set in the timer register TA0REG match. It is also inverted when a 2^n-1 counter overflow occurs (n = 6, 7 or 8 as specified in TA01MOD<PWM01:PWM00>). The up-counter UC0 is cleared to 0 upon the occurrence of a 2^n-1 counter overflow.

Before PWM mode can be used, the following conditions must be satisfied:

(TA0REG set value) < (2^n-1 counter overflow set value)

(TA0REG set value) $\neq$ 0
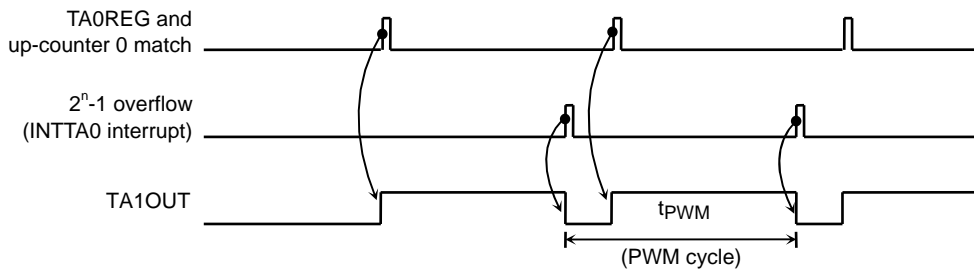
Figure 3.9.21 8-Bit PWM Output Waveform

Figure 3.9.22 shows a block diagram of 8-bit PWM output mode.
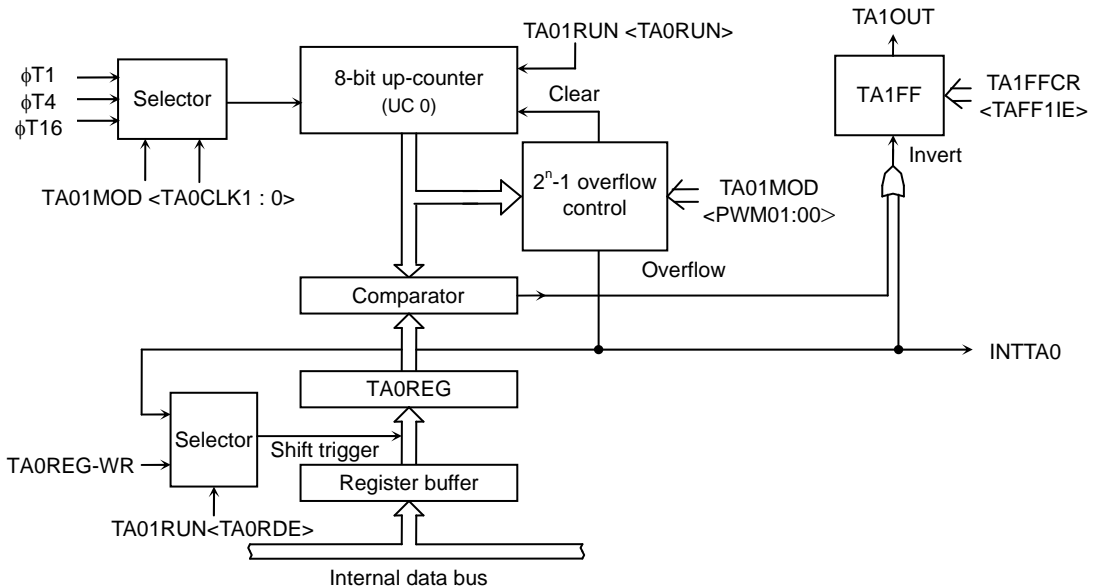
Figure 3.9.22 8-Bit PWM Output Mode Block Diagram

If TA0REG has its double-buffer enabled in this mode, the value in the register buffer is shifted into TA0REG upon the detection of a 2^n-1 overflow. Using the double-buffer facilitates satisfying the requirements for small duty cycle waveforms.
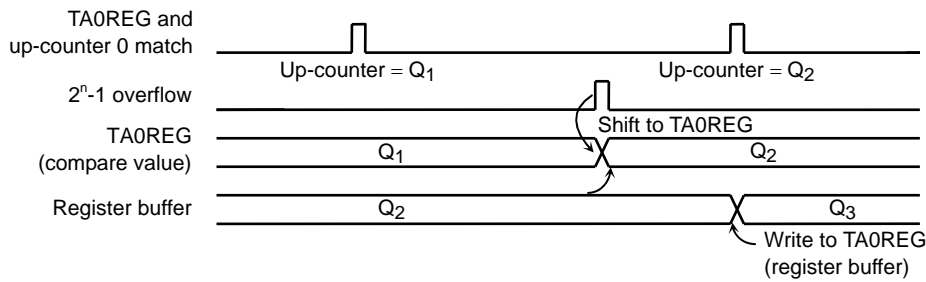
Figure 3.9.23 Register Buffer Operation

E×ample: To output the following PWM waveform on the TA1OUT pin using TMRA0 when fc = 32 MHz
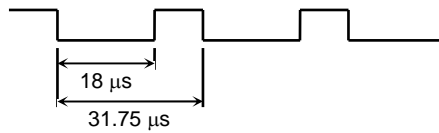
*Clock conditions
System clock: High-speed (fc)
High-speed clock gear: × 1 (fc)
Prescaler clock: fperiph/4 (fperiph = fsys)

To achieve a PWM cycle of 31.75 μs with $\phi T1 = 0.25$ μs (at fc = 32 MHz), the following equation must be satisfied:

31.75 μs/0.25 μs = 127 = 2^n-1

Therefore, n must be set to 7.

Since the Low-level period is 18 μs and $\phi T1 = 0.25$ μs,

18 μs/0.25 μs = 72 = 48H

Therefore, TA0REG must be set to 48H.

```
          MSB              LSB
          7  6  5  4  3  2  1  0
┌ TA01RUN  ← −  X  X  X  −  −  −  0    Stops TMRA0 and clear it to 0.
│ TA01MOD  ← 1  1  1  0  −  −  0  1    Select 8-bit PWM mode (cycle = 2⁷-1) and set input clock to
│                                      φT1.
│ TA0REG   ← 0  1  0  0  1  0  0  0    Write 48H.
│ TA1FFCR  ← X  X  X  X  1  0  1  X    Clear TA1FF and enable inversion.
│
│ P7CR     ← −  −  −  −  −  −  1  −  ⎫
│ P7FC     ← −  −  −  −  −  −  1  −  ⎬ Set PA6 to TA1OUT output pin.
└ TA01RUN  ← 1  X  X  X  −  1  −  1    Start TMRA0.
```

The code block above: TA01MOD cycle = 2^7-1.

Note: X = Don't care; "—" = No change

Table 3.9.4 PWM Periods

@fc = 32 MHz

Peripheral Clock Selection <FPSEL>	Clock Gear Value <GEAR1:0>	Selected Prescaler Clock <PRCK1:0>	PWM Period								
			$2^6 - 1$			$2^7 - 1$			$2^8 - 1$		
			φT1	φT4	φT16	φT1	φT4	φT16	φT1	φT4	φT16
0 (fgear)	00 (fc)	00 (fperiph/4)	15.8 μs	63 μs	252 μs	31.8 μs	127 μs	508 μs	63.8 μs	255 μs	1020 μs
		01 (fperiph/2)	7.9 μs	31.5 μs	126 μs	15.9 μs	63.5 μs	254 μs	31.9 μs	127.5 μs	510 μs
		10 (fperiph)	—	15.8 μs	63 μs	—	31.8 μs	127 μs	—	63.8 μs	255 μs
	01 (fc/2)	00 (fperiph/4)	31.5 μs	126 μs	504 μs	63.5 μs	254 μs	1016 μs	127.5 μs	510 μs	2040 μs
		01 (fperiph/2)	15.8 μs	63 μs	252 μs	31.8 μs	127 μs	508 μs	63.8 μs	255 μs	1020 μs
		10 (fperiph)	—	31.5 μs	126 μs	—	63.5 μs	254 μs	—	127.5 μs	510 μs
	10 (fc/4)	00 (fperiph/4)	63 μs	252 μs	1008 μs	127 μs	508 μs	2032 μs	255 μs	1020 μs	4080 μs
		01 (fperiph/2)	31.5 μs	126 μs	504 μs	63.5 μs	254 μs	1016 μs	127.5 μs	510 μs	2040 μs
		10 (fperiph)	—	63 μs	252 μs	—	127 μs	508 μs	—	255 μs	1020 μs
	11 (fc/8)	00 (fperiph/4)	126 μs	504 μs	2016 μs	254 μs	1016 μs	4064 μs	510 μs	2040 μs	8160 μs
		01 (fperiph/2)	63 μs	252 μs	1008 μs	127 μs	508 μs	2032 μs	255 μs	1020 μs	4080 μs
		10 (fperiph)	—	126 μs	504 μs	—	254 μs	1016 μs	—	510 μs	2040 μs
1 (fc)	00 (fc)	00 (fperiph/4)	15.8 μs	63 μs	252 μs	31.8 μs	127 μs	508 μs	63.8 μs	255 μs	1020 μs
		01 (fperiph/2)	7.9 μs	31.5 μs	126 μs	15.9 μs	63.5 μs	254 μs	31.9 μs	127.5 μs	510 μs
		10 (fperiph)	—	15.8 μs	63 μs	—	31.8 μs	127 μs	—	63.8 μs	255 μs
	01 (fc/2)	00 (fperiph/4)	15.8 μs	63 μs	252 μs	31.8 μs	127 μs	508 μs	63.8 μs	255 μs	1020 μs
		01 (fperiph/2)	—	31.5 μs	126 μs	—	63.5 μs	254 μs	—	127.5 μs	510 μs
		10 (fperiph)	—	15.8 μs	63 μs	—	31.8 μs	127 μs	—	63.8 μs	255 μs
	10 (fc/4)	00 (fperiph/4)	—	63 μs	252 μs	—	127 μs	508 μs	—	255 μs	1020 μs
		01 (fperiph/2)	—	31.5 μs	126 μs	—	63.5 μs	254 μs	—	127.5 μs	510 μs
		10 (fperiph)	—	—	63 μs	—	—	127 μs	—	—	255 μs
	11 (fc/8)	00 (fperiph/4)	—	63 μs	252 μs	—	127 μs	508 μs	—	255 μs	1020 μs
		01 (fperiph/2)	—	—	126 μs	—	—	254 μs	—	—	510 μs
		10 (fperiph)	—	—	63 μs	—	—	127 μs	—	—	255 μs

Note 1: The prescaler's output clock φTn must be selected such that the relationship φTn < fsys/2 is satisfied (i.e., φTn must be slower than fsys/2).

Note 2: Do not change the clock gear value while the timer is running.

Note 3: The — character means "Don't use".

(5) Summary of operating mode settings

Table 3.9.5 summarizes the settings for TMRA01 for each mode.

Table 3.9.5 Register Settings for Each Timer Mode

Register Name	TA01MOD				TA1FFCR
Register Field Name	<TA01M1 : 0>	<PWM01 : 00>	<TA1CLK1 : 0>	<TA0CLK1 : 0>	TAFF1IS
Function	Timer mode	PWM period	High-order timer input clock	Low-order timer input clock	Timer F/F inverting signal selection
8-bit timer × 2 channels	00	—	Low-order timer match, ϕT1, ϕT16, ϕT256 (00, 01, 10, 11)	External, ϕT1, ϕT4, ϕT16 (00, 01, 10, 11)	0: Low-order timer output 1: High-order timer output
16-bit timer mode	01	—	—	External, ϕT1, ϕT4, ϕT16 (00, 01, 10, 11)	—
8-bit PPG × 1 channel	10	—	—	External, ϕT1, ϕT4, ϕT16 (00, 01, 10, 11)	—
8-bit PWM × 1 channel 8-bit timer × 1 channel (Note)	11	$2^6 - 1$, $2^7 - 1$, $2^8 - 1$ (01, 10, 11)	ϕT1, ϕ16, ϕT256 (01, 10, 11)	External, ϕT1, ϕT4, ϕT16 (00, 01, 10, 11)	PWM output

"—" = Don't care

Note: In 8-bit PWM generation mode, the UC1 can be used as an 8-bit timer. However, the match-detect output from the UC0 can not be used as a clock source for the UC1, and the timer output is not available for the UC1.

3.10   16-Bit Timers/Event Counters (TMRBn)

The TMP1942 contains fourteen multi-function 16-bit timer/event counter channels (TMRB0-TMRBD). TMRBn can operate in the following four modes:

- 16-bit interval timer mode

- 16-bit event counter mode

- 16-bit programmable square wave output (PPG) mode

- 2-phase pulse input counter mode (only for TMRB2 and TMRB3)

In addition, when used in combination with the capture function, TMRBn can be run in the following modes:

- Frequency measurement mode

- Pulse width measurement mode

- Time difference measurement mode

Each channel consists primarily of a 16-bit up-counter, two 16-bit timer registers (one with a double-buffer structure), two 16-bit capture registers, two comparators, capture input controller, and a timer flip-flop with accompanying control circuit. Timer operating modes and flip-flops are controlled by eleven registers.

All channels TMRB0 to TMRBD operate independently of each other. Because each channel functions the same way except for the 2-phase pulse counter function and a few other differences as shown in Tables 3.10.1 to 3.10.2, operation of the TMRB0 only is described here, with an explanation of the 2-phase pulse counter function for TMRB2 and TMRB3.

Table 3.10.1 Specification Differences Among the TMRB Channels

Specification		Channel			
		TMRB0	TMRB1	TMRB2	TMRB3
External pins	External clock/ capture trigger input pins	TB0IN0 (Shared with PA0) TB0IN1 (Shared with PA1)	TB1IN0 (Shared with PA3) TB1IN1 (Shared with PA4)	TB2IN0 (Shared with PB0) TB2IN1 (Shared with PB1)	TB3IN0 (Shared with PB3) TB3IN1 (Shared with PB4)
	Capture trigger timer	TA3OUT	TA3OUT	TA3OUT	TA3OUT
	Timer flip-flop output pin	TB0OUT (Shared with PA2)	TB1OUT (Shared with PA5)	TB2OUT (Shared with PB2)	TB3OUT (Shared with PB5)
Register name (address)	Timer run register	TB0RUN (0xFFFF_F140)	TB1RUN (0xFFFF_F150)	TB2RUN (0xFFFF_F160)	TB3RUN (0xFFFF_F170)
	Timer mode register	TB0MOD (0xFFFF_F142)	TB1MOD (0xFFFF_F152)	TB2MOD (0xFFFF_F162)	TB3MOD (0xFFFF_F172)
	Timer flip-flop control register	TB0FFCR (0xFFFF_F143)	TB1FFCR (0xFFFF_F153)	TB2FFCR(0xFFFF_F163)	TB3FFCR (0xFFFF_F173)
	Timer registers	TB0RG0L (0xFFFF_F148) TB0RG0H (0xFFFF_F149) TB0RG1L (0xFFFF_F14A) TB0RG1H (0xFFFF_F14B)	TB1RG0L (0xFFFF_F158) TB1RG0H (0xFFFF_F159) TB1RG1L (0xFFFF_F15A) TB1RG1H (0xFFFF_F15B)	TB2RG0L (0xFFFF_F168) TB2RG0H (0xFFFF_F169) TB2RG1L (0xFFFF_F16A) TB2RG1H (0xFFFF_F16B)	TB3RG0L (0xFFFF_F178) TB3RG0H (0xFFFF_F179) TB3RG1L (0xFFFF_F17A) TB3RG1H (0xFFFF_F17B)
	Capture registers	TB0CP0L (0xFFFF_F14C) TB0CP0H (0xFFFF_F14D) TB0CP1L (0xFFFF_F14E) TB0CP1H (0xFFFF_F14F)	TB1CP0L (0xFFFF_F15C) TB1CP0H (0xFFFF_F15D) TB1CP1L (0xFFFF_F15E) TB1CP1H (0xFFFF_F15F)	TB2CP0L (0xFFFF_F16C) TB2CP0H (0xFFFF_F16D) TB2CP1L (0xFFFF_F16E) TB2CP1H (0xFFFF_F16F)	TB3CP0L (0xFFFF_F17C) TB3CP0H (0xFFFF_F17D) TB3CPIL (0xFFFF_F17E) TB3CPIH (0xFFFF_F17F)

Table 3.10.2 Specification Differences Among the TMRB Channels

Specification		Channel			
		TMRB4	TMRB5	TMRB6	TMRB7
External pins	External clock/ capture trigger input pins	TB4IN0 (Shared with PB2) TB4IN1 (Shared with PB5)	—	—	TB4IN0 (Shared with P95) TB4IN1 (Shared with P96)
	Capture trigger timer	TA3OUT	TA3OUT	TA3OUT	TA3OUT
	Timer flip-flop output pin	TB4OUT (Shared with P92)	TB5OUT (Shared with P93)	TB6OUT (Shared with P94)	TB7OUT (Shared with P97)
Register name (address)	Timer run register	TB4RUN (0xFFFF_F180)	TB5RUN (0xFFFF_F190)	TB6RUN (0xFFFF_F1A0)	TB7RUN (0xFFFF_F1B0)
	Timer mode register	TB4MOD (0xFFFF_F182)	TB5MOD (0xFFFF_F192)	TB6MOD (0xFFFF_F1A2)	TB7MOD (0xFFFF_F1B2)
	Timer flip-flop control register	TB4FFCR (0xFFFF_F183)	TB5FFCR (0xFFFF_F193)	TB6FFCR(0xFFFF_F1A3)	TB7FFCR (0xFFFF_F1B3)
	Timer registers	TB4RG0L (0xFFFF_F188) TB4RG0H (0xFFFF_F189) TB4RG1L (0xFFFF_F18A) TB4RG1H (0xFFFF_F18B)	TB5RG0L (0xFFFF_F198) TB5RG0H (0xFFFF_F199) TB5RG1L (0xFFFF_F19A) TB5RG1H (0xFFFF_F19B)	TB6RG0L (0xFFFF_F1A8) TB6RG0H (0xFFFF_F1A9) TB6RG1L (0xFFFF_F1AA) TB6RG1H (0xFFFF_F1AB)	TB7RG0L (0xFFFF_F1B8) TB7RG0H (0xFFFF_F1B9) TB7RG1L (0xFFFF_F1BA) TB7RG1H (0xFFFF_F1BB)
	Capture registers	TB4CP0L (0xFFFF_F18C) TB4CP0H (0xFFFF_F18D) TB4CP1L (0xFFFF_F18E) TB4CP1H (0xFFFF_F18F)	TB5CP0L (0xFFFF_F19C) TB5CP0H (0xFFFF_F19D) TB5CP1L (0xFFFF_F19E) TB5CP1H (0xFFFF_F19F)	TB6CP0L(0xFFFF_F1AC) TB6CP0H(0xFFFF_F1AD) TB6CP1L (0xFFFF_F1AE) TB6CP1H (0xFFFF_F1AF)	TB7CP0L (0xFFFF_F1BC) TB7CP0H(0xFFFF_F1BD) TB7CPIL (0xFFFF_F1BE) TB7CPIH (0xFFFF_F1BF)

Table 3.10.3 Specification Differences Among the TMRB Channels

Specification		Channel			
		TMRB8	TMRB9	TMRBA	TMRBB
External pins	External clock/ capture trigger input pins	TB8IN0 (Shared with PC6) TB8IN1 (Shared with PC7)	TB9IN0 (Shared with PD0) TB8IN1 (Shared with PD1)	TBAIN0 (Shared with PD5) TBAIN1 (Shared with PD6)	—
	Capture trigger timer	TA5OUT	TA5OUT	TA5OUT	TA5OUT
	Timer flip-flop output pin	—	—	—	—
Register name (address)	Timer run register	TB8RUN (0xFFFF_F1C0)	TB9RUN (0xFFFF_F1D0)	TBARUN (0xFFFF_F1E0)	TBBRUN (0xFFFF_F1F0)
	Timer mode register	TB8MOD (0xFFFF_F1C2)	TB9MOD (0xFFFF_F1D2)	TBAMOD (0xFFFF_F1E2)	TBBMOD (0xFFFF_F1F2)
	Timer flip-flop control register	—	—	—	—
	Timer registers	TB8RG0L (0xFFFF_F1C8) TB8RG0H (0xFFFF_F1C9) TB8RG1L (0xFFFF_F1CA) TB8RG1H (0xFFFF_F1CB)	TB9RG0L (0xFFFF_F1D8) TB9RG0H (0xFFFF_F1D9) TB9RG1L (0xFFFF_F1DA) TB9RG1H (0xFFFF_F1DB)	TBARG0L (0xFFFF_F1E8) TBARG0H (0xFFFF_F1E9) TBARG1L (0xFFFF_F1EA) TBARG1H (0xFFFF_F1EB)	TBBRG0L (0xFFFF_F1F8) TBBRG0H (0xFFFF_F1F9) TBBRG1L (0xFFFF_F1FA) TBBRG1H (0xFFFF_F1FB)
	Capture registers	TB8CP0L (0xFFFF_F1CC) TB8CP0H (0xFFFF_F1CD) TB8CP1L (0xFFFF_F1CE) TB8CP1H (0xFFFF_F1CF)	TB9CP0L (0xFFFF_F1DC) TB9CP0H (0xFFFF_F1DD) TB9CP1L (0xFFFF_F1DE) TB9CP1H (0xFFFF_F1DF)	TBACP0L (0xFFFF_F1EC) TBACP0H (0xFFFF_F1ED) TBACP1L (0xFFFF_F1EE) TBACP1H (0xFFFF_F1EF)	TBBCP0L (0xFFFF_F1FC) TBBCP0H (0xFFFF_F1FD) TBBCPIL (0xFFFF_F1FE) TBBCPIH (0xFFFF_F1FF)

Table 3.10.4 Specification Differences Among the TMRB Channels

Specification		Channel	
		TMRBC	TMRBD
External pins	External clock/ capture trigger input pins		
	Capture trigger timer	TA5OUT	TA5OUT
	Timer flip-flop output pin		
Register name (address)	Timer run register	TBCRUN (0xFFFF_F200)	TBDRUN (0xFFFF_F210)
	Timer mode register	TBCMOD (0xFFFF_F202)	TBDMOD (0xFFFF_F212)
	Timer flip-flop control register		
	Timer registers	TBCRG0L (0xFFFF_F208) TBCRG0H (0xFFFF_F209) TBCRG1L (0xFFFF_F20A) TBCRG1H (0xFFFF_F20B)	TBDRG0L (0xFFFF_F218) TBDRG0H (0xFFFF_F219) TBDRG1L (0xFFFF_F21A) TBDRG1H (0xFFFF_F21B)
	Capture registers	TBCCP0L (0xFFFF_F20C) TBCCP0H (0xFFFF_F20D) TBCCP1L (0xFFFF_F20E) TBCCP1H (0xFFFF_F20F)	TBDCP0L (0xFFFF_F21C) TBDCP0H (0xFFFF_F21D) TBDCP1L (0xFFFF_F21E) TBDCP1H (0xFFFF_F21F)

3.10.1   Block Diagrams
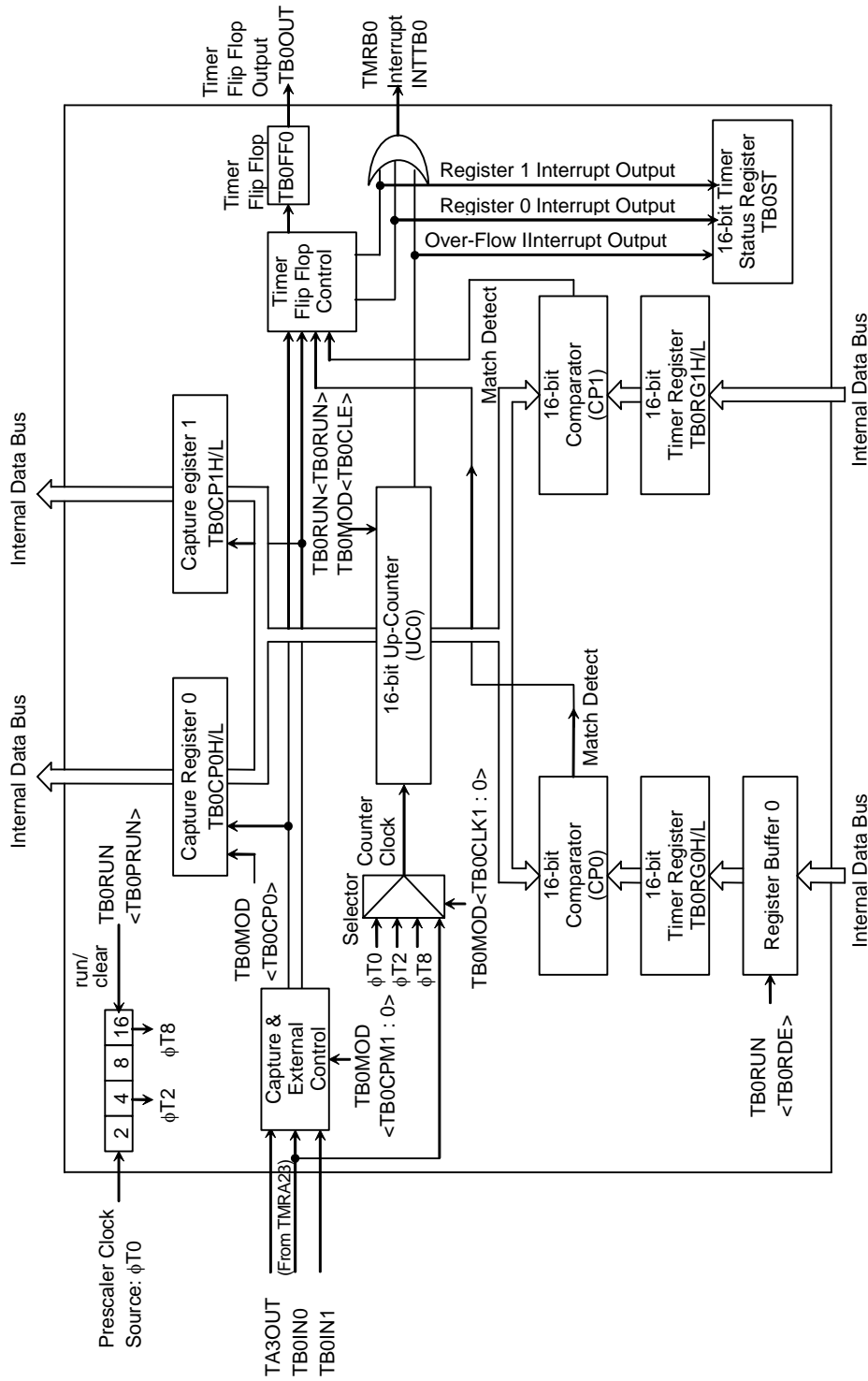
Figure 3.10.1 TMRB0/1 and TMRB to TMRBD Block Diagram
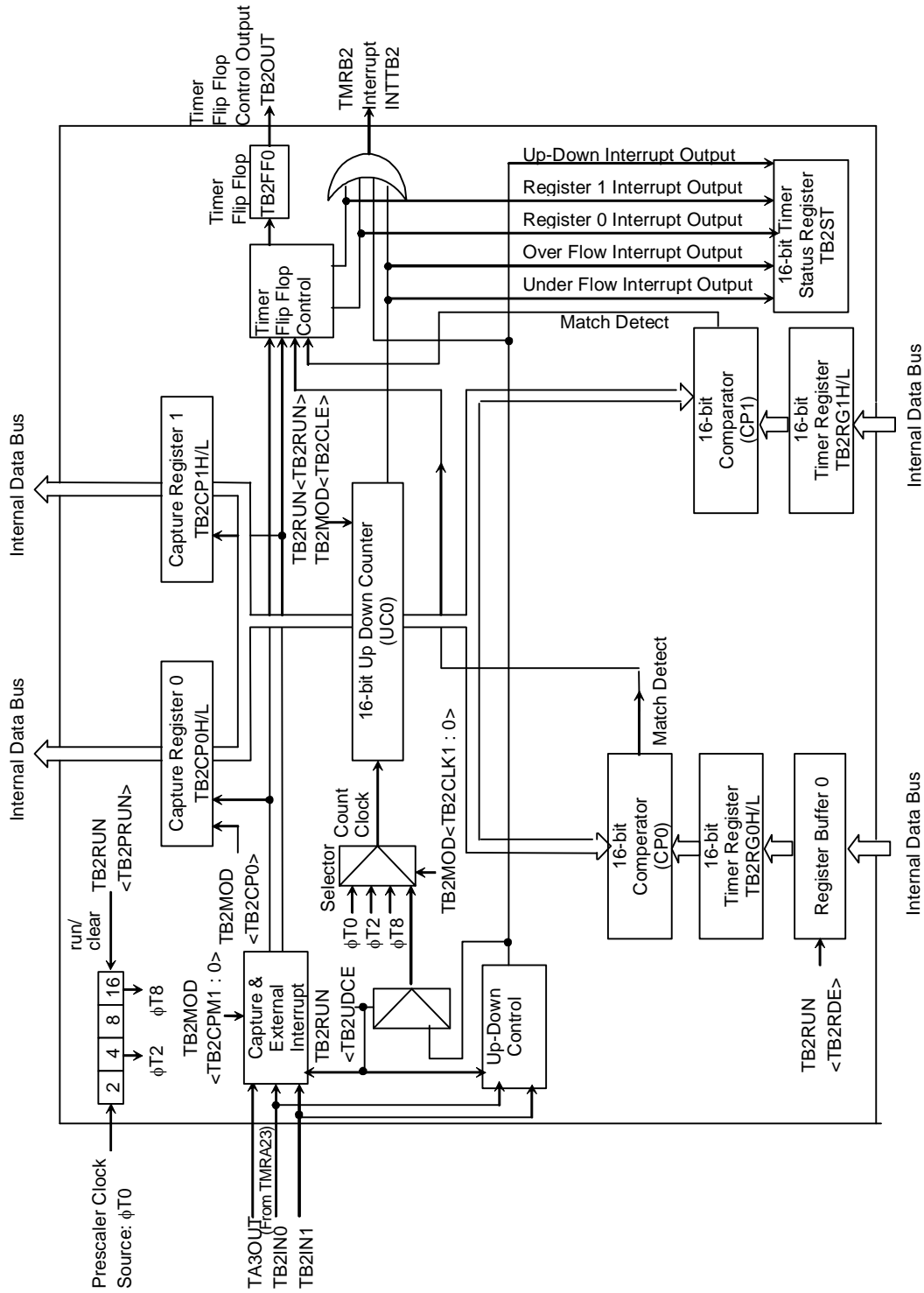
Figure 3.10.2 TMRB2/3 Block Diagram

3.10.2 Function description of each circuit

(1) Prescaler

The TMP1942 has a 5-bit prescaler to supply a clock to TMRB0. The prescaler's input clock ϕT0 has a frequency of fperiph, fperiph/2, or fperiph/4 as selected by SYSCR0<PRCK1:PRCK0> in the CG block.

Fperiph is either the clock fgear as selected by SYSCR1<FPSEL> in the CG block or the clock fc before division by the clock gear.

The prescaler is set to either run or stop by TA01RUN<TA0PRUN>. Writing a 1 to this bit causes the prescaler to start counting and writing 0 causes it to clear itself and stop counting. Table 3.10.5 shows the resolutions of the prescaler output clocks.

Table 3.10.5 Prescaler Output Clock Resolutions

@fc = 32 MHz

Peripheral Clock Selection <FPSEL>	Clock Gear Value <GEAR1:0>	Selected Prescaler Clock <PRCK1:0>	Prescaler Output Clock Resolution		
			ϕT1	ϕT4	ϕT16
0 (fgear)	00 (fc)	00 (fperiph/4)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
		01 (fperiph/2)	$fc/2^2$ (0.125 μs)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)
		10 (fperip)	—	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)
	01 (fc/2)	00 (fperiph/4)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
		01 (fperiph/2)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
		10 (fperip)	—	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)
	10 (fc/4)	00 (fperiph/4)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16 μs)
		01 (fperiph/2)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
		10 (fperip)	—	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
	11 (fc/8)	00 (fperiph/4)	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)	$fc/2^{10}$ (32 μs)
		01 (fperiph/2)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)	$fc/2^9$ (16 μs)
		10 (fperip)	—	$fc/2^6$ (2.0 μs)	$fc/2^8$ (8.0 μs)
1 (fc)	00 (fc)	00 (fperiph/4)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
		01 (fperiph/2)	$fc/2^2$ (0.125 μs)	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)
		10 (fperip)	—	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)
	01 (fc/2)	00 (fperiph/4)	$fc/2^3$ (0.25 μs)	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
		01 (fperiph/2)	—	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)
		10 (fperip)	—	$fc/2^3$ (0.25 μs)	$fc/25$ (1.0 μs)
	10 (fc/4)	00 (fperiph/4)	—	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
		01 (fperiph/2)	—	$fc/2^4$ (0.5 μs)	$fc/2^6$ (2.0 μs)
		10 (fperip)	—	—	$fc/2^5$ (1.0 μs)
	11 (fc/8)	00 (fperiph/4)	—	$fc/2^5$ (1.0 μs)	$fc/2^7$ (4.0 μs)
		01 (fperiph/2)	—	—	$fc/2^6$ (2.0 μs)
		10 (fperip)	—	—	$fc/2^5$ (1.0 μs)

Note 1: The prescaler's output clock ϕTn must be selected such that the relationship ϕTn < fsys/2 is satisfied (i.e., ϕTn must be slower than fsys/2).
Note 2: Do not change the clock gear value while the timer is running.
Note 3: The — character means "Don't use".

(2) Up-counter (UC0)

UC0 is a 16-bit binary counter which counts up synchronously with the input clock selected by TB0MOD<TB0CLK1:TB0CLK0>.

The input clock for UC0 is either the external clock entered via the TB0IN0 pin or one of the three prescaler output clocks, $\phi T1$, $\phi T4$ or $\phi T16$. The setting of TB0RUN<TB0RUN> either causes the up-counter UC0 to count, or stops and clears it. If the value in the up-counter UC0 matches the value in the timer register TB0RG1H/L while clearing is enabled, UC0 is cleared to 0. Clearing of UC0 can be enabled or disabled by setting TB0MOD<TB0CLE> accordingly.

If clearing is disabled, the counter functions as a free-running counter. In addition, when UC0 overflows, it generates an overflow interrupt INTTB01.

TMRB2 and TMRB3 support the 2-phase pulse input counter feature. When 2-phase pulse counter mode is selected with the setting of TB2RUN<TB2UDCE>, UC0 functions as an up/down-counter with an initial value of 0x7FFF. When the counter overflows, it is reloaded with an initial value of 0x0000. When the counter underflows, it is reloaded with an initial value of 0xFFFF. UC0 only functions as an up-counter in other modes.

> Note: Programming the TB0CLK[1:0] and TB0CLE bits in the TB0MOD register should only be attempted when the timer is not running.

(3) Timer registers (TB0RG0H/L and TB0RG1H/L)

Each channel incorporates two 16-bit registers used to set a counter value. When the value set in one of these timer registers matches the value of up-counter UC0, the comparator's match detection signal becomes active.

Timer registers TB0RG0H/L and TB0RG1H/L can be written in a single operation using a 2-byte data transfer instruction, or in two operations (the eight low-order bits first and then the eight high-order bits) using a 1-byte data transfer instruction.

The timer register TB0RG0 has a double-buffer structure, being paired with register buffer 0. The setting of TB0RUN<TB0RDE> enables or disables the register's double-buffer facility. The double-buffer is disabled when <TB0RDE> = 0 and enabled when <TB0RDE> = 1. When the double-buffer is enabled, data transfer from register buffer 0 to the timer register TB0RG0 is initiated by a match of UC0 and TB0RG1.

When reset, the contents of the timer registers TB0RG0 and TB0RG1 are undefined; hence, data must be written to the timer registers before the 16-bit timers can be used. A reset initializes TB0RUN<TB0RDE> to 0, disabling the double-buffer. To use the double-buffer, write data to the timer registers and set <TB0RDE> to 1, then write the following data in the register buffer.

TB0RG0 and its register buffer both have the same addresses, 0xFFFF_F188 and 0xFFFF_F189, allocated to them. When <TB0RDE> = 0, the same value is written to TB0RG0 and its register buffer; when <TB0RDE> = 1, the value is only written to the register buffer. Therefore, the register buffer must be disabled before the initial value is written to the timer register.

> Note: Programming the TB0RDE bit should only be attempted when the timer is not running.

(4) Capture registers (TB0CP0H/L and TB0CP1H/L)

TB0CP0H/L and TB0CP1H/L are 16-bit registers used to latch the value of the up-counter UC0. Data may be read out from a capture register in a single operation using a 2-byte data transfer instruction, or in two operations (the eight low-order bits first and then the eight high-order bits) using a 1-byte data transfer instruction.

(5) Capture controller

This circuit controls the timing at which the value in the up-counter UC0 is latched into the capture registers (TB0CP0 and TB0CP1). The capture register latch timing is set using TB0MOD<TB0CMPM1:TB0CMPM0>.

In addition, the value of the up-counter UC0 can be latched into the capture registers by software. Each time TB0MOD<TB0CP0> is set to 0, the UC0 value at that point is latched into TB0CP0. Before this function can be used, the prescaler must be placed in the Run state by setting TB0RUN<TB0PRUN> to 1.

In 2-phase pulse counter mode (only for TMRB2 and TMRB3), the counter value is latched by software capture.

Note1: Reading the eight low-order bits of the capture register disables capture operation. Subsequently reading the eight high-order bits of the capture register enables capture operation.

Note2: If the timer is stopped when only the eight low-order bits have been read, capture operation is not enabled even after the timer is restarted. Do not stop the timer until both the eight low-order and eight high-order bits are read.

Note3: When the TB0IN0 pin is selected as a capture trigger input, it can not function as a timer clock.

---

(6) Comparators (CP0 and CP1)

The two 16-bit comparators compare the value of the up-counter UC0 with the values set in the timer registers TB0RG0 and TB0RG1 to detect a match. If the value in either TB0RG0 or TB0RG1 matches the value in UC0, the corresponding comparator generates an INTTB0 interrupt.

(7) Timer flip-flop (TB0FF0)

The timer flip-flop TB0FF0 is designed to be inverted by a match detection signal from the comparator or by a latch signal to the capture registers. Inversion can be enabled or disabled by setting TB0FFCR<TB0C1T1,TB0C0T1,TB0E1T1,TB0E0T1> accordingly.

When reset, the TB0FF0 value is undefined. Writing 00 to TB0FFCR<TB0FF0C1:TB0FF0C0> inverts the value of the flip-flop; writing 01 to TB0FFCR<TB0FF0C1,TB0FF0C0> sets the flip-flop to 1; writing 10 to TB0FFCR<TB0FF0C1,TB0FF0C0> clears the flip-flop to 0.

The TB0FF0 value can be forwarded to the timer output pin, TB0OUT (shared with PA2). When timer output is needed, this pin must be set for that purpose by using the port A registers PACR and PAFC.

Note: Programming the TB0FF0C1[1:0] field should only be attempted when the timer is not running.

3.10.3 Register description

TMRB0 run register

		7	6	5	4	3	2	1	0
TB0RUN (0xFFFF_ F140)	Bit symbol	TB0RDE	—	—	—	I2TB0	TB0PRUN	—	TB0RUN
	Read/Write	R/W	R/W	—	R/W	R/W	R/W	—	R/W
	After reset	0	0	—	0	0	0	—	0
	Function	Double Buffer 0: Disable 1: Enable	Must always be set to 0.		Must always be set to 0.	IDLE 0: Idle 1: Operate	Timer Run/Stop Control 0: Stop and cleared 1: Count		

I2TB0:   Operation in IDLE mode
TB0PRUN: Operation of the prescaler
TB0RUN: Operation of timer B0

> Note: TB0RUN bits 1 and 5 are undefined when read.

TMRB1 run register

		7	6	5	4	3	2	1	0
TB1RUN (0xFFFF_ F150)	Bit symbol	TB1RDE	—	—	—	I2TB1	TB1PRUN	—	TB1RUN
	Read/Write	R/W	R/W	—	R/W	R/W	R/W	—	R/W
	After reset	0	0	—	0	0	0	—	0
	Function	Double Buffer 0: Disable 1: Enable	Must always be set to 0.		Must always be set to 0.	IDLE 0: Idle 1: Operate	16-Bit Timer Run/Stop Control 0: Stop and cleared 1: Count		

I2TB1:   Operation in IDLE mode
TB1PRUN: Operation of the prescaler
TB1RUN: Operation of timer B1

> Note: TB1RUN bits 1 and 5 are undefined when read.

Figure 3.10.3 TMRB Registers

TMRB2 run register

		7	6	5	4	3	2	1	0
TB2RUN (0xFFFF_ F160)	Bit symbol	TB2RDE	—	UD2CK	TB2UDCE	I2TB2	TB2PRUN		TB2RUN
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W		R/W
	After reset	0	0	0	0	0	0		0
	Function	Double Buffer 0: Disable 1: Enable	Must always be set to 0.	Sampling clock 0: fs 1: fsys/2	2-phase counter enable 0: Disable 1: Enable	IDLE 0: Idle 1: Operate	Timer Run/Stop Control 0: Stop and cleared 1: Count		

I2TB2:   Operation in IDLE mode
TB2PRUN: Operation of the prescaler
TB2RUN: Operation of timer B2
TB2UDCE: Operation of the 2-phase pulse input counter
UD2CK: Sampling clock selection for 2-phase pulse input

Note 1: TB2RUN bits 1 and 5 are undefined when read.
Note 2: Setting TB2RUN bit 4 to 1 selects 2-phase pulse input counter mode, causing the counter to operate as an up/down-counter. Clearing the bit to 0 restores normal timer mode, causing the timer to operate as an up-counter.

TMRB3 run register

		7	6	5	4	3	2	1	0
TB3RUN (0xFFFF_ F170)	Bit symbol	TB3RDE	—	UD3CK	TB3UDCE	I2TB3	TB3PRUN	—	TB3RUN
	Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	—	R/W
	After reset	0	0	0	0	0	0	—	0
	Function	Double Buffer 0: Disable 1: Enable	Must always be set to 0.	Sampling clock 0: fs 1: fsys/2	2-phase counter enable 0: Disable 1: Enable	IDLE 0: Idle 1: Operate	16-Bit Timer Run/Stop Control 0: Stop and cleared 1: Count		

I2TB3:   Operation in IDLE mode
TB3PRUN: Operation of the prescaler
TB3RUN: Operation of timer B3
TB3UDCE: Operation of the 2-phase pulse input counter
UD3CK: Sampling clock selection for 2-phase pulse input

Note 1: TB3RUN bits 1 and 5 are undefined when read.
Note 2: Setting TB3RUN bit 4 to 1 selects 2-phase pulse input counter mode, causing the counter to operate as an up/down-counter. Clearing the bit to 0 restores normal timer mode, causing the timer to operate as an up-counter.

Figure 3.10.4 TMRB Registers

TMRB4 run register

		7	6	5	4	3	2	1	0
TB4RUN (0xFFFF_ F180)	Bit symbol	TB4RDE	—	—	—	I2TB4	TB4PRUN	—	TB4RUN
	Read/Write	R/W	R/W	—	R/W	R/W	R/W	—	R/W
	After reset	0	0	—	0	0	0	—	0
	Function	Double Buffer 0: Disable 1: Enable	Must always be set to 0.		Must always be set to 0.	IDLE 0: Idle 1: Operate	16-Bit Timer Run/Stop Control 0: Stop and cleared 1: Count		

I2TB4:   Operation in IDLE mode
TB4PRUN: Operation of the prescaler
TB4RUN: Operation of timer B4

> Note: TB4RUN bits 1 and 5 are undefined when read.

TMRB5 run register

		7	6	5	4	3	2	1	0
TB5RUN (0xFFFF_ F190)	Bit symbol	TB5RDE	—	—	—	I2TB5	TB5PRUN	—	TB5RUN
	Read/Write	R/W	R/W	—	R/W	R/W	R/W	—	R/W
	After reset	0	0	—	0	0	0	—	0
	Function	Double Buffer 0: Disable 1: Enable	Must always be set to 0.		Must always be set to 0.	IDLE 0: Idle 1: Operate	16-Bit Timer Run/Stop Control 0: Stop and cleared 1: Count		

I2TB5:   Operation in IDLE mode
TB5PRUN: Operation of the prescaler
TB5RUN: Operation of timer B5

> Note: TB5RUN bits 1 and 5 are undefined when read.

TMRB6 run register

		7	6	5	4	3	2	1	0
TB6RUN (0xFFFF_ F1A0)	Bit symbol	TB6RDE	—	—	—	I2TB6	TB6PRUN	—	TB6RUN
	Read/Write	R/W	R/W	—	R/W	R/W	R/W	—	R/W
	After reset	0	0	—	0	0	0	—	0
	Function	Double Buffer 0: Disable 1: Enable	Must always be set to 0.		Must always be set to 0.	IDLE 0: Idle 1: Operate	16-Bit Timer Run/Stop Control 0: Stop and cleared 1: Count		

I2TB6: Operation in IDLE mode
TB6PRUN: Operation of the prescaler
TB6RUN: Operation of timer B6

> Note: TB6RUN bits 1 and 5 are undefined when read.

Figure 3.10.5 TMRB Registers

TMRB7 run register

	7	6	5	4	3	2	1	0
Bit symbol	TB7RDE	—	—	—	I2TB7	TB7PRUN	—	TB7RUN
Read/Write	R/W	R/W	—	—	R/W	R/W	—	R/W
After reset	0	0	—	—	0	0	—	0
Function	Double Buffer 0: Disable 1: Enable	Must always be set to 0.		Must always be set to 0.	IDLE 0: Idle 1: Operate	16-Bit Timer Run/Stop Control 0: Stop and cleared 1: Count		

TB7RUN (0xFFFF_F1B0)

I2TB7:   Operation in IDLE mode
TB7PRUN: Operation of the prescaler
TB7RUN: Operation of timer B7

Note: TB7RUN bits 1 and 5 are undefined when read.

TMRB8 run register

	7	6	5	4	3	2	1	0
Bit symbol	TB8RDE	—	—	—	I2TB8	TB8PRUN	—	TB8RUN
Read/Write	R/W	R/W	—	—	R/W	R/W	—	R/W
After reset	0	0	—	—	0	0	—	0
Function	Double Buffer 0: Disable 1: Enable	Must always be set to 0.		Must always be set to 0.	IDLE 0: Idle 1: Operate	16-Bit Timer Run/Stop Control 0: Stop and cleared 1: Count		

TB8RUN (0xFFFF_F1C0)

I2TB8:   Operation in IDLE mode
TB8PRUN: Operation of the prescaler
TB8RUN: Operation of timer B8

Note: TB8RUN bits 1 and 5 are undefined when read.

TMRB9 run register

	7	6	5	4	3	2	1	0
Bit symbol	TB9RDE	—	—	—	I2TB9	TB9PRUN	—	TB9RUN
Read/Write	R/W	R/W	—	—	R/W	R/W	—	R/W
After reset	0	0	—	—	0	0	—	0
Function	Double Buffer 0: Disable 1: Enable	Must always be set to 0.		Must always be set to 0.	IDLE 0: Idle 1: Operate	16-Bit Timer Run/Stop Control 0: Stop and cleared 1: Count		

TB9RUN (0xFFFF_F1D0)

I2TB9:   Operation in IDLE mode
TB9PRUN: Operation of the prescaler
TB9RUN: Operation of timer B9

Note: TB9RUN bits 1 and 5 are undefined when read.

Figure 3.10.6 TMRB Registers

TMRBA run register

		7	6	5	4	3	2	1	0
TBARUN (0xFFFF_ F1E0)	Bit symbol	TBARDE	—	—	—	I2TBA	TBAPRUN	—	TBARUN
	Read/Write	R/W	R/W	—	—	R/W	R/W	—	R/W
	After reset	0	0	—	—	0	0	—	0
	Function	Double Buffer 0: Disable 1: Enable	Must always be set to 0.		Must always be set to 0.	IDLE 0: Idle 1: Operate	16-Bit Timer Run/Stop Control 0: Stop and cleared 1: Count		

I2TBA: Operation in IDLE mode
TBAPRUN: Operation of the prescaler
TBARUN: Operation of timer BA

Note: TBARUN bits 1 and 5 are undefined when read.

TMRBB run register

		7	6	5	4	3	2	1	0
TBBRUN (0xFFFF_ F1F0)	Bit symbol	TBBRDE	—	—	—	I2TBB	TBBPRUN	—	TBBRUN
	Read/Write	R/W	R/W	—	—	R/W	R/W	—	R/W
	After reset	0	0	—	—	0	0	—	0
	Function	Double Buffer 0: Disable 1: Enable	Must always be set to 0.		Must always be set to 0.	IDLE 0: Idle 1: Operate	16-Bit Timer Run/Stop Control 0: Stop and cleared 1: Count		

I2TBB: Operation in IDLE mode
TBBPRUN: Operation of the prescaler
TBBRUN: Operation of timer BB

Note: TBBRUN bits 1 and 5 are undefined when read.

TMRBC run register

		7	6	5	4	3	2	1	0
TBCRUN (0xFFFF_ F200)	Bit symbol	TBCRDE	—	—	—	I2TBC	TBCPRUN	—	TBCRUN
	Read/Write	R/W	R/W	—	—	R/W	R/W	—	R/W
	After reset	0	0	—	—	0	0	—	0
	Function	Double Buffer 0: Disable 1: Enable	Must always be set to 0.		Must always be set to 0.	IDLE 0: Idle 1: Operate	16-Bit Timer Run/Stop Control 0: Stop and cleared 1: Count		

I2TBC: Operation in IDLE mode
TBCPRUN: Operation of the prescaler
TBCRUN: Operation of timer BC

Note: TBCRUN bits 1 and 5 are undefined when read.

Figure 3.10.7 TMRB Registers

TMRBD run register

		7	6	5	4	3	2	1	0
TBDRUN (0xFFFF_ F210)	Bit symbol	TBDRDE	—	—	—	I2TBD	TBDPRUN	—	TBDRUN
	Read/Write	R/W	R/W	—	—	R/W	R/W	—	R/W
	After reset	0	0	—	—	0	0	—	0
	Function	Double Buffer 0: Disable 1: Enable	Must always be set to 0.		Must always be set to 0.	IDLE 0: Idle 1: Operate	16-Bit Timer Run/Stop Control 0: Stop and cleared 1: Count		

I2TBD: Operation in IDLE mode
TBDPRUN: Operation of the prescaler
TBDRUN: Operation of timer BD

Note: TBDRUN bits 1 and 5 are undefined when read.

Figure 3.10.8 TMRB Registers

TMRB0 mode register

TB0MOD (0xFFFF_F142)		7	6	5	4	3	2	1	0
	Bit symbol	—	—	TB0CP0	TB0CPM1	TB0CPM0	TB0CLE	TB0CLK1	TB0CLK0
	Read/Write	—		W	R/W				
	After reset	0	0	1	0	0	0	0	0
	Function	Must always be set to 00.		Software capture control 0: Software capture 1: Don't care	Capture timing 00: Disabled 01: TB0IN0 ↑ TB0IN1 ↑ 10: TB0IN0 ↑ TB0IN0 ↓ 11: TA3OUT ↑ TA3OUT ↓		Up-counter control 0: Clearing disabled 1: Clearing enabled	Source clock selection 00: TB0IN0 pin input 01: φT1 10: φT4 11: φT16	

Clearing up-counter (UC0)

0	Clearing of up-counter disabled
1	Up-counter cleared when it matches TB0RG1

Capture timing

	Capture control
00	Capture disabled
01	Latches value into TB0CP0 at rise of TB0IN0. Latches value into TB0CP1 at rise of TB0IN1.
10	Latches value into TB0CP0 at rise of TB0IN0. Latches value into TB0CP1 at fall of TB0IN0.
11	Latches value into TB0CP0 at rise of TA3OUT. Latches value into TB0CP1 at fall of TA3OUT.

Software capture

0	Latches up-counter value into TB0CP0.
1	Don't care

TMRB1 mode register

TB1MOD (0xFFFF_F152)		7	6	5	4	3	2	1	0
	Bit symbol	—	—	TB1CP0	TB1CPM1	TB1CPM0	TB1CLE	TB1CLK1	TB1CLK0
	Read/Write	R/W		W*	R/W				
	After reset	0	0	1	0	0	0	0	0
	Function	Must always be set to 00.		Software capture control 0: Software capture 1: Don't care	Capture timing 00: Disabled 01: TB1N0 ↑ TB1IN1 ↑ 10: TB1IN0 ↑ TB1IN0 ↓ 11: TA3OUT ↑ TA3OUT ↓		Up-counter control 0: Clearing disabled 1: Clearing enabled	Source clock selection 00: TB1IN0 pin input 01: φT1 10: φT4 11: φT16	

Clearing up-counter (UC0)

0	Clearing of up-counter disabled
1	Up-counter cleared when it matches TB1RG1

Capture timing

	Capture control
00	Capture disabled
01	Latches value into TB1CP0 at rise of TB1IN0. Latches value into TB1CP1 at rise of TB1IN1.
10	Latches value into TB1CP0 at rise of TB1IN0. Latches value into TB1CP1 at fall of TB1IN0.
11	Latches value into TB1CP0 at rise of TA3OUT. Latches value into TB1CP1 at fall of TA3OUT.

Software capture

0	Latches up-counter value into TB1CP0.
1	Don't care

Figure 3.10.9 TMRB Registers

TMRB2 mode register

TB2MOD (0xFFFF_ F162)		7	6	5	4	3	2	1	0
	Bit symbol	—	—	TB2CP0	TB2CPM1	TB2CPM0	TB2CLE	TB2CLK1	TB2CLK0
	Read/Write	—		W*	R/W				
	After reset	0	0	1	0	0	0	0	0
	Function	Must always be set to 00.		Software capture control 0: Software capture 1: Don't care	Capture timing 00: Disabled 01: TB2IN0↑ TB2IN1↑ 10: TB2IN0↑ TB2IN0↓ 11: TA3OUT↑ TA3OUT↓		Up-counter control 0: Clearing disabled 1: Clearing enabled	Source clock selection 00: TB2IN0 pin input 01: φT1 10: φT4 11: φT16	

Clearing up-counter (UC0)

0	Clearing of up-counter disabled
1	Up-counter cleared when it matches TB2RG1

Capture timing

	Capture control
00	Capture disabled
01	Latches value into TB2CP0 at rise of TB2IN0. Latches value into TB2CP1 at rise of TB2IN1.
10	Latches value into TB2CP0 at rise of TB2IN0. Latches value into TB2CP1 at fall of TB2IN0.
11	Latches value into TB2CP0 at rise of TA3OUT. Latches value into TB2CP1 at fall of TA3OUT.

Software capture

0	Latches up-counter value into TB2CP0.
1	Don't care

TMRB3 mode register

TB3MOD (0xFFFF_ F172)		7	6	5	4	3	2	1	0
	Bit symbol	—	—	TB3CP0	TB3CPM1	TB3CPM0	TB3CLE	TB3CLK1	TB3CLK0
	Read/Write	—		W*	R/W				
	After reset	0	0	1	0	0	0	0	0
	Function	Must always be set to 00.		Software capture control 0: Software capture 1: Don't care	Capture timing 00: Disabled 01: TB3IN0↑ TB3IN1↑ 10: TB3IN0↑ TB3IN0↓ 11: TA3OUT↑ TA3OUT↓		Up-counter control 0: Clearing disabled 1: Clearing enabled	Source clock selection 00: TB3IN0 pin input 01: φT1 10: φT4 11: φT16	

Clearing up-counter (UC0)

0	Clearing of up-counter disabled
1	Up-counter cleared when it matches TB3RG1

Capture timing

	Capture control
00	Capture disabled
01	Latches value into TB3CP0 at rise of TB3IN0. Latches value into TB3CP1 at rise of TB3IN1.
10	Latches value into TB3CP0 at rise of TB3IN0. Latches value into TB3CP1 at fall of TB3IN0.
11	Latches value into TB3CP0 at rise of TA3OUT. Latches value into TB3CP1 at fall of TA3OUT.

Software capture

0	Latches up-counter value into TB3CP0.
1	Don't care

Figure 3.10.10 TMRB Registers

TMRB4 mode register

		7	6	5	4	3	2	1	0
TB4MOD (0xFFFF_ F182)	Bit symbol	—	—	TB4CP0	TB4CPM1	TB4CPM0	TB4CLE	TB4CLK1	TB4CLK0
	Read/Write	—		W∗	R/W				
	After reset	0	0	1	0	0	0	0	0
	Function	Must always be set to 00.		Software capture control 0: Software capture 1: Don't care	Capture timing 00: Disabled 01: TB4IN0 ↑ TB4IN1 ↑ 10: TB4IN0 ↑ TB4IN0 ↓ 11: TA3OUT ↑ TA3OUT ↓		Up-counter control 0: Clearing disabled 1: Clearing enabled	Source clock selection 00: TB4IN0 pin input 01: φT1 10: φT4 11: φT16	

Clearing up-counter (UC0)

0	Clearing of up-counter disabled
1	Up-counter cleared when it matches TB4RG1

Capture timing

	Capture control
00	Capture disabled
01	Latches value into TB4CP0 at rise of TB4IN0. Latches value into TB4CP1 at rise of TB4IN1.
10	Latches value into TB4CP0 at rise of TB4IN0. Latches value into TB4CP1 at fall of TB4IN0.
11	Latches value into TB4CP0 at rise of TA3OUT. Latches value into TB4CP1 at fall of TA3OUT.

Software capture

0	Latches up-counter value into TB4CP0.
1	Don't care

TMRB5 mode register

		7	6	5	4	3	2	1	0
TB5MOD (0xFFFF_ F192)	Bit symbol	—	—	TB5CP0	TB5CPM1	TB5CPM0	TB5CLE	TB5CLK1	TB5CLK0
	Read/Write	—		W∗	R/W				
	After reset	0	0	1	0	0	0	0	0
	Function	Must always be set to 00.		Software capture control 0: Software capture 1: Don't care	Capture timing 00: Disabled 01: Setting prohibited 10: Setting prohibited 11: TA3OUT↑TA3OUT↓		Up-counter control 0: Clearing disabled 1: Clearing enabled	Source clock selection 00: Setting prohibited 01: φT1 10: φT4 11: φT16	

Clearing up-counter (UC0)

0	Clearing of up-counter disabled
1	Up-counter cleared when it matches TB5RG1

Capture timing

00	Capture disabled
01	Capture disabled
10	Capture disabled
11	Latches value into TB5CP0 at rise of TA3OUT. Latches value into TB5CP1 at fall of TA3OUT.

Software capture

0	Latches up-counter value into TB5CP0.
1	Don't care

Figure 3.10.11 TMRB Registers

TMRB6 mode register

TB6MOD (0xFFFF_ F1A2)		7	6	5	4	3	2	1	0
	Bit symbol	—	—	TB6CP0	TB6CPM1	TB6CPM0	TB6CLE	TB6CLK1	TB6CLK0
	Read/Write	—		W∗	R/W				
	After reset	0	0	1	0	0	0	0	0
	Function	Must always be set to 00.		Software capture control 0: Software capture 1: Don't care	Capture timing 00: Disabled 01: Setting prohibited 10: Setting prohibited 11: TA3OUT↑TA3OUT↓		Up-counter control 0: Clearing disabled 1: Clearing enabled	Source clock selection 00: Setting prohibited 01: φT1 10: φT4 11: φT16	

Clearing up-counter (UC0)

0	Clearing of up-counter disabled
1	Up-counter cleared when it matches TB6RG1

Capture timing

00	Capture disabled
01	Capture disabled
10	Capture disabled
11	Latches value into TB6CP0 at rise of TA3OUT. Latches value into TB6CP1 at fall of TA3OUT.

Software capture

0	Latches up-counter value into TB6CP0.
1	Don't care

TMRB7 mode register

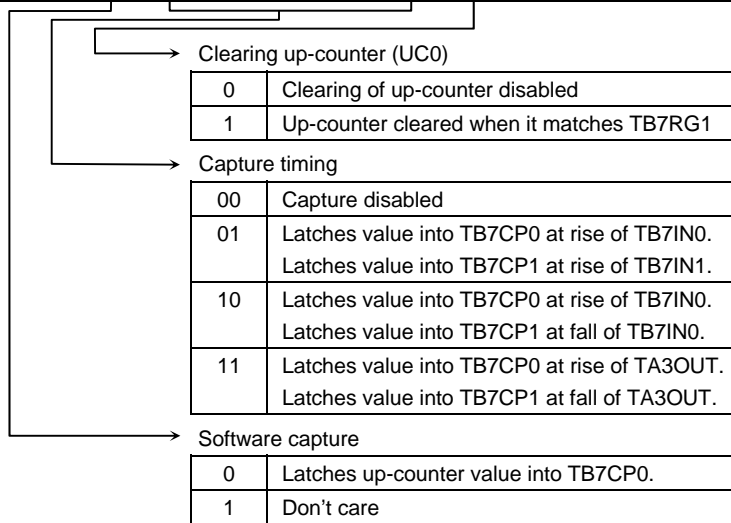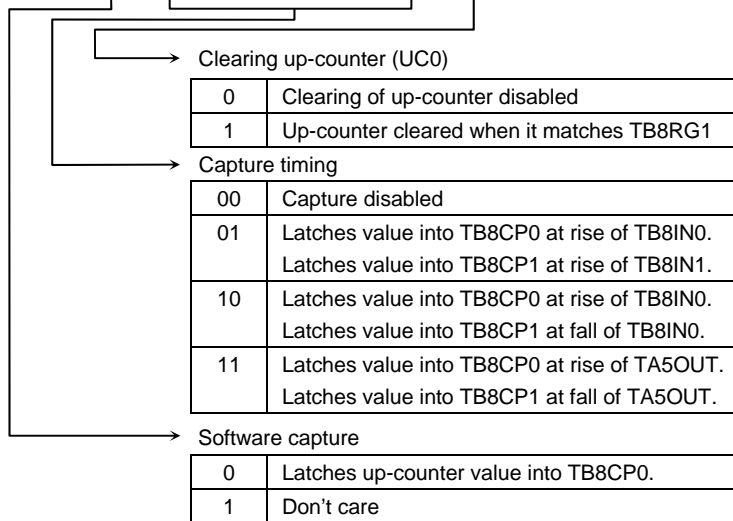| TB7MOD (0xFFFF_ F1B2)		7	6	5	4	3	2	1	0
	Bit symbol	—	—	TB7CP0	TB7CPM1	TB7CPM0	TB7CLE	TB7CLK1	TB7CLK0
	Read/Write	—		W∗	R/W				
	After reset	0	0	1	0	0	0	0	0
	Function	Must always be set to 00.		Software capture control 0: Software capture 1: Don't care	Capture timing 00: Disabled 01: TB7IN0 ↑ TB7IN1 ↑ 10: TB7IN0 ↑ TB7IN0 ↓ 11: TA3OUT ↑ TA3OUT		Up-counter control 0: Clearing disabled 1: Clearing enabled	Source clock selection 00: TB7IN0 pin input 01: φT1 10: φT4 11: φT16	

Clearing up-counter (UC0)

0	Clearing of up-counter disabled
1	Up-counter cleared when it matches TB7RG1

Capture timing

00	Capture disabled
01	Latches value into TB7CP0 at rise of TB7IN0. Latches value into TB7CP1 at rise of TB7IN1.
10	Latches value into TB7CP0 at rise of TB7IN0. Latches value into TB7CP1 at fall of TB7IN0.
11	Latches value into TB7CP0 at rise of TA3OUT. Latches value into TB7CP1 at fall of TA3OUT.

Software capture

0	Latches up-counter value into TB7CP0.
1	Don't care

Figure 3.10.12 TMRB Registers

TMRB8 mode register

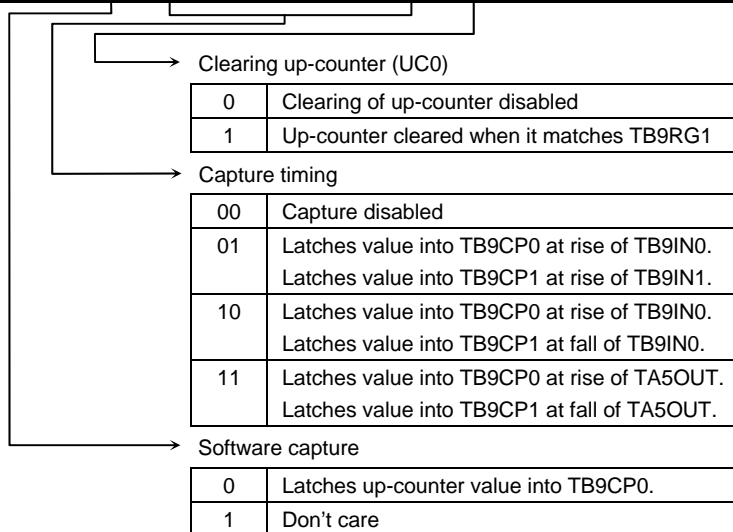|	7	6	5	4	3	2	1	0
Bit symbol	—	—	TB8CP0	TB8CPM1	TB8CPM0	TB8CLE	TB8CLK1	TB8CLK0
Read/Write	—		W*	R/W				
After reset	0	0	1	0	0	0	0	0
Function	Must always be set to 00.		Software capture control 0: Software capture 1: Don't care	Capture timing 00: Disabled 01: TB8IN0 ↑ TB8IN1 ↑ 10: TB8IN0 ↑ TB8IN0 ↓ 11: TA5OUT ↑ TA5OUT		Up-counter control 0: Clearing disabled 1: Clearing enabled	Source clock selection 00: TB8IN0 pin input 01: ϕT1 10: ϕT4 11: ϕT16	

TB8MOD (0xFFFF_ F1C2)

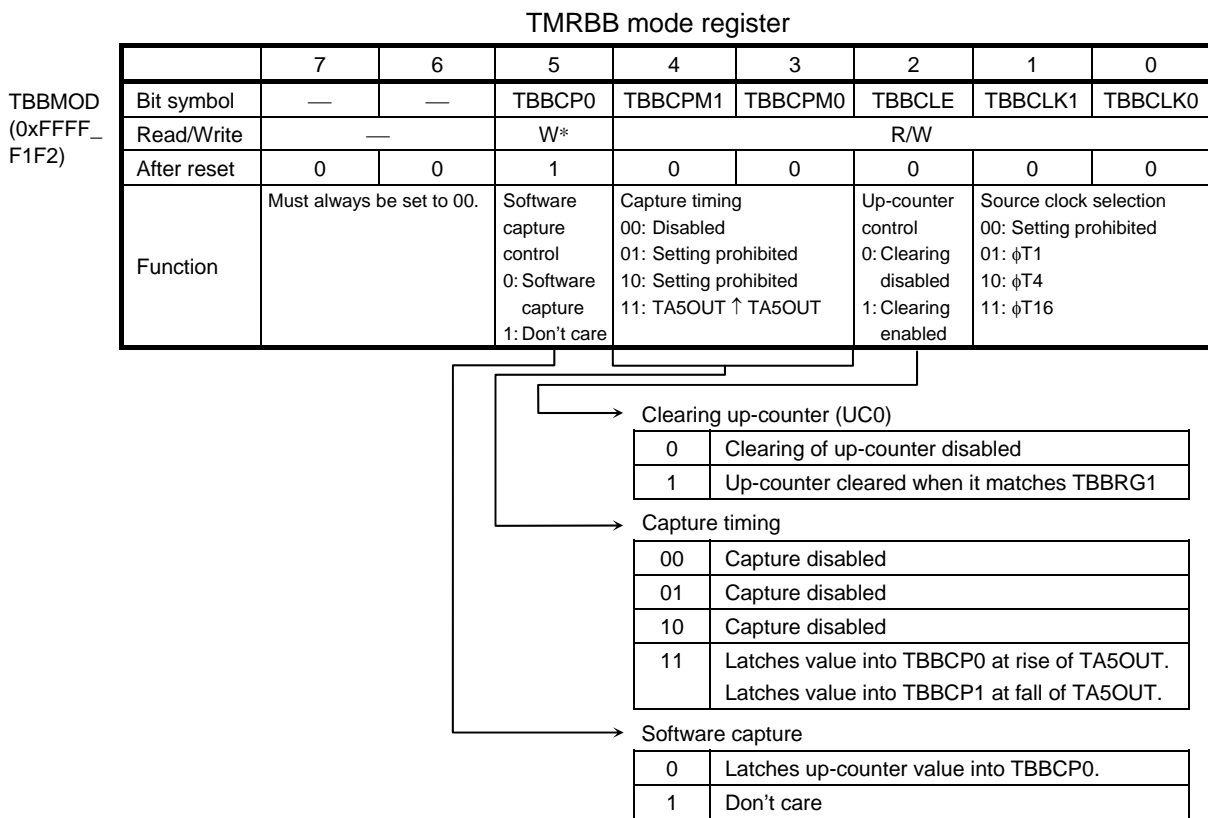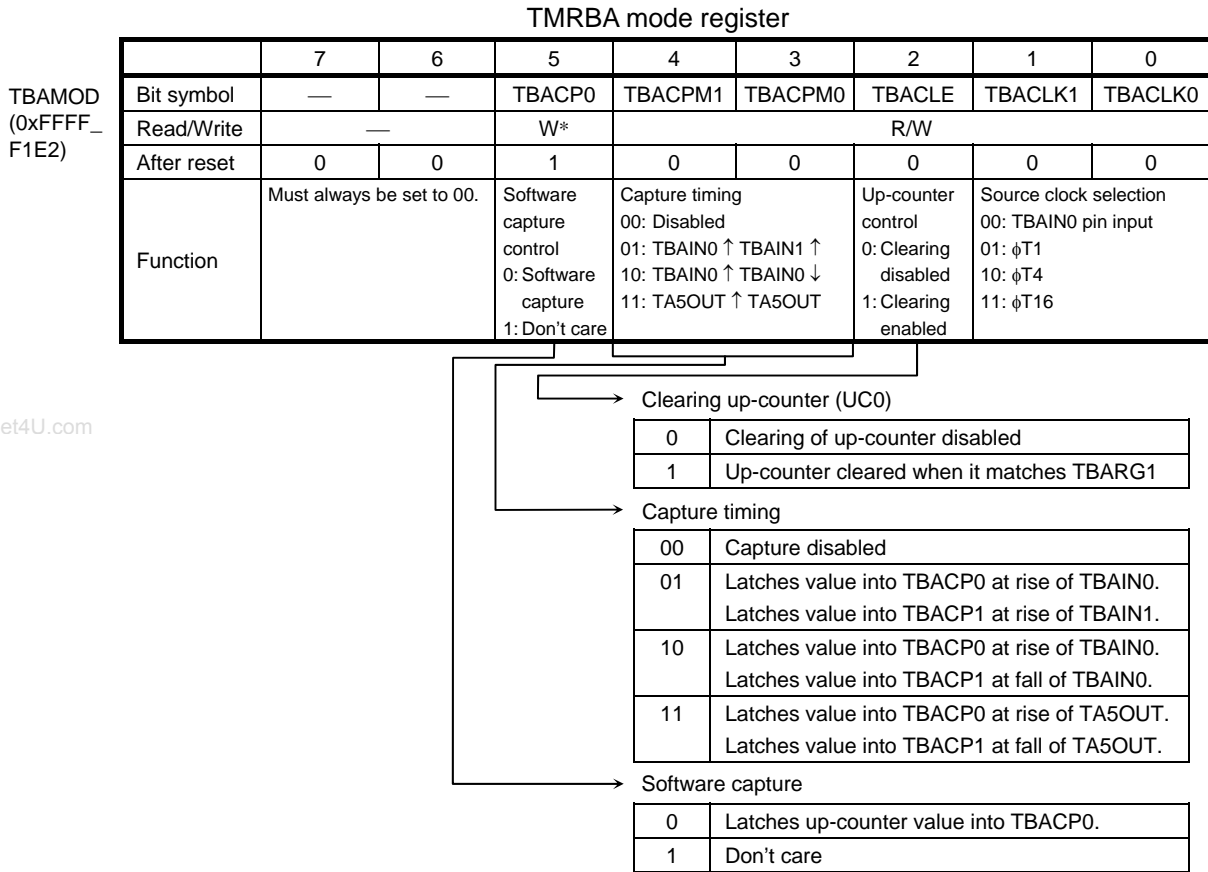Clearing up-counter (UC0)

0	Clearing of up-counter disabled
1	Up-counter cleared when it matches TB8RG1

Capture timing

00	Capture disabled
01	Latches value into TB8CP0 at rise of TB8IN0. Latches value into TB8CP1 at rise of TB8IN1.
10	Latches value into TB8CP0 at rise of TB8IN0. Latches value into TB8CP1 at fall of TB8IN0.
11	Latches value into TB8CP0 at rise of TA5OUT. Latches value into TB8CP1 at fall of TA5OUT.

Software capture

0	Latches up-counter value into TB8CP0.
1	Don't care

TMRB9 mode register

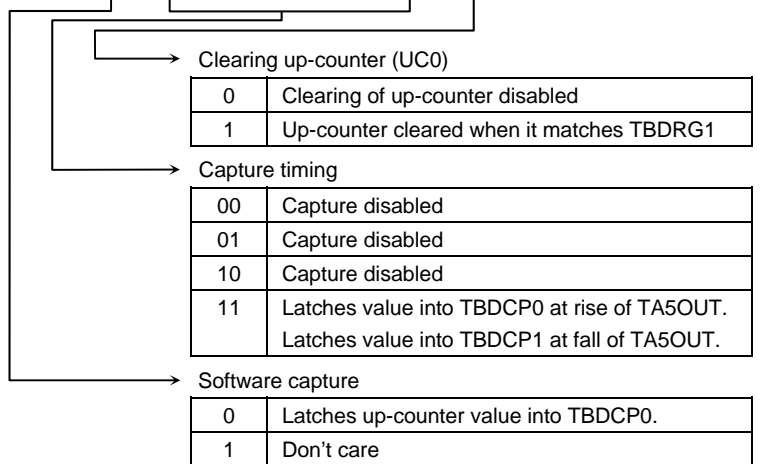|	7	6	5	4	3	2	1	0
Bit symbol	—	—	TB9CP0	TB9CPM1	TB9CPM0	TB9CLE	TB9CLK1	TB9CLK0
Read/Write	—		W*	R/W				
After reset	0	0	1	0	0	0	0	0
Function	Must always be set to 00.		Software capture control 0: Software capture 1: Don't care	Capture timing 00: Disabled 01: TB9IN0 ↑ TB9IN1 ↑ 10: TB9IN0 ↑ TB9IN0 ↓ 11: TA5OUT ↑ TA5OUT		Up-counter control 0: Clearing disabled 1: Clearing enabled	Source clock selection 00: TB9IN0 pin input 01: ϕT1 10: ϕT4 11: ϕT16	

TB9MOD (0xFFFF_ F1D2)

Clearing up-counter (UC0)

0	Clearing of up-counter disabled
1	Up-counter cleared when it matches TB9RG1

Capture timing

00	Capture disabled
01	Latches value into TB9CP0 at rise of TB9IN0. Latches value into TB9CP1 at rise of TB9IN1.
10	Latches value into TB9CP0 at rise of TB9IN0. Latches value into TB9CP1 at fall of TB9IN0.
11	Latches value into TB9CP0 at rise of TA5OUT. Latches value into TB9CP1 at fall of TA5OUT.

Software capture

0	Latches up-counter value into TB9CP0.
1	Don't care

Figure 3.10.13 TMRB Registers

TMRBA mode register

TBAMOD (0xFFFF_F1E2)		7	6	5	4	3	2	1	0
	Bit symbol	—	—	TBACP0	TBACPM1	TBACPM0	TBACLE	TBACLK1	TBACLK0
	Read/Write	—		W*	R/W				
	After reset	0	0	1	0	0	0	0	0
	Function	Must always be set to 00.		Software capture control 0: Software capture 1: Don't care	Capture timing 00: Disabled 01: TBAIN0 ↑ TBAIN1 ↑ 10: TBAIN0 ↑ TBAIN0 ↓ 11: TA5OUT ↑ TA5OUT		Up-counter control 0: Clearing disabled 1: Clearing enabled	Source clock selection 00: TBAIN0 pin input 01: φT1 10: φT4 11: φT16	

Clearing up-counter (UC0)

0	Clearing of up-counter disabled
1	Up-counter cleared when it matches TBARG1

Capture timing

00	Capture disabled
01	Latches value into TBACP0 at rise of TBAIN0. Latches value into TBACP1 at rise of TBAIN1.
10	Latches value into TBACP0 at rise of TBAIN0. Latches value into TBACP1 at fall of TBAIN0.
11	Latches value into TBACP0 at rise of TA5OUT. Latches value into TBACP1 at fall of TA5OUT.

Software capture

0	Latches up-counter value into TBACP0.
1	Don't care

TMRBB mode register

TBBMOD (0xFFFF_F1F2)		7	6	5	4	3	2	1	0
	Bit symbol	—	—	TBBCP0	TBBCPM1	TBBCPM0	TBBCLE	TBBCLK1	TBBCLK0
	Read/Write	—		W*	R/W				
	After reset	0	0	1	0	0	0	0	0
	Function	Must always be set to 00.		Software capture control 0: Software capture 1: Don't care	Capture timing 00: Disabled 01: Setting prohibited 10: Setting prohibited 11: TA5OUT ↑ TA5OUT		Up-counter control 0: Clearing disabled 1: Clearing enabled	Source clock selection 00: Setting prohibited 01: φT1 10: φT4 11: φT16	

Clearing up-counter (UC0)

0	Clearing of up-counter disabled
1	Up-counter cleared when it matches TBBRG1

Capture timing

00	Capture disabled
01	Capture disabled
10	Capture disabled
11	Latches value into TBBCP0 at rise of TA5OUT. Latches value into TBBCP1 at fall of TA5OUT.

Software capture

0	Latches up-counter value into TBBCP0.
1	Don't care

Figure 3.10.14 TMRB Registers

TMRBC mode register

	7	6	5	4	3	2	1	0
Bit symbol	—	—	TBCCP0	TBCCPM1	TBCCPM0	TBCCLE	TBCCLK1	TBCCLK0
Read/Write	—		W*	R/W				
After reset	0	0	1	0	0	0	0	0
Function	Must always be set to 00.		Software capture control 0: Software capture 1: Don't care	Capture timing 00: Disabled 01: Setting prohibited 10: Setting prohibited 11: TA5OUT ↑ TA5OUT		Up-counter control 0: Clearing disabled 1: Clearing enabled	Source clock selection 00: Setting prohibited 01: φT1 10: φT4 11: φT16	

TBCMOD
(0xFFFF_F202)

Clearing up-counter (UC0)

0	Clearing of up-counter disabled
1	Up-counter cleared when it matches TBCRG1

Capture timing

00	Capture disabled
01	Capture disabled
10	Capture disabled
11	Latches value into TBCCP0 at rise of TA5OUT. Latches value into TBCCP1 at fall of TA5OUT.

Software capture

0	Latches up-counter value into TBCCP0.
1	Don't care

TMRBD mode register

	7	6	5	4	3	2	1	0
Bit symbol	—	—	TBDCP0	TBDCPM1	TBDCPM0	TBDCLE	TBDCLK1	TBDCLK0
Read/Write			W*	R/W				
After reset	0	0	1	0	0	0	0	0
Function	Must always be set to 00.		Software capture control 0: Software capture 1: Don't care	Capture timing 00: Disabled 01: Disabled 10: Disabled 11: TA5OUT ↑ TA5OUT		Up-counter control 0: Clearing disabled 1: Clearing enabled	Source clock selection 00: Setting prohibited 01: φT1 10: φT4 11: φT16	

TBDMOD
(0xFFFF_F212)

Clearing up-counter (UC0)

0	Clearing of up-counter disabled
1	Up-counter cleared when it matches TBDRG1

Capture timing

00	Capture disabled
01	Capture disabled
10	Capture disabled
11	Latches value into TBDCP0 at rise of TA5OUT. Latches value into TBDCP1 at fall of TA5OUT.

Software capture

0	Latches up-counter value into TBDCP0.
1	Don't care

Figure 3.10.15 TMRB Registers

TMRB0 flip-flop control register

		7	6	5	4	3	2	1	0
TB0FFCR (0xFFFF_ F143)	Bit symbol	—	—	TB0C1T1	TB0C0T1	TB0E1T1	TB0E0T1	TB0FF0C1	TB0FF0C0
	Read/Write	W*		R/W				W*	
	After reset	1	1	0	0	0	0	1	1
	Function	Must always be set to 11. * These bits are always 11 when read.		TB0FF0 inversion trigger 0: Trigger disabled 1: Trigger enabled				TB0FF0 control 00: Invert 01: Set 10: Clear 11: Don't care * These bits are always 11 when read.	
				When up-counter value is latched into TB0CP1	When up-counter value is latched into TB0CP0	When up-counter and TB0RG1 values match	When up-counter and TB0RG0 values match		

Control of timer flip-flop (TB0FF0)

00	TB0FF0 value inverted (soft inversion)
01	TB0FF0 set to 1
10	TB0FF0 set to 0
11	Don't care (read as 11)

Trigger for inverting timer flip-flop (TB0FF0) when up-counter and TB0RG0 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB0FF0) when up-counter and TB0RG1 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB0FF0) when up-counter value is latched into TB0CP0

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB0FF0) when up-counter value is latched into TB0CP1

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Figure 3.10.16 TMRB Registers

TMRB1 flip-flop control register

		7	6	5	4	3	2	1	0
TB1FFCR (0xFFFF_ F153)	Bit symbol	—	—	TB1C1T1	TB1C0T1	TB1E1T1	TB1E0T1	TB1FF0C1	TB1FF0C0
	Read/Write	W*		R/W				W*	
	After reset	1	1	0	0	0	0	1	1
	Function	Must always be set to 11. * These bits are always 11 when read.		TB1FF0 inversion trigger 0: Trigger disabled 1: Trigger enabled				TB1FF0 control 00: Invert 01: Set 10: Clear 11: Don't care * These bits are always 11 when read.	
				When up-counter value is latched into TB1CP1	When up-counter value is latched into TB1CP0	When up-counter and TB1RG1 values match	When up-counter and TB1RG0 values match		

Control of timer flip-flop (TB1FF0)

00	TB1FF0 value inverted (soft inversion)
01	TB1FF0 set to 1
10	TB1FF0 set to 0
11	Don't care (read as 11)

Trigger for inverting timer flip-flop (TB1FF0) when up-counter and TB1RG0 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB1FF0) when up-counter and TB1RG1 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB1FF0) when up-counter value is latched into TB1CP0

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB1FF0) when up-counter value is latched into TB1CP1

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Figure 3.10.17 TMRB Registers

TMRB2 flip-flop control register

		7	6	5	4	3	2	1	0
TB2FFCR (0xFFFF_ F163)	Bit symbol	—	—	TB2C1T1	TB2C0T1	TB2E1T1	TB2E0T1	TB2FF0C1	TB2FF0C0
	Read/Write	W*		R/W				W*	
	After reset	1	1	0	0	0	0	1	1
	Function	Must always be set to 11. * These bits are always 11 when read.		TB2FF0 inversion trigger 0: Trigger disabled 1: Trigger enabled				TB2FF0 control 00: Invert 01: Set 10: Clear 11: Don't care * These bits are always 11 when read.	
				When up-counter value is latched into TB2CP1	When up-counter value is latched into TB2CP0	When up-counter and TB2RG1 values match	When up-counter and TB2RG0 values match		

Control of timer flip-flop (TB2FF0)

00	TB2FF0 value inverted (soft inversion)
01	TB2FF0 set to 1
10	TB2FF0 set to 0
11	Don't care (read as 11)

Trigger for inverting timer flip-flop (TB2FF0) when up-counter and TB2RG0 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB2FF0) when up-counter and TB2RG1 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB2FF0) when up-counter value is latched into TB2CP0

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB2FF0) when up-counter value is latched into TB2CP1

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Figure 3.10.18 TMRB Registers

TMRB3 flip-flop control register

		7	6	5	4	3	2	1	0
TB3FFCR (0xFFFF_ F173)	Bit symbol	—	—	TB3C1T1	TB3C0T1	TB3E1T1	TB3E0T1	TB3FF0C1	TB3FF0C0
	Read/Write	W*		R/W				W*	
	After reset	1	1	0	0	0	0	1	1
	Function	Must always be set to 11. * These bits are always 11 when read.		TB3FF0 inversion trigger 0: Trigger disabled 1: Trigger enabled				TB3FF0 control 00: Invert 01: Set 10: Clear 11: Don't care * These bits are always 11 when read.	
				When up-counter value is latched into TB3CP1	When up-counter value is latched into TB3CP0	When up-counter and TB3RG1 values match	When up-counter and TB3RG0 values match		

Control of timer flip-flop (TB3FF0)

00	TB3FF0 value inverted (soft inversion)
01	TB3FF0 set to 1
10	TB3FF0 set to 0
11	Don't care (read as 11)

Trigger for inverting timer flip-flop (TB3FF0) when up-counter and TB3RG0 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB3FF0) when up-counter and TB3RG1 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB3FF0) when up-counter value is latched into TB3CP0

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB3FF0) when up-counter value is latched into TB3CP1

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Figure 3.10.19 TMRB Registers

TMRB4 flip-flop control register

TB4FFCR (0xFFFF_ F183)		7	6	5	4	3	2	1	0
	Bit symbol	—	—	TB4C1T1	TB4C0T1	TB4E1T1	TB4E0T1	TB4FF0C1	TB4FF0C0
	Read/Write	W*		R/W				W*	
	After reset	1	1	0	0	0	0	1	1
	Function	Must always be set to 11. * These bits are always 11 when read.		TB4FF0 inversion trigger 0: Trigger disabled 1: Trigger enabled				TB4FF0 control 00: Invert 01: Set 10: Clear 11: Don't care * These bits are always 11 when read.	
				When up-counter value is latched into TB4CP1	When up-counter value is latched into TB4CP0	When up-counter and TB4RG1 values match	When up-counter and TB4RG0 values match		

Control of timer flip-flop (TB4FF0)

00	TB4FF0 value inverted (soft inversion)
01	TB4FF0 set to 1
10	TB4FF0 set to 0
11	Don't care (read as 11)

Trigger for inverting timer flip-flop (TB4FF0) when up-counter and TB4RG0 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB4FF0) when up-counter and TB4RG1 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB4FF0) when up-counter value is latched into TB4CP0

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB4FF0) when up-counter value is latched into TB4CP1

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Figure 3.10.20 TMRB Registers

TMRB5 flip-flop control register

		7	6	5	4	3	2	1	0
TB5FFCR (0xFFFF_ F193)	Bit symbol	—	—	TB5C1T1	TB5C0T1	TB5E1T1	TB5E0T1	TB5FF0C1	TB5FF0C0
	Read/Write	W*		R/W				W*	
	After reset	1	1	0	0	0	0	1	1
	Function	Must always be set to 11. * These bits are always 11 when read.		TB5FF0 inversion trigger 0: Trigger disabled 1: Trigger enabled When up-counter value is latched into TB5CP1	When up-counter value is latched into TB5CP0	When up-counter and TB5RG1 values match	When up-counter and TB5RG0 values match	TB5FF0 control 00: Invert 01: Set 10: Clear 11: Don't care * These bits are always 11 when read.	

Control of timer flip-flop (TB5FF0)

00	TB5FF0 value inverted (soft inversion)
01	TB5FF0 set to 1
10	TB5FF0 set to 0
11	Don't care (read as 11)

Trigger for inverting timer flip-flop (TB5FF0) when up-counter and TB5RG0 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB5FF0) when up-counter and TB5RG1 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB5FF0) when up-counter value is latched into TB5CP0

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB5FF0) when up-counter value is latched into TB5CP1

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Figure 3.10.21 TMRB Registers

TMRB6 flip-flop control register

		7	6	5	4	3	2	1	0
TB6FFCR (0xFFFF_ F1A3)	Bit symbol	—	—	TB6C1T1	TB6C0T1	TB6E1T1	TB6E0T1	TB6FF0C1	TB6FF0C0
	Read/Write	W*		R/W				W*	
	After reset	1	1	0	0	0	0	1	1
	Function	Must always be set to 11. * These bits are always 11 when read.		TB6FF0 inversion trigger 0: Trigger disabled 1: Trigger enabled When up-counter value is latched into TB6CP1	When up-counter value is latched into TB6CP0	When up-counter and TB6RG1 values match	When up-counter and TB6RG0 values match	TB6FF0 control 00: Invert 01: Set 10: Clear 11: Don't care * These bits are always 11 when read.	

Control of timer flip-flop (TB6FF0)

00	TB6FF0 value inverted (soft inversion)
01	TB6FF0 set to 1
10	TB6FF0 set to 0
11	Don't care (read as 11)

Trigger for inverting timer flip-flop (TB6FF0) when up-counter and TB6RG0 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB6FF0) when up-counter and TB6RG1 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB6FF0) when up-counter value is latched into TB6CP0

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB6FF0) when up-counter value is latched into TB6CP1

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Figure 3.10.22 TMRB Registers

TMRB6 flip-flop control register

		7	6	5	4	3	2	1	0
TB6FFCR (0xFFFF_ F1A3)	Bit symbol	—	—	TB6C1T1	TB6C0T1	TB6E1T1	TB6E0T1	TB6FF0C1	TB6FF0C0
	Read/Write	W*		R/W				W*	
	After reset	1	1	0	0	0	0	1	1
	Function	Must always be set to 11. * These bits are always 11 when read.		TB6FF0 inversion trigger 0: Trigger disabled 1: Trigger enabled When up-counter value is latched into TB6CP1	When up-counter value is latched into TB6CP0	When up-counter and TB6RG1 values match	When up-counter and TB6RG0 values match	TB6FF0 control 00: Invert 01: Set 10: Clear 11: Don't care * These bits are always 11 when read.	

Control of timer flip-flop (TB6FF0)

00	TB6FF0 value inverted (soft inversion)
01	TB6FF0 set to 1
10	TB6FF0 set to 0
11	Don't care (read as 11)

Trigger for inverting timer flip-flop (TB6FF0) when up-counter and TB6RG0 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB6FF0) when up-counter and TB6RG1 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB6FF0) when up-counter value is latched into TB6CP0

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB6FF0) when up-counter value is latched into TB6CP1

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Figure 3.10.22 TMRB Registers

TMRB7 flip-flop control register

		7	6	5	4	3	2	1	0
TB7FFCR (0xFFFF_F1B3)	Bit symbol	—	—	TB7C1T1	TB7C0T1	TB7E1T1	TB7E0T1	TB7FF0C1	TB7FF0C0
	Read/Write	W*		R/W				W*	
	After reset	1	1	0	0	0	0	1	1
	Function	Must always be set to 11. * These bits are always 11 when read.		TB7FF0 inversion trigger 0: Trigger disabled 1: Trigger enabled When up-counter value is latched into TB7CP1	When up-counter value is latched into TB7CP0	When up-counter and TB7RG1 values match	When up-counter and TB7RG0 values match	TB7FF0 control 00: Invert 01: Set 10: Clear 11: Don't care * These bits are always 11 when read.	

Control of timer flip-flop (TB7FF0)

00	TB7FF0 value inverted (soft inversion)
01	TB7FF0 set to 1
10	TB7FF0 set to 0
11	Don't care (read as 11)

Trigger for inverting timer flip-flop (TB7FF0) when up-counter and TB7RG0 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB7FF0) when up-counter and TB7RG1 values match

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB7FF0) when up-counter value is latched into TB7CP0

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Trigger for inverting timer flip-flop (TB7FF0) when up-counter value is latched into TB7CP1

0	Trigger disabled (inversion disabled)
1	Trigger enabled (inversion enabled)

Figure 3.10.23 TMRB Registers

TMRB0 status register

		7	6	5	4	3	2	1	0
TB0ST (0xFFFF_ F144)	Bit symbol	—	—	—	—	—	INTTBOF0	INTTB01	INTTB00
	Read/Write	—	—	—	—	—	R		
	After reset	—	—	—	—	—	0	0	0
	Function						0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated

INTTB00: Interrupt generated upon detection of match with timer register TB0RG0

INTTB01: Interrupt generated upon detection of match with timer register TB0RG1

INTTBOF0: Interrupt generated upon detection of up-counter overflow

TMRB1 status register

		7	6	5	4	3	2	1	0
TB1ST (0xFFFF_ F154)	Bit symbol	—	—	—	—	—	INTTBOF1	INTTB11	INTTB10
	Read/Write	—	—	—	—	—	R		
	After reset	—	—	—	—	—	0	0	0
	Function						0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated

INTTB10: Interrupt generated upon detection of match with timer register TB1RG0

INTTB11: Interrupt generated upon detection of match with timer register TB1RG1

INTTBOF1: Interrupt generated upon detection of up-counter overflow

Figure 3.10.1 TMRB Registers

TMRB2 status register

a. When TB2RUN<TB2UDCE> = 0: Normal timer mode

		7	6	5	4	3	2	1	0
TB2ST (0xFFFF_ F164)	Bit symbol	—	—	—	—	—	INTTBOF2	INTTB21	INTTB20
	Read/Write	—	—	—	—	—	R		
	After reset	—	—	—	—	—	0	0	0
	Function						0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated

INTTB20: Interrupt generated upon detection of match with timer register TB2RG0

INTTB21: Interrupt generated upon detection of match with timer register TB2RG1

INTTBOF2: Interrupt generated upon detection of up-counter overflow

b. When TB2RUN<TB2UDCE> = 1: 2-phase pulse input counter mode

		7	6	5	4	3	2	1	0
TB2ST (0xFFFF_ F164)	Bit symbol	—	—	—	INTTBUD2	INTTBUDF2	INTTBOUF2	—	—
	Read/Write	—	—	—	R			—	—
	After reset	—	—	—	0	0	0	—	—
	Function				Up or down count 0: Not detected 1: Detected	Underflow 0: Not detected 1: Detected	Overflow 0: Not detected 1: Detected		

INTTBUDF2: Interrupt generated upon detection of up-down counter underflow

INTTBOVF2: Interrupt generated upon detection of up-down counter overflow

INTTBUD2: Interrupt generated upon detection of up-down counter increment or decrement

Figure 3.10.25 TMRB Registers

TMRB3 status register

a.   When TB3RUN<TB3UDCE> = 0: Normal timer mode

		7	6	5	4	3	2	1	0
TB3ST (0xFFFF_ F174)	Bit symbol	—	—	—	—	—	INTTBOF3	INTTB31	INTTB30
	Read/Write	—	—	—	—	—	R		
	After reset	—	—	—	—	—	0	0	0
	Function						0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated

INTTB30: Interrupt generated upon detection of match with timer register TB3RG0
INTTB31: Interrupt generated upon detection of match with timer register TB3RG1
INTTBOF3: Interrupt generated upon detection of up-counter overflow

b.   When TB3RUN<TB3UDCE> = 1: 2-phase pulse input counter mode

		7	6	5	4	3	2	1	0
TB3ST (0xFFFF_ F174)	Bit symbol	—	—	—	INTTBUD3	INTTBUDF3	INTTBOUF3	—	—
	Read/Write	—	—	—	R			—	—
	After reset	—	—	—	0	0	0	—	—
	Function				Up or down count 0: Not detected 1: Detected	Underflow 0: Not detected 1: Detected	Overflow 0: Not detected 1: Detected		

INTTBUDF3: Interrupt generated upon detection of up-down counter underflow
INTTBOVF3: Interrupt generated upon detection of up-down counter overflow
INTTBUD3: Interrupt generated upon detection of up-down counter increment or decrement

TMRB4 status register

		7	6	5	4	3	2	1	0
TB4ST (0xFFFF_ F184)	Bit symbol	—	—	—	—	—	INTTBOF4	INTTB41	INTTB40
	Read/Write	—	—	—	—	—	R		
	After reset	—	—	—	—	—	0	0	0
	Function						0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated

INTTB40: Interrupt generated upon detection of match with timer register TB4RG0
INTTB41: Interrupt generated upon detection of match with timer register TB4RG1
INTTBOF4: Interrupt generated upon detection of up-counter overflow

Figure 3.10.26 TMRB Registers

TMRB5 status register

		7	6	5	4	3	2	1	0
TB5ST (0xFFFF_F194)	Bit symbol	—	—	—	—	—	INTTBOF4	INTTB41	INTTB40
	Read/Write	—	—	—	—	—	R		
	After reset	—	—	—	—	—	0	0	0
	Function						0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated

INTTB50: Interrupt generated upon detection of match with timer register TB5RG0

INTTB51: Interrupt generated upon detection of match with timer register TB5RG1

INTTBOF5: Interrupt generated upon detection of up-counter overflow

TMRB6 status register

		7	6	5	4	3	2	1	0
TB6ST (0xFFFF_F1A4)	Bit symbol	—	—	—	—	—	INTTBOF6	INTTB61	INTTB60
	Read/Write	—	—	—	—	—	R		
	After reset	—	—	—	—	—	0	0	0
	Function						0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated

INTTB60: Interrupt generated upon detection of match with timer register TB6RG0

INTTB61: Interrupt generated upon detection of match with timer register TB6RG1

INTTBOF6: Interrupt generated upon detection of up-counter overflow

TMRB7 status register

		7	6	5	4	3	2	1	0
TB7ST (0xFFFF_F1B4)	Bit symbol	—	—	—	—	—	INTTBOF7	INTTB71	INTTB70
	Read/Write	—	—	—	—	—	R		
	After reset	—	—	—	—	—	0	0	0
	Function						0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated

INTTB70: Interrupt generated upon detection of match with timer register TB7RG0

INTTB71: Interrupt generated upon detection of match with timer register TB7RG1

INTTBOF7: Interrupt generated upon detection of up-counter overflow

Figure 3.10.27 TMRB Registers

TMRB8 status register

TB8ST (0xFFFF_ F1C4)		7	6	5	4	3	2	1	0
	Bit symbol	—	—	—	—	—	INTTBOF8	INTTB81	INTTB80
	Read/Write	—	—	—	—	—	R		
	After reset	—	—	—	—	—	0	0	0
	Function						0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated

INTTB80:   Interrupt generated upon detection of match with timer register TB8RG0

INTTB81:   Interrupt generated upon detection of match with timer register TB8RG1

INTTBOF8: Interrupt generated upon detection of up-counter overflow

TMRB9 status register

TB9ST (0xFFFF_ F1D4)		7	6	5	4	3	2	1	0
	Bit symbol	—	—	—	—	—	INTTBOF9	INTTB91	INTTB90
	Read/Write	—	—	—	—	—	R		
	After reset	—	—	—	—	—	0	0	0
	Function						0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated

INTTB90:   Interrupt generated upon detection of match with timer register TB9RG0

INTTB91:   Interrupt generated upon detection of match with timer register TB9RG1

INTTBOF9: Interrupt generated upon detection of up-counter overflow

TMRBA status register

TBAST (0xFFFF_ F1E4)		7	6	5	4	3	2	1	0
	Bit symbol	—	—	—	—	—	INTTBOFA	INTTBA1	INTTBA0
	Read/Write	—	—	—	—	—	R		
	After reset	—	—	—	—	—	0	0	0
	Function						0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated

INTTBA0:   Interrupt generated upon detection of match with timer register TBARG0

INTTBA1:   Interrupt generated upon detection of match with timer register TBARG1

INTTBOFA: Interrupt generated upon detection of up-counter overflow

Figure 3.10.28 TMRB Registers

TMRBB status register

		7	6	5	4	3	2	1	0
TBBST (0xFFFF_ F1F4)	Bit symbol	—	—	—	—	—	INTTBOFB	INTTBB1	INTTBB0
	Read/Write	—	—	—	—	—	R		
	After reset	—	—	—	—	—	0	0	0
	Function						0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated

INTTBB0: Interrupt generated upon detection of match with timer register TBBRG0

INTTBB1: Interrupt generated upon detection of match with timer register TBBRG1

INTTBOFB: Interrupt generated upon detection of up-counter overflow

TMRBC status register

		7	6	5	4	3	2	1	0
TBCST (0xFFFF_ F204)	Bit symbol	—	—	—	—	—	INTTBOF9	INTTB91	INTTB90
	Read/Write	—	—	—	—	—	R		
	After reset	—	—	—	—	—	0	0	0
	Function						0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated

INTTBC0: Interrupt generated upon detection of match with timer register TBCRG0

INTTBC1: Interrupt generated upon detection of match with timer register TBCRG1

INTTBOFC: Interrupt generated upon detection of up-counter overflow

TMRBD status register

		7	6	5	4	3	2	1	0
TBDST (0xFFFF_ F214)	Bit symbol	—	—	—	—	—	INTTBOFD	INTTBD1	INTTBD0
	Read/Write	—	—	—	—	—	R		
	After reset	—	—	—	—	—	0	0	0
	Function						0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated

INTTBD0: Interrupt generated upon detection of match with timer register TBDRG0

INTTBD1: Interrupt generated upon detection of match with timer register TBDRG1

INTTBOFD: Interrupt generated upon detection of up-counter overflow

Figure 3.10.29 TMRB Registers

3.10.4   Functional description for each mode

(1)   16-bit interval timer mode

To generate interrupts at certain intervals, set the interval time in the timer register TB0RG1 and enable INTTB01 interrupts.

```
              7  6  5  4  3  2  1  0
TB0RUN    ←   0  0  X  X  −  0  X  0        Stop TMRB0.

IMC7LH    ←   X  X  1  1  0  1  0  0        Enable INTTB0 and set its priority level to 4.
TB0FFCR   ←   1  1  0  0  0  0  1  1        Disable trigger.
TB0MOD    ←   0  0  1  0  0  1  *  *        Select prescaler output clock as input clock and disable
              (** = 01, 10, 11)            capture function.
TB0RG1    ←   *  *  *  *  *  *  *  *        Set interval time.
              *  *  *  *  *  *  *  *        (16 bits)
TB0RUN    ←   0  0  X  X  −  1  X  1        Start TMRB0.
```

Note: X = Don't care; "−" = No change

---

(2)   16-bit event counter mode

The timer can be used as an event counter by selecting an external clock (input to the TB0IN0 pin) as its input clock.

The up-counter is incremented on each rising edge of the TB0IN0 pin input. The count value can be read by capturing it by software and reading the captured value.

```
              7  6  5  4  3  2  1  0
TB0RUN    ←   0  0  X  X  −  0  X  0  ⎫     Stop TMRB0.
PACR      ←   −  −  −  −  −  −  −  0  ⎬     Set PA0 to input mode.
PAFC      ←   −  −  −  −  −  −  −  1  ⎭

IMC7LH    ←   X  X  1  1  0  1  0  0        Enable INTTB0 and set its priority level to 4.
TB0FFCR   ←   1  1  0  0  0  0  1  1        Disable trigger.
TB0MOD    ←   0  0  1  0  0  1  0  0        Select TB0IN0 pin input as input clock.
TB0RG1    ←   *  *  *  *  *  *  *  *        Set count (16 bits).
TB0RUN    ←   0  0  X  X  −  1  X  1        Start TMRB0.
```

Note: X = Don't care; "−" = No change
When the timer is used as an event counter, the prescaler must be set to run (i.e. TB0RUN<TB0PRUN> must be set to 1).

---

(3) 16-bit PPG (programmable square wave) output mode

A square wave of any frequency with any duty cycle (programmable square wave) can be output. Either Low-active or High-active output pulses can be selected.

This mode is used to output a programmable square wave on the TB0OUT pin by triggering inversion of the timer flip-flop (TB0FF) when the values in the up-counter (UC0) and one of the timer registers (TB0RG0 or TB0RG1) match. However, the values set in TB0RG0 and TB0RG1 must satisfy the following condition:

(TB0RG0 set value) < (TB0RG1 set value)
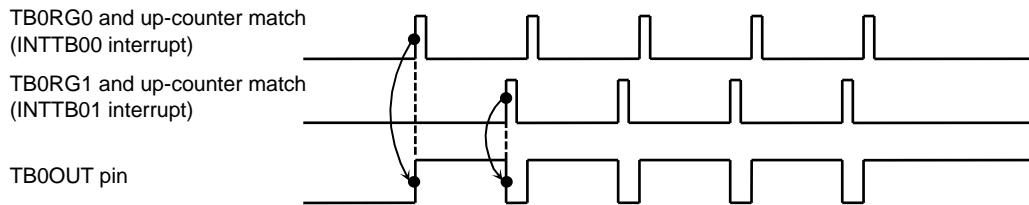
Figure 3.10.30 Example PPG Output Waveform

If TB0RG0 has its double-buffer enabled in this mode, the value in register buffer 0 is shifted into TB0RG0 when TB0RG1 and UC0 match. Using the double-buffer facilitates satisfying the requirements for small duty cycle waveforms.
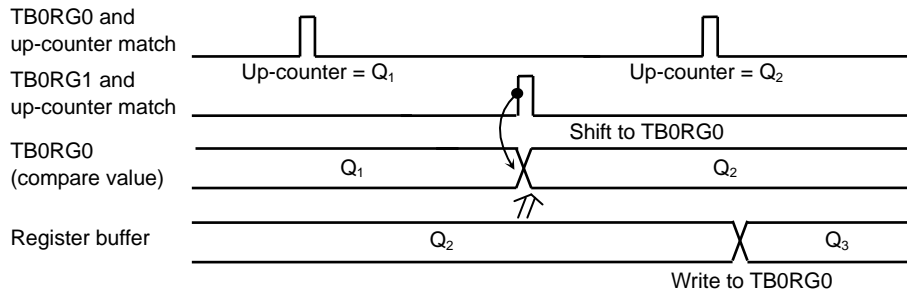
Figure 3.10.31 Register Buffer Operation

Figure 3.10.32 shows a block diagram of 16-bit PPG output mode.
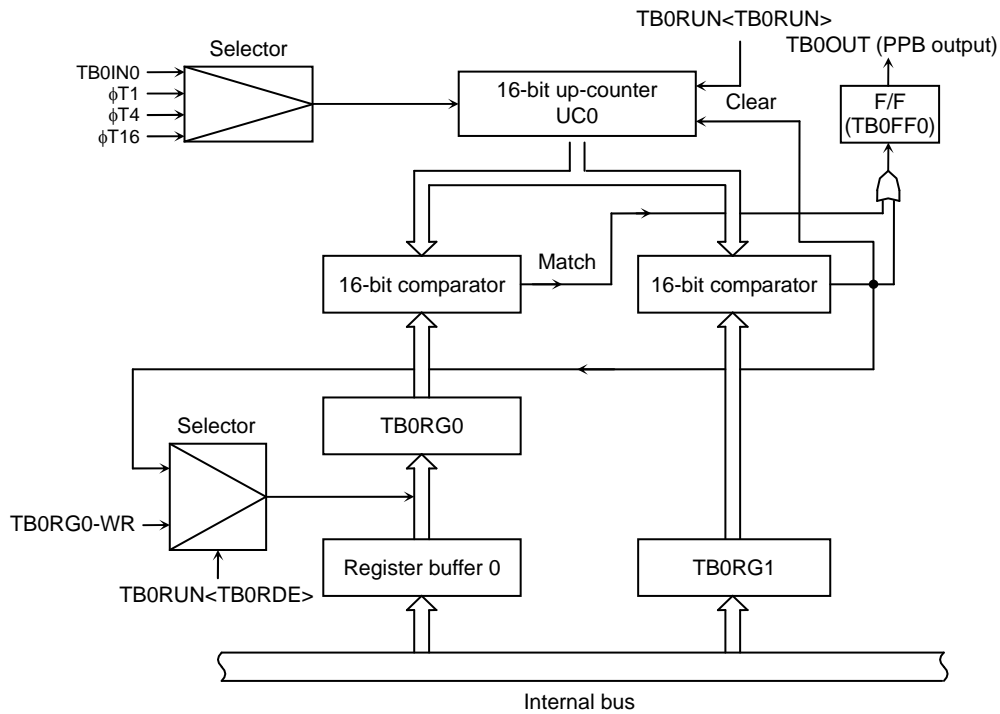
Figure 3.10.32 16-Bit PPG Output Mode Block Diagram

To use the timer in 16-bit PPG output mode, set each register as shown below.

```
              7   6   5   4   3   2   1   0
TB0RUN    ←   0   0   X   X   –   0   X   0     Disable TB0RG0 double-buffer and stop TMRB0.
TB0RG0    ←   *   *   *   *   *   *   *   *     Set duty cycle (16 bits).
TB0RG1    ←   *   *   *   *   *   *   *   *     Set PPG cycle (16 bits).
TB0RUN    ←   1   0   X   X   –   0   X   0     Enable TB0RG0 double-buffer.
                                               (Duty cycle and PPG cycle are changed by INTTB01 interrupt.)
TB0FFCR   ←   X   X   0   0   1   1   1   0     Set TB0FF0 so that its value will be inverted on detecting a match
                                               with TB0RG0 or TB0RG1. Also initialize TB0FF0 to 0.
TB0MOD    ←   0   0   1   0   0   1   *   *     Select prescaler output clock as input clock and disable capture
                      (** = 01, 10, 11)        function.
P7CR      ←   –   1   –   –   –   –   –   –  ⎫  Set PA2 pin to TB0OUT.
P7FC      ←   –   1   –   –   –   –   –   –  ⎭
TB0RUN    ←   1   0   X   X   –   1   X   1     Start TMRB0.
```

Note1: X = Don't care; "–" = No change
Note2: Please do not stop a timer in PPG mode at the time of duty change (please use a double buffer).
   In order to change a timer-related setup, please perform the following setup, when you resume
   after a stop (<TBnRUN>="0") (<TBnRUN>="1").
(1) Change a timer output terminal to a PORT function.
(2) Timer stop (<TBnRUN>="0")
(3) Forbid a reversal output at the time of coincidence with a Up Counter and a Timer Register
   (TBnFFCR<TBnE1T1:TBnE0T1>=00).
(4) Start a timer (<TBnRUN>="1").
(5) Suspend a timer (<TBnRUN>="0").
(6) A setup of the contents of change.
(7) Change an output terminal from a PORT function to a timer output.
(8) Start a timer (<TBnRUN>="1").

(4) Application examples using the capture function

With its capture function enabled, TMRB can be used for various applications including those presented in the examples given below:

a. One-shot pulse output using an external trigger pulse

b. Frequency measurement

c. Pulse width measurement

d. Time difference measurement

a. One-shot pulse output using an external trigger pulse

To output a one-shot pulse using an external trigger pulse, follow the procedure described below.

Let the 16-bit up-counter UC0 count up in free-running mode using the prescaler output clock. Enter an external trigger pulse via the TB0IN0 pin and use the capture function to latch the up-counter value into the capture register (TB0CP0) at the rising edge of the external trigger pulse.

Set INTC so that an INT3 interrupt is generated when the external trigger pulse goes High. During this interrupt write the sum of the TB0CP0 value (c) and the delay time (d) to the timer register TB0RG0. Similarly, write the sum of the TB0RG0 value and the one-shot pulse width (p), i.e. (c + d + p), to the timer register TB0RG1.

Then, set the relevant field in the timer flip-flop control register (TB0FFCR <TB0E1T1,TB0E0T1>) to 11, enabling the trigger so that the timer flip-flop (TB0FF0) will be inverted on detection of a match between the value of UC0 and the value of TB0RG0 or TB0RG1. After a one-shot pulse is output, disable inversion during INTTB0 interrupt handling.

The terms (c), (d) and (p) in the above explanation correspond to c, d and p in Figure 3.10., One-Shot Pulse Output (with Delay).
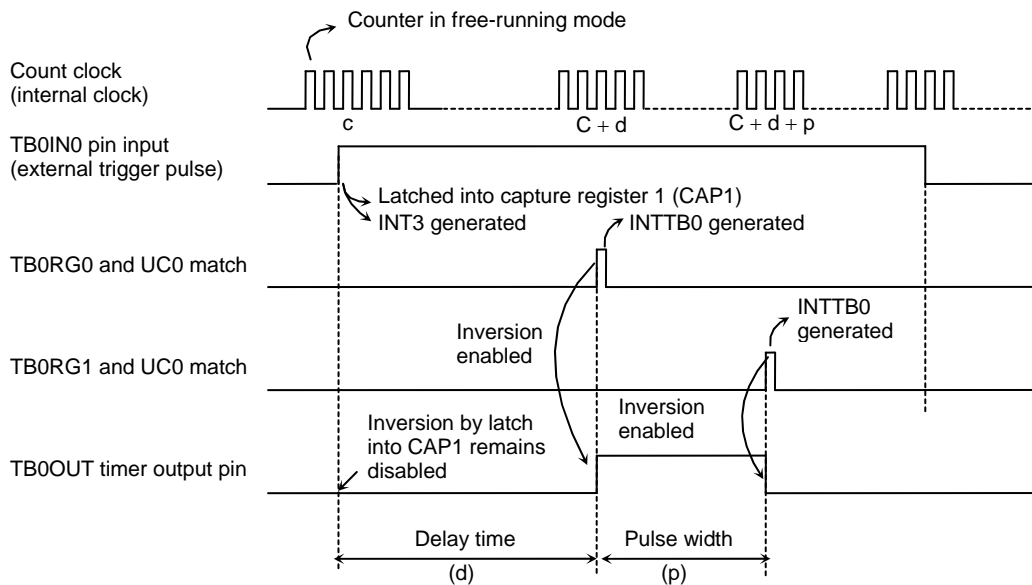
Figure 3.10.33 One-Shot Pulse Output (with Delay)

Set-up example: To output a 2 ms one-shot pulse with a delay time of 3 ms after an external trigger pulse
on the TB0IN0 pin

∗ Clock conditions

System clock: High-speed (fc)
High-speed clock gear: × 1 (fc)
Prescaler clock: fperiph/4 (fperiph = fsys)

Settings in the main routine

```
              7  6  5  4  3  2  1  0
```
→ Place counter in free-running mode.

→ Use φT1 as clock source for counting.

```
TB0MOD    ←  X  X  1  0  1  0  0  1
```

→ Latch count into TB0CP0 on rise of TB0IN0 input.

```
TB0FFCR   ←  X  X  0  0  0  0  1  0
```

→ Clear TB0FF0 to 0.

→ Disable inversion of TB0FF0.

```
PACR      ←  -  -  -  -  -  1  -  -
PAFC      ←  -  -  -  -  -  1  -  -
```
Set PA2 pin to TB0OUT.

```
IMC0HL    ←  X  X  1  1  0  1  0  0
```

Enable INT3 and disable INTTB0

```
IMC7LH    ←  X  X  1  1  0  0  0  0
TB0RUN    ←  -  0  X  X  -  1  X  1
```
Start TMRB0.

Settings in INT3

```
TB0RG0    ←  TB0CP0 + 3ms/φT1
TB0RG1    ←  TB0RG0 + 2ms/φT1
TB0FFCR   ←  X  X  -  -  1  1  -  -
```

→ Enable TB0FF0 inversion when up-counter value matches
TB0RG0 or TB0RG1.

```
IMC7LH    ←  X  X  1  1  0  1  0  0
```
Enable INTTB0.

Settings in INTTB0

```
TB0FFCR   ←  X  X  -  -  0  0  -  -
```

→ Disable TB0FF0 inversion when up-counter value
matches TB0RG0 or TB0RG1.

```
IMC7LH    ←  X  X  1  1  0  0  0  0
```
Disable INTTB0.

Note: X = Don't care; "–" = No change

If a delay is not necessary, invert TB0FF0 by latching the counter value into TB0CP0; then,
during the INT3 interrupt, write the sum of the TB0CP0 value (c) and the one-shot pulse width
(p) to TB0RG1. Enable the trigger so that TB0FF0 will be inverted on detection of a match
between the value of UC0 and the value of TB0RG1. TB0FF0 inversion should be disabled
during INTTB0 interrupt handling.
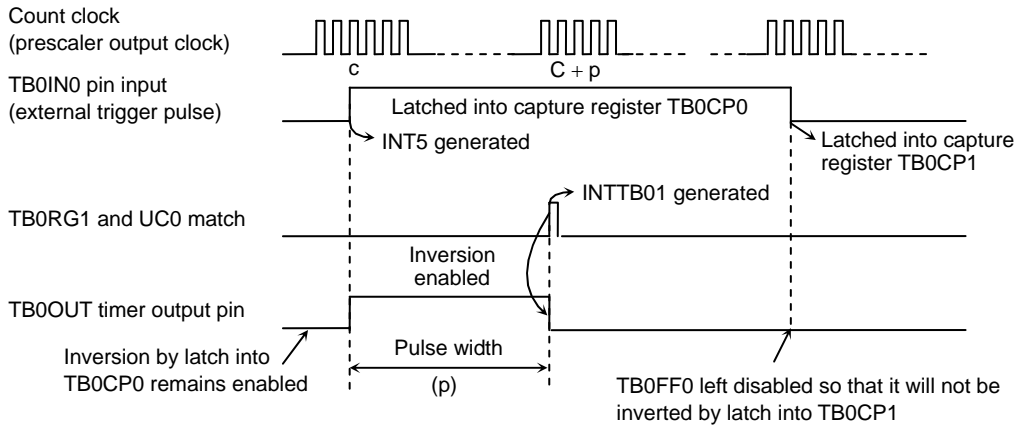
Figure 3.10.34 One-Shot Pulse Output Using an External Trigger Pulse (without Delay)

b. Frequency measurement

With its capture function enabled, the timer can be used to measure the frequency of an external clock.

The frequency is measured using a combination of a 16-bit timer/event counter and 8-bit timers (TMRA01). (TMRA01 determines the measurement time by inverting TA1FF.)

Select TB0IN0 pin input as the count clock for TMRB0 so that it counts up synchronously with the external clock pulses. Set TB0MOD<TB0CPM1:TB0CPM0> to 11. This setting causes the count value of the 16-bit up-counter UC0 to be latched into the capture register TB0CP0 when the 8-bit timer (TMRA01) flip-flop (TA1FF) output goes High, and to be latched into the capture register TB0CP1 when the TA1FF output goes Low.

The frequency is calculated from the difference between the loaded values in TB0CP0 and TB0CP1 based on the measurement time determined by an 8-bit timer interrupt INTTA0 or INTTA1.
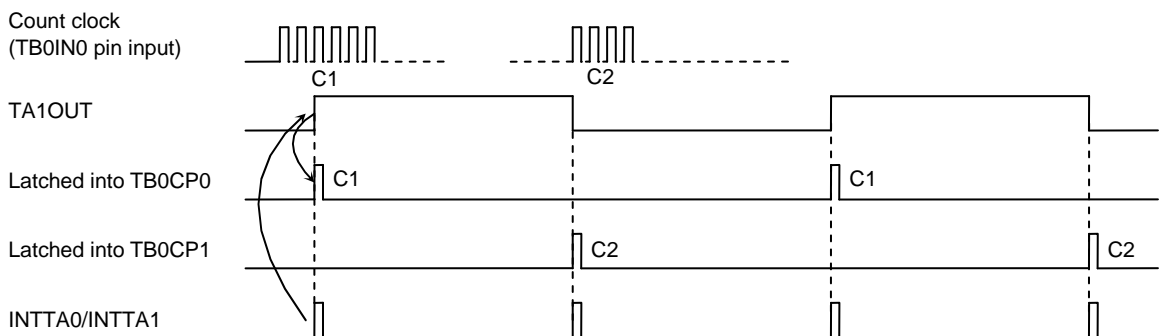
Figure 3.10.35 Frequency Measurement

For example, if the 8-bit timers set the High level width of TA1FF to 0.5 s and the difference between TB0CP0 and TB0CP1 is 100, then the frequency is 100/0.5 s = 200 Hz.

c. Pulse width measurement

With its capture function enabled, the timer can be used to measure the high-level duration of an external pulse. Enter an external pulse via the TB0IN0 pin and let the up-counter (UC0) count up in free-running mode using the prescaler output clock. Then, using the capture function, latch the up-counter value into the capture registers TB0CP0 and TB0CP1 on the rising and falling edges of the external pulse, respectively. Set INTC so that INT5 is generated when the TB0IN0 pin goes Low.

The High-level duration of the pulse can be obtained by finding the difference between TB0CP0 and TB0CP1 and multiplying the resulting value by the internal clock period.

For example, if the difference between TB0CP0 and TB0CP1 is 100 and the prescaler output clock period is 0.5 μs, then the pulse width will be $100 \times 0.5\ \mu s = 50\ \mu s$.

Additionally, the pulse width which exceeds the UC0 maximum count time specified by the clock source can be measured by software coding.
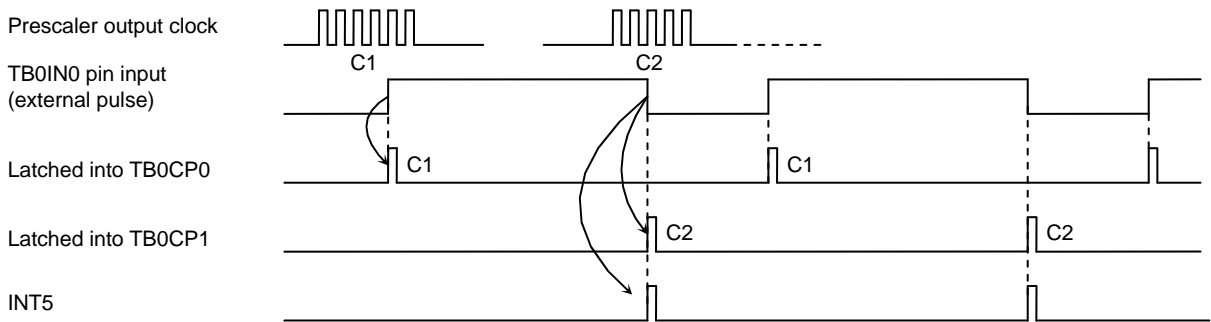
Figure 3.10.36 Pulse Width Measurement

The Low-level duration can be measured using the time difference measurement function shown in Figure 3.10.. The Low-level duration is obtained by multiplying the difference between the first C2 and the second C1 by the prescaler output clock period during the handling of the second INT5 interrupt.

d. Time difference measurement

With its capture function enabled, the timer can be used to measure the difference in time between two events. Let the up-counter (UC0) count up in free-running mode using the prescaler output clock. Latch the UC0 value into the capture register TB0CP0 at the rising edge of the pulse input on the TB0IN0 pin. Set INTC so that an INT3 interrupt is generated at that point.

Latch the UC0 value into the capture register (TB0CP1) at the rising edge of the pulse input on the TB0IN1 pin. Set INTC so that an INT4 interrupt is generated at that point.

The difference in time can be obtained by subtracting the value in TB0CP0 from the value in TB0CP1 after the values have been latched into the capture registers, and then multiplying the difference by the internal clock period.
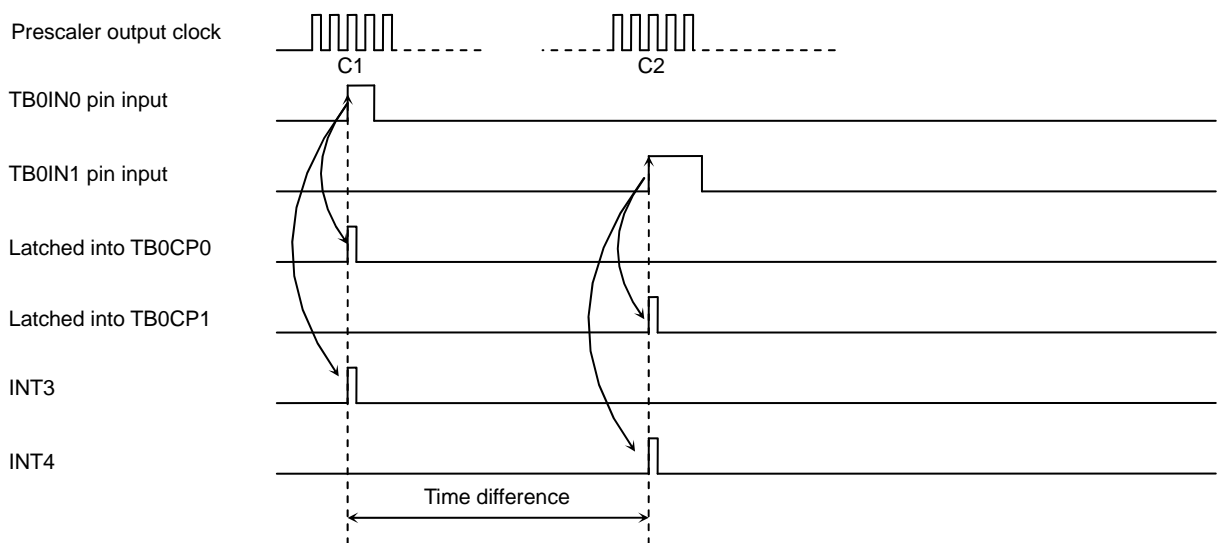
Figure 3.10.37 Time Difference Measurement

(5) 2-phase pulse input counter mode (TMRB2 and TMRB3)

(The function operates in the same way for TMRB2 and TMRB3. Only TMRB2 is described here.)

In this mode, the counter is either incremented or decremented by one according to the state transition of 2-phase clock pulses, with a phase difference of 90 degrees, input from TB2IN0 and TB2IN1. An interrupt is generated when the up/down-counter overflows or underflows, or when it is incremented or decremented.

a. Count operation

- Counting up
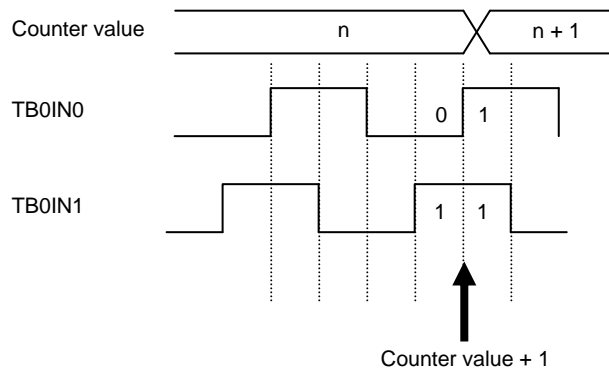
Figure 3.10.38 Counting Up

- Counting down
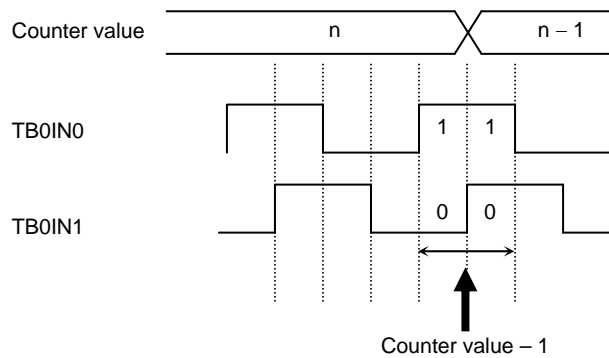
Figure 3.10.39 Counting Down

- Sampling clock

TMRB2 run register (TB2RUN)

	7	6	5	4	3	2	1	0
Bit symbol	TB2RDE	—	UD2CK	TB2UDCE	I2TB2	TB2PRUN	—	TB2RUN
Read/Write	R/W	—	R/W	R/W	R/W	R/W	—	R/W
After reset	0	—	0	0	0	0	—	0
Function	Double Buffer 0: Disable 1: Enable		Sampling clock selection 0: fs 1: fsys/2	2-phase counter enable 0: Disable 1: Enable	IDLE 0: Idle 1: Operate	Timer Run/Stop Control 0: Stop and cleared 1: Count		

Figure 3.10.40 Register for Setting 2-Phase Pulse Input Counter Mode

Bit 5 of the TB2RUN register (UD2CK) determines the sampling clock to be used.

UD2CK (sampling clock selection) = 0: Selects fs (32 kHz) (8 kHz sampling)

1: Selects fsys/2 (fsys/8 Hz sampling)

1)   Exiting from STOP mode

Because an 2-phase timer interrupt cannot be used to terminate STOP mode, an interrupt on the INTB or INTC shared pin is used to terminate STOP mode.

Terminate STOP/SLEEP mode using this path.
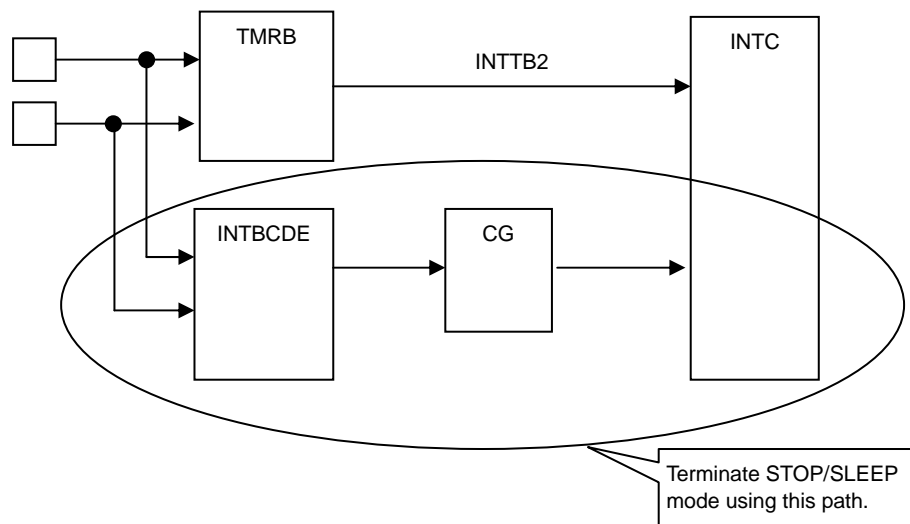
The 2-phase counter enters STOP mode while retaining its previous state. Therefore, if the relationship between the state of the input used to terminate STOP mode and the retained state satisfies the condition for counting up or down, the counter value is incremented or decremented after STOP mode has been terminated. (The counter value remains unchanged if the condition is not satisfied.) If it is necessary to obtain a constant counter state after exiting from STOP mode, initialize the 2-phase counter after STOP mode is terminated (clearing TB2RUN<TB2UDCE> to 0 and then setting it to 1 initializes the counter to 0x7FFF).

2)   Exiting from SLEEP mode

Because an 2-phase timer interrupt cannot be used to terminate SLEEP mode, an interrupt on the INTBCDE shared pin is used to terminate SLEEP mode. Whether the 2-phase counter is incremented or decremented depends on the state of the input used to terminate SLEEP mode. If it is necessary to obtain a constant counter state after exiting from SLEEP mode, initialize the 2-phase counter after SLEEP mode is terminated (clearing TB2RUN<TB2UDCE> to 0 and then setting it to 1 initializes the counter to 0x7FFF).

b.   Operating mode

Use appropriate register bits to determine whether the external input signals on the TB2IN0 and TB2IN1 input pins will be sent to the ordinary 16-bit timer or to the up/down-counter.

- In up/down-counter mode, only software capture is available; capture based on external clock timing is not enabled.

- In up/down-counter mode, the comparator is disabled; comparison with timer registers is not performed.

- In up/down-counter mode, ordinary INTB to INTE interrupts (interrupts other than those used to terminate STOP/SLEEP mode) cannot be used.

- The input clock signals are sampled at fs (32 kHz) or based on the high-speed clock (system clock). When fs is used, the maximum input frequency is 8 kHz. When the high-speed clock is used, the maximum input frequency is fsys/8 Hz.

Setting the up/down-counter

Set TB2MOD<TB2CLK0,TB2CLK1> to "00" (prescaler disabled). Next, set bit 4 of the TB2RUN register (TB2UDCE) to determine whether the counter should operate as a up/down counter or as an ordinary up-counter based on external clock input.

TB2UDCE (up/down-counter enable) = 0: Normal 16-bit timer operation

1: Up/down-counter operation

TMRB2 run register (TB2RUN)

	7	6	5	4	3	2	1	0
Bit symbol	TB2RDE	—	UD2CK	TB2UDCE	I2TB2	TB2PRUN	—	TB2RUN
Read/Write	R/W	—	R/W	R/W	R/W	R/W	—	R/W
After reset	0	—	0	0	0	0	—	0
Function	Double Buffer 0: Disable 1: Enable		Sampling clock selection 0: fs 1: fsys/2	2-phase counter enable 0: Disable 1: Enable	IDLE 0: Idle 1: Operate	Timer Run/Stop Control 0: Stop and cleared 1: Count		

Figure 3.10.41 Register for Setting the Up/Down-Counter

c. Interrupts

- In NORMAL or SLOW mode

Enable INTTB2 interrupts in the interrupt controller (INTC). An INTTB2 interrupt will occur when the counter either counts up or down. You can determine whether an overflow or underflow has occurred at the same time by reading the status register TB2ST during the handling of the interrupt. An overflow has occurred if TB2ST<INTTBOVF2> = 1. An underflow has occurred if TB2ST<INTTBUDF2> = 1. This register is cleared when read. An overflow causes the counter to be initialized to 0x0000 and an underflow causes the counter to be initialized to 0xFFFF, allowing the counter to continue counting.

		7	6	5	4	3	2	1	0
TB2ST (0xFFFF_ F164)	Bit symbol	—	—	—	INTTBUD2	INTTBUDF2	INTTBOUF2	—	—
	Read/Write	—	—	—	R			—	—
	After reset	—	—	—	0	0	0	—	—
	Function				Up or down count 0: Not detected 1: Detected	Underflow 0: Not detected 1: Detected	Overflow 0: Not detected 1: Detected		

Figure 3.10.42 TMRB2 Status Register

- In SLEEP mode

The 2-phase pulse input counter operates. Enable the INTBCDE release input in the clock generator (CG). Use INTnST of the INTBCDE circuit to set the active level for each interrupt input. An up or down counter input generates an INTB or INTC interrupt, causing the counter to exit from SLEEP mode. The interrupt source is determined by reading the flag register INTFLG. The flag is cleared when read. Whether this releasing interrupt source causes the counter to count up or down depends on whether the state of the releasing input satisfies the condition for counting up or down.

		7	6	5	4	3	2	1	0
INTFLG (0xFFFF_F384)	Bit symbol	—	—	—	—	INTES	INTDS	INTCS	INTBS
	Read/Write	—	—	—	—	R			
	After reset	—	—	—	—	0	0	0	0
	Function					0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated	0: No interrupt generated 1: Interrupt generated

Figure 3.10.43 INTFLG Register

- In STOP mode

The 2-phase pulse input counter stops. Enable the INTBCDE release input in the clock generator (CG). An up or down counter input generates an INTB or INTC interrupt, causing the counter to exit from STOP mode. Whether this releasing input causes the counter to count up or down depends on the relationship between the input state prior to entering STOP mode and the state of the releasing input.

After the releasing input is asserted, the device warms up for the specified period before entering NORMAL or SLOW mode to restart counting.

d. Up/down-counter

When 2-phase input counter mode is selected (TB2RUN<TB2UDCE> = 1), the up-counter is initialized to 0x7FFF and operates as an up/down-counter. If the counter overflows, it is initialized to 0x0000 and continues counting. If the counter underflows, it is initialized to 0xFFFF and continues counting. Therefore, you can determine the state of the counter by reading the counter value and the status flag TB2ST after an interrupt occurs.

Sampling clock	
Count up input	
Up/down-counter value	0x3FFF 0x4000 0x4001
Up/down interrupt	
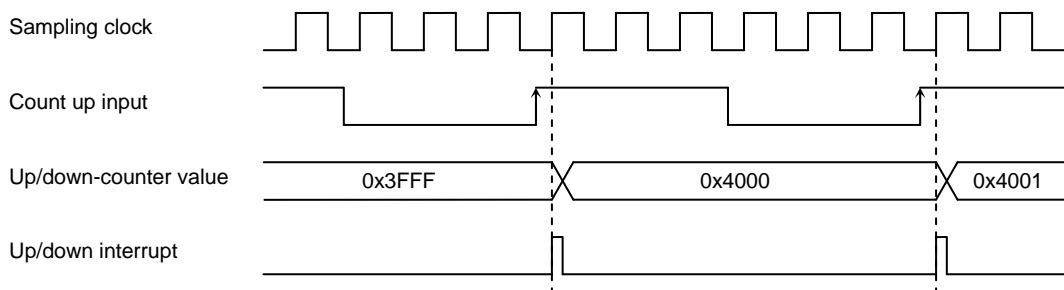
Note 1: Ensure that the count up (down) input is High before and after it is input.

Note 2: The counter value must be read during exception handling for INTTB2. If the counter value is read during exception handling for INTB or INTC used to terminate SLEEP or STOP mode, the counter value varies depending on whether the condition is satisfied or not and the difference in time between SLEEP/STOP mode being terminated and counting being restarted.

3.11 Serial Channels (SIO)

The TMP1942 contains five serial input/output channels: SIO0, SIO1, SIO3, SIO4 and SIO5. Each channel can be operated in (asynchronous) UART mode or (synchronous) I/O interface mode, as shown below.

- I/O interface mode ——————— Mode 0: Transmit and receive I/O data using the sync signal (SCLK) for extended I/O operation.

- Asynchronous (UART) mode ———— Mode 1: Transmit and receive 7-bit data.
  Mode 2: Transmit and receive 8-bit data.
  Mode 3: Transmit and receive 9-bit data.

In modes 1 and 2 a parity bit can be added. Mode 3 supports a wake-up function which is used by the master controller in a multi-controller system to initiate communication with a slave controller via a serial link. Figure 3.11.2 shows a block diagram for SIO0.

Each channel consists of a prescaler, a serial clock generator, a receive buffer and its accompanying control circuit, and a transmit buffer and its accompanying control circuit. All channels operate independently of each other. Because all channels operate in the same way, this section consists only of an explanation for SIO0.

- Mode 0 (I/O interface mode): LSB first

← Direction of transfer

- Mode 0 (I/O interface mode): MSB first

← Direction of transfer

- Mode 1 (7-bit UART mode)

Without parity

With parity

- Mode 2 (8-bit UART mode)

Without parity

With parity

- Mode 3 (9-bit UART mode)

Address (selection code) when bit 8 = 1
Data when bit 8 = 0
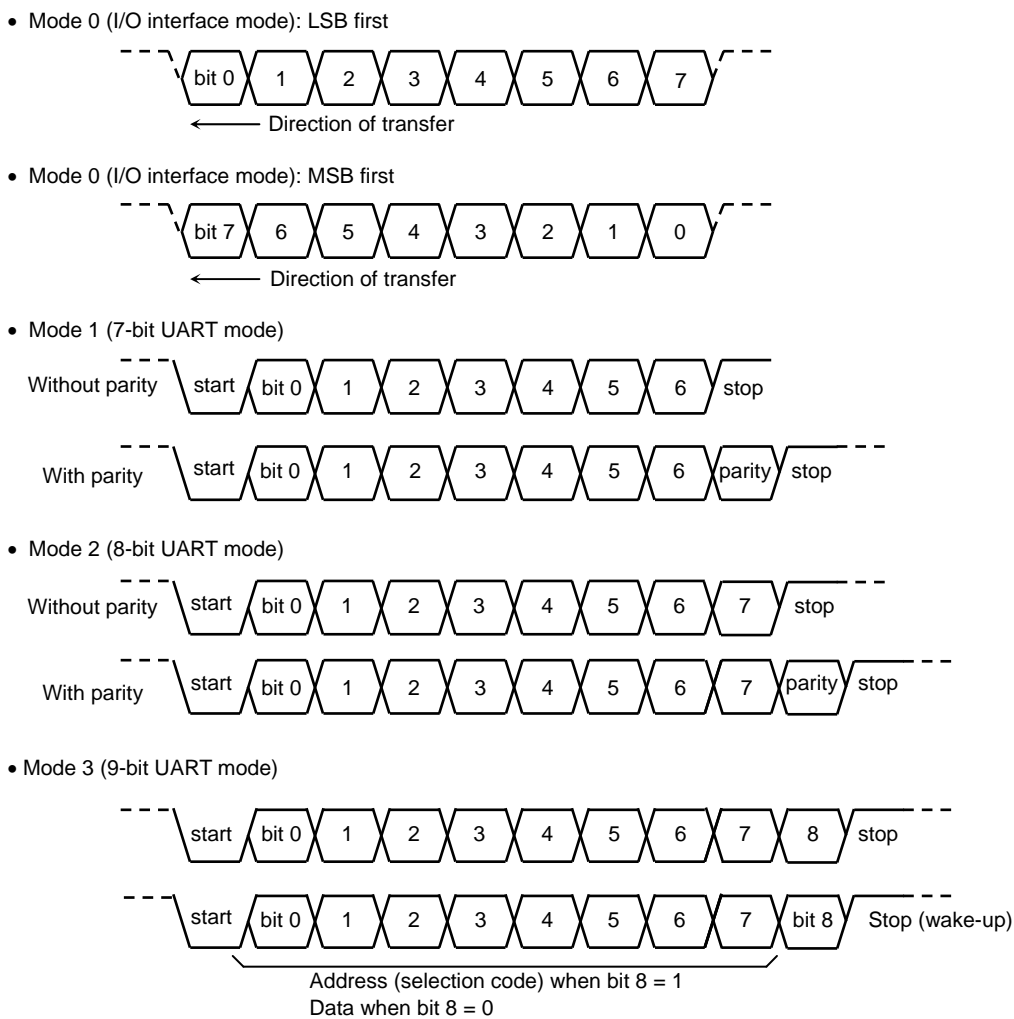
Figure 3.11.1 Data Formats

3.11.1 Block diagram (for channel 0 as an example)
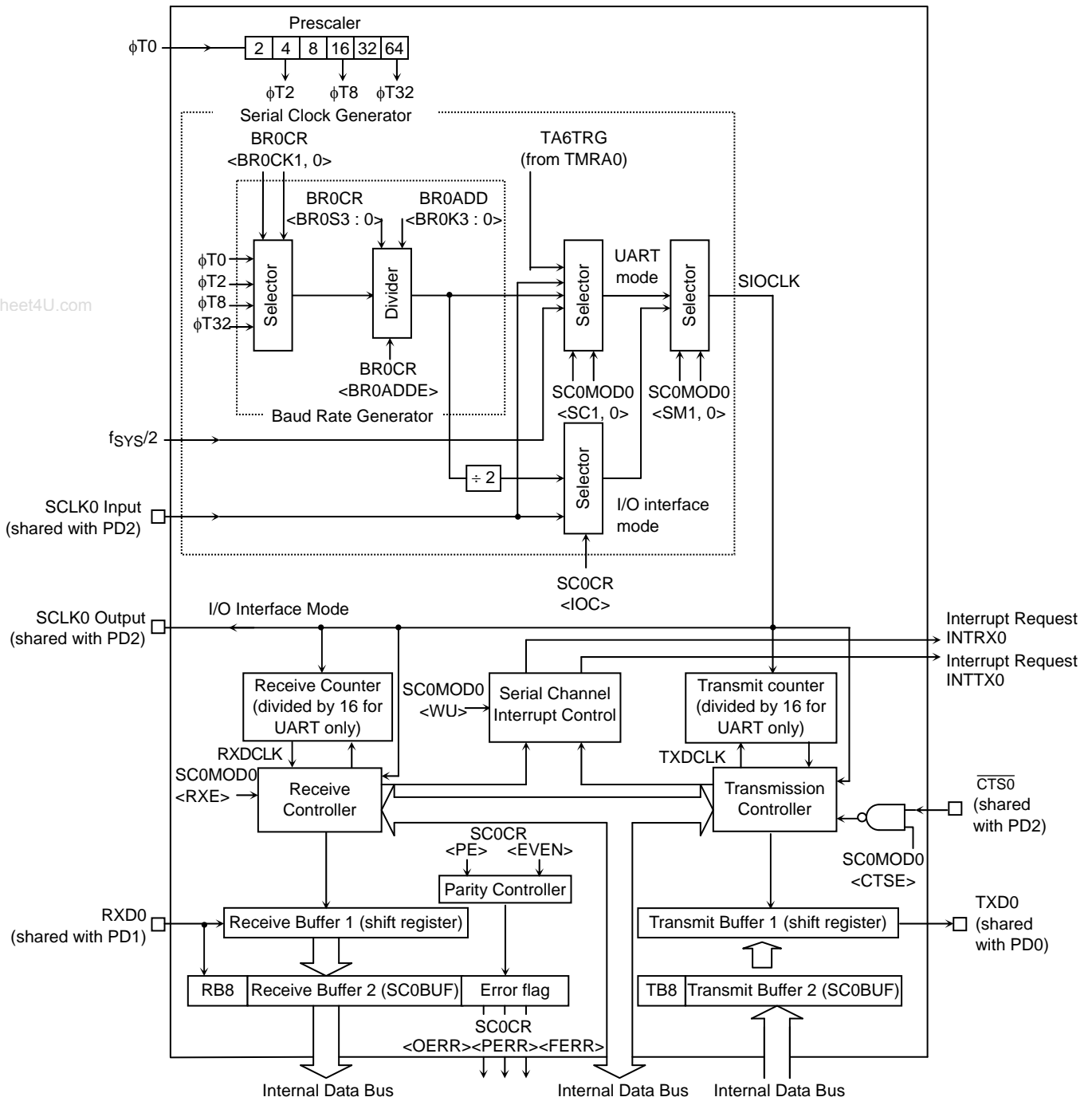
Figure 3.11.2 SIO0 Block Diagram

3.11.2 Functional description of each circuit (for channel 0 as an example)

(1) Prescaler

The TMP1942 has a 6-bit prescaler to supply an operating clock to SIO0. The prescaler's input clock φT0 has a frequency of fperiph, fperiph/2 or fperiph/4 as selected by SYSCR0 <PRCK1:PRCK0> in the CG block.

fperiph is either the clock fgear as selected by SYSCR1<FPSEL> in the CG block or the clock fc before division by the clock gear.

The prescaler operates only when the baud rate generator has been specified as the serial transfer clock. Table 3.11.1 shows the resolutions of the prescaler output clocks.

Table 3.11.1 Baud Rate Generator Input Clock Resolutions

@ = 32 MHz

Peripheral Clock Selection <FPSEL>	Clock Gear Value <GEAR1:0>	Selected Prescaler Clock <PRCK1:0>	Prescaler Output Clock Resolution			
			φT0	φT2	φT8	φT32
0 (fgear)	00 (fc)	00 (fperiph/4)	$fc/2^2$ (0.125 µs)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)
		01 (fperiph/2)	—	$fc/2^3$ (0.25 µs)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)
		10 (fperiph)	—	$fc/2^2$ (0.125 µs)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)
	01 (fc/2)	00 (fperiph/4)	$fc/2^3$ (0.25 µs)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16 µs)
		01 (fperiph/2)	—	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)
		10 (fperiph)	—	$fc/2^3$ (0.25 µs)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)
	10 (fc/4)	00 (fperiph/4)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)	$fc/2^{10}$ (32 µs)
		01 (fperiph/2)	—	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16 µs)
		10 (fperiph)	—	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)
	11 (fc/8)	00 (fperiph/4)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16 µs)	$fc/2^{11}$ (64 µs)
		01 (fperiph/2)	—	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)	$fc/2^{10}$ (32 µs)
		10 (fperiph)	—	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)	$fc/2^9$ (16 µs)
1 (fc)	00 (fc)	00 (fperiph/4)	$fc/2^2$ (0.125 µs)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)
		01 (fperiph/2)	—	$fc/2^3$ (0.25 µs)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)
		10 (fperiph)	—	$fc/2^2$ (0.125 µs)	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)
	01 (fc/2)	00 (fperiph/4)	—	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)
		01 (fperiph/2)	—	$fc/2^3$ (0.25 µs)	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)
		10 (fperiph)	—	—	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)
	10 (fc/4)	00 (fperiph/4)	—	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)
		01 (fperiph/2)	—	—	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)
		10 (fperiph)	—	—	$fc/2^4$ (0.5 µs)	$fc/2^6$ (2.0 µs)
	11 (fc/8)	00 (fperiph/4)	—	—	$fc/2^6$ (2.0 µs)	$fc/2^8$ (8.0 µs)
		01 (fperiph/2)	—	—	$fc/2^5$ (1.0 µs)	$fc/2^7$ (4.0 µs)
		10 (fperiph)	—	—	—	$fc/2^6$ (2.0 µs)

Note 1: The prescaler's output clock φTn must be selected such that the relationship φTn < fsys/2 is satisfied (i.e., φTn must be slower than fsys/2).

Note 2: Do not change the clock gear value while the timer is running

Note 3: The — character means "Don't use"

One of prescaler output clocks - φT0, φT2, φT8 or φT32 - is used for the serial interface baud rate generator.

(2) Baud rate

The baud rate generator is used to generate the transmit/receive clock which determines the rate at which data is transferred via serial channels.

The clock source fed to the baud rate generator is the clock $\phi T0$, $\phi T2$, $\phi T8$ or $\phi T32$ as output by the 6-bit prescaler. This input clock is selected by setting the bits BR0CR<BR0CK1:BR0CK0> in the baud rate generator control register.

The baud rate generator contains a divider which can divide the input clock frequency by 1, n + $\frac{m}{16}$ (n = 2-15, m = 0-15) or 16. The input clock frequency is divided according to the values set in BR0CR<BR0ADDE,BR0S3:BR0S0> and BR0ADD<BR0K3:BR0K0> to specify the rate of transfer.

- For UART mode

1) When BR0CR<BR0ADDE> = 0

The value set in BR0ADD<BR0K3:BR0K0> is ignored and the input clock is divided by the value N set in BR0CR<BR0S3:BR0S0> (N = 1, 2, 3... 16).

2) When BR0CR<BR0ADDE> = 1

The input clock is divided by N+(16-K)/16, where the value of N is specified by BR0CR <BR0S3:BR0S0> (N = 2, 3... 15) and the value of K is specified by BR0ADD<BR0K3:0> (K = 1, 2, 3... 15).

Note: When N = 1 or 16, division by N+(16-K)/16 is disabled. In that case, always set BR0CR<BR0ADDE> to 0.

- For I/O interface mode

Division by N+(16-K)/16 cannot be used in I/O interface mode. In this mode always set BR0CR<BR0ADDE> to 0 so that the input clock will be divided by N.

The baud rate is calculated as follows:

- In UART mode

$$\text{Baud rate} = \frac{\text{Baud rate generator input clock}}{\text{Baud rate generator divisor}} \div 16$$

The maximum baud rate which can be generated by the baud rate generator is 500 kbps and is generated when $\phi T0$ = 8 MHz.

In addition to an output from the baud rate generator, fsys/2 can also be used as a serial clock. If fsys/2 is used as a serial clock, the maximum baud rate is 1 Mbps, which is generated when fsys = 32 MHz.

- In I/O interface mode

$$\text{Baud rate} = \frac{\text{Baud rate generator input clock}}{\text{Baud rate generator divisor}} \div 2$$

• When dividing the input clock by an integer (N)

If φT2 is chosen as the input clock to the baud rate generator with the divisor N (BR0CR<BR0S3:BR0S0>) set to 10 and BR0CR<BR0ADDE> set to 0 after fc = 24.576 MHz has been specified as fperiph and φT0 has been set to fperiph/4, then the baud rate in UART mode is calculated as follows:

∗ Clock conditions ⎰ System clock   : High-speed (fc)
                       ⎧ High-speed clock gear : × 1 (fc)
                       ⎩ Prescaler clock   : fperiph/4 (fperiph = fsys)

$$\text{Baud rate} = \frac{\text{fc}/16}{10} \div 16$$

$$= 24.576 \times 10^6 \div 16 \div 10 \div 16 = 9600 \text{ (bps)}$$

Note: Since division by N+(16-K)/16 is disabled, the value set in BR0ADD<BR0K3:BR0K0> is ignored.

• When dividing the input clock by N+(16-K)/16 (UART mode only)

If φT2 is chosen as the input clock to the baud rate generator with the divisor N (BR0CR<BR0S3:BR0S0>) set to 7, K (BR0ADD<BR0K3:BR0K0>) set to 3 and BR0CR<BR0ADDE> set to 1 after fc = 19.2 MHz has been specified as fperiph and φT0 has been set to fperiph/4, the baud rate is calculated as follows:

∗ Clock conditions ⎰ System clock   : High-speed (fc)
                       ⎧ High-speed clock gear : × 1 (fc)
                       ⎩ Prescaler clock   : fperiph/4 (fperiph = fsys)

$$\text{Baud rate} = \frac{\text{fc}/16}{7 + \dfrac{16 - 3}{16}} \div 16$$

$$= 19.2 \times 10^6 \div 16 \div (7 + \frac{13}{16}) \div 16 = 9600 \text{ (bps)}$$

Tables 3.11.2 and 3.11.3 show example baud rates in UART mode.

Instead of a prescaler output, a clock input from an external source can also be used as the serial clock. In this case the baud rate is calculated as follows:

• UART mode
Baud rate = external clock input/16
However, the period of the external clock must be greater than or equal to 4/fsys.

• I/O interface mode
Baud Rate = external clock input
However, the period of the external clock must be greater than or equal to 16/fsys.

Table 3.11.2 UART Baud Rate Selection
(When the baud rate generator is used and BR0CR<BR0ADDE> = 0)

Units: kbps

fc [MHz]	Divisor N (Specified with BR0CR<BR0S3:BR0S0>)	Input Clock			
		φT0 (fc/4)	φT2 (fc/16)	φT8 (fc/64)	φT32 (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: The values shown in the table above are applied when the system clock frequency = fc, the clock gear = fc/1 and the prescaler clock frequency = fperiph/4.

Table 3.11.3 UART Baud Rate Selection
(When TMRA6 timer trigger output is used and TMRA6 input clock = φT1)

Units: kbps

TA0REG	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

Calculate the baud rate as follows (when timer TMRA6 is used):

$$\text{Transfer rate} = \frac{\text{Clock frequency specified with SYSCR0} < PRCK1:PRCK0>}{TA0REG \times 2 \times 16}$$

(when TMRA6 input clock = φT1)

Note 1: The trigger signal from timer TMRA6 cannot be used as the transfer clock in I/O interface mode.

Note 2: The values shown in the table above are applied when the system clock frequency = fc, the clock gear = fc/1, and the prescaler clock frequency = fperiph/4.

(3) Serial clock generator

This circuit generates a basic clock used to transmit and receive data.

- For I/O interface mode

    In SCLK output mode (when SC0CR<IOC> = 0), the basic clock is generated by dividing the baud rate generator output, described above, by 2.

    In SCLK input mode (when SC0CR<IOC> = 1), the basic clock is generated by detecting either the rising or falling edges of the SCLK input, as specified by the SC0CR<SCLKS> setting.

- For asynchronous (UART) mode

    One of the following four sources is selected to generate the basic clock SIOCLK: the clock output by the baud rate generator as described above, the system clock (fsys/2), the trigger output signal from timer TMRA6, or the external clock (on the SCLK0 pin). The setting of SC0MOD0<SC1:SC0> specifies which source is selected.

(4) Receive counter

The receive counter is a 4-bit binary counter which is used in asynchronous (UART) mode. This counter is incremented every time a SIOCLK pulse is detected. Receiving one bit of data requires 16 SIOCLK pulses and data is sampled three times: at the seventh, eighth and ninth pulses. The received data is determined from the three samples by majority rule.

(5) Receive controller

- For I/O interface mode

    In SCLK output mode (when SC0CR<IOC> = 0), the RXD0 pin is sampled at the rising edge of the shift clock which is output to the SCLK0 pin.

    In SCLK input mode (when SC0CR<IOC> = 1), the RXD0 pin is sampled at either the rising or falling edge of the SCLK input, as specified by the setting of SC0CR<SCLKS>.

- For asynchronous (UART) mode

    The receive controller incorporates a start bit detection circuit so that it can start receive operation upon the detection of a valid start bit.

(6) Receive buffer

The receive buffer has double-buffer structure to prevent overrun errors. Received data is stored one bit at a time in receive buffer 1 (a shift register). When all bits of data have been received, the data is transferred to another receive buffer, receive buffer 2 (SC0BUF), at which point an INTRX0 interrupt is generated. Also, the receive buffer full flag (SC0MOD2<RBFLL>) is set to 1 simultaneously, indicating that receive buffer 2 contains valid data.

The CPU reads data from receive buffer 2 (SC0BUF). This read causes the RBFLL flag to be cleared to 0. Next received data can be stored in receive buffer 1 even before the CPU reads the data out from receive buffer 2 (SC0BUF).

When SCLK output is selected in I/O interface mode, receive buffer 2 (SCOBUF) can be enabled or disabled by setting SC0MOD2<WBUF> accordingly. Disabling receive buffer 2 allows the device to handshake with the remote device it is communicating with, so that it stops CLK output every time it has sent a single frame. In that case, the CPU reads data from receive buffer 1. This read causes CLK output to restart. When receive buffer 2 is enabled in I/O interface mode, operation is as follows: The first received data is transferred from receive buffer 1 to receive buffer 2. CLK output stops when the next data has been received and both receive buffers 1 and 2 contain valid data. Once the CPU has read data from receive buffer 2, the data in receive buffer 1 is transferred to receive buffer 2, at which point an INTRX0 interrupt is generated and CLK output is restarted. Therefore, no overrun error occurs in SCLK output I/O interface mode, regardless of the WBUG setting.

> Note: In this mode the SC0CR OEER flag has no meaning, resulting in undefined operation. Be sure to read SC0CR to initialize this flag before changing the mode from SCLK output mode.

In other operating modes, receive buffer 2 is always enabled to improve performance for continuous transfer. However, if the CPU has not read the data out from receive buffer 2 (SC0BUF) by the time all the bits of the next data item have been received into receive buffer 1, an overrun error will occur. If an overrun error occurs, the contents of receive buffer 1 will be lost; the contents of receive buffer 2 and SC0CR<RB8> will be retained.

SC0CR<RB8> stores either the parity bit which is added to 8-bit UART data or the most significant bit of 9-bit UART data.

In 9-bit UART mode, slave controller wake-up operation can be enabled by setting SC0MOD0<WU> to 1. In this case, an INTRX0 interrupt is only generated if SC0CR<RB8> = 1.

(7) Transmit counter

The transmit counter is a 4-bit binary counter used in asynchronous (UART) mode. Like the receive counter, this counter is incremented every time a SIOCLK pulse is detected and generates a transmit clock (TXDCLK) pulse every 16 SIOCLK pulses.
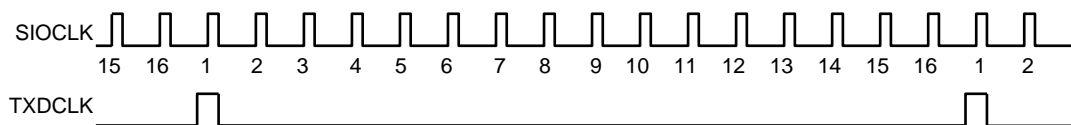
Figure 3.11.3 Generating a Transmit Clock

(8) Transmit controller

- In I/O interface mode

    In SCLK output mode (when SC0CR<IOC> = 0), data is output from the transmit buffer to the TXD0 pin one bit at a time at each rising edge of the shift clock output on the SCLK0 pin.

    In SCLK input mode (when SC0CR<IOC> = 1), data is output from the transmit buffer to the TXD0 pin one bit at a time, either at each rising edge or each falling edge of the SCLK input as specified by the setting of SC0CR<SCLKS>.

- In asynchronous (UART) mode

    After transmit data has been written to the transmit buffer by the CPU, the transmit controller will start transmitting the data at the next rising edge of TXDCLK, thus generating a transmit shift clock (TXDSFT).

Handshaking function

The device has a $\overline{\text{CTS}}$ pin, which makes it possible to transmit data in frame units, preventing overrun errors from occurring. This function can be enabled or disabled using SC0MOD<CTSE>.

When the $\overline{\text{CTS0}}$ pin goes High, the transmitter stops transmission after it has finished sending the current data and remains idle until the $\overline{\text{CTS0}}$ pin goes back to Low. The transmit controller generates an INTTX0 interrupt to request the next transmission of data from the CPU and, after writing the data to the transmit buffer, will wait for the new data to be sent.

Although the device does not have an $\overline{\text{RTS}}$ pin, the handshaking function can be implemented in the following way: One of the receiver's ports can be assigned to the function and when the receiver has finished receiving data, it drives that port High (using the receive interrupt routine), thereby requesting the transmitter to temporarily suspend transmission.
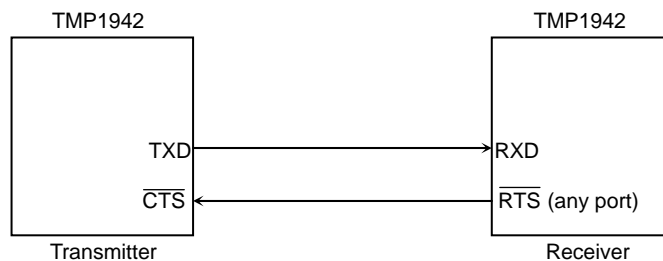
Figure 3.11.4 Handshaking Function
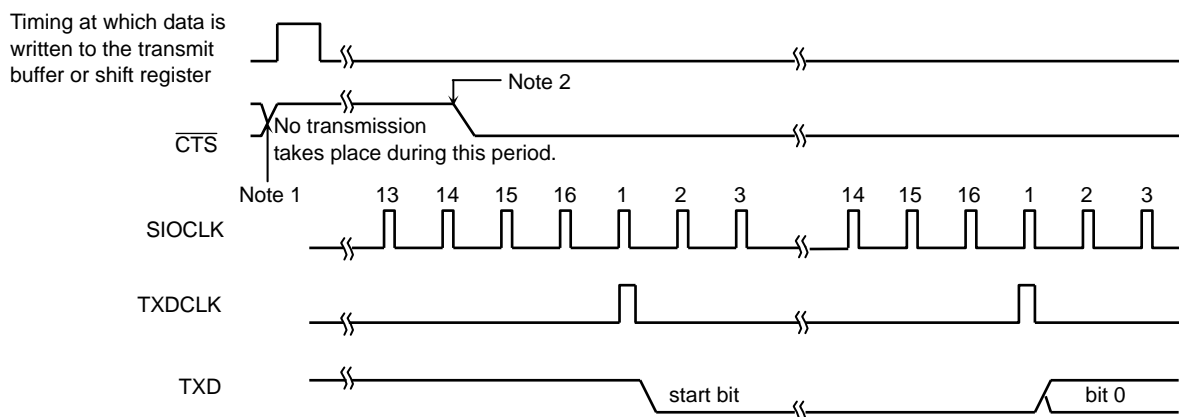
| Note 1: | When $\overline{\text{CTS}}$ goes High during transmission, the transmitter will stop sending data upon the completion of transmitting the current data item. |
| Note 2: | The transmitter starts sending data at the first falling edge of the TXDCLK clock after the $\overline{\text{CTS}}$ signal has been pulled Low. |

Figure 3.11.5 Clear to Send ($\overline{\text{CTS}}$) Signal Timing

(9) Transmit buffer

The transmit buffer (SC0BUF) has double-buffer structure. The double-buffer can be enabled or disabled by setting SC0MOD1<WBUF> accordingly. When the double-buffer is enabled, data written to transmit buffer 2 (SC0BUF) is transferred to transmit buffer 1 (a shift register), at which point an INTTX interrupt is generated. Also, the SCnMOD2 TBEMP flag is set to 1 simultaneously, indicating that transmit buffer 2 is empty so that next transmit data can be written. The TBEMP flag is cleared to 0 when next transmit data is written to transmit buffer 2. When the double-buffer is disabled, the CPU writes transmit data to transmit buffer 1 and an INTTX interrupt occurs upon the completion of transmission.

> Note: In this mode the SC0CR UEER flag has no meaning, resulting in undefined operation. Be sure to read SC0CR to initialize this flag before changing the mode from SCLK output mode.

If it is necessary to handshake with the remote device, set WBUF to 0 to disable transmit buffer 2. To perform continuous transmission without handshaking, you can improve performance by setting WBUF to 1 to enable transmit buffer 2.

(10) Parity controller

Data transmission with parity is enabled by setting the PE bit of the serial channel control register SC0CR to 1. Note, however, that parity can only be used in 7-bit UART mode or 8-bit UART mode. The SC0CR<EVEN> bit can be used to select even or odd parity.

During transmission the parity controller automatically generates parity bits from the data written to the transmit buffer (SC0BUF). Upon the completion of transmitting the data, it stores the parity in SC0BUF<TB7> in 7-bit UART mode or SC0MOD0<TB8> in 8-bit UART mode. The PE and EVEN bits in the SC0CR register must be set before the transmit data is written to the transmit buffer.

During reception the parity controller automatically generates parity bits from the data which has been shifted in and transferred from receive buffer 1 to receive buffer 2 (SC0BUF), and compares it with the parity stored in SC0BUF<RB7> in 7-bit UART mode or SC0CR<RB8> in 8-bit UART mode. If the parities do not match, a parity error is generated, setting the SC0CR<PERR> flag.

In I/O interface mode, SC0CR<PERR> is not a parity flag but functions as an underrun error flag.

(11) Error flags

Three error flags are available for the purpose of increasing the reliability of the received data.

1. Overrun error <OERR>

In both UART and I/O interface modes, an overrun error occurs when all bits of the next frame have been received before data stored in the receive buffer is read out completely. An overrun error causes the OERR flag to be set. Reading the flag clears it to 0. If SCLK output is selected in I/O interface mode, however, this flag is undefined because no overrun error will occur.

2. Parity error/underrun error <PERR>

In UART mode, the PERR flag is set to 1 when a parity error occurs. A parity error occurs if the parity calculated from the received data differs from the received parity bit. Reading the PERR flag clears it to 0.

In I/O interface mode, the PERR bit indicates an underrun error. When SC0MOD2<WBUF> is set to 1, an underrun error occurs in the following case: In SCLK input mode, it occurs if data stored in the transmit shift register has been transmitted but no data is set in the transmit double-buffer before the next transfer clock is input. In SCLK output mode, this flag is undefined because no underrun error will occur. The PERR flag is not set when transmit buffer 2 is disabled. Reading the flag clears it to 0.

3. Framing error <FERR>

In UART mode, the FERR flag is set to 1 when a framing error occurs. Reading the flag clears it to 0. A framing error occurs if the stop bit in the received data is detected as being 0 when sampled around the center.

Operating Mode	Error Flag	Description
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	Undefined
	PERR	Undefined
	FERR	Fixed to 0

Note: FERR reading occurs during the interruption handling must be executed before a receive buffer reading. Polling for reading FERR is prohibited.
See the example in 3.11.4 (3) Mode 2 (8-bit UART Mode) for the details.

(12) Direction of data transfer

In I/O interface mode, the direction of transfer can be toggled between MSB first and LSB first by setting SCnMOD2<DRCHG>. Do not change the direction of transfer while data is being transferred.

(13) STOP bit length

In UART mode, the STOP bit length in transmit data can be toggled between one bit and two bits by setting SCnMOD2<SBLEN>.

(14) Status flag

The SCnMOD2<RBFLL> bit is a flag which indicates that the receive buffer is full when the double-buffer is enabled (WBUF = 1). Once a single frame of data has been received and the data has been transferred from receive buffer 1 to receive buffer 2, this flag is set to 1, indicating that buffer 2 is full (contains data). When the CPU/DMAC reads the receive buffer, the flag is cleared to 0. When WBUF = 0, the RBFLL bit has no meaning and cannot be used as a status flag. TBEMP is a flag which indicates that transmit buffer 2 is empty when the double-buffer is enabled (WBUF = 1). Once data has been transferred from transmit buffer 2 to transmit buffer 1 (a shift register), this flag is set to 1, indicating that transmit buffer 2 is empty. When the CPU/DMAC writes data to the transmit buffer, the flag is cleared to 0. When WBUF = 0, the TBEMP bit has no meaning and cannot be used as a status flag.

(15) Transmit/receiver buffer configuration

		WBUF = 0	WBUF = 1
UART	Transmit	SINGLE	DOUBLE
	Receive	DOUBLE	DOUBLE
I/O interface (SCLK input)	Transmit	SINGLE	DOUBLE
	Receive	DOUBLE	DOUBLE
I/O interface (SCLK output)	Transmit	SINGLE	DOUBLE
	Receive	SINGLE	DOUBLE

(16) Signal generation timing

   1)   UART mode

   Reception

	Mode		
	9 bits	8 bits + parity	8 bits, 7 bits + parity, or 7 bits
Interrupt generation timing	Center of first stop bit	Center of first stop bit	Center of first stop bit
Framing error generation timing	Center of stop bit	Center of stop bit	Center of stop bit
Parity error generation timing	—	Center of last bit (parity bit)	Center of last bit (parity bit)
Overrun error generation timing	Center of stop bit	Center of stop bit	Center of stop bit

   Transmission

	Mode		
	9 bits	8 bits + parity	8 bits, 7 bits + parity, or 7 bits
Interrupt generation timing (WBUF = 0)	Immediately before stop bit is sent	Immediately before stop bit is sent	Immediately before stop bit is sent
Interrupt generation timing (WBUF = 1)	Immediately after data is transferred to transmit buffer 1 (immediately before start bit is sent)	Immediately after data is transferred to transmit buffer 1 (immediately before start bit is sent)	Immediately after data is transferred to transmit buffer 1 (immediately before start bit is sent)

2)　I/O interface mode

Reception

Interrupt generation timing (WBUF = 0)	SCLK output mode	Immediately after rise of last SCLK pulse
	SCLK input mode	Immediately after rise of last SCLK pulse (rise mode); in fall mode, immediately after fall of last SCLK pulse
Interrupt generation timing (WBUF = 1)	SCLK output mode	Immediately after rise of last SCLK pulse (immediately after data is transferred to receive buffer 2) or immediately after data is read from receive buffer 2
	SCLK input mode	Immediately after rise of last SCLK pulse (rise mode); in fall mode, immediately after fall of last SCLK pulse (immediately after data is transferred to receive buffer 2)
Overrun error generation timing	SCLK output mode	Immediately after rise of last SCLK pulse
	SCLK input mode	Immediately after rise of last SCLK pulse (rise mode); in fall mode, immediately after fall of last SCLK pulse

Transmission

Interrupt generation timing (WBUF = 0)	SCLK output mode	Immediately after rise of last SCLK pulse
	SCLK input mode	Immediately after rise of last SCLK pulse (rise mode); in fall mode, immediately after fall of last SCLK pulse
Interrupt generation timing (WBUF = 1)	SCLK output mode	Immediately after rise of last SCLK pulse or immediately after data is transferred to transmit buffer 1
	SCLK input mode	Immediately after rise of last SCLK pulse (rise mode); in fall mode, immediately after fall of last SCLK pulse; or immediately after data is transferred to transmit buffer 1
Underrun error generation timing	SCLK output mode	Immediately after rise of last SCLK pulse
	SCLK input mode	Immediately after rise of next SCLK pulse (rise mode); in fall mode, immediately after fall of next SCLK pulse

Note 1:　Do not modify any control register during transmission or reception (while reception is enabled).

Note 2:　Do not disable reception (by setting SC0MOD0<RXE> to 0) while data is being received.

3.11.3   Register description

SC0MOD0
(0xFFFF_F232)

	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	Handsha-king function control 0:Disable CTS 1:Enable CTS	Receive control 0:Disable reception 1:Enable reception	Wake-up function 0: Disable 1: Enable	Serial transfer mode 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial transfer clock (for UART) 00: Timer TA6TRG 01: Baud rate generator 10: Internal clock f_{sys}/2 11: External clock (SCLK0 input)	

Note: In I/O interface mode, the clock is selected using the serial control register (SC0CR).

Wake-up function

	9-bit UART mode	Other modes
0	Interrupt when data is received	Don't care
1	Interrupt only when RB8 = 1	

Handshaking function ($\overline{CTS}$ pin) enable

0	Disable (continuous transmission allowed)
1	Enable

Note: Do not set RXE to 1 while setting each mode register (SC0MOD0, SC0MOD1, and SC0MOD2). Set RXE to 1 after setting all other register bits.

Figure 3.11.6 Serial Mode Control Register 0 (SC0MOD0, for SIO0)

SC0MOD1
(0xFFFF_F235)

	7	6	5	4	3	2	1	0
Bit symbol	I2S0	FDPX0	SIOEN	—	—	—	—	—
Read/Write	R/W	R/W	R/W	—	—	—	—	—
After reset	0	0	0	—	—	—	—	—
Function	IDLE 0: Idle 1: Running	Sync format 0:Half-duplex 1:Full-duplex	SIO operation 0: Disable 1: Enable					

<SIOEN>: Enables or disables a clock supply to SIO module components other than registers.

Note: When setting SIOEN to 1, set it before setting I2S0 and FDPX0.

Figure 3.11.7 Serial Mode Control Register 1 (SC0MOD1, for SIO0)

		7	6	5	4	3	2	1	0
SC0MOD2 (0xFFFF_F236)	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	Transmi-ssion in progress flag 0:Stopped 1:Transmi-tting	STOP bit length 0: 1 bit 1: 2 bits	Direction of transfer 0: LSB first 1: MSB first	Double-buff er enable 0: Disable 1: Enable	Soft reset Writing 10 then 01 triggers a reset.	

<SWRST1:0>: Writing 10 and 01 in this order triggers a software reset. This initializes the mode register bits SC0MOD0<RXE>, SC0MOD2<TBEMP>, <RBFIL> and <TXRUN>, control register bits SC0CR<OERR>, <PERR> and <FERR>, and the internal logic.

<WBUF>: Enables or disables the double-buffer for transmission (SCLK output/input) and reception (SCLK output) in I/O interface mode and transmission in UART mode. In other modes, the double-buffer is always enabled regardless of the setting.

<DRCHG>: Specifies the direction of transfer in I/O interface mode. In UART mode, this bit is fixed to 0 (LSB first).

<TXRUN>: This bit is a status flag which indicates whether transmission shift operation is in progress. When this bit is set to 1, it indicates that data is being transmitted. When this bit is set to 0, it indicates that transmission is completely finished (if TBEMP = 1) or that the device is waiting with next transmit data stored in the transmit buffer (if TBEMP = 0).

<RBFLL>: This bit is a flag which indicates whether the receive double-buffer is full. RBFIL is set to 1 when data has been transferred from the receive shift register to the receive double-buffer. It is cleared to 0 when the data has been read. This flag has no meaning if the double-buffer is disabled.

<TBEMP>: This bit is a flag which indicates whether the transmit double-buffer is empty. TBEMP is set to 1 when data has been transferred from the transmit double-buffer to the transmit shift register, resulting in the transmit double-buffer being empty. It is cleared to 0 when transmit data has been written to the double-buffer. This flag has no meaning if the double-buffer is disabled.

<SBLEN>: Specifies the transmit STOP bit length in UART mode. During reception, the device always recognizes a single STOP bit regardless of the setting of this bit.

Note: If it is necessary to perform a soft reset during transmission, perform it twice consecutively.

Figure 3.11.8 Serial Mode Control Register 2 (SC0MOD2, for SIO0)

SC1MOD0
(0xFFFF_F23A)

	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	1: Enable CTS	Receive control 0:Disable reception 1:Enable reception	Wake-up function 0: Disable 1: Enable	Serial transfer mode 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial transfer clock (for UART) 00:Timer TA6TRG 01:Baud rate generator 10:Internal clock $f_{sys}/2$ 11:External clock (SCLK1 input)	

Note: In I/O interface mode, the clock is selected using the serial control register (SC1CR).

Wake-up function

	9-bit UART mode	Other modes
0	Interrupt when data is received	Don't care
1	Interrupt only when RB8 = 1	

Note: Do not set RXE to 1 while setting each mode register (SC1MOD0, SC1MOD1, and SC1MOD2). Set RXE to 1 after setting all other register bits.

Figure 3.11.9 Serial Mode Control Register 0 (SC1MOD0, for SIO1)

SC1MOD1
(0xFFFF_F23D)

	7	6	5	4	3	2	1	0
Bit symbol	I2S0	FDPX0	SIOEN	—	—	—	—	—
Read/Write	R/W	R/W	R/W	—	—	—	—	—
After reset	0	0	0	—	—	—	—	—
Function	I Idle 0: Idle 1: Running	Sync format 0:Half-duplex 1:Full-duplex	SIO operation 0: Disable 1: Enable					

<SIOEN>: Enables or disables a clock supply to SIO module components other than registers.

Note: When setting SIOEN to 1, set it before setting I2S0 and FDPX0.

Figure 3.11.10 Serial Mode Control Register 1 (SC1MOD1, for SIO1)

SC0MOD2
(0xFFFF_F23E)

	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	Transmi-ssion in progress flag 0: Stopped 1:Transmi-tting	STOP bit length 0: 1 bit 1: 2 bits	Direction of transfer 0: LSB first 1: MSB first	Double-buffer enable 0: Disable 1: Enable	Soft reset Writing 10 then 01 triggers a reset.	

<SWRST1:0>: Writing 10 and 01 in this order triggers a software reset. This initializes the mode register bits SC1MOD0<RXE>, SC1MOD2<TBEMP>, <RBFIL> and <TXRUN>, control register bits SC1CR<OERR>, <PERR> and <FERR>, and the internal logic.

<WBUF>: Enables or disables the double-buffer for transmission (SCLK output/input) and reception (SCLK output) in I/O interface mode and transmission in UART mode. In other modes, the double-buffer is always enabled regardless of the setting.

<DRCHG>: Specifies the direction of transfer in I/O interface mode. In UART mode, this bit is fixed to 0 (LSB first).

<TXRUN>: This bit is a status flag which indicates whether transmission shift operation is in progress. When this bit is set to 1, it indicates that data is being transmitted. When this bit is set to 0, it indicates that transmission is completely finished (if TBEMP = 1) or that the device is waiting with next transmit data stored in the transmit buffer (if TBEMP = 0).

<RBFLL>: This bit is a flag which indicates whether the receive double-buffer is full. RBFIL is set to 1 when data has been transferred from the receive shift register to the receive double-buffer. It is cleared to 0 when the data has been read. This flag has no meaning if the double-buffer is disabled.

<TBEMP>: This bit is a flag which indicates whether the transmit double-buffer is empty. TBEMP is set to 1 when data has been transferred from the transmit double-buffer to the transmit shift register, resulting in the transmit double-buffer being empty. It is cleared to 0 when transmit data has been written to the double-buffer. This flag has no meaning if the double-buffer is disabled.

<SBLEN>: Specifies the transmit STOP bit length in UART mode. During reception, the device always recognizes a single STOP bit regardless of the setting of this bit.

Note: If it is necessary to perform a soft reset during transmission, perform it twice consecutively.

Figure 3.11.11 Serial Mode Control Register 2 (SC1MOD2, for SIO1)

SC3MOD0
(0x0FFFF_F282)

	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	1:Enable CTS	Receive control 0:Disable reception 1:Enable reception	Wake-up function 0: Disable 1: Enable	Serial transfer mode 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial transfer clock (for UART) 00: Timer TA6TRG 01: Baud rate generator 10: Internal clock $f_{sys}/2$ 11: External clock (SCLK1 input)	

Wake-up function

	9-bit UART mode	Other modes
0	Interrupt when data is received	Don't care
1	Interrupt only when RB8 = 1	

Note: Do not set RXE to 1 while setting each mode register (SC3MOD0, SC3MOD1, and SC3MOD2). Set RXE to 1 after setting all other register bits.

Figure 3.11.12 Serial Mode Control Register 0 (SC3MOD0, for SIO3)

SC3MOD1
(0xFFFF_F285)

	7	6	5	4	3	2	1	0
Bit symbol	I2S0	FDPX0	SIOEN	—	—	—	—	—
Read/Write	R/W	R/W	R/W	—	—	—	—	—
After reset	0	0	0	—	—	—	—	—
Function	Idle 0: Idle 1: Running	Sync format 0:Half-duplex 1:Full-duplex	SIO operation 0: Disable 1: Enable					

<SIOEN>: Enables or disables a clock supply to SIO module components other than registers.

Note: When setting SIOEN to 1, set it before setting I2S0 and FDPX0.

Figure 3.11.13 Serial Mode Control Register 1 (SC3MOD1, for SIO3)

	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	Transmi-ssion in progress flag 0: Stopped 1:Transmi-tting	STOP bit length 0: 1 bit 1: 2 bits	Direction of transfer 0: LSB first 1: MSB first	Double-buffer enable 0: Disable 1: Enable	Soft reset Writing 10 then 01 triggers a reset.	

<SWRST1:0>: Writing 10 and 01 in this order triggers a software reset. This initializes the mode register bits SC3MOD0<RXE>, SC3MOD2<TBEMP>, <RBFIL> and <TXRUN>, control register bits SC3CR<OERR>, <PERR> and <FERR>, and the internal logic.

<WBUF>: Enables or disables the double-buffer for transmission (SCLK output/input) and reception (SCLK output) in I/O interface mode and transmission in UART mode. In other modes, the double-buffer is always enabled regardless of the setting.

<DRCHG>: Specifies the direction of transfer in I/O interface mode. In UART mode, this bit is fixed to 0 (LSB first).

<TXRUN>: This bit is a status flag which indicates whether transmission shift operation is in progress. When this bit is set to 1, it indicates that data is being transmitted. When this bit is set to 0, it indicates that transmission is completely finished (if TBEMP = 1) or that the device is waiting with next transmit data stored in the transmit buffer (if TBEMP = 0).

<RBFLL>: This bit is a flag which indicates whether the receive double-buffer is full. RBFIL is set to 1 when data has been transferred from the receive shift register to the receive double-buffer. It is cleared to 0 when the data has been read. This flag has no meaning if the double-buffer is disabled.

<TBEMP>: This bit is a flag which indicates whether the transmit double-buffer is empty. TBEMP is set to 1 when data has been transferred from the transmit double-buffer to the transmit shift register, resulting in the transmit double-buffer being empty. It is cleared to 0 when transmit data has been written to the double-buffer. This flag has no meaning if the double-buffer is disabled.

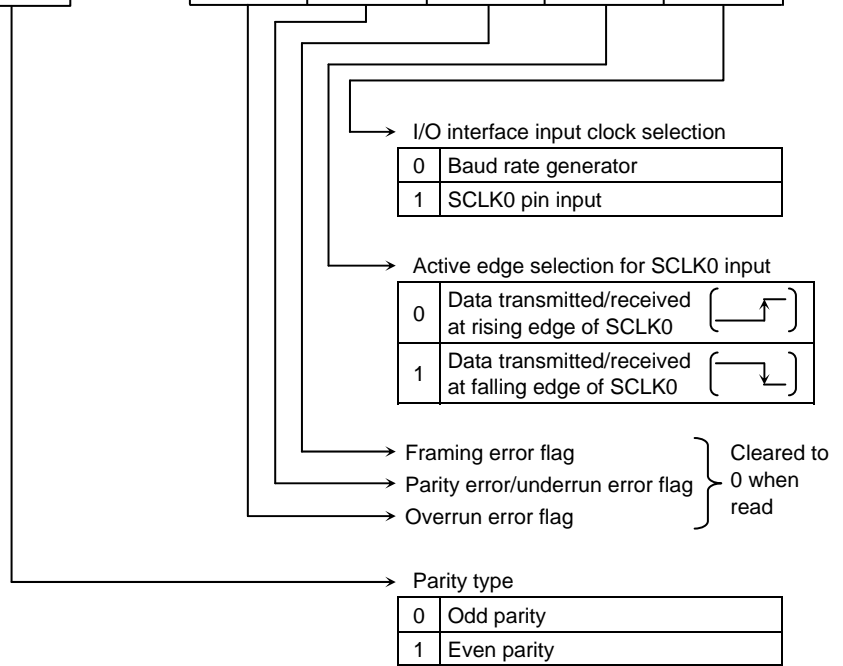<SBLEN>: Specifies the transmit STOP bit length in UART mode.

Note: If it is necessary to perform a soft reset during transmission, perform it twice consecutively.

Figure 3.11.14 Serial Mode Control Register 2 (SC3MOD2, for SIO3)

SC4MOD0
(0xFFFF_F28A)

	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	1: Enable CTS	Receive control 0: Disable reception 1: Enable reception	Wake-up function 0: Disable 1: Enable	Serial transfer mode 00: Reserved 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial transfer clock (for UART) 00: Timer TA6TRG 01: Baud rate generator 10: Internal clock $f_{sys}/2$ 11: Don't care	

Wake-up function

	9-bit UART mode	Other modes
0	Interrupt when data is received	Don't care
1	Interrupt only when RB8 = 1	

Note: Do not set RXE to 1 while setting each mode register (SC4MOD0, SC4MOD1, and SC4MOD2). Set RXE to 1 after setting all other register bits.

Figure 3.11.15 Serial Mode Control Register 0 (SC4MOD0, for SIO4)

SC4MOD1
(0xFFFF_F28D)

	7	6	5	4	3	2	1	0
Bit symbol	I2S0	FDPX0	SIOEN	—	—	—	—	—
Read/Write	R/W	R/W	R/W	—	—	—	—	—
After reset	0	0	0	—	—	—	—	—
Function	Idle 0: Idle 1: Running	Sync format 0:Half-duplex 1:Full-duplex	SIO operation 0: Disable 1: Enable					

<SIOEN>: Enables or disables a clock supply to SIO module components other than registers.

Note: When setting SIOEN to 1, set it before setting I2S0 and FDPX0.

Figure 3.11.16 Serial Mode Control Register 1 (SC4MOD1, for SIO4)

SC4MOD2
(0xFFFF_F28E)

	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	Transmi-ssion in progress flag 0: Stopped 1:Transmi-tting	STOP bit length 0: 1 bit 1: 2 bits	Direction of transfer 0: LSB first 1: MSB first	Double-buffer enable 0: Disable 1: Enable	Soft reset Writing 10 then 01 triggers a reset.	

<SWRST1:0>: Writing 10 and 01 in this order triggers a software reset. This initializes the mode register bits SC4MOD0<RXE>, SC4MOD2<TBEMP>, <RBFIL> and <TXRUN>, control register bits SC4CR<OERR>, <PERR> and <FERR>, and the internal logic.

<WBUF>: Enables or disables the double-buffer for transmission (SCLK output/input) and reception (SCLK output) in I/O interface mode and transmission in UART mode. In other modes, the double-buffer is always enabled regardless of the setting.

<DRCHG>: Specifies the direction of transfer in I/O interface mode. In UART mode, this bit is fixed to 0 (LSB first).

<TXRUN>: This bit is a status flag which indicates whether transmission shift operation is in progress. When this bit is set to 1, it indicates that data is being transmitted. When this bit is set to 0, it indicates that transmission is completely finished (if TBEMP = 1) or that the device is waiting with next transmit data stored in the transmit buffer (if TBEMP = 0).

<RBFLL>: This bit is a flag which indicates whether the receive double-buffer is full. RBFIL is set to 1 when data has been transferred from the receive shift register to the receive double-buffer. It is cleared to 0 when the data has been read. This flag has no meaning if the double-buffer is disabled.

<TBEMP>: This bit is a flag which indicates whether the transmit double-buffer is empty. TBEMP is set to 1 when data has been transferred from the transmit double-buffer to the transmit shift register, resulting in the transmit double-buffer being empty. It is cleared to 0 when transmit data has been written to the double-buffer. This flag has no meaning if the double-buffer is disabled.

<SBLEN>: Specifies the transmit STOP bit length in UART mode. During reception, the device always recognizes a single STOP bit regardless of the setting of this bit.

Note: If it is necessary to perform a soft reset during transmission, perform it twice consecutively.

Figure 3.11.17 Serial Mode Control Register 2 (SC4MOD2, for SIO4)

SC5MOD0
(0xFFFF_F292)

	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	1:Enable CTS	Receive control 0:Disable reception 1:Enable reception	Wake-up function 0: Disable 1: Enable	Serial transfer mode 00: Reserved 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial transfer clock (for UART) 00: Timer TA6TRG 01: Baud rate generator 10: Internal clock $f_{sys}/2$ 11: Don't care	

Wake-up function

	9-bit UART mode	Other modes
0	Interrupt when data is received	Don't care
1	Interrupt only when RB8 = 1	

Note: Do not set RXE to 1 while setting each mode register (SC5MOD0, SC5MOD1, and SC5MOD2). Set RXE to 1 after setting all other register bits.

Figure 3.11.18 Serial Mode Control Register 0 (SC5MOD0, for SIO5)

SC5MOD1
(0xFFFF_F295)

	7	6	5	4	3	2	1	0
Bit symbol	I2S0	FDPX0	SIOEN	—	—	—	—	—
Read/Write	R/W	R/W	R/W	—	—	—	—	—
After reset	0	0	0	—	—	—	—	—
Function	I Idle 0: Idle 1: Running	Sync format 0:Half-duplex 1:Full-duplex	SIO operation 0: Disable 1: Enable					

<SIOEN>: Enables or disables a clock supply to SIO module components other than registers.

Note: When setting SIOEN to 1, set it before setting I2S0 and FDPX0.

Figure 3.11.19 Serial Mode Control Register 1 (SC5MOD1, for SIO5)

SC5MOD2
(0xFFFF_F296)

	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	Transmi-ssion in progress flag 0: Stopped 1: Transmi-tting	STOP bit length 0: 1 bit 1: 2 bits	Direction of transfer 0: LSB first 1: MSB first	Double-buffer enable 0: Disable 1: Enable	Soft reset Writing 10 then 01 triggers a reset.	

<SWRST1:0>: Writing 10 and 01 in this order triggers a software reset. This initializes the mode register bits SC5MOD0<RXE>, SC5MOD2<TBEMP>, <RBFIL> and <TXRUN>, control register bits SC5CR<OERR>, <PERR> and <FERR>, and the internal logic.

<WBUF>: Enables or disables the double-buffer for transmission (SCLK output/input) and reception (SCLK output) in I/O interface mode and transmission in UART mode. In other modes, the double-buffer is always enabled regardless of the setting.

<DRCHG>: Specifies the direction of transfer in I/O interface mode. In UART mode, this bit is fixed to 0 (LSB first).

<TXRUN>: This bit is a status flag which indicates whether transmission shift operation is in progress. When this bit is set to 1, it indicates that data is being transmitted. When this bit is set to 0, it indicates that transmission is completely finished (if TBEMP = 1) or that the device is waiting with next transmit data stored in the transmit buffer (if TBEMP = 0).

<RBFLL>: This bit is a flag which indicates whether the receive double-buffer is full. RBFIL is set to 1 when data has been transferred from the receive shift register to the receive double-buffer. It is cleared to 0 when the data has been read. This flag has no meaning if the double-buffer is disabled.

<TBEMP>: This bit is a flag which indicates whether the transmit double-buffer is empty. TBEMP is set to 1 when data has been transferred from the transmit double-buffer to the transmit shift register, resulting in the transmit double-buffer being empty. It is cleared to 0 when transmit data has been written to the double-buffer. This flag has no meaning if the double-buffer is disabled.

<SBLEN>: Specifies the transmit STOP bit length in UART mode. During reception, the device always recognizes a single STOP bit regardless of the setting of this bit.

Note: If it is necessary to perform a soft reset during transmission, perform it twice consecutively.

Figure 3.11.20 Serial Mode Control Register 2 (SC5MOD2, for SIO5)

	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
Function	Receive data bit 8	Parity type 0: Odd 1: Even	Parity 0: Disabled 1: Enabled	1: Error			0: SCLK0 1: SCLK0	0:Baud rate generator 1:SCLK0 pin input
				Overrun	Parity /underrun	Framing		

SC0CR
(0xFFFF_F231)

I/O interface input clock selection

0	Baud rate generator
1	SCLK0 pin input

Active edge selection for SCLK0 input

0	Data transmitted/received at rising edge of SCLK0
1	Data transmitted/received at falling edge of SCLK0

Framing error flag
Parity error/underrun error flag } Cleared to 0 when read
Overrun error flag

Parity type

0	Odd parity
1	Even parity

Note 1: All error flags are cleared to 0 when read.

Note 2: For SCLK output operation, set SCLKS to 0 (rising edge).

Figure 3.11.21 Serial Control Register (SC0CR, for SIO0)

	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
Function	Receive data bit 8	Parity type 0: Odd 1: Even	Parity 0: Disabled 1: Enabled	1: Error			0: SCLK0 ⌐↱ 1: SCLK0 ⌐↳	0: Baud rate generator 1: SCLK0 pin input
				Overrun	Parity/ underrun	Framing		

SC1CR
(0xFFFF_F239)

I/O interface input clock selection

0	Baud rate generator
1	SCLK0 pin input

Active edge selection for SCLK0 input

0	Data transmitted/received at rising edge of SCLK0
1	Data transmitted/received at falling edge of SCLK0

Framing error flag
Parity error/underrun error flag
Overrun error flag

Cleared to 0 when read

Parity type

0	Odd parity
1	Even parity

Note 1: All error flags are cleared to 0 when read.

Note 2: For SCLK output operation, set SCLKS to 0 (rising edge).

Figure 3.11.22 Serial Control Register (SC1CR, for SIO1)

	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
Function	Receive data bit 8	Parity type 0: Odd 1: Even	Parity 0: Disabled 1: Enabled	1: Error			0: SCLK0 ↱ 1: SCLK0 ↳	0:Baud rate generator 1:SCLK0 pin input
				Overrun	Parity/ underrun	Framing		

SC3CR (0xFFFF_F281)

I/O interface input clock selection

0	Baud rate generator
1	SCLK0 pin input

Active edge selection for SCLK0 input

0	Data transmitted/received at rising edge of SCLK0
1	Data transmitted/received at falling edge of SCLK0

Framing error flag
Parity error/underrun error flag   } Cleared to 0 when read
Overrun error flag

Parity type

0	Odd parity
1	Even parity

<OERR>: In both UART and I/O interface modes, an overrun error occurs when all bits of the next frame have been received before data stored in the receive buffer is read out completely. An overrun error causes the OERR flag to be set.

<PERR>: In UART mode, the PERR flag is set to 1 when a parity error occurs. Reading the PERR flag clears it to 0. In I/O interface mode, the PERR bit indicates an underrun error. When SC0MOD2<WBUF> is set to 1, an underrun error occurs in the following case: In SCLK input mode, it occurs if data stored in the transmit shift register has been transmitted but no data is set in the transmit double-buffer. In other modes, this flag is not set. Reading the flag clears it to 0.

<FERR>: In UART mode, the FERR flag is set to 1 when a framing error occurs. Reading the flag clears it to 0.

Note 1: All error flags are cleared to 0 when read.

Note 2: For SCLK output operation, set SCLKS to 0 (rising edge).

Figure 3.11.23 Serial Control Register (SC3CR, for SIO3)

SC4CR
(0xFFFF_F289)

	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
Function	Receive data bit 8	Parity type 0: Odd 1: Even	Parity 0: Disabled 1: Enabled	1: Error			0: SCLK0 ⌐↑ 1: SCLK0 ⌐↓	0:Baud rate generator 1:SCLK0 pin input
				Overrun	Parity/ underrun	Framing		

I/O interface input clock selection

0	Baud rate generator
1	SCLK0 pin input

Active edge selection for SCLK0 input

0	Data transmitted/received at rising edge of SCLK0
1	Data transmitted/received at falling edge of SCLK0

Framing error flag
Parity error/underrun error flag   } Cleared to 0 when read
Overrun error flag

Parity type

0	Odd parity
1	Even parity

Note 1:   All error flags are cleared to 0 when read.

Note 2:   For SCLK output operation, set SCLKS to 0 (rising edge).

Figure 3.11.24 Serial Control Register (SC4CR, for SIO4)

SC5CR
(0xFFFF_F293)

	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
Function	Receive data bit 8	Parity type 0: Odd 1: Even	Parity 0: Disabled 1: Enabled	1: Error			0: SCLK0 ⟋ 1: SCLK0 ⟍	0:Baud rate generator 1:SCLK0 pin input
				Overrun	Parity/ underrun	Framing		

I/O interface input clock selection

0	Baud rate generator
1	SCLK0 pin input

Active edge selection for SCLK0 input

0	Data transmitted/received at rising edge of SCLK0 ⟋
1	Data transmitted/received at falling edge of SCLK0 ⟍

Framing error flag
Parity error/underrun error flag   } Cleared to 0 when read
Overrun error flag

Parity type

0	Odd parity
1	Even parity

Note 1: All error flags are cleared to 0 when read.

Note 2: For SCLK output operation, set SCLKS to 0 (rising edge).

Figure 3.11.25 Serial Control Register (SC5CR, for SIO5)

BR0CR
(0xFFFF_F233)

	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	Must always be set to 0.	Division by N+(16-K)/16 0: Disable 1: Enable	00: φT0 01: φT2 10: φT8 11: φT32		Sets value of divisor N			

Selects baud rate generator input clock

00	Internal clock φT0
01	Internal clock φT2
10	Internal clock φT8
11	Internal clock φT32

BR0ADD
(0xFFFF_F234)

	7	6	5	4	3	2	1	0
Bit symbol	—	—	—	—	BR0K3	BR0K2	BR0K1	BR0K0
Read/Write	—	—	—	—		R/W		
After reset	—	—	—	—	0	0	0	0
Function					Sets K value for division by N + (16-K)/16			

Sets divisor value for baud rate generator

BR0ADD<BR0K3:0>	BR0CR<BR0ADDE> = 1		BR0CR<BR0ADDE> = 0
	BR0CR<BR0S3:0>		
	0000 (N = 16) ≀ 0001 (N = 1)	0010 (N = 2) ≀ 1111 (N = 15)	0001 (N = 1) (ONLY UART) ≀ 1111 (N = 15) 0000 (N = 16)
0000	Invalid	Invalid	—
0001 (K = 1) ≀ 1111 (K = 15)	Invalid	Divided by N + (16-K)/16	Divided by N

Note 1: The baud rate generator divisor cannot be set to 1 in UART mode if division by N+(16-K)/16 is being used. It cannot be set to 1 at all in I/O interface mode.

Note 2: When using division by N+(16-K)/16, be sure to set K (1 to 15) in BR0ADD <BR0K3:BR0K0> before setting BR0CR<BR0ADDE> to 1. However, if BR0CR <BR0S3:BR0S0> = 0000 or 0001 (i.e. if N = 16 or 1), do not use division by N+(16-K)/16.

Note 3: Division by N+(16-K)/16 can only be used in UART mode. In I/O interface mode, set BR0CR<BR0ADDE> to 0 to disable division by N+(16-K)/16.

Figure 3.11.26 Baud Rate Generator Control Registers (BR0CR and BR0ADD, for SIO0)
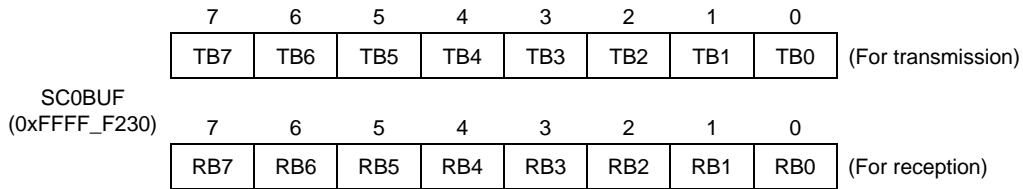
BR1CR (0xFFFF_F23B)		7	6	5	4	3	2	1	0
	Bit symbol	—	BR1ADDE	BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0
	Function	Must always be set to 0.	Division by N+(16-K)/16 0: Disable 1: Enable	00: φT0 01: φT2 10: φT8 11: φT32		Sets value of divisor N			

Selects baud rate generator input clock

00	Internal clock φT0
01	Internal clock φT2
10	Internal clock φT8
11	Internal clock φT32

BR1ADD (0xFFFF_F23C)		7	6	5	4	3	2	1	0
	Bit symbol	—	—	—	—	BR1K3	BR1K2	BR1K1	BR1K0
	Read/Write	—	—	—	—	R/W			
	After reset	—	—	—	—	0	0	0	0
	Function					Sets K value for division by N+(16-K)/16			

Sets divisor value for baud rate generator

	BR0CR<BR0ADDE> = 1		BR0CR<BR0ADDE> = 0
BR0ADD<BR0K3:0>	BR0CR<BR0S3:0>		
	0000 (N = 16) ≀ 0001 (N = 1)	0010 (N = 2) ≀ 1111 (N = 15)	0001 (N = 1) (ONLY UART) ≀ 1111 (N = 15) 0000 (N = 16)
0000	Invalid	Invalid	—
0001 (K = 1) ≀ 1111 (K = 15)	Invalid	Divided by N + (16-K)/16	Divided by N

Note 1: The baud rate generator divisor cannot be set to 1 in UART mode if division by N+(16-K)/16 is being used. It cannot be set to 1 at all in I/O interface mode.

Note 2: When using division by N+(16-K)/16, be sure to set K (1 to 15) in BR1ADD <BR1K3:BR1K0> before setting BR1CR<BR1ADDE> to 1. However, if BR1CR <BR1S3:BR1S0> = 0000 or 0001 (i.e. if N = 16 or 1), do not use division by N+(16-K)/16.

Note 3: Division by N+(16-K)/16 can only be used in UART mode. In I/O interface mode, set BR1CR<BR1ADDE> to 0 to disable division by N+(16-K)/16.

Figure 3.11.27 Baud Rate Generator Control Registers (BR1CR and BR1ADD, for SIO1)

www.DataSheet4U.com

	7	6	5	4	3	2	1	0
Bit symbol	—	BR3ADDE	BR3CK1	BR3CK0	BR3S3	BR3S2	BR3S1	BR3S0
Read/Write				R/W				
After reset	0	0	0	0	0	0	0	0
Function	Must always be set to 0.	Division by N+(16-K)/16 0: Disable 1: Enable	00: φT0 01: φT2 10: φT8 11: φT32		Sets value of divisor N			

BR3CR (0xFFFF_F283)

Selects baud rate generator input clock

00	Internal clock φT0
01	Internal clock φT2
10	Internal clock φT8
11	Internal clock φT32

	7	6	5	4	3	2	1	0
Bit symbol	—	—	—	—	BR3K3	BR3K2	BR3K1	BR3K0
Read/Write	—	—	—	—		R/W		
After reset	—	—	—	—	0	0	0	0
Function					Sets K value for division by N+(16-K)/16			

BR3ADD (0xFFFF_F284)

Sets divisor value for baud rate generator

	BR0CR<BR0ADDE> = 1		BR0CR<BR0ADDE> = 0
BR0ADD<BR0K3:0>	BR0CR<BR0S3:0>		
	0000 (N = 16) ₹ 0001 (N = 1)	0010 (N = 2) ₹ 1111 (N = 15)	0001 (N = 1) (ONLY UART) ₹ 1111 (N = 15) 0000 (N = 16)
0000	Invalid	Invalid	—
0001 (K = 1) ₹ 1111 (K = 15)	Invalid	Divided by N + (16-K)/16	Divided by N

Note 1: The baud rate generator divisor cannot be set to 1 in UART mode if division by N+(16-K)/16 is being used. It cannot be set to 1 at all in I/O interface mode.

Note 2: When using division by N+(16-K)/16, be sure to set K (1 to 15) in BR3ADD <BR3K3:BR3K0> before setting BR3CR<BR3ADDE> to 1. However, if BR3CR <BR3S3:BR3S0> = 0000 or 0001 (i.e. if N = 16 or 1), do not use division by N+(16-K)/16.

Note 3: Division by N+(16-K)/16 can only be used in UART mode. In I/O interface mode, set BR3CR<BR3ADDE> to 0 to disable division by N+(16-K)/16.

Figure 3.11.28 Baud Rate Generator Control Registers (BR3CR and BR3ADD, for SIO3)

	7	6	5	4	3	2	1	0
Bit symbol	—	BR4ADDE	BR4CK1	BR4CK0	BR4S3	BR4S2	BR4S1	BR4S0
Read/Write				R/W				
After reset	0	0	0	0	0	0	0	0
Function	Must always be set to 0.	Division by N+(16-K)/16 0: Disable 1: Enable	00: φT0 01: φT2 10: φT8 11: φT32		Sets value of divisor N			

BR4CR (0xFFFF_F28B)

Selects baud rate generator input clock

00	Internal clock φT0
01	Internal clock φT2
10	Internal clock φT8
11	Internal clock φT32

	7	6	5	4	3	2	1	0
Bit symbol	—	—	—	—	BR4K3	BR4K2	BR4K1	BR4K0
Read/Write	—	—	—	—		R/W		
After reset	—	—	—	—	0	0	0	0
Function					Sets K value for division by N+(16-K)/16			

BR4ADD (0xFFFF_F28C)

Sets divisor value for baud rate generator

BR0ADD<BR0K3:0>	BR0CR<BR0ADDE> = 1		BR0CR<BR0ADDE> = 0
	BR0CR<BR0S3:0>		
	0000 (N = 16) ≀ 0001 (N = 1)	0010 (N = 2) ≀ 1111 (N = 15)	0001 (N = 1) (ONLY UART) ≀ 1111 (N = 15) 0000 (N = 16)
0000	Invalid	Invalid	—
0001 (K = 1) ≀ 1111 (K = 15)	Invalid	Divided by N + (16-K)/16	Divided by N

Note 1: The baud rate generator divisor cannot be set to 1 in UART mode if division by N+(16-K)/16 is being used. It cannot be set to 1 at all in I/O interface mode.

Note 2: When using division by N+(16-K)/16, be sure to set K (1 to 15) in BR4ADD <BR4K3:BR4K0> before setting BR4CR<BR4ADDE> to 1. However, if BR4CR <BR4S3:BR4S0> = 0000 or 0001 (i.e. if N = 16 or 1), do not use division by N+(16-K)/16.

Note 3: Division by N+(16-K)/16 can only be used in UART mode. In I/O interface mode, set BR4CR<BR4ADDE> to 0 to disable division by N+(16-K)/16.

Figure 3.11.29 Baud Rate Generator Control Registers (BR4CR and BR4ADD, for SIO4)

	7	6	5	4	3	2	1	0
Bit symbol	—	BR5ADDE	BR5CK1	BR5CK0	BR5S3	BR5S2	BR5S1	BR5S0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Must always be set to 0.	Division by N+(16-K)/16 0: Disable 1: Enable	00: φT0 01: φT2 10: φT8 11: φT32		Sets value of divisor N			

BR5CR
(0xFFFF_F293)

Selects baud rate generator input clock

00	Internal clock φT0
01	Internal clock φT2
10	Internal clock φT8
11	Internal clock φT32

	7	6	5	4	3	2	1	0
Bit symbol	—	—	—	—	BR5K3	BR5K2	BR5K1	BR5K0
Read/Write	—	—	—	—	R/W			
After reset	—	—	—	—	0	0	0	0
Function					Sets K value for division by N+(16-K)/16			

BR5ADD
(0xFFFF_F294)

Sets divisor value for baud rate generator ←

	BR0CR<BR0ADDE> = 1		BR0CR<BR0ADDE> = 0
BR0ADD<BR0K3:0>	BR0CR<BR0S3:0>		
	0000 (N = 16) ⟨ 0001 (N = 1)	0010 (N = 2) ⟨ 1111 (N = 15)	0001 (N = 1) (ONLY UART) ⟨ 1111 (N = 15) 0000 (N = 16)
0000	Invalid	Invalid	—
0001 (K = 1) ⟨ 1111 (K = 15)	Invalid	Divided by N + (16-K)/16	Divided by N

Note 1: The baud rate generator divisor cannot be set to 1 in UART mode if division by N+(16-K)/16 is being used. It cannot be set to 1 at all in I/O interface mode.

Note 2: When using division by N+(16-K)/16, be sure to set K (1 to 15) in BR5ADD <BR5K3:BR5K0> before setting BR5CR<BR5ADDE> to 1. However, if BR5CR <BR5S3:BR5S0> = 0000 or 0001 (i.e. if N = 16 or 1), do not use division by N+(16-K)/16.

Note 3: Division by N+(16-K)/16 can only be used in UART mode. In I/O interface mode, set BR5CR<BR5ADDE> to 0 to disable division by N+(16-K)/16.

Figure 3.11.30 Baud Rate Generator Control Registers (BR5CR and BR5ADD, for SIO5)

7	6	5	4	3	2	1	0	
TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(For transmission)

SC0BUF
(0xFFFF_F230)

7	6	5	4	3	2	1	0	
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(For reception)

Figure 3.11.31 Serial Transmit/Receive Buffer Register (SC0BUF, for SIO0)

7	6	5	4	3	2	1	0	
TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(For transmission)

SC1BUF
(0xFFFF_F238)

7	6	5	4	3	2	1	0	
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(For reception)
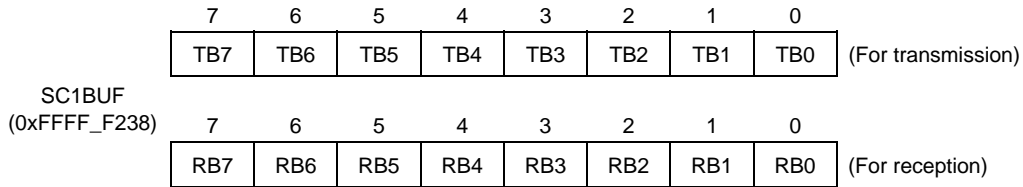
Figure 3.11.32 Serial Transmit/Receive Buffer Register (SC1BUF, for SIO1)

7	6	5	4	3	2	1	0	
TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(For transmission)

SC3BUF
(0xFFFF_F280)

7	6	5	4	3	2	1	0	
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(For reception)
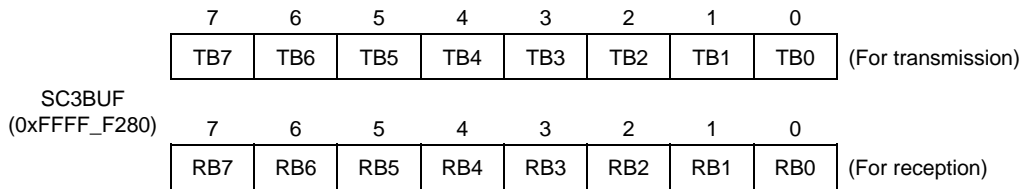
Figure 3.11.33 Serial Transmit/Receive Buffer Register (SC3BUF, for SIO3)

7	6	5	4	3	2	1	0	
TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(For transmission)

SC4BUF
(0xFFFF_F288)

7	6	5	4	3	2	1	0	
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(For reception)
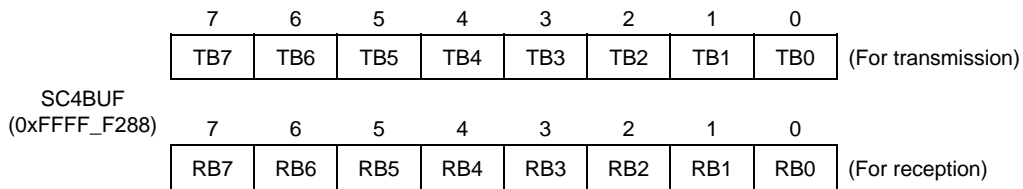
Figure 3.11.34 Serial Transmit/Receive Buffer Register (SC4BUF, for SIO4)

7	6	5	4	3	2	1	0	
TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0	(For transmission)

SC5BUF
(0xFFFF_F290)

7	6	5	4	3	2	1	0	
RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0	(For reception)
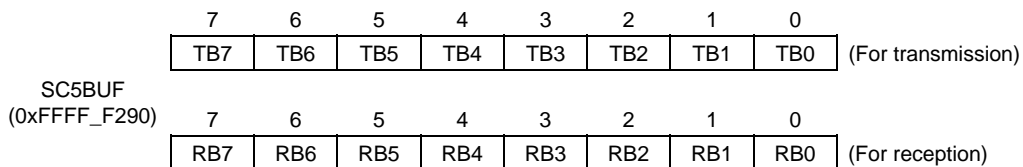
Figure 3.11.35 Serial Transmit/Receive Buffer Register (SC5BUF, for SIO5)

3.11.4　Functional description for each mode

(1)　Mode 0 (I/O interface mode)

　　　This mode comprises two submodes: SCLK output mode, in which the synchronizing clock SCLK is generated internally by the device, and SCLK input mode, in which the synchronizing clock SCLK is input from an external source.

1)　Transmission

　　　If WBUF = 0, that is, the transmit double-buffer is disabled in SCLK output mode, 8 bits of data and the synchronizing clock signal are output on the TXD0 and SCLK0 pins, respectively, each time the CPU writes data to the transmit buffer. When all the data bits have been output, an INTTX0 interrupt is generated.

　　　If WBUF = 1, that is, the transmit double-buffer is enabled, data is transferred from transmit buffer 2 to transmit buffer 1 when the CPU writes data to transmit buffer 2 while transmission is stopped or when data has been transmitted from transmit buffer 1 (shift register). Simultaneously, SC0MOD2<TBEMP> is set to 1 and an INTTX0 interrupt occurs. If transmit buffer 2 does not contain data to be transferred to transmit buffer 1, SCLK0 output is stopped without generating an INTTX0 interrupt.
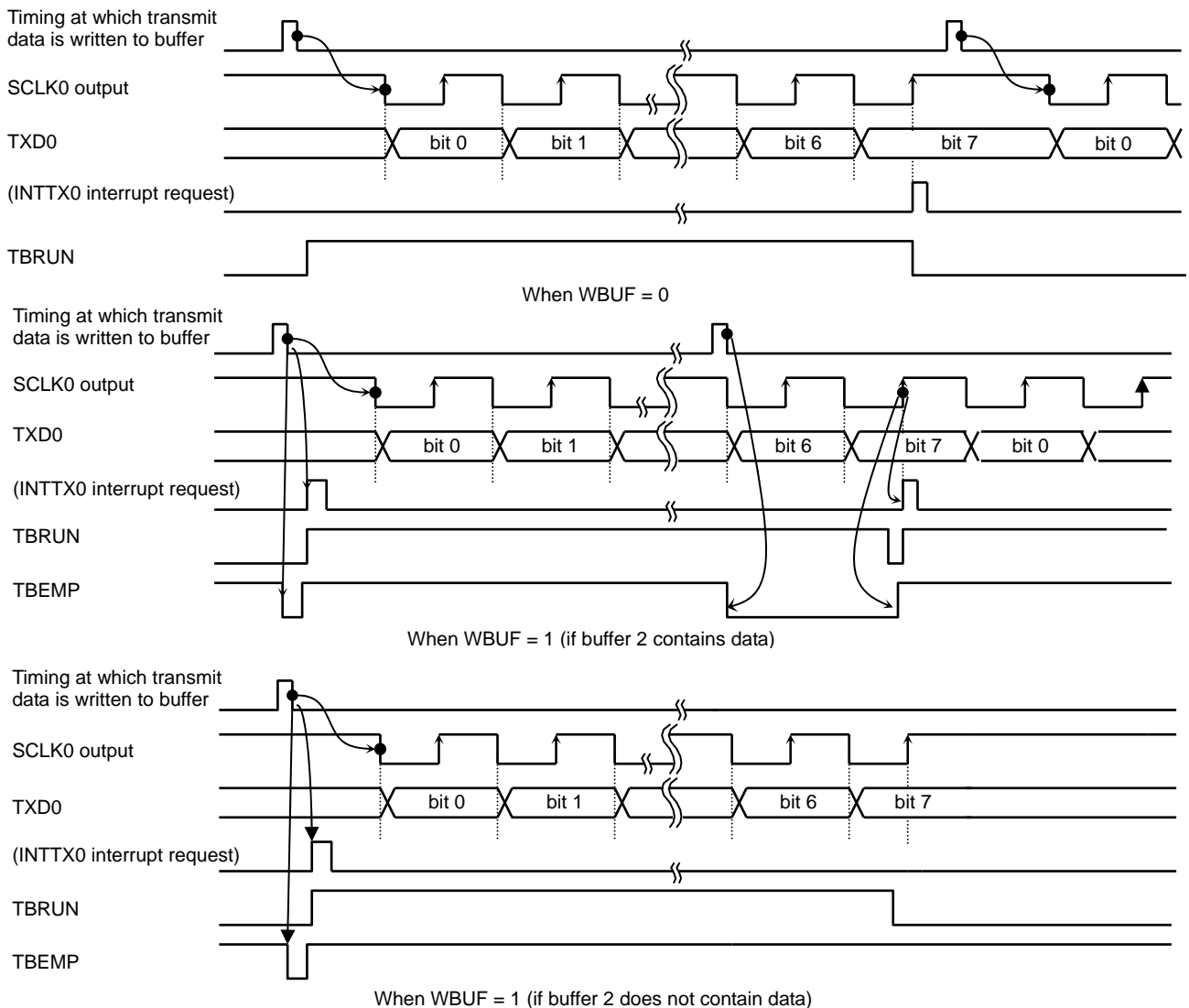
Figure 3.11.36　Transmit Operation in I/O Interface Mode (SCLK0 Output Mode)

If WBUF = 0, that is, the transmit double-buffer is disabled in SCLK input mode, 8 bits of data are output on the TXD0 pin when the SCLK0 input becomes active with data present in the transmit buffer. When all the data bits have been output, an INTTX0 interrupt is generated. Writing the next transmit data must be completed before point A in the shown below.

If WBUF = 1, that is, the transmit double-buffer is enabled, data is transferred from transmit buffer 2 to transmit buffer 1 when the CPU writes data to the transmit buffer before the SCLK0 input becomes active or when data has been transmitted from transmit buffer 1 (shift register). Simultaneously, SC0MOD2<TBEMP> is set to 1 and an INTTX0 interrupt occurs. If the SCLK0 input becomes active when transmit buffer 2 does not contain data, the internal bit counter starts counting but an underrun error flag is set, causing 8 bits of dummy data (FFh) to be transmitted.

When WBUF = 0

When WBUF = 1 (if buffer 2 contains data)

When WBUF = 1 (if buffer 2 does not contain data)
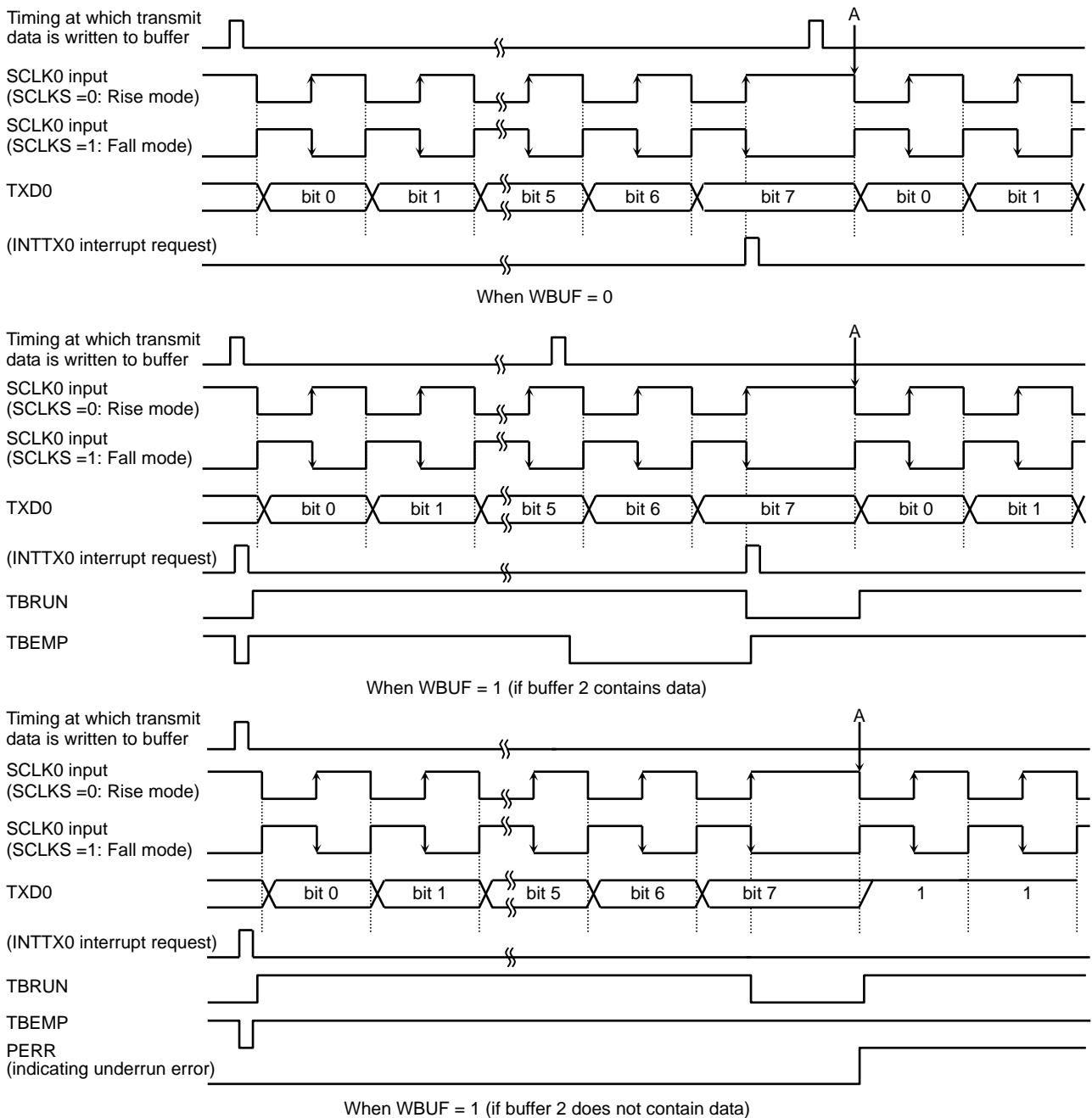
Figure 3.11.37 Transmit Operation in I/O Interface Mode (SCLK0 Input Mode)

2) Reception

If WBUF = 0, that is, the receive double-buffer is disabled in SCLK output mode, the synchronizing clock is output on the SCLK0 pin and the next data item is shifted into receive buffer 1 each time the received data is read by the CPU. When 8 bits of data have been received, an INTRX0 interrupt is generated.

SCLK output is initiated by setting SC0MOD0<RXE> to 1. If WBUF = 1, that is, the receive double-buffer is enabled, the received frame is transferred to transmit buffer 2 and then the next frame is received into receive buffer 1. When data has been transferred from receive buffer 1 to receive buffer 2, SCnMOD2<RBFLL> is set to 1 and an INTRX0 interrupt occurs.

If the CPU/DMAC does not read data from receive buffer 2 before the next eight bits of data have been received, an overrun error occurs, setting SCnCR<OERR>. In that case, SCLK0 output is stopped without generating an INTRX0 interrupt. After an overrun error occurs, reading data from receive buffer 2 causes the data in receive buffer 1 to be transferred to receive buffer 2, generating an INTRX0 interrupt to restart reception.

When WBUF = 0

When WBUF = 1 (if data is read from buffer 2)

When WBUF = 1 (if data is not read from buffer 2)
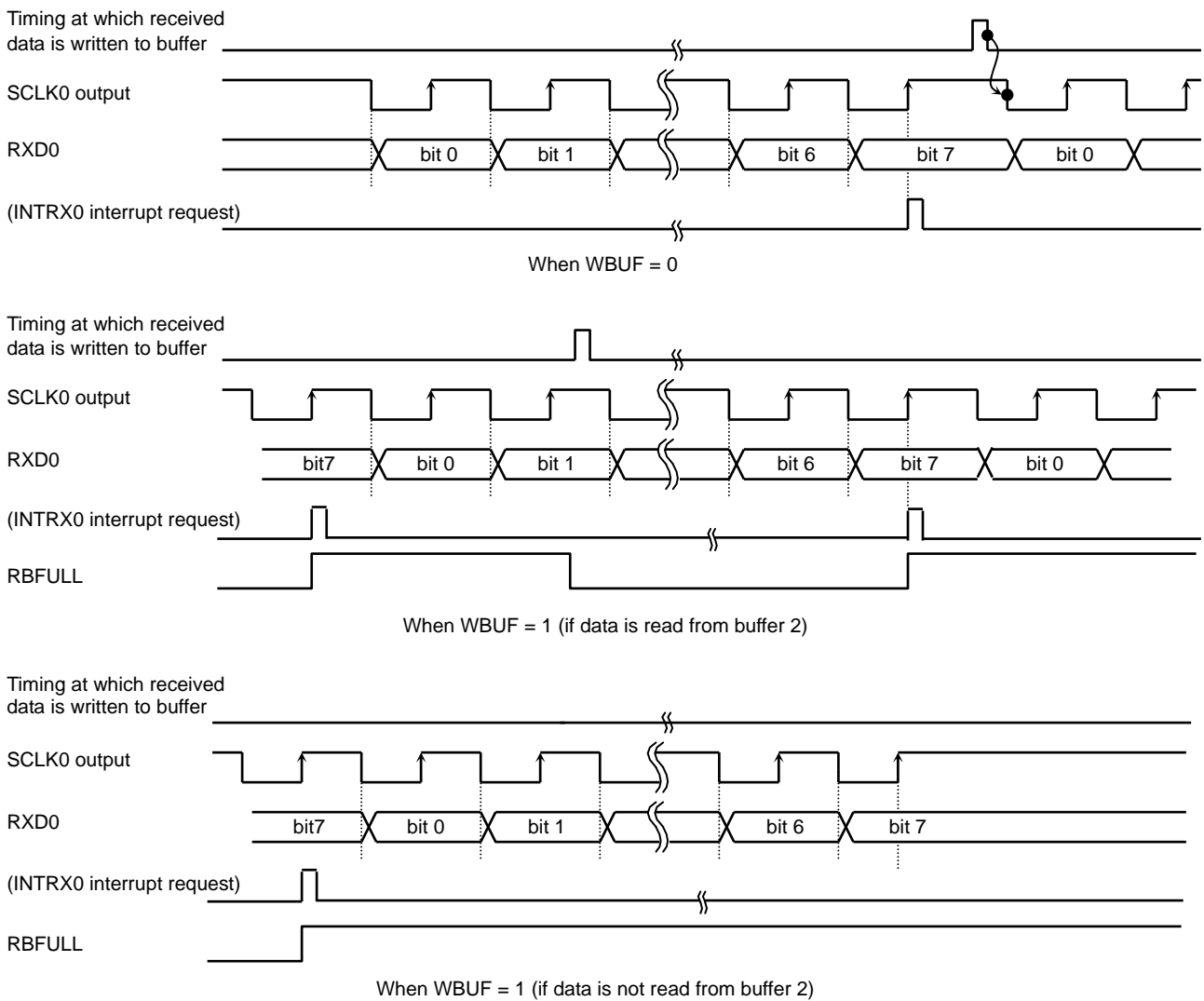
Figure 3.11.38 Receive Operation in I/O Interface Mode (SCLK0 Output Mode)

In SCLK input mode, the receive double-buffer is always enabled. The received frame is transferred to receive buffer 2, so that receive buffer 1 can receive the next frame immediately.

Each time received data has been transferred to receive buffer 2, an INTRX0 interrupt occurs.

If data is read from buffer 2
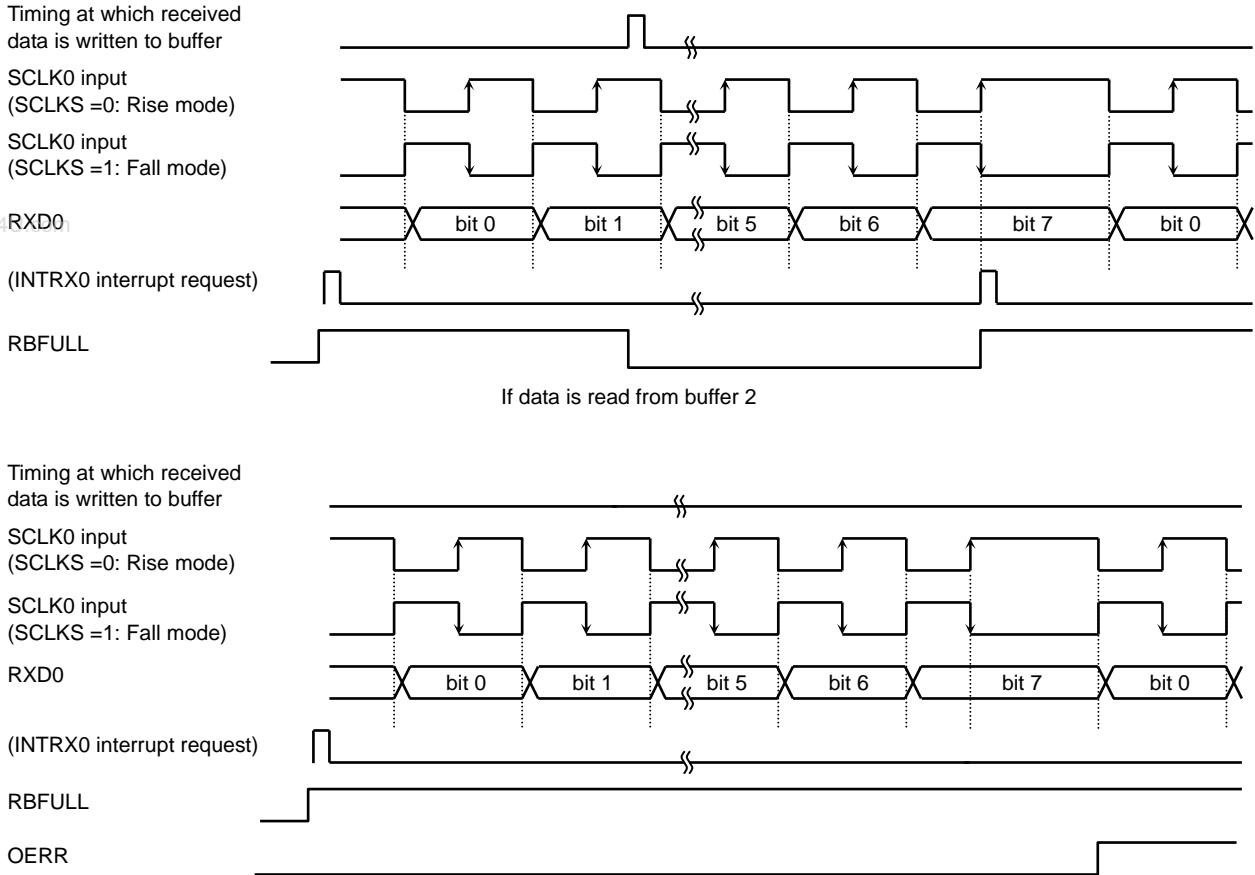
Figure 3.11.39 Receive Operation in I/O Interface Mode (SCLK0 Input Mode)

Note: Before receive operation can be performed in either SCLK input mode or SCLK output mode, reception must be enabled by setting SC0MOD<RXE> to 1.

3)   Transmission/reception (full-duplex)

Setting SC0MOD1<FDPX0> to 1 enables full-duplex communication.

If WBUF = 0, that is, both the transmit and receive double-buffers are disabled in SCLK output mode, writing data to the transmit buffer initiates SCLK output and shifts the received 8-bit data into receive buffer 1, generating a receive interrupt (INTRX0). Simultaneously, the 8-bit data written to the transmit buffer is output on the TXD0 pin. When all bits of data have been transmitted, a transmit interrupt (INTTX0) is generated, causing SCLK output to stop. When the CPU subsequently reads the receive buffer and writes data to the transmit buffer, next transmission/reception starts. Transmission/reception is restarted when the CPU has performed both the read and write, regardless of their sequence.

If WBUF = 1, that is, both the transmit and receive double-buffers are enabled, writing data to the transmit buffer initiates SCLK output and shifts the received 8-bit data into receive buffer 1, which is then transferred to receive buffer 2, generating a receive interrupt (INTRX0). Simultaneously, the 8-bit data written to the transmit buffer is output on the TXD0 pin. When all bits of data have been transmitted, a transmit interrupt (INTTX0) is generated and the next data is transferred from transmit buffer 2 to transmit buffer 1. If transmit buffer 2 does not contain data to be transferred (TBEMP = 1) or receive buffer 2 contains data (RBFLL = 1), SCLK output is stopped. When the CPU subsequently reads the receive buffer and writes data to the transmit buffer, SCLK output is restarted and next transmission/reception starts.
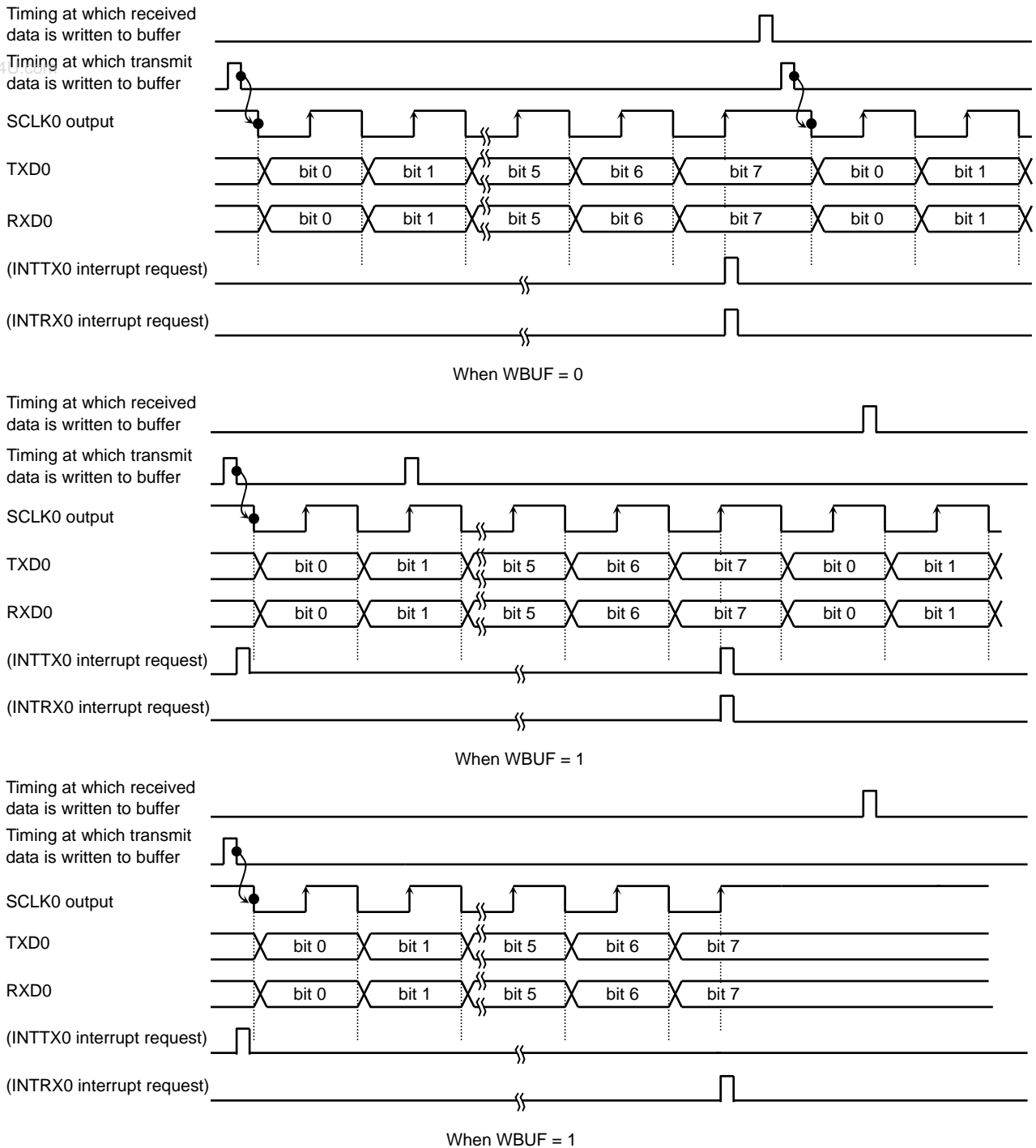
Figure 3.11.40 Transmit/Receive Operation in I/O Interface Mode (SCLK0 Output Mode)

If WBUF = 0, that is, the transmit double-buffer is disabled in SCLK input mode (the receive double-buffer is always enabled in SCLK input mode), 8-bit data is output on the TXD0 pin and 8-bit data is shifted into the receive buffer simultaneously when the SCLK input becomes active with data present in the transmit buffer. When all bits of data have been transmitted, a transmit interrupt (INTTX0) is generated. When all bits of data have been received and then transferred from receive buffer 1 to receive buffer 2, a receive interrupt (INTRX0) is generated. Next transmit data must be written to the transmit buffer before the SCLK pulse for the next frame is input, that is, before point A in the figure below. Because the receive double-buffer is enabled, the received data must be read before the reception of the next frame is completed.

If WBUF = 1, that is, both the transmit and receive double-buffers are enabled, the data in transmit buffer 2 is transferred to transmit buffer 1, generating a transmit interrupt (INTTX0), when all bits of data in transmit buffer 1 have been transmitted. When the received 8-bit data has been shifted into receive buffer 1, the data is transferred to receive buffer 2, generating a receive interrupt (INTRX0). Then, the SCLK input pulse for the next frame initiates the transmission of the data transferred from transmit buffer 2 to transmit buffer 1 and the reception of data into receive buffer 1. If the data in receive buffer 2 is not read before the last bit of the frame is received, an overrun error occurs. If transmit data is not written to transmit buffer 2 before the SCLK pulse for the next frame is input, an underrun error occurs.

When WBUF = 0

Timing at which received
data is written to buffer

Timing at which transmit
data is written to buffer

SCLK0 input

TXD0

| | bit 0 | bit 1 | bit 5 | bit 6 | bit 7 | bit 0 | bit 1 | |

RXD0

| | bit 0 | bit 1 | bit 5 | bit 6 | bit 7 | bit 0 | bit 1 | |

(INTTX0 interrupt request)

(INTRX0 interrupt request)

When WBUF = 1 (without error)

Timing at which received
data is written to buffer

Timing at which transmit
data is written to buffer

SCLK0 input

TXD0

| | bit 0 | bit 1 | bit 5 | bit 6 | bit 7 | bit 0 | bit 1 | |

RXD0

| | bit 0 | bit 1 | bit 5 | bit 6 | bit 7 | bit 0 | bit 1 | |

(INTTX0 interrupt request)

(INTRX0 interrupt request)

PERR (underrun error)

When WBUF = 1 (if error occurs)

(2) Mode 1 (7-bit UART mode)

Setting the SM1 and SM0 bits of the serial channel mode register SC0MOD to 01 places the device into 7-bit UART mode.

In this mode, a parity bit can be used. Parity can be enabled or disabled using the PE bit of the serial channel control register SC0CR. When PE = 1 (parity enabled), even or odd parity can be selected using SC0CR<EVEN>. The STOP bit length can also be specified using SCnMOD2<SBLEN>.

Example: To transmit data in the following format, set the control registers as shown below.

```
......        ........
  \  start / bit 0 X 1 X 2 X 3 X 4 X 5 X 6 X even  \ stop
                                              parity
```

←——— Direction of transfer (transfer rate = 2400 bps at fc = 24.576 MHz)

∗ Clock conditions

System clock:	High-speed (fc)
High-speed clock gear:	× 1 (fc)
Prescaler clock:	$f_{periph}/4$ ($f_{periph} = f_{sys}$)

```
              7 6 5 4 3 2 1 0
PDCR    ← – – – – – – – 1        ⎫  Set PD0 to TXD0 pin.
PDFC    ← – – – – – – – 1        ⎭
SC0MOD  ← X 0 – X 0 1 0 1        Select 7-bit UART mode.
SC0CR   ← X 1 1 X X X 0 0        Select even parity.
BR0CR   ← 0 0 1 0 1 0 1 0        Set transfer rate to 2400 bps.
IMCCLH  ← – – 1 1 0 1 0 0        Enable INTTX0 interrupt and set its priority level to 4.
SC0BUF  ← ∗ ∗ ∗ ∗ ∗ ∗ ∗ ∗        Set transmit data.
```

Note: X = Don't care; "—" = No change

(3) Mode 2 (8-bit UART mode)

Setting the SM1 and SM0 bits of SC0MOD to 10 places the device into 8-bit UART mode. In this mode, a parity bit can be used. Parity can be enabled or disabled using SC0CR<PE>. When PE = 1 (parity enabled), even or odd parity can be selected using SC0CR<EVEN>.

Example: To transmit data in the following format, set the control registers as shown below.
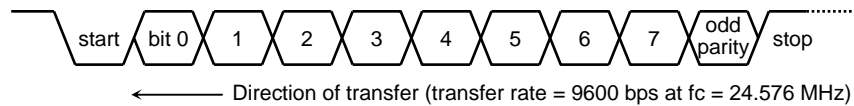
```
                                                        .........
  \ start / bit 0 X 1 X 2 X 3 X 4 X 5 X 6 X 7 X odd   \ stop
                                                parity
```

←——— Direction of transfer (transfer rate = 9600 bps at fc = 24.576 MHz)

∗ Clock conditions

System clock:	High-speed (fc)
High-speed clock gear:	× 1 (fc)
Prescaler clock:	$f_{periph}/4$ ($f_{periph} = f_{sys}$)

• Setting in the main routine

```
              7 6 5 4 3 2 1 0
PDCR    ← – – – – – – 0 –        Set PD1 (RxD0) to input pin.
SC0MOD  ← – 0 0 X 1 0 0 1        Select 8-bit UART mode.
SC0CR   ← X 0 1 X X X 0 0        Select odd parity.
BR0CR   ← 0 0 0 1 0 1 0 1        Set transfer rate to 9600 bps.
IMCCLL  ← – – 1 1 0 1 0 0        Enable INTRX0 interrupt and set its priority level to 4.
SC0MOD  ← – – 1 X – – – –        Enable reception.
```

• Example of interrupt routine processing

```
INTCLR  ← X X 1 1 0 0 0 0               Clear interrupt request.
Reg.    ← SC0CR AND 0x1C        ⎫
if Reg. ≠ 0 then ERROR processing  ⎬  Check for errors.
Reg.    ← SC0BUF                        Read received data.
End of interrupt processing
```

Note: X = Don't care; "—" = No change

(4) Mode 3 (9-bit UART mode)

Setting SC0MOD0<SM1:SM0> to 11 places the device into 9-bit UART mode. In this mode a parity bit cannot be used; hence, parity should be disabled by setting SC0CR<PE> to 0.

During transmission the most significant bit (the 9th bit) is written to the TB8 bit of the serial channel mode register SC0MOD0. During reception the bit is stored in the RB8 bit of the serial channel control register SC0CR. Data is always written to or read from the buffer register the most significant bit first and then the rest of the data from SC0BUF. The STOP bit length can be specified using SCnMOD2<SBLEN>.

Wake-up function

In 9-bit UART mode, slave controller wake-up can be enabled by setting SC0MOD0<WU> to 1. An INTRX0 interrupt will only be generated if RB8 = 1.
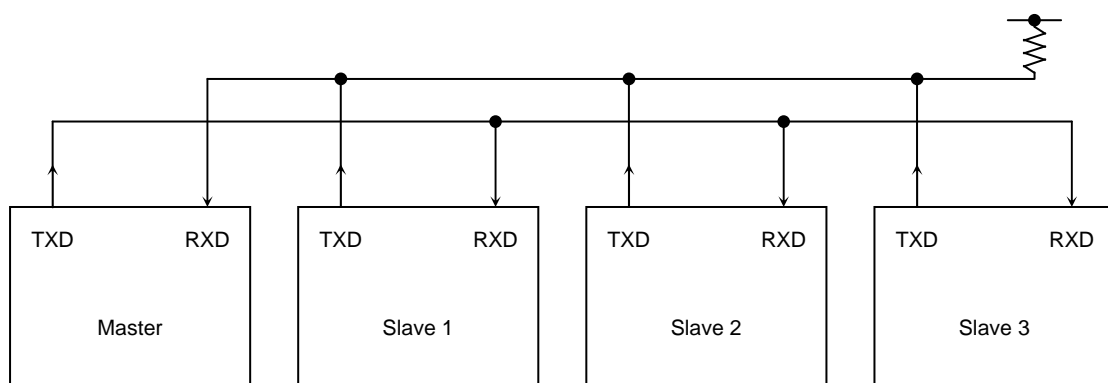
Note: The slave controller's TXD pin must always be placed in open-drain output mode by setting the ODE register accordingly.

Figure 3.11.41 Serial Link Using the Wake-Up Function

Protocol

1) The master and slave controllers are placed in 9-bit UART mode.

2) Each slave controller is enabled for reception by setting SC0MOD0<WU> to 1.

3) The master controller transmits one frame of data including the 8-bit slave controller selection code. At this point the most significant bit (bit 8: TB8) is set to 1.

| start | bit 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | stop |
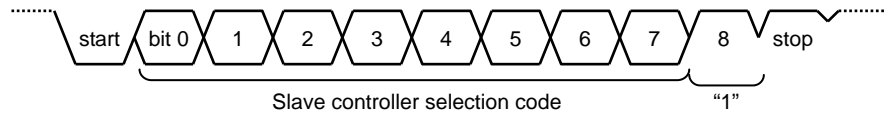
Slave controller selection code   "1"

4) Each slave controller receives the above frame. The slave controller whose selection code matches the transmitted selection code clears its WU bit to 0.

5) The master controller transmits data to the selected slave controller (the one whose SC0MOD0<WU> bit has been cleared to 0). At this point the most significant bit (bit 8: TB8) is set to 0.

| start | bit 0 | 1 | 2 | 3 | 4 | 5 | 6 | 7 | bit 8 | stop |

Data   "0"
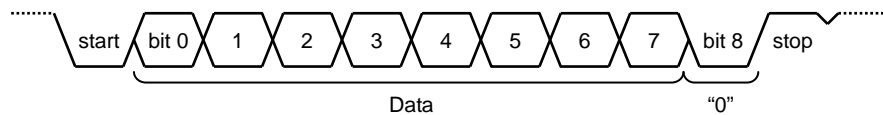
6) No interrupt (INTRX0) is generated for the slave controllers whose WU bit remains 1 because the most significant bit of the received data (bit 8: RB8) = 0. These slave controllers ignore the received data. The slave controller whose WU bit has been cleared to 0 can transmit data to the master controller so as to notify the master controller that it has finished receiving.

Example settings: Serial link with two slave controllers using the internal clock $f_{sys/2}$ as the transfer clock

| TXD   RXD | TXD   RXD | TXD   RXD |
| Master | Slave 1 | Slave 2 |

Selection code   Selection code
00000001   00001010
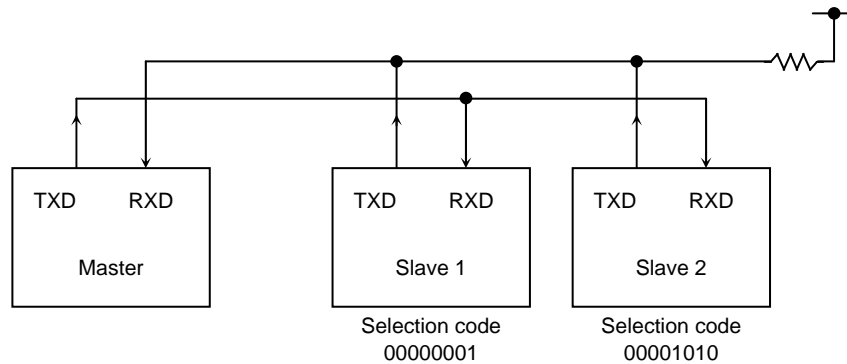
- Master controller settings

    Main routine
    ```
                  7 6 5 4 3 2 1 0
    PDCR      ← – – – – – – 0 1
    PDFC      ← – – – – – – X 1
    IMCCLL    ← – – 1 1 0 1 0 1
    IMCCLH    ← – – 1 1 0 1 0 0
    SC0MOD0   ← 1 0 1 0 1 1 1 0
    SC0BUF    ← 0 0 0 0 0 0 0 1
    ```
    Set PD0 to TXD0 and PD1 to RXD0.

    Enable INTRX0 and set interrupt level to 5.
    Enable INTTX0 and set interrupt level to 4.
    Select 9-bit UART mode and set transfer clock to $f_{sys/2}$.
    Set selection code for slave 1.

    Interrupt routine (INTTX0)

    ```
    INTCLR    ← X X 1 1 0 0 0 1
    SC0MOD0   ← 0 – – – – – – –
    SC0BUF    ← * * * * * * * *
    ```
    Clear interrupt request.
    Set TB8 to 0.
    Set transmit data.

    End of interrupt processing

- Slave settings

    Main routine
    ```
                  7 6 5 4 3 2 1 0
    PDCR      ← – – – – – – 0 1
    PDFC      ← – – – – – – X 1
    ODE       ← X X – – – – – 1
    IMCCLL    ← – – 1 1 0 1 1 0
    IMCCLH    ← – – 1 1 0 1 0 1
    SC0MOD0   ← 0 0 1 1 1 1 1 0
    ```
    Set PD0 to TXD (open-drain output) and PD1 to RXD.

    Enable INTTX0 and INTRX0.

    Select 9-bit UART mode and set transfer clock to $f_{sys/2}$ and WU to 1.

    Interrupt routine (INTRX0)
    ```
                  7 6 5 4 3 2 1 0
    INTCLR    ← X X 1 1 0 0 0 0
    Reg.      ← SC0BUF
    if  Reg. = selection code
    Then
    SC0MOD0   ← – – – 0 – – – –
    ```
    Clear interrupt request.

    Clear WU to 0.

3.12 Serial Bus Interface (SBI)

The TMP1942 contains one serial bus interface (SBI) channel. The serial bus interface has the following two operating modes:

- I^{2}C bus mode (multi-master)
- Clock-synchronous 8-bit SIO mode

In I^{2}C bus mode, the serial bus interface can be connected to external devices via PF4 (SDA) and PF5 (SCL). In clock-synchronous 8-bit SIO mode, it can be connected to external devices via PF3 (SCK), PF4 (SO) and PF5 (SI).

The following table shows the pin settings for each mode:

	ODE <ODEF5, F4>	PFCR <PF5C, PF4C, PF3C>	PAFC <PF5F, PF4F, PF3F>
I^{2}C bus mode	11	11X	110
Clock-synchronous 8-bit SIO mode	XX	011 010	111

X: Don't care

3.12.1 Configuration
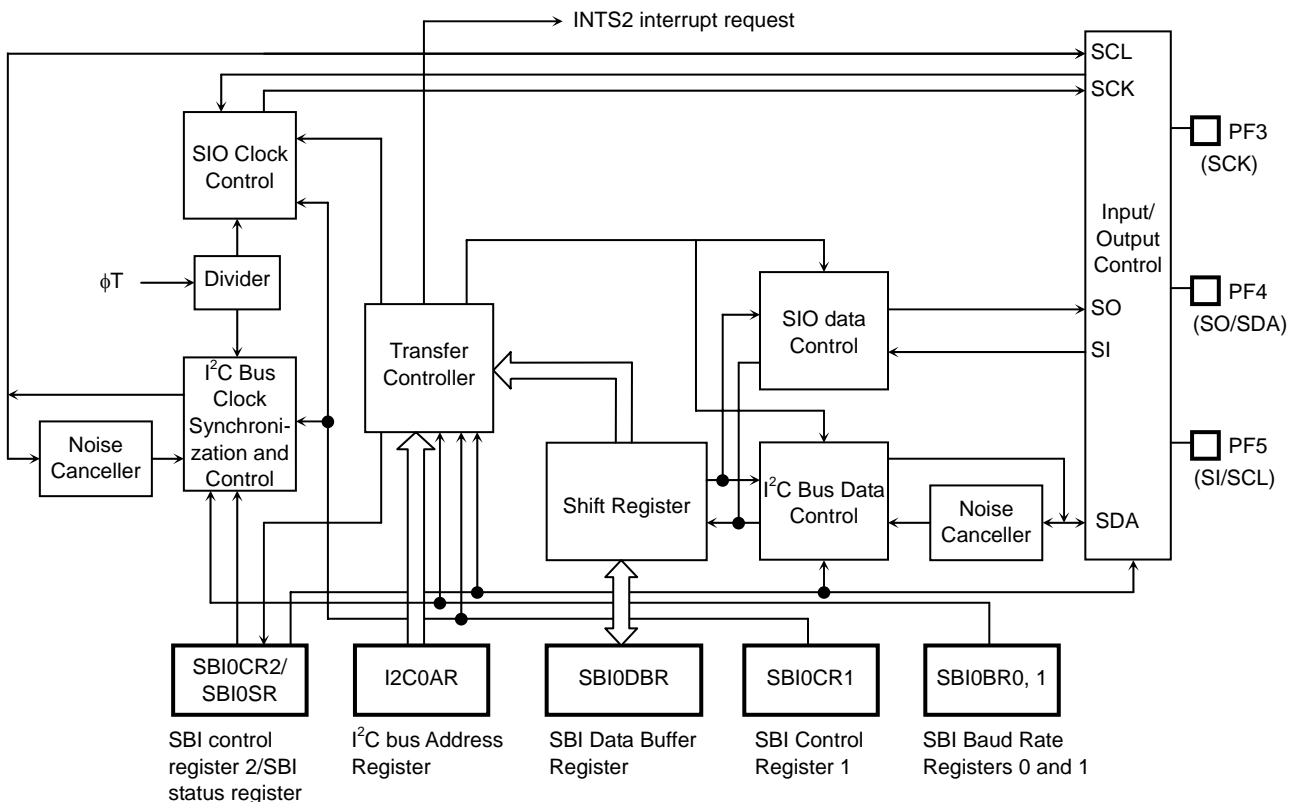
3.12.2   Control

The following registers are used to control the serial bus interface and monitor its operating status:

- Serial bus interface control register 1 (SBI0CR1)
- Serial bus interface control register 2 (SBI0CR2)
- Serial bus interface data buffer register (SBI0DBR)
- I^2C bus address register (I2C0AR)
- Serial bus interface status register (SBI0SR)
- Serial bus interface status register 0 (SBI0BR0)
- Serial bus interface status register 1 (SBI0BR1)

The functions of the above registers vary according to the current operating mode of the serial bus interface. For details, refer to Section 3.12.4, "Control in I^2C bus mode", and Section 3.12.7, "Control in clock-synchronous 8-bit SIO mode".

3.12.3 I^2C Bus Mode Data Formats

Figure 3.12.1 shows the serial bus interface data formats used in I^2C bus mode.

(a)   Addressing format
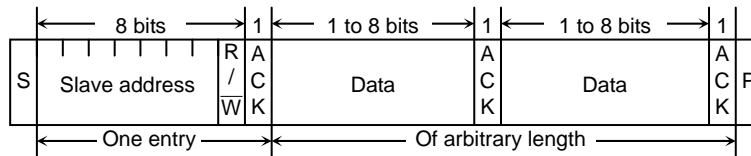
(b)   Addressing format (with restart)
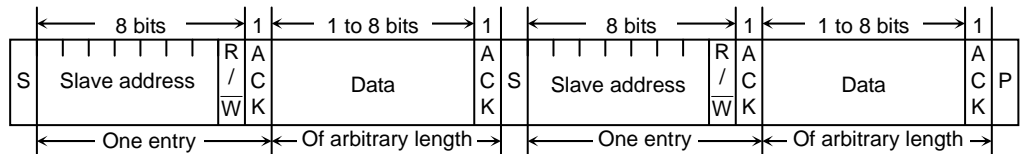
(c)   Free data format (format used to transfer data from master device to slave device)
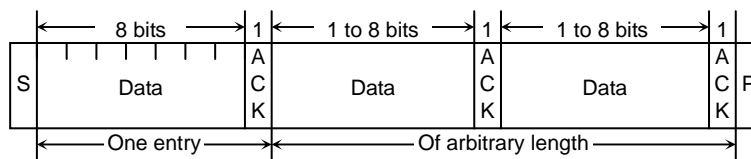
S: Start condition

R/W: Direction bit

ACK: Acknowledge bit

P: Stop condition

Figure 3.12.1 I^2C Bus Mode Data Formats

3.12.4 I²C Bus Mode Control Registers

When the serial bus interface is operated in I²C bus mode, the following registers are used to control the interface and to monitor its operating status:

Serial Bus Interface Control Register 1

SBI0CR1
(0xFFFF_F240)

		7	6	5	4	3	2	1	0
	Bit symbol	BC2	BC1	BC0	ACK	—	SCK2	SCK1	SCK0/ SWRMON
	Read/Write		W		R/W			W	R/W
	After Reset	0	0	0	0		0	0	1
	Function	Selects number of bits to be transferred (Note 1)			Ack clock 0: Do not generate 1: Generate		Selects internal SCL output clock frequency (Note 2) and monitors reset state		

Selects internal SCL output clock frequency <SCK2:SCK0> (for write)

000	n=4	400	kHz
001	n=5	222	kHz
010	n=6	118	kHz
011	n=7	60.6	kHz
100	n=8	30.8	kHz
101	n=9	15.5	kHz
110	n=10	7.78	kHz
111		reserved	

System clock:: fc(=32 MHz)
Clock gear: : fc/1
ϕT0 = fperiph/4 (= 8 MHz)
Frequency = $\dfrac{\phi T0}{2n+4}$ [Hz]

Software reset state monitor <SWRMON> (for read)

0	Software reset in progress
1	Software reset not in progress

Selects number of bits to be transferred

<BC2:0>	<ACK> = 0		<ACK> = 1	
	Number of clock cycles	Data length	Number of clock cycles	Data length
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 SBI0CR1<BC2:BC0> to 000 before switching the device to clock-synchronous 8-bit SIO mode.

Note 2: For details of the SCL line clock frequency, refer to Section 3.12.5 (3), "Serial clock".

Figure 3.12.2 I²C Bus Mode Registers

Serial Bus Interface Control Register 2

		7	6	5	4	3	2	1	0
SBI0CR2	Bit symbol	MST	TRX	BB	PIN	SBIM1	SBIM0	SWRST1	SWRST0
(0xFFFF_F243)	Read/Write			W			W (Note 1)		W (Note 1)
	After Reset	0	0	0	1	0	0	0	0
	Function	Selects master/ slave 0: Slave 1: Master	Selects transmit/ receive 0: Receive 1: Transmit	Start/stop generation 0: Generate stop state 1: Generate start state	Cancels INTSBI interrupt request 0: — 1: Cancel interrupt request	Selects serial bus interface operating mode (Note 2) 00: Port mode 01: SIO mode 10: I^2C bus mode 11: (Reserved)		Generates software reset A reset can be generated by writing 10 and then 01 to these bits.	

Selects 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: When read, this register functions as the SBI0SR register.

Note 2: Check to see that the bus is free before switching the device to port mode. Also, check that input signals on the ports are High before switching from port mode to I^2C bus mode or clock-synchronous 8-bit SIO mode.

Figure 3.12.3 I^2C Bus Mode Registers

Table 3.12.1 Output Clock (ϕT0) Resolutions

@fc=32 MHz

Peripheral Clock Selection <FPSEL>	Clock Gear Value <GEAR1:0>	Selected Prescaler Clock <PRCK1:0>	Prescaler Output Clock Resolution ϕT0
0 (fgear)	00 (fc)	00 (fperiph/4)	$fc/2^2$ (0.125 μs)
		01 (fperiph/2)	—
		10 (fperiph)	—
	01 (fc/2)	00 (fperiph/4)	$fc/2^3$ (0.25 μs)
		01 (fperiph/2)	—
		10 (fperiph)	—
	10 (fc/4)	00 (fperiph/4)	$fc/2^4$ (0.5 μs)
		01 (fperiph/2)	—
		10 (fperiph)	—
	11 (fc/8)	00 (fperiph/4)	$fc/2^5$ (1.0 μs)
		01 (fperiph/2)	—
		10 (fperiph)	—
1 (fc)	00 (fc)	00 (fperiph/4)	$fc/2^2$ (0.25 μs)
		01 (fperiph/2)	—
		10 (fperiph)	—
	01 (fc/2)	00 (fperiph/4)	—
		01 (fperiph/2)	—
		10 (fperiph)	—
	10 (fc/4)	00 (fperiph/4)	—
		01 (fperiph/2)	—
		10 (fperiph)	—
	11 (fc/8)	00 (fperiph/4)	—
		01 (fperiph/2)	—
		10 (fperiph)	—

Note: The – character means "Don't use".

Serial Bus Interface Status Register

		7	6	5	4	3	2	1	0
SBI0CR (0xFFFF_F243)	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 0: Slave 1: Master	Transmit/r eceive selection 0: Receive 1: Transmit	I2C bus status 0: Bus free 1: Bus busy	INTS2 interrupt request status 0: Interrupt request generated 1: Interrupt request cancelled	Arbitration lost detection 0: — 1: Detected	Slave address match detection 0: — 1: Detected	General call detection 0: — 1: Detected	Last received bit 0: 0 1: 1

Last received bit

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

Slave address match detection

0	—
1	Matching slave address or general call has been detected

Arbitration lost detection

0	—
1	Arbitration lost has been detected
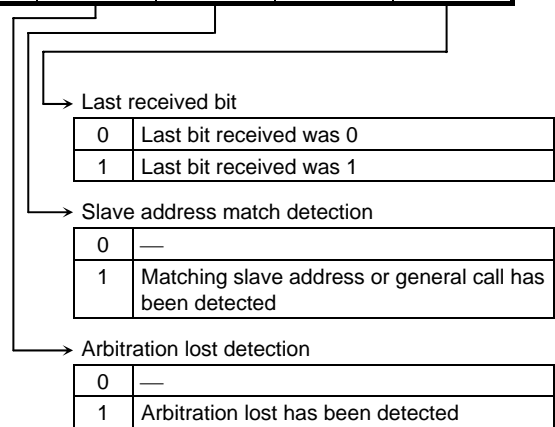
Note: When written, this register functions as the SBI0CR2 register.

Figure 3.12.4 I^{2}C Bus Mode Registers

Serial Bus Interface Baud Rate Register 0

		7	6	5	4	3	2	1	0
SBI0BR0	Bit symbol	—	I2SBI0	—	—	—	—	—	—
(0xFFFF_F244)	Read/Write	—	R/W	—	—	—	—	—	W
	After Reset	—	0	—	—	—	—	—	—
	Function		IDLE 0: Idle 1: Operate						Must always be set to 0.

Operation in IDLE mode

0	Idle
1	Operate

Serial Bus Interface Baud Rate Register 1

		7	6	5	4	3	2	1	0
SBI0BR1	Bit symbol	P4EN	—	—	—	—	—	—	—
(0xFFFF_F245)	Read/Write	R/W	—	—	—	—	—	—	W
	After Reset	0	—	—	—	—	—	—	—
	Function	Internal clock 0: Stopped 1: Operate							

Operation in IDLE mode

0	Idle
1	Operate

Serial Bus Interface Baud Rate Register

		7	6	5	4	3	2	1	0
SBI0DBR	Bit symbol	DB7	DB6	DB5	DB4	DB3	DB2	DB1	DB0
(0xFFFF_F241)	Read/Write	R (receive)/W (transmit)							
	After Reset	Undefined							

Note: When writing transmit data, make sure that the data is MSB-justified (bit 7 is the MSB).

I^2C Bus Address Register

		7	6	5	4	3	2	1	0
SBI0BR1	Bit symbol	SA6	SA5	SA4	SA3	SA2	SA1	SA0	ALS
(0xFFFF_F242)	Read/Write	W							
	After Reset	0	0	0	0	0	0	0	0
	Function	Specifies slave address when device is operating as slave device							Specifies address recognition mode

Operation in IDLE mode

0	Idle
1	Operate

Figure 3.12.5 I^2C Bus Mode Registers

3.12.5 Control in I²C Bus Mode

(1) Specifying acknowledgment mode

Setting SBI0CR1<ACK> to 1 causes the serial bus interface to operate in acknowledgment mode. When operating as the master device, the device allows one extra clock cycle for an acknowledge signal. In transmitter mode, the device releases the SDA pin during this clock cycle so that it can receive an acknowledge signal from the receiver. In receiver mode, the device pulls the SDA pin Low during this clock cycle, thus generating an acknowledge signal.

Setting SBI0CR1<ACK> to 0 causes the serial bus interface to operate in non-acknowledgment mode, in which case the device will not generate an extra clock cycle for an acknowledge signal.

(2) Selecting the number of bits to be transferred

SBI0CR1<BC2:BC0> can be used to specify the number of bits in the next data item to be transmitted or received.

Since the BC2:BC0 bits are cleared to 000 as a start condition, the slave address and direction bit are always transferred as eight bits. In all other cases, the BC2:BC0 bits hold the value which has been set.

(3) Serial clock

1) Clock source

The SBI0CR1<SCK2:SCK0> bits are used to select the maximum transfer frequency of the serial clock which is output on the SCL pin in master mode.
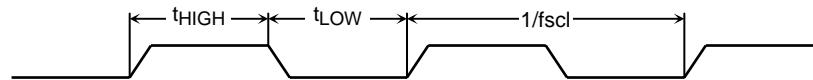
$t_{LOW} = 2^n/\phi T0$

$t_{HIGH} = 2^n/\phi T0 + 4/\phi T0$

$fscl = 1/(t_{Low} + t_{HIGH})$

$= \dfrac{\phi T0}{2n + 4}$

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

Figure 3.12.6 Clock Source

2)   Clock synchronization

In I²C bus mode, a master device which first pulls the clock line Low will disable the clocks of other master devices which are outputting a High clock pulse, thus implementing wired-AND bus configuration. Therefore, any master which is outputting a High clock pulse must detect the situation and take appropriate action.

Since the serial bus interface has a clock synchronization function, transfers are always performed correctly even when multiple master devices are present on the bus.

The clock synchronization procedure is described below using an example in which there are two masters on the bus.
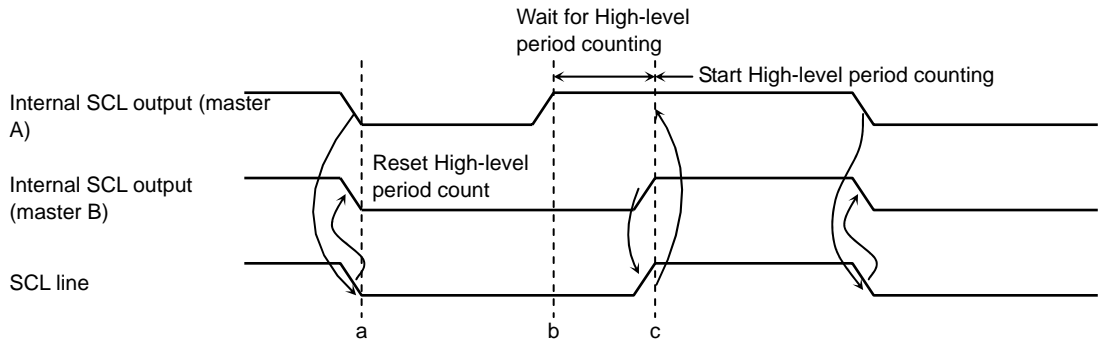
Figure 3.12.7 Clock Synchronization Example

Master A pulls the internal SCL output Low at point 'a' so that the bus SCL line goes Low. Master B detects this and resets its High-level period count before pulling its internal SCL output Low.

Master A finishes Low-level period counting at point 'b', releasing its internal SCL output back High. However, since master B is still holding the SCL line Low, master A does not start High-level period counting. At point "c", when master B has released its internal SCL output back High and the bus SCL line goes High, master A detects these conditions and starts High-level period counting.

Thus, the bus clock frequency is determined by the master connected to the bus which has the shortest High-level period and the master connected to the bus which has the longest Low-level period.

(4)   Setting the slave address and selecting address recognition mode

To operate the device as a slave device, set the slave address in I2C0AR<SA6:SA0> and <ALS>. Setting ALS to 0 selects address recognition mode.

(5)   Specifying a master or slave

Setting SBI0CR2<MST> to 1 causes the device to operate as a master device.

Setting SBI0CR2<MST> to 0 causes the device to operate as a slave device. If a stop condition or arbitration lost is detected on the bus, SBI0CR2<MST> is automatically cleared to 0 by hardware.

(6) Selecting a transmitter or receiver

Setting SBI0CR2<TRX> to 1 causes the device to operate as a transmitter. Setting SBI0CR2<TRX> to 0 causes the device to operate as a receiver.

In slave mode,

- when transferring data in addressing format

- when the received slave address is the same as the value set in I2C0AR

- when a general call (all 8 bits of data after a start condition are 0) is received

TRX is set to 1 by hardware when the direction bit ($R/\overline{W}$) sent from the master device is 1 or set to 0 when the direction bit is 0.

In master mode, when acknowledgement is returned from a slave device, TRX changes to 0 by hardware if the transmitted direction bit is 1 or changes to 1 if the transmitted direction bit is 0. When no acknowledgement is returned, TRX remains unchanged.

If a stop condition or arbitration lost is detected on the bus, SBI0CR2<TRX> is automatically cleared to 0 by hardware.

(7) Generating a start/stop condition

When SBI0SR<BB> = 0, writing 1s to SBI0CR2<MST, TRX, BB, PIN> causes a start condition and 8-bit data to appear on the bus. Ensure that SBI0CR1<ACK> has been set to 1 beforehand.
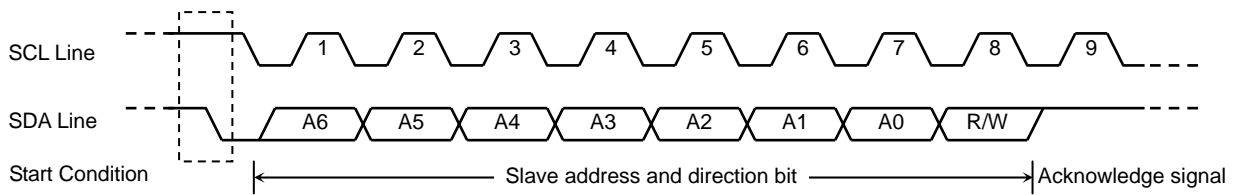
Figure 3.12.8 Generating a Start Condition and Slave Address

When BB = 1, writing 1s to SBI0CR2<MST, TRX, PIN> and a 0 to SBI0CR2<BB> initiates a stop condition output sequence on the bus. Do not change the contents of SBI0CR2<MST, TRX, BB, PIN> until a stop condition has been generated on the bus.
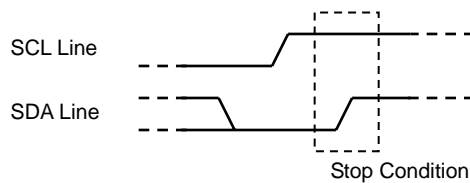
Figure 3.12.9 Generating a Stop Condition

The bus status can be determined by reading SBI0SR<BB>. SBI0SR<BB> is set to 1 (bus busy state) upon the detection of a start condition on the bus or reset to 0 (bus free state) upon the detection of a stop condition.

(8) Requesting interrupt service and canceling requests

When a serial bus interface interrupt request (INTS2) occurs, SBI0CR2<PIN> is reset to 0. The SCL line is held Low while SBI0CR2<PIN> = 0.

PIN is reset to 0 when the device has finished transmitting or receiving one word of data, and set to 1 when data is written to or read from SBI0DBR. There is a delay of tLOW between PIN being set to 1 and the SCL line being released.

In address recognition mode (i.e. when I2C0CR<ALS> = 0), PIN is reset to 0 when the slave address received matches the value set in I2C0AR or when a general call is received (i.e. when the eight data bits after the start condition are all 0). Writing a 1 to SBI0CR2<PIN> in the program sets it to 1; however, writing a 0 to PIN does not clear it to 0.

(9) Serial bus interface operating mode

The SBI0CR2<SBIM1:SBIM0> bits are used to set the operating mode of the serial bus interface. To use the serial bus interface in I^2C bus mode, set SBI0CR2<SBIM1:SBIM0> to 10. Ensure that the bus is free before switching from this mode to port mode.

(10) Monitoring detection of arbitration lost

Since multi-master operation is possible in I^2C bus mode (i.e. two or more masters may exist on the bus simultaneously), a procedure for arbitrating among masters contending for bus control is needed in order to guarantee the integrity of data being transferred.

Any attempt to generate a start condition in the bus busy state will result in "arbitration lost"; data is not output on the SCL or SDA line. The data on the SDA line is used for bus arbitration in I^2C bus mode.

The arbitration procedure is described below using an example in which two masters are residing on the bus simultaneously. Masters A and B output the same data until the bit at point "a", at which point master A outputs a Low signal and master B a High signal. Since the SDA line of the bus has wired-AND configuration, it is pulled Low by master A. When the SCL line goes High at point "b", the slave device latches the SDA line data (i.e. the data output by master A). The data output by master B at this time has no effect and is ignored. This condition of master B is referred to as "arbitration lost". Master B releases the SDA pin so that it will not affect data output by other masters. If more than one master transmits the same first data word, the arbitration procedure will be continued on the next and subsequent words.
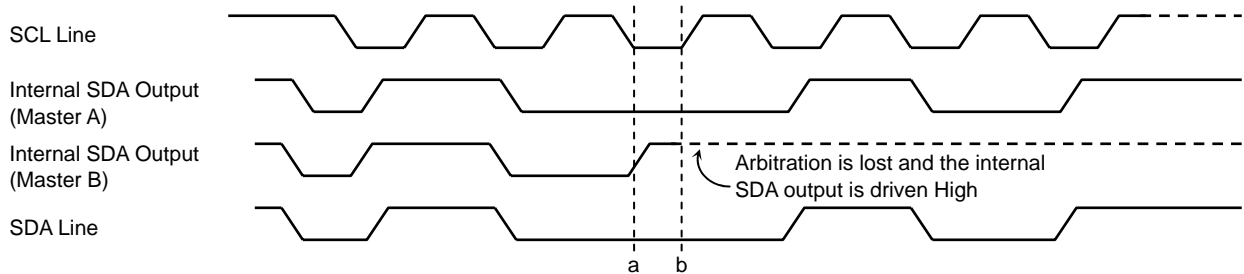
Figure 3.12.10 Arbitration Lost

The internal SDA output level for each master is compared with the level of the bus SDA line at the rising edge of the SCL clock. If the levels do not match, it is assumed that arbitration is lost and SBI0SR<AL> will be set to 1.

At this point the SBI0SR<MST,TRX> bits are reset to 00, thus placing the master into slave receiver mode. SBI0SR<AL> is cleared to 0 by writing data to or reading data from SBI0DBR, or by writing data to SBI0CR2.
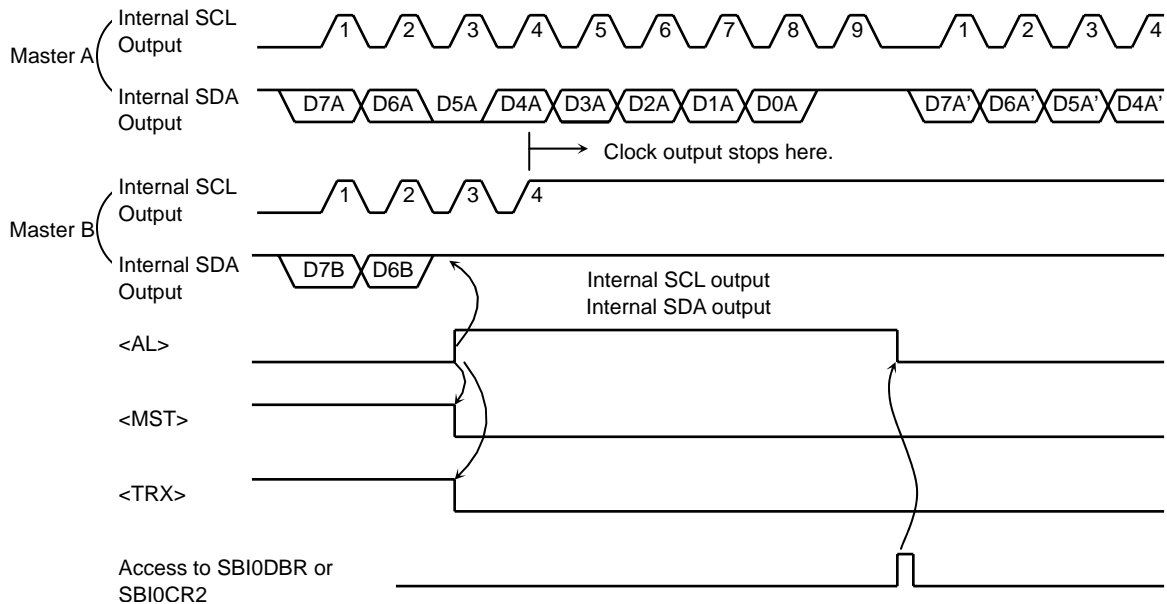
Figure 3.12.11 Example for Master B (D7A = D7B, D6A = D6B)

(11) Monitoring detection of a slave address match

If the device is operating as a slave device in address recognition mode (I2C0AR<ALS> = 0) and receives a general call or a slave address of the same value as that set in I2C0AR, SBI0SR<AAS> will be set to 1. If I2C0AR<ALS> = 1, SBI0SR<AAS> will be set to 1 upon the reception of the first word. The AAS flag is cleared to 0 by writing data to or reading data from SBI0DBR.

(12) Monitoring detection of a general call

SBI0SR<AD0> is set to 1 when a general call is received (i.e. when the eight data bits after the start condition are all 0) in slave mode, and is reset to 0 when a start or stop condition is detected on the bus.

(13) Monitoring the last bit received

SBI0SR<LRB> holds the value of the SDA line which is latched at the rising edge of the SCL clock. In acknowledgment mode, the value read from SBI0SR<LRB> immediately after an INTS2 interrupt request has been generated is equivalent to the value of the ACK signal.

(14) Software reset

If the serial bus interface circuit locks due to noise from external sources, it can be initialized using the software reset function.

Writing 10 and then 01 to SBI0CR2<SWRST1:SWRST0> causes a reset signal pulse to be applied to the serial bus interface circuit, initializing it. All control registers and status flags are initialized to their reset values. SBI0CR2<SWRST1:SWRST0> are automatically cleared to 00 upon the initialization of the serial bus interface.

Note: A software reset also resets the selection of the operating mode, causing a transition from I^2C bus mode to clock-synchronous 8-bit SIO mode.

(15) Serial bus interface data buffer register (SBI0DBR)

Reading received data from and writing transmit data to the serial bus interface circuit are accomplished by reading from and writing to SBI0DBR. In addition, in master mode the slave address and the direction bit are set in this register, after which a start condition is generated.

(16) I^2C bus address register (I2C0AR)

When the device is operating as a slave device, the I2C0AR<SA6:SA0> bits are used to set the slave address. In addition, when I2C0AR<ALS> = 0, the device recognizes the slave address output by the master device, and data is sent in addressing format. When I2C0AR<ALS> = 1, the device will not recognize the slave address output by the master device, and data will be sent in free format.

(17) Baud rate register (SBI0BR1)

Before the I^2C bus can be used, the P4EN bit of the baud rate circuit control register (SBI0BR1) must be set to 1.

(18) IDLE2 setting register (SBI0BR0)

The SBI0BR0<I2SBI0> bit enables or disables device operation after the device has entered IDLE mode. This bit must be set before the instruction to enter standby mode is executed.

3.12.6 Data Transfer Procedure in I²C Bus Mode

(1) Initializing the device

First, set SBI0BR1<P4EN> and SBI0CR1<ACK, SCK2:SCK0>. Set SBI0BR1<P4EN> to 1 and clear bits 7-5 and 3 of SBI0CR1 to 0.

Next, set the slave address in I2C0AR<SA6:SA0> and set I2C0AR<ALS> to 0 for addressing format.

Then, to initialize the device to slave receiver mode, set SBI0CR2<MST, TRX, BB> to 000, SBI0CR2<PIN> to 1, SBI0CR2<SBIM1:SBIM0> to 10 and clear bits 1 and 0 of SBI0CR2 to 00.

```
               7 6 5 4 3 2 1 0
SBI0BR1  ← 1 0 0 0 0 0 0 0        Operate internal baud rate generator.
SBI0CR1  ← 0 0 0 X 0 X X X        Set ACK and SCL clocks.
I2C0AR   ← X X X X X X X X        Set slave address and address recognition mode.
SBI0CR2  ← 0 0 0 1 1 0 0 0        Select slave receiver mode.
 (Note)  X: Don't care
```

(2) Generating a start condition and slave address

1) In master mode

Follow the procedure described below to generate a start condition and slave address in master mode:

First, check that the bus is free (SBI0SR<BB> = 0). Next, place the serial bus into acknowledgment mode by setting SBI0CR1<ACK> to 1. Also, write the slave address and direction bit to SBI0DBR.

While SBI0SR<BB> = 0, set SBI0CR2<MST, TRX, BB, PIN> to 1111 to generate a start condition on the bus. Then output nine clock pulses on the SCL pin. For the first eight clock pulses, output the slave address and direction bit which have been set in SBI0DBR. Release the SDA line on the ninth clock pulse to receive an acknowledge signal from the slave device.

An INTS2 interrupt request is generated at the falling edge of the ninth clock pulse, resetting SBI0CR2<PIN> to 0. In master mode, the SCL line is held Low while PIN = 0. In addition, only when an acknowledge signal is returned from the slave device, the generation of an INTS2 interrupt request causes SBI0CR2<TRX> to change state according to the transmitted direction bit.

Settings in the main routine

```
               7 6 5 4 3 2 1 0
→ Reg.       ← SBI0SR
│  Reg.       ← Reg. e 0x20
│  if Reg. ≠ 0x00                  Check that bus is free.
└ Then
   SBI0CR1  ← X X X 1 0 X X X       Select acknowledgement mode.
   SBI0DR1  ← X X X X X X X X       Set slave address and direction for target slave.
   SBI0CR2  ← 1 1 1 1 1 0 0 0       Generate start condition.
```

Example of INTS2 interrupt routine processing

```
INTCLR    ← 0X34                   Clear interrupt request.
Processing
End of interrupt processing
```

2)   In slave mode

In slave mode a start condition and slave address are received.

The slave address and the direction bit are received from the master device with the first eight clock pulses on the SCL line after the start condition. The start condition is also received from the master device. When a general call or an address identical to the slave address which has been set in I2C0AR is received, the SDA line is pulled Low on the ninth clock pulse to output an acknowledge signal.

An INTS2 interrupt request is generated at the falling edge of the ninth clock pulse, resetting SBI0CR2<PIN> to 0. In slave mode, the SCL line is held Low while PIN = 0.

> Note: DMA transfer can be used only when the following conditions are satisfied:
> - A single master corresponds to a single slave.
> - Continuous transmission or reception is possible.
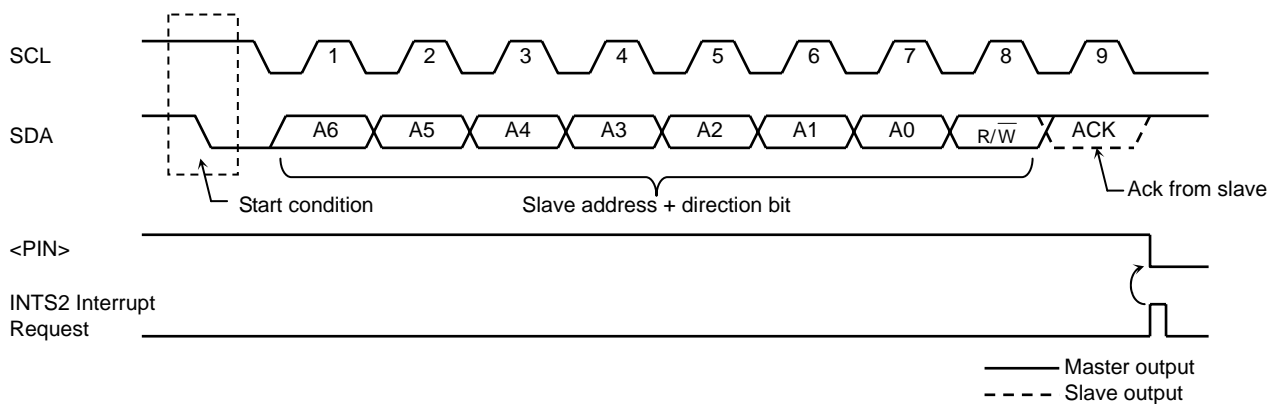
Figure 3.12.12 Generating a Start Condition and Slave Address

(3)   Transferring one word of data

During the INTS2 interrupt processing which takes place after the device has finished transferring one word of data, SBI0SR<MST> is tested to determine whether the device is placed in master mode or slave mode.

1)   In master mode (when SBI0SR<MST> = 1)

SBI0SR<TRX> is tested to determine whether the device is a transmitter or a receiver.

In transmitter mode (when SBI0SR<TRX> = 1)

SBI0SR<LRB> is tested. If SBI0SR<LRB> = 1, the receiver is not requesting data; therefore, a sequence for generating a stop condition (described later) should be performed to terminate the data transfer.

If SBI0SR<LRB> = 0, the receiver is requesting the next data item. If the next data item to be transferred is 8 bits long, write the transfer data to SBI0DBR. If it is not 8 bits long, set SBI0CR1<BC2:BC0> and SBI0CR1<ACK> before writing the transfer data to SBI0DBR. When data is written to the data buffer register, SBI0CR2<PIN> is set to 1, the serial clock for transferring the next word of data is generated from the input on the SCL pin, and one word of data is output on the SDA pin. When the device has finished transferring data, an INTS2 interrupt request is generated, SBI0CR2<PIN> is reset to 0 and the SCL pin is pulled Low. To transfer more than one word, repeat the above procedure starting from the test of SBI0SR<LRB>.

INTS2 interrupt

if MST = 0

Then go to slave mode processing

if TRX = 0

Then go to receiver mode processing

if LRB = 0

Then go to processing for generating stop condition

```
SBI0CR1 ← X X X X 0 X X X        Set number of bits to be transferred and ACK.
SBI0DBR ← X X X X X X X X        Write transfer data.
```
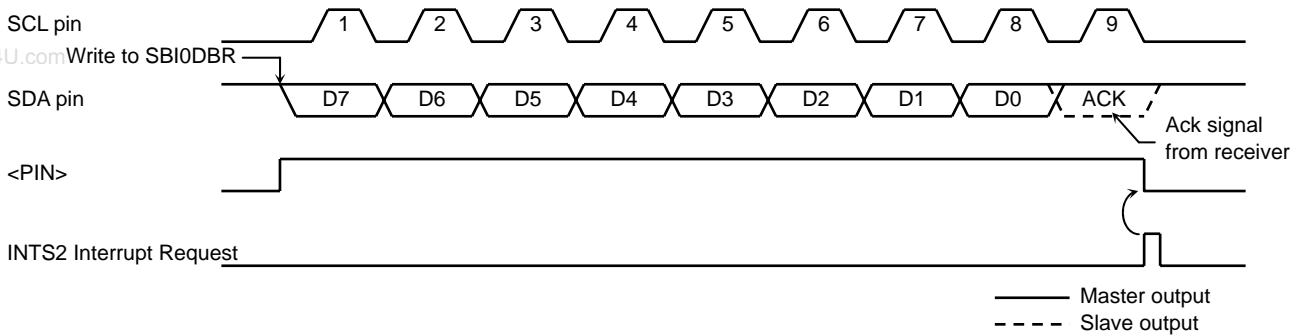
End of interrupt processing

Note: X: Don't care

Figure 3.12.13 Example in Which SBI0CR1<BC2:BC0> = 000 and SBI0CR1<ACK> = 1 (Transmitter Mode)

In receiver mode (when SBI0SR<TRX> = 0)

If the next data item to be transferred is 8 bits long, write the transfer data to SBI0DBR. If it is not 8 bits long, set SBI0CR1<BC2:BC0> and SBI0CR1<ACK> and then read the received data from SBI0DBR in order to release the SCL line. (The data read out immediately after the transmission of the slave address is undefined.) When data is read from the data buffer register, SBI0CR2<PIN> is set to 1. The serial clock for transferring the next word of data is output on the SCL pin. The SDA pin is pulled Low at the final bit when the acknowledge signal goes Low.

An INTS2 interrupt request is now generated, SBI0CR2<PIN> is reset to 0 and the SCL pin is pulled Low. Each time received data is read from SBI0DBR, a clock pulse for one-word data transfer and an acknowledge signal are output.
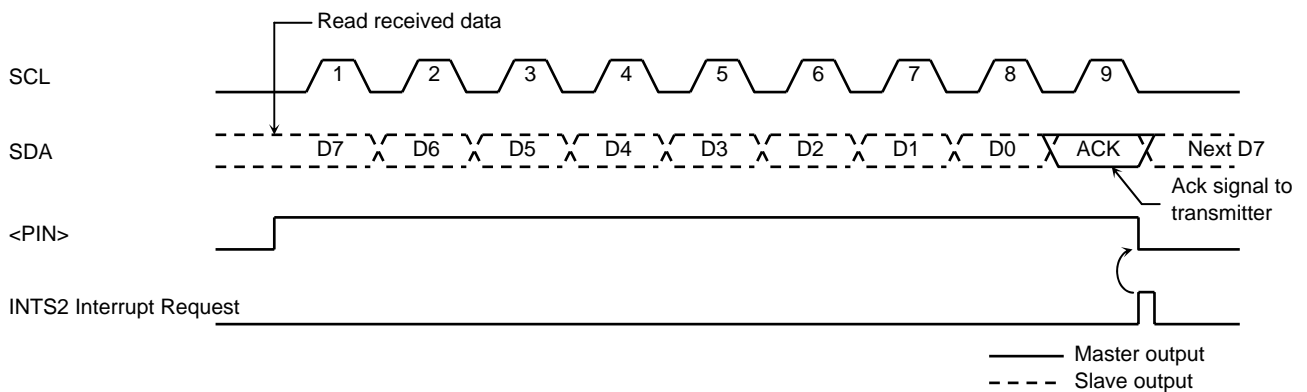
Figure 3.12.14 Example in Which SBI0CR1<BC2:BC0> = 000 and SBI0CR1<ACK> = 1 (Receiver Mode)

To instruct the transmitter to terminate data transmission, set SBI0CR1<ACK> to 0 before reading the data which is one word before the last data to be received. This disables generation of an acknowledge clock pulse for the last data. As part of the processing after the generation of an end-of-transfer interrupt request, set SBI0CR1<BC2:BC0> to 001 and read out data, at which time a clock pulse for one-bit data transfer is generated. Since the master at this time is a receiver, it will hold the bus SDA line High. The transmitter receives this High-level signal as an ACK signal, so that the receiver can request the transmitter to terminate transmission.

As part of the processing after the interrupt request generated upon the completion of receiving this one bit, generate a stop condition to terminate data transfer.
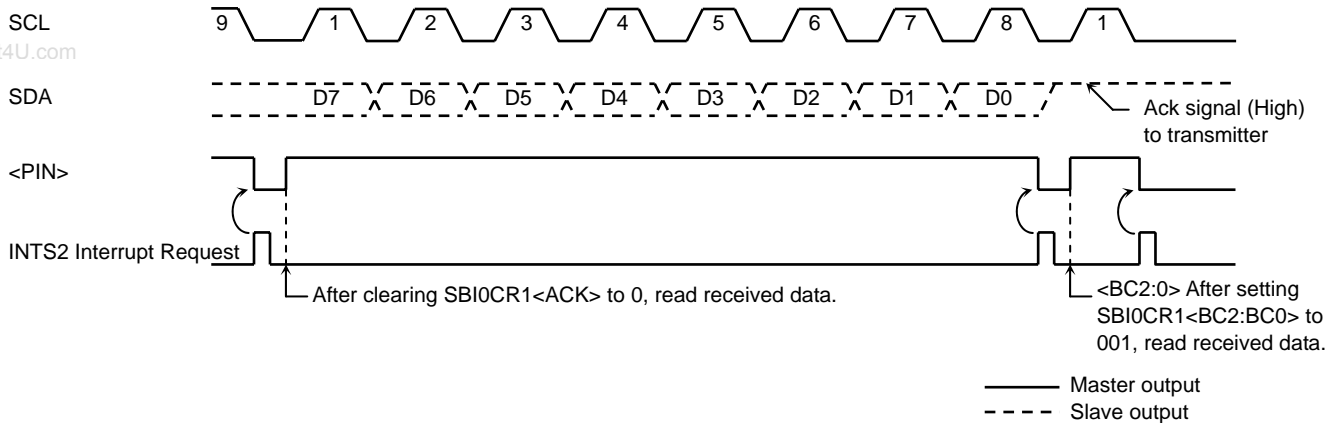
Figure 3.12.15 Terminating Data Transmission in Master Receiver Mode

Example: When receiving data N times

INTS2 interrupt (after transmitting data)

```
           7 6 5 4 3 2 1 0
SBI0CR1 ← X X X X 0 X X X      Set number of bits received and ACK.
Reg.    ← SBI0CBR               Read dummy data.
End of interrupt
```

INTS2 interrupt (first to (N-2)th data reception)

```
           7 6 5 4 3 2 1 0
Reg.    ← SBI0DBR               Read first to (N-2)th received data.
End of interrupt
```

INTS2 interrupt ((N-1)th data reception)

```
           7 6 5 4 3 2 1 0
SBI0CR1 ← X X X 0 0 X X X       Disable generation of clock for acknowledge signal.
Reg.    ← SBI0DBR               Read (N-1)th received data.
End of interrupt
```

INTS2 interrupt (Nth data reception)

```
           7 6 5 4 3 2 1 0
SBI0CR1 ← 0 0 1 0 0 X X X       Generate clock for 1-bit transfer.
Reg.    ← SBI0DBR               Read Nth received data.
End of interrupt
```

INTS2 interrupt (after receiving data)

```
Processing for generating stop condition      Terminate data transfer.
End of interrupt
Note:  X: Don't care
```

2)   In slave mode (SBI0SR<MST> = 0)

In slave mode, an INTS2 interrupt request is generated when a slave address or general call sent by the master is received, or when the data transfer is completed after a general call is received or the received slave address is found to match the device's address. Also, if the arbitration-lost condition is detected in master mode, the device will operate in slave mode, in which case an INTS2 interrupt request will be generated when the device has finished transferring the word in which the arbitration-lost condition was detected. When an INTS2 interrupt request occurs, SBI0CR2<PIN> is set to 0 and the SCL pin is pulled Low. The SCL pin is released tLOW after data is written to or read from SBI0DBR, or tLOW after SBI0CR2<PIN> is set to 1.

In slave mode, perform the processing which normally needs to be performed in slave mode or any processing which needs to be performed after the device has entered slave mode upon detecting the arbitration-lost condition.

In each case, test SBI0SR<AL, TRX, AAS, AD0> to determine the necessary processing. Table 3.12.1 shows the various slave mode statuses and the necessary processing for each.

Example: When the slave address is matched and the direction bit is 1 in slave receiver mode

INTS2 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
SBI0CR1 ← X X X 1 0 X X X        Set number of bits to be transmitted.
SBI0DBR ← X X X X 0 X X X        Set transmit data.
Note:  X: Don't care
```

Table 3.12.2 Processing in Slave Mode

<TRX>	<AL>	<AAS>	<AD0>	Status	Processing
1	1	1	0	The arbitration-lost condition was detected while the slave address was being sent and the device received a slave address sent by another master for which the direction bit was 1.	Set SBI0CR1<BC2:BC0> to the number of bits in one word and write the data to be transmitted to SBI0DBR.
	0	1	0	In slave receiver mode, the device received a slave address sent by another master for which the direction bit was 1.	
		0	0	The device has finished sending one data word in slave transmitter mode.	Test SBI0SR<LRB>. If it is set to 1, indicating that the receiver is not requesting the next data item, set SBI0CR2<PIN> to 1 and reset <TRX> to 0 to release the bus. If SBI0SR<LRB> = 0, indicating that the receiver is requesting the next data item, set SBI0CR1<BC2:BC0> to the number of bits in one word and write the data to be transmitted to SBI0DBR.
0	1	1	1/0	The arbitration-lost condition was detected while the slave address was being sent and the device received either a slave address sent by another master for which the direction bit was 0 or a general call.	Read SBI0DBR to set SBI0CR2<PIN> to 1 (a dummy read), or set it by writing a 1 to it.
		0	0	The arbitration-lost condition was detected while the slave address or data was being sent and the device finished sending the word.	
	0	1	1/0	In slave receiver mode, the device received either a slave address sent by another master for which the direction bit was 0 or a general call.	
		0	1/0	In slave receiver mode, the device has finished receiving one word of data.	Set SBI0CR1<BC2:BC0> to the number of bits in one word and read the received data from SBI0DBR.

(4) Generating a stop condition

If SBI0SR<BB> = 1, set SBI0CR2<MST,TRX,PIN> to 111 and reset SBI0CR2<BB> to 0. The device starts a sequence for outputting a stop condition to the bus. Do not rewrite the contents of SBI0CR2<MST,TRX,BB,PIN> until the stop condition appears on the bus.

Note, however, that if the bus SCL line has been pulled Low by some other device, the device will wait until the SCL line is released High again; when SCL is High again, the device will drive the SDA pin High, thereby generating a stop condition.

```
              7 6 5 4 3 2 1 0
SBI0CR2  ←  1 1 0 1 1 0 0 0        Generate stop condition.
```
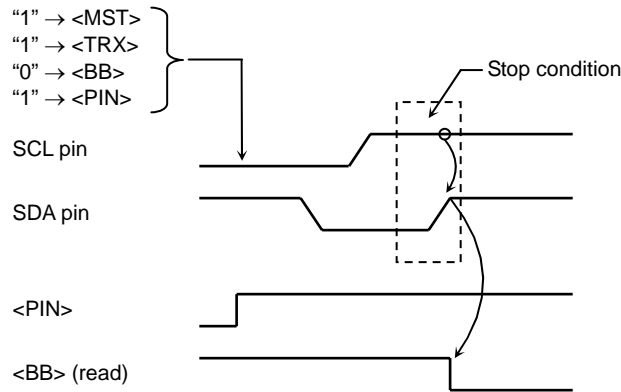

Figure 3.12.16 Generating a Stop Condition

(5) Restart procedure

　　Restart is used by a master device to change the direction of transfer with respect to a slave device without terminating data transfer. The following shows how to trigger a restart when the device is operating in master mode.

　　First, reset SBI0CR2<MST,TRX,BB> to 000 and set SBI0CR2<PIN> to 1 to release the bus. Since at this time the SDA pin is held High and the SCL pin is released, no stop condition is generated on the bus, with the result that the bus appears to other masters to be in busy state still. Then, test SBI0SR<BB> and wait until it becomes 0, confirming that the SCL pin has been released. Next, test SBI0SR<LRB> and wait until it becomes 1, confirming that no other device is pulling the bus SCL line Low. After using the above procedures to confirm that the bus is free, generate a start condition by following the procedure described earlier in (2).

　　Note, however, that in order to yield the necessary restart set-up time, a wait time of at least 4.7 μs must be generated by software between the bus free state being confirmed and a start condition being generated.

```
                  7 6 5 4 3 2 1 0
  SBI0CR2 ← 0 0 0 1 1 0 0 0          Release bus.
→ if SBI0SR<BB> ≠ 0                   Check that SCL pin is released.
└ Then
→ if SBI0SR<LRB> ≠ 1                  Check that no other device is pulling SCL line Low.
└ Then
  4.7 μs Wait
  SBI0CR1 ← X X X 1 0 X X X          Select acknowledgement mode.
  SBI0DBR ← X X X X X X X X          Set slave address and direction for target slave.
  SBI0CR2 ← 1 1 1 1 1 0 0 0          Generate start condition.
  Note:  X: Don't care
```
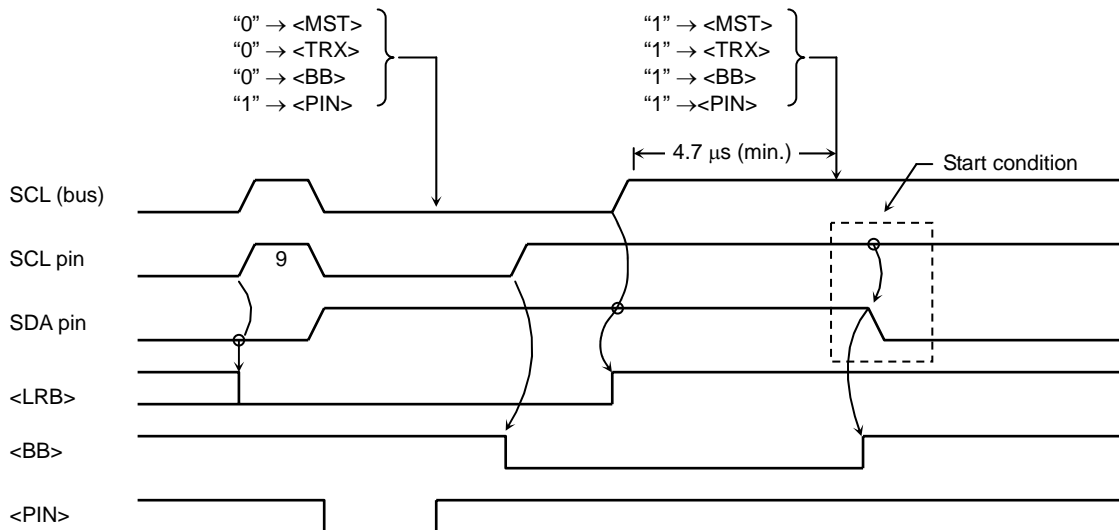

Figure 3.12.17 Restart Generation Timing

Note : Please do not carry out the light of <MST>="0" in the state of <MST>="0" (it cannot re-start).

3.12.7 Control in clock-synchronous 8-bit SIO mode

The following section describes the registers which are used to control the serial bus interface and to monitor its operating status when it is used in clock-synchronous 8-bit SIO mode

Serial Bus Interface Baud Rate Register 0

SBI0CR1 (0xFFFF_F240)		7	6	5	4	3	2	1	0
	Bit symbol	SIOS	SIOINH	SIOM1	SIOM0	—	SCK2	SCK1	SCK0
	Read/Write	W				—	W		R/W
	After Reset	0	0	0	0	—	0	0	1
	Function	Indicate transfer start/stop 0: Stop 1: Start	Continue/ abort transfer 0: Continue transfer 1: Abort transfer	Transfer mode selection 00: Transmit mode 01: (Reserved) 10: Transmit/receive mode 11: Receive mode			Selects serial clock frequency and monitors reset state		

Selects serial clock frequency <SCK2:SCK0> (for write)

000	n = 3	1.25 kHz
001	n = 4	625 kHz
010	n = 5	312.5 kHz
011	n = 6	156.3 kHz
100	n = 7	78.13 kHz
101	n = 8	39.06 kHz
110	n = 9	19.53 kHz
111	—	External clock

System clock:: fc(=40 MHz)
Clock gear: : fc/1
$\phi T0 = fperiph/4 (= 10 MHz)$
$Frequency = \dfrac{\phi T0}{2n}$ [Hz]

Note: Set SBI0CR1<SIOS> to 0 and SBI0CR1<SIOINH> to 1 before setting the transfer mode and the serial clock frequency.

Serial Bus Interface Data Buffer Register

SBI0DBR (0xFFFF_F241)		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							

Figure 3.12.18 SIO Mode Registers

Serial Bus Interface Control Register 2

		7	6	5	4	3	2	1	0
SBI0CR2	Bit symbol	—	—	—	—	SBIM1	SBIM0	—	—
(0xFFFF_F243)	Read/Write	—	—	—	—	W		—	—
	After Reset	—	—	—	—	0	0	—	—
	Function					Selects serial bus interface operating mode 00: Port mode 01: Clock-synchronous 8-bit SIO mode 10: I²C bus mode 11: (Reserved)			

Selects Serial Bus Interface Operating Mode

		7	6	5	4	3	2	1	0
SBI0SR	Bit symbol	—	—	—	—	SIOF	SEF	—	—
(0xFFFF_F243)	Read/Write	—	—	—	—	R		—	—
	After Reset	—	—	—	—	0	0	—	—
	Function					Serial transfer operation status 0: Transfer terminated 1: Transfer in progress	Shift operation status 0: Shift operation terminated 1: Shift operation in progress		

Serial Bus Interface Baud Rate Register 0

		7	6	5	4	3	2	1	0
SBI0BR0	Bit symbol	—	I2SBI0	—	—	—	—	—	—
(0xFFFF_F244)	Read/Write	—	R/W	—	—	—	—	—	W
	After Reset	—	0	—	—	—	—	—	—
	Function		IDLE 0: Idle 1: Operate						Must always be set to 0.

Serial Bus Interface Baud Rate Register 1

		7	6	5	4	3	2	1	0
SBI0BR1	Bit symbol	P4EN	—	—	—	—	—	—	—
(0xFFFF_F245)	Read/Write	R/W	—	—	—	—	—	—	—
	After Reset	0	—	—	—	—	—	—	—
	Function	Internal clock 0: Stopped 1: Operate							

Figure 3.12.19 SIO Mode Registers

(1) Serial clock

1) Clock source

Clock sources can be selected using SBI0CR1<SCK2:SCK0> as described below.

Internal clock

In internal clock mode, one of seven clock source frequencies can be selected. The serial clock is output to external devices on the SCK pin. Note that when a transfer starts, the SCK pin is driven High.

The device has an automatic wait function which works as follows: if an operation to write data (during transmission) or read data (during reception) in a program cannot keep up with the serial clock rate, the device will automatically stop the serial clock and suspend the next shift operation until reading or writing has been completed.
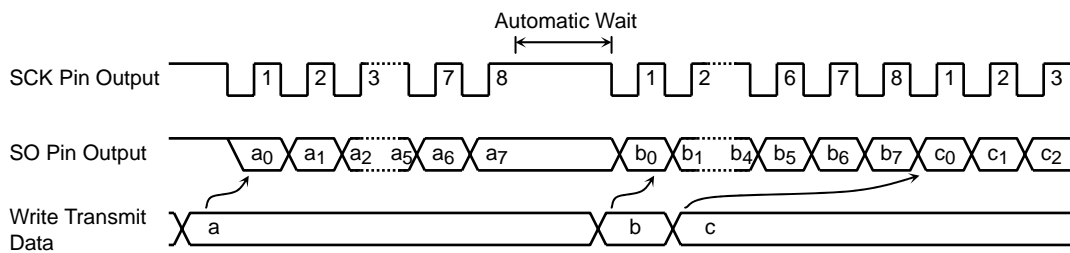
Figure 3.12.20 Automatic Wait Function

External clock (SBI0CR1<SCK2:SCK0> = 111)

A clock signal from an external source input via the SCK pin can be used as the serial clock. To ensure that shift operations will be performed without fail, the High-level and the Low-level durations of the serial clock must satisfy the pulse width conditions given below.

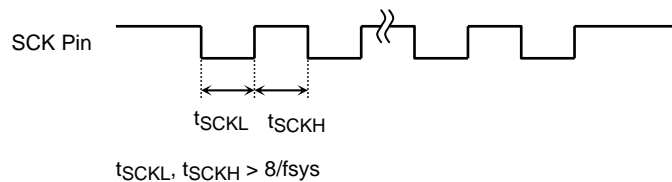t_{SCKL}, $t_{SCKH} > 8/f_{sys}$

Figure 3.12.21 Maximum Transfer Frequency for External Clock Input

2)   Edges used for shifting

     During transmission data is shifted at the leading edge; during reception data is shifted at the trailing edge.

<u>Leading-edge shift</u>

     Data is shifted at the leading edge of the serial clock (the falling edge of the SCK pin input/output).

<u>Trailing-edge shift</u>

     Data is shifted at the trailing edge of the serial clock (the rising edge of the SCK pin input/output).
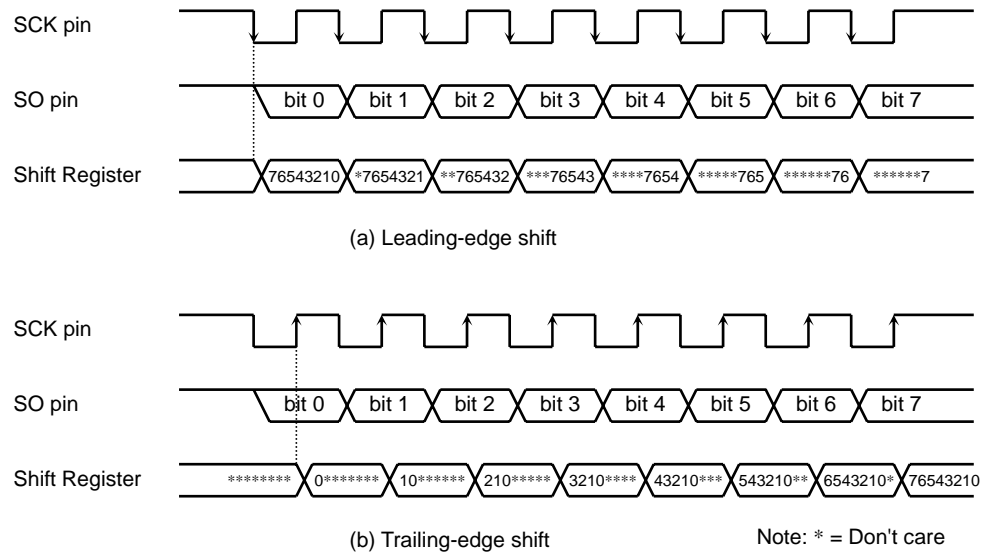
SCK pin								
SO pin	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7
Shift Register	76543210	*7654321	**765432	***76543	****7654	*****765	******76	*******7

(a) Leading-edge shift

SCK pin								
SO pin	bit 0	bit 1	bit 2	bit 3	bit 4	bit 5	bit 6	bit 7
Shift Register	********	0*******	10******	210*****	3210****	43210***	543210**	6543210*	76543210

(b) Trailing-edge shift   Note: * = Don't care

Figure 3.12.22 Edges Used for Shifting

(2) Transfer mode

SBI0CR1<SIOM1:SIOM0> is used to select the mode of transfer among transmit mode, receive mode and transmit/receive mode.

1) 8-bit transmit mode

After selecting transmit mode in the control register, write transmit data to SBI0DBR.

Transmission is initiated by setting SBI0CR1<SIOS> to 1 after writing transmit data to the data buffer register. The transmit data is transferred from SBI0DBR to the shift register, from which the data is shifted out to the SO pin synchronously with the serial clock, starting with the least significant bit (LSB). Once the transmit data has been transferred from SBI0DBR to the shift register, SBI0DBR becomes empty and generates an INTSBI (buffer empty) interrupt request to request the next transmit data.

If an internal clock is being used, unless the next data item has been set in the data buffer register after the transmission of all 8 bits of data, the device will automatically stop the serial clock and suspend processing. The automatic wait is released when the next transmit data is written into the data buffer register.

If an external clock is being used, data must be written into SBI0DBR before the next data item can be shifted. The transfer rate thus depends on the maximum delay between an interrupt request being generated and data being written into SBI0DBR by an interrupt service routine.

When transmission is started, after the SBI0SR<SIOF> goes High, the SO pin outputs the final bit of the last transferred data until the falling edge of SCK.

To terminate transmission write a 0 to SBI0CR1<SIOS> or a 1 to SBI0CR1<SIOINH> in the interrupt service routine for the INTS2 interrupt. Once SBI0CR1<SIOS> has been cleared, transmission will be terminated when all the data has been output. Check SBI0SR<SIOF> in the program to determine whether transmission has been terminated. SBI0SR<SIOF> is reset to 0 upon the termination of transmission. If SBI0CR1<SIOINH> has been set to 1, transmission will be aborted immediately and SBI0SR<SIOF> cleared to 0.

Furthermore, if an external clock is being used, SBI0CR1<SIOS> must be cleared to 0 before the device can start shifting out the next transmit data. Unless SBI0CR1<SIOS> has been cleared to 0 before the device shifts out the data, dummy data will be transmitted and transmit operation ends.

```
          7 6 5 4 3 2 1 0
SBI0CR1 ← 0 1 0 0 0 X X X          Select transmit mode.

SBI0DBR ← X X X X X X X X          Write transmit data.
SBI0CR1 ← 1 0 0 0 0 X X X          Start transmission.
```

INTS2 interrupt
```
SBI0DBR ← X X X X X X X X          Write transmit data.
```

(a) Internal clock

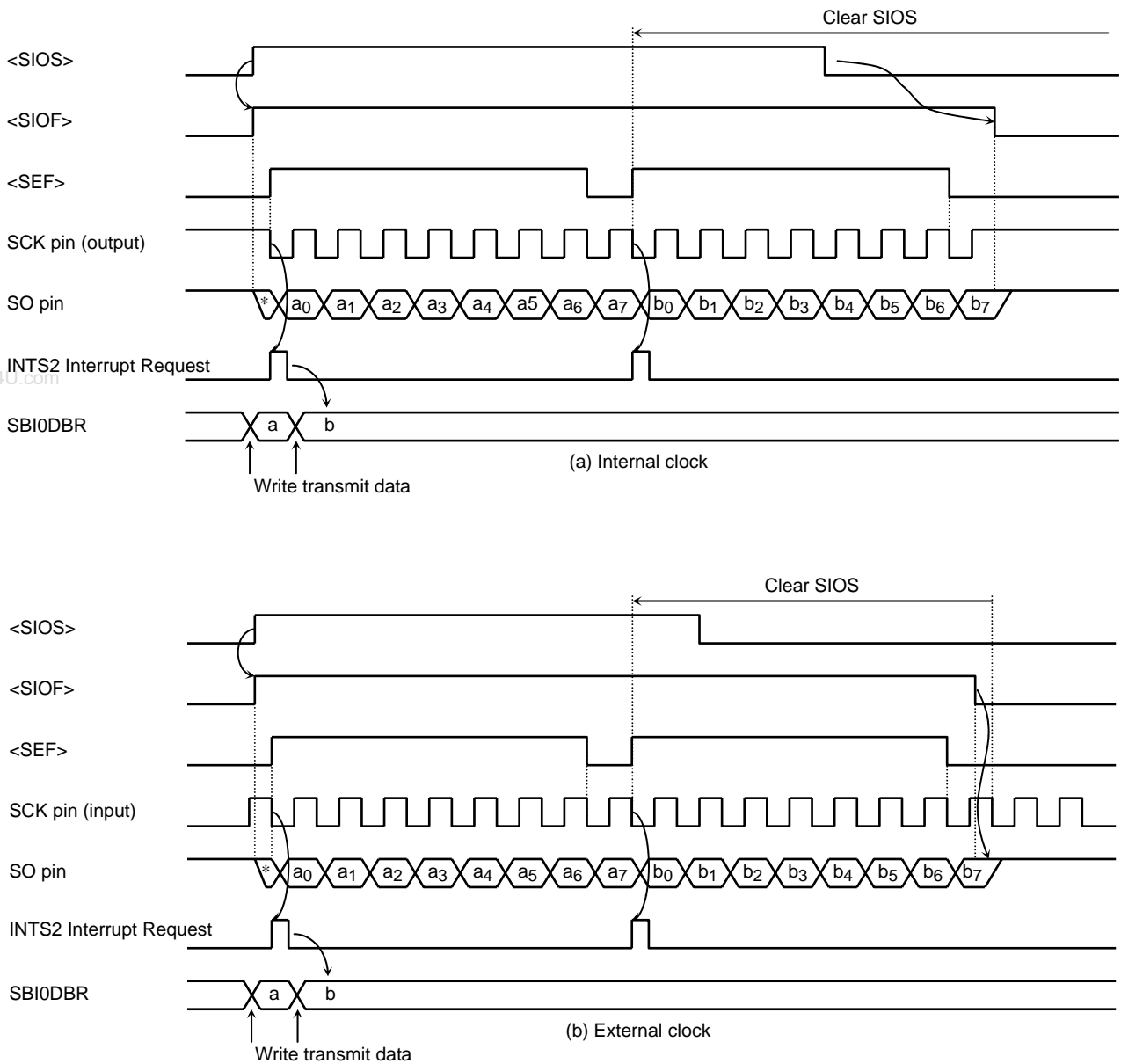
(b) External clock

Figure 3.12.23 Transmit Mode

Example: Sample program (MIPS16) that instructs termination of SIO transmission (with external clock)

```
          ADDIU   r3, r0,   0x04
STEST1  : LB      r2, (SBI0SR)              ; If SBI0SR<SEF> = 1 then loop
          AND     r2, r3
          BNEZ    r2,  STEST1
          ADDIU   r3, r0,   0x20
STEST2  : LB      r2, (PA)                  ; If SCK = 0 then loop
          AND     r2, r3
          BEQZ    r2,  STEST2
          ADDIU   r3, r0,   0y00000111
          STB     r3, (SBI0CR1)             ; <SIOS> ← 0
```
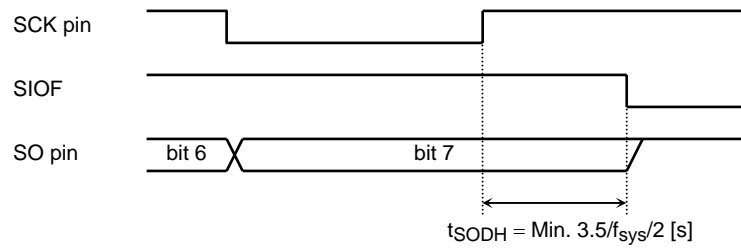
$$t_{SODH} = \text{Min. } 3.5/f_{sys}/2 \ [s]$$

Figure 3.12.24 Transmit Data Retention Time When Terminating Transmission

2) 8-bit receive mode

After selecting receive mode in the control register, write a 1 to SBI0CR1<SIOS>, enabling the device to receive data. Data is read into the shift register from the SI pin synchronously with the serial clock, beginning with the least significant bit. When 8 bits of data have been read, the received data is transferred from the shift register to SBI0DBR and an INTS2 (buffer full) interrupt request is generated, requesting that the received data be read out. The received data is read out from SBI0DBR by an interrupt service routine.

If an internal clock is being used, the automatic wait function is activated, halting the serial clock until the receive data is read out from SBI0DBR.

If an external clock is being used, shift operation is synchronized to the externally sourced clock. The maximum transfer rate for external clock operation thus depends on the maximum delay between an interrupt request being generated and the received data being read out.

To terminate reception write a 0 to SBI0CR1<SIOS> or a 1 to SBI0CR1<SIOINH> in the interrupt service routine for the INTS2 interrupt. Once SBI0CR1<SIOS> has been cleared, reception will be terminated when all the received data bits have been written into SBI0DBR. Check SBI0SR<SIOF> in the program to determine whether reception has been terminated. SBI0SR<SIOF> is reset to 0 upon the termination of reception. After confirming that reception has been terminated, read out the last data item received. If SBI0CR1<SIOINH> has been set to 1, reception will be aborted immediately and SBI0SR<SIOF> cleared to 0. (In that case, the received data is invalid and need not be read out.)

Note: If the transfer mode is changed during receive operation, the contents of SBI0DBR will be lost. If it is necessary to change the transfer mode, first terminate reception (by writing a 0 to SBI0CR1<SIOS>) and read out the last data received.

```
              7 6 5 4 3 2 1 0
  SBI0CR1 ← 0 1 1 1 0 X X X        Select receive mode.

  SBI0CR1 ← 1 0 1 1 0 0 0 0        Start reception.

INTS2 interrupt
  Reg.    ← SBI0DBR                Read received data.
```
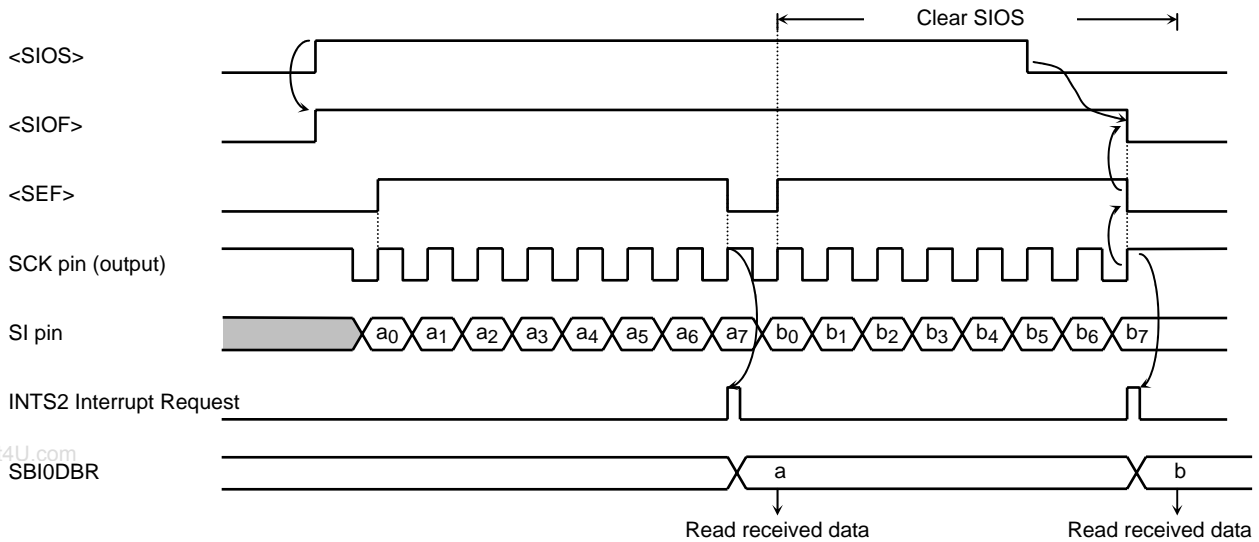
Figure 3.12.25 Receive Mode (example with internal clock)

3) 8-bit transmit/receive mode

   After selecting transmit/receive mode in the control register, write the transmit data to SBI0DBR. Then set SBI0CR1<SIOS> to 1, enabling the device to transmit/receive data. Transmit data is output on the SO pin at the falling edge of the serial clock starting with the least significant bit, while the received data is read into the shift register from the SI pin at the rising edge of the serial clock. When 8 bits of data have been read, the received data is transferred from the shift register to SBI0DBR and an INTS2 interrupt request is generated. Use an interrupt service routine to read out the received data from the data buffer register, then write transmit data to the data buffer register. Since SBI0DBR is shared for transmission and reception, always be sure to read out the received data before writing transmit data to the data buffer register.

   If an internal clock is being used, the automatic wait function is activated, halting the serial clock until the received data has been read out and the next transmit data has been written into the data buffer register.

   If an external clock is being used, since shift operation is synchronized to the externally sourced clock, the received data must be read out and the next transmit data written into the data buffer register before the next shift operation can start. The maximum transfer rate for external clock operation thus depends on the maximum delay between an interrupt request being generated and the received data being read out.

   When transmission is started, after the SBI0SR<SIOF> goes High, the SO pin outputs the final bit of the last transferred data until the falling edge of SCK.

   To terminate transmission/reception write a 0 to SBI0CR1<SIOS> or a 1 to SBI0CR1<SIOINH> in the interrupt service routine for the INTS2 interrupt. Once SBI0CR1<SIOS> has been cleared, transmission/reception will be terminated when all the received data bits have been written into SBI0DBR. Check SBI0SR<SIOF> in the program to determine whether transmission/reception has been terminated. SBI0SR<SIOF> is reset to 0 upon the termination of transmission/reception. If SBI0CR1<SIOINH> has been set to 1, transmission/reception will be aborted immediately and SBI0SR<SIOF> cleared to 0.

Note: If the transfer mode is changed during transmit/receive operation, the contents of SBI0DBR will be lost. If it is necessary to change the transfer mode, first terminate transmission/reception (by writing a 0 to SBI0CR1<SIOS>) and read out the last received data.
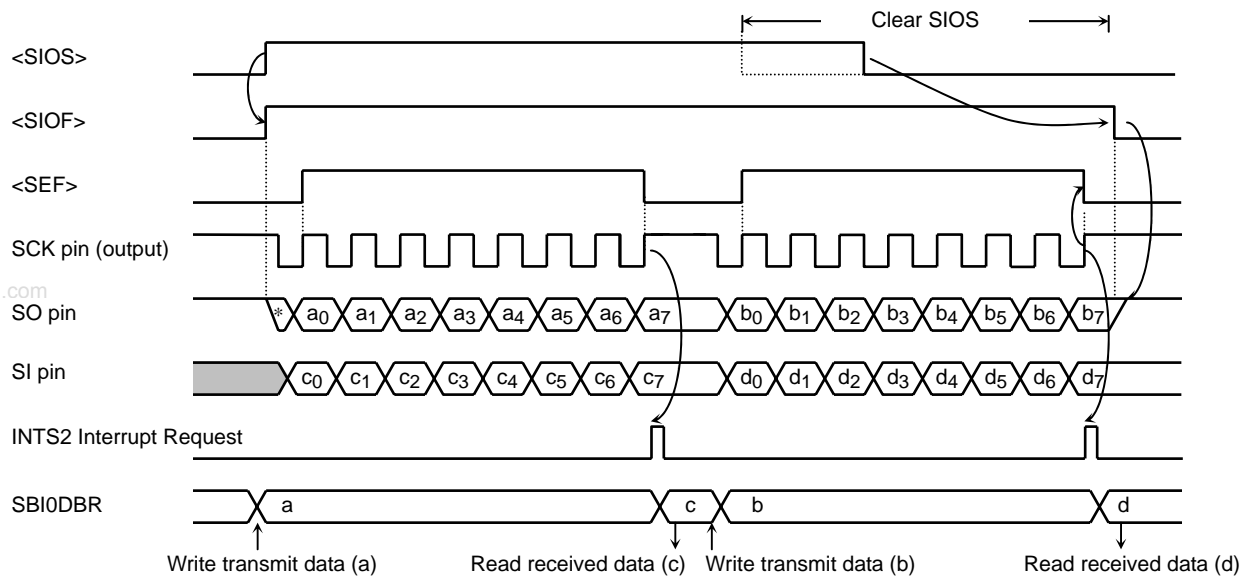

Figure 3.12.26 Transmit/Receive Mode (example with internal clock)

$t_{SODH} = $ Min. $4/f_{sys}/2$ [s]
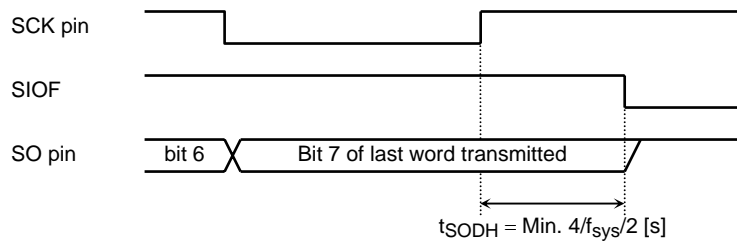
Figure 3.12.27 Transmit Data Retention Time When Terminating Transmission/Reception
(in Transmit/Receive Mode)

```
              7 6 5 4 3 2 1 0
SBI0CR1  ←  0 1 0 0 0 X X X        Select transmit mode.

SBI0DBR  ←  X X X X X X X X        Write transmit data.
SBI0CR1  ←  1 0 0 0 0 X X X        Start transmission/reception.
```

INTS2 interrupt

```
Reg.     ←  SBI0DBR               Read received data.
SBI0DBR  ←  X X X X X X X X        Write transmit data.
```

3.13 ANALOG/DIGITAL CONVERTER

The TMP1942 contains a 10-bit Half Flash analog/digital converter (A/D converter) with sixteen analog input channels. In addition to normal conversion, the converter supports highest-priority conversion mode, in which continuous conversion can be interrupted by conversion for a specific analog channel. The converter also supports an A/D monitor function, which allows the device to compare the value in the specified conversion result register with the value set in the compare register to determine which is greater. This function enables the device to monitor analog quantities without software intervention.

Figure 3.13.1 shows a block diagram of the A/D converter. The pins for the sixteen analog input channels (AN0-AN15) are also used as input-only port pins and/or key input pins.

> Note: When placing the device into IDLE, SLEEP or STOP mode to reduce the device's current consumption, check that the A/D converter has stopped operating before executing the instruction to enter a standby mode. This is necessary because, with some timings, the internal comparator may remain enabled while the device is in a standby mode. When placing the device into SLOW mode, stop the operation of the A/D converter beforehand.
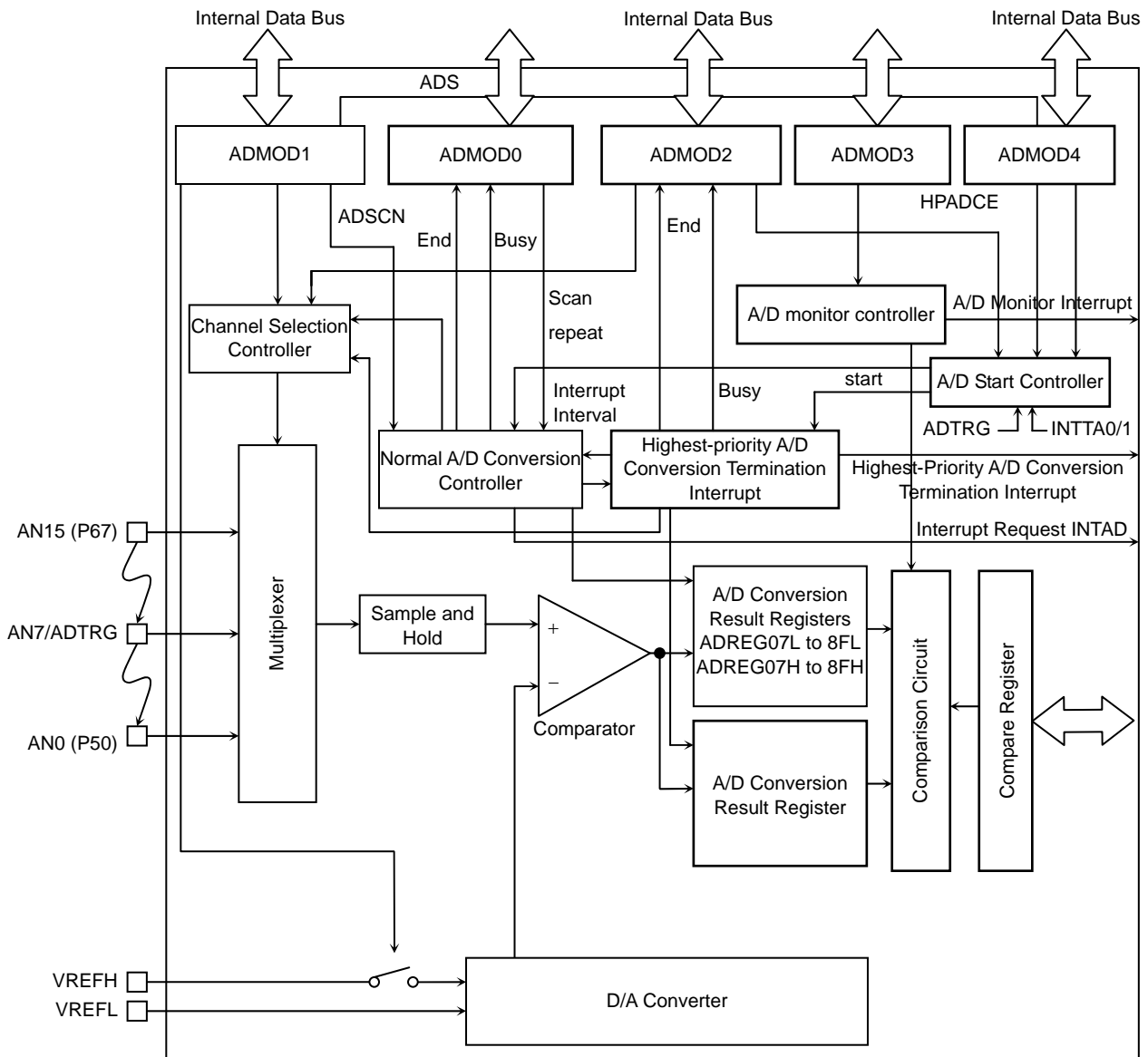

Figure 3.13.1 A/D Converter Block Diagram

3.13.1 Control Registers

The A/D converter is controlled by the A/D mode control registers (ADMOD0, ADMOD1, ADMOD2, ADMOD3 and ADMOD4). Also, the A/D conversion results are stored in the sixteen A/D conversion result upper/lower registers: ADREG08H/L to ADREG7FH/L. The results of highest-priority conversion are stored in ADREGSPH/L.

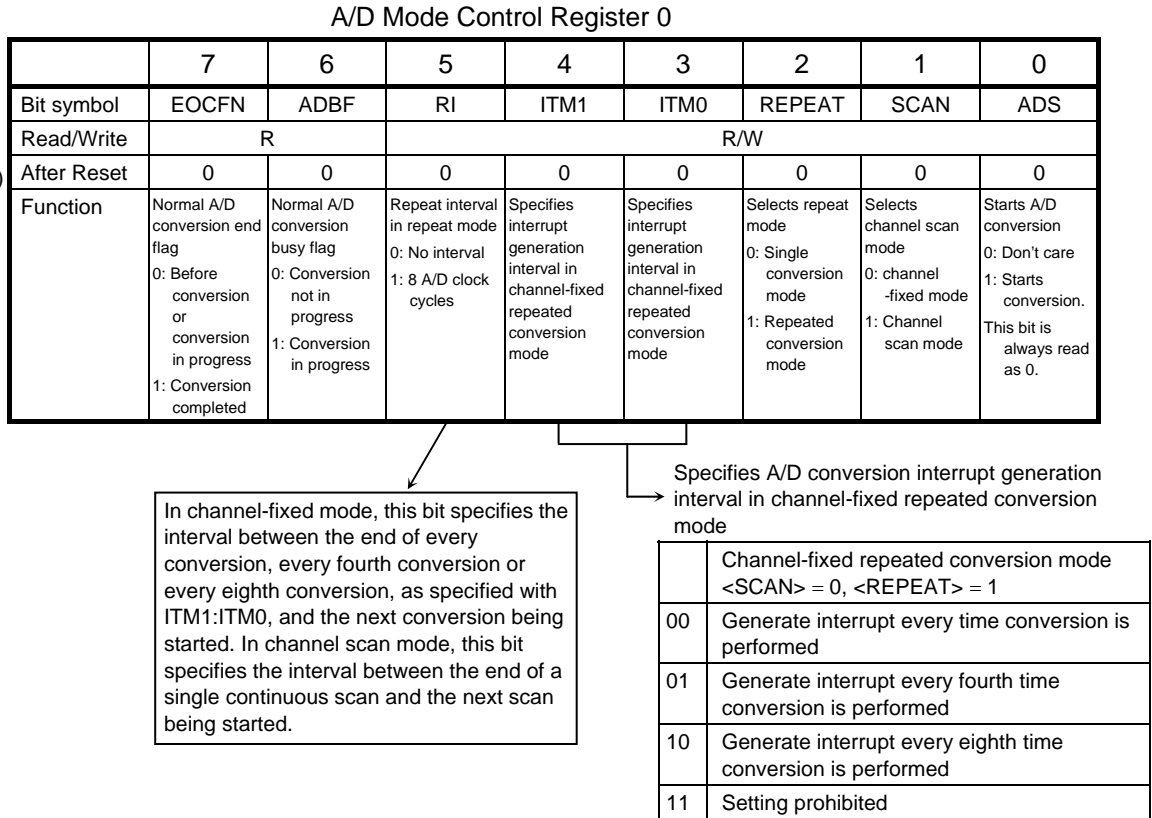Figure 3.13.2 shows the registers associated with the A/D converter.

A/D Mode Control Register 0

	7	6	5	4	3	2	1	0
Bit symbol	EOCFN	ADBF	RI	ITM1	ITM0	REPEAT	SCAN	ADS
Read/Write	R		R/W					
After Reset	0	0	0	0	0	0	0	0
Function	Normal A/D conversion end flag 0: Before conversion or conversion in progress 1: Conversion completed	Normal A/D conversion busy flag 0: Conversion not in progress 1: Conversion in progress	Repeat interval in repeat mode 0: No interval 1: 8 A/D clock cycles	Specifies interrupt generation interval in channel-fixed repeated conversion mode	Specifies interrupt generation interval in channel-fixed repeated conversion mode	Selects repeat mode 0: Single conversion mode 1: Repeated conversion mode	Selects channel scan mode 0: channel -fixed mode 1: Channel scan mode	Starts A/D conversion 0: Don't care 1: Starts conversion. This bit is always read as 0.

ADMOD0
(0xFFFF_F318)

In channel-fixed mode, this bit specifies the interval between the end of every conversion, every fourth conversion or every eighth conversion, as specified with ITM1:ITM0, and the next conversion being started. In channel scan mode, this bit specifies the interval between the end of a single continuous scan and the next scan being started.

Specifies A/D conversion interrupt generation interval in channel-fixed repeated conversion mode

	Channel-fixed repeated conversion mode <SCAN> = 0, <REPEAT> = 1
00	Generate interrupt every time conversion is performed
01	Generate interrupt every fourth time conversion is performed
10	Generate interrupt every eighth time conversion is performed
11	Setting prohibited

Figure 3.13.2 A/D Converter Registers (1/12)

A/D Mode Control Register 1

	7	6	5	4	3	2	1	0
Bit symbol	—	I2AD	—	ADSCN	ADCH3	ADCH2	ADCH1	ADCH0
Read/Write	—	R/W	—	R/W	R/W			
After Reset	—	0	—	0	0	0	0	0
Function		IDLE 0: Idle 1: Operate		Selects channel scan operating mode 0: 4-channel scan 1: 8-channel scan	Analog input channel selection			

ADMOD1
(0xFFFF_F319)

Selects Analog Input Channel

<ADCH3.2, 1, 0>	<SCAN>		
	0 Channel-fixed	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 AN6
0110	AN6	AN4 to AN6	AN0 to AN6
0111 (Note)	AN7	AN4 to AN7	AN0 to AN7
1000	AN8	AN8	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	AN13	AN12 to AN13	AN8 to AN13
1110	AN14	AN12 to AN14	AN8 to AN14
1111	AN15	AN12 to AN15	AN8 to AN15

Note: The AN7 pin is shared with the $\overline{\text{ADTRG}}$ input. Therefore, do not set <ADCH3:ADCH0> to 0111 when using the $\overline{\text{ADTRG}}$ input with ADTRGE set to 1.

Figure 3.13.2 A/D Converter Registers (2/12)

A/D Mode Control Register 2

	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
Function	Highest-priority A/D conversion end flag 0: Before conversion or conversion in progress 1: Conversion completed	Highest-priority A/D conversion busy flag 0: Conversion not in progress 1: Conversion in progress		Starts highest-priority A/D conversion 0: Don't care 1: Starts conversion. This bit is always read as 0.		Analog input channel selection for highest-priority A/D conversion		

ADMOD2 (0xFFFF_F31A)

<HPADCH3.2, 1, 0>	Analog Input Channel for Highest-Priority A/D 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	AN13
1110	AN14
1111	AN15

Figure 3.13.2 A/D Converter Registers (3/12)

A/D Mode Control Register 3

	7	6	5	4	3	2	1	0
Bit symbol	—	—	ADOBIC	REGS3	REGS2	REGS1	REGS0	ADOBSV
Read/Write	R/W	—	R/W					
After Reset	0	—	0	0	0	0	0	0
Function	Must always be set to 0.		A/D monitor interrupt generation condition 0: Less than compare register 1: Greater than compare register	Selects A/D conversion result register to be compared with compare register when A/D monitor function is enabled				A/D monitor function 0: Disable 1: Enable

ADMOD3 (0xFFFF_F31B)

<REGS.2, 1, 0>	A/D Conversion Result Register to Be Compared
0000	ADREG08
0001	ADREG19
0010	ADREG2A
0011	ADREG3B
0100	ADREG4C
0101	ADREG5D
0110	ADREG6E
0111	ADREG7F
1XXX	ADREGSP

Figure 3.13.2 A/D Converter Registers (4/12)

A/D Mode Control Register 4

	7	6	5	4	3	2	1	0
Bit symbol	HADHS	HADHTG	ADHS	ADHTG	—	—	ADRST1	ADRST0
Read/Write	R/W				—	—	W	W
After Reset	0	0	0	0	—	—	—	—
Function	Hardware start source for highest-priority A/D conversion 0: External trigger 1: INTTA1 interrupt	Hardware start of highest-priority A/D conversion 0: Disable 1: Enable	Hardware start source for normal A/D conversion 0: External trigger 1: INTTA0 interrupt	Hardware start of normal A/D conversion 0: Disable 1: Enable			Writing 10 and then 01 triggers the software reset of the A/D converter.	

ADMOD4 (0xFFFF_F31C)

Note 1: When performing A/D conversion using a hardware start resource by setting ADHTG or HADHTG to 1, observe the following procedure: To use an external trigger, first set P5FC<P57F> to 1 (ADTRG) before enabling hardware start. To use an 8-bit timer, first set ADHS or HADHS to 1 to select the use of a timer interrupt. Then, enable hardware start and finally operate the timer to enable A/D conversion to start at constant intervals.

Note 2: To change the hardware start resource (from an 8-bit timer to external trigger, or vice versa), first perform a software reset before changing the setting.

Note 3: To stop using an external trigger (ADTRG) to start A/D conversion, first disable hardware start (by setting ADHTG or HADHTG to 0) and then set P5FC<P57F> to 0 to set the pin to a general-purpose port.

Figure 3.13.2 A/D Converter Registers (5/12)

A/D Conversion Result Lower Register 08

		7	6	5	4	3	2	1	0
ADREG08L (0xFFFF_F300)	Bit symbol	ADR01	ADR00	—	—	—	—	OVR0	ADR0RF
	Read/Write	R		—	—	—	—	R	R
	After Reset	Undefined		—	—	—	—	0	0
	Function	Stores lower 2 bits of A/D conversion result						Overrun flag 0: No overrun occurred 1: Overrun occurred	A/D conversion result store flag 1: Conversion result stored

A/D Conversion Result Upper Register 08

		7	6	5	4	3	2	1	0
ADREG08H (0xFFFF_F301)	Bit symbol	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
	Read/Write	R							
	After Reset	Undefined							
	Function	Stores upper 8 bits of A/D conversion result							

A/D Conversion Result Lower Register 19

|		7	6	5	4	3	2	1	0
ADREG19L (0xFFFF_F302)	Bit symbol	ADR11	ADR10	—	—	—	—	OVR1	ADR1RF
	Read/Write	R		—	—	—	—	R	R
	After Reset	Undefined		—	—	—	—	0	0
	Function	Stores lower 2 bits of A/D conversion result						Overrun flag 0: No overrun occurred 1: Overrun occurred	A/D conversion result store flag 1: Conversion result stored

A/D Conversion Result Upper Register 19

		7	6	5	4	3	2	1	0
ADREG19H (0xFFFF_F303)	Bit symbol	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
	Read/Write	R							
	After Reset	Undefined							
	Function	Stores upper 8 bits of A/D conversion result							

Bits 2 to 5 are always read as 1s.

Note1: Bit 0 is the A/D conversion result store flag ADRxRF. This bit is set to 1 when an A/D converted value is stored in the register pair. This bit is cleared to 0 when the lower register (ADREGxL) is read.

Note2: Bit 1 is the overrun flag OVRx. This bit is set to 1 when the next conversion result is written before both conversion result registers (ADREGxH and ADREGxL) have been read. Reading the flag clears the bit.

Figure 3.13.2 A/D Converter Registers (6/12)

A/D Conversion Result Lower Register 2A

ADREG2AL
(0xFFFF_F304)

	7	6	5	4	3	2	1	0
Bit symbol	ADR21	ADR20	—	—	—	—	OVR2	ADR2RF
Read/Write	R		—	—	—	—	R	R
After Reset	Undefined		—	—	—	—	0	0
Function	Stores lower 2 bits of A/D conversion result						Overrun flag 0: No overrun occurred 1: Overrun occurred	A/D conversion result store flag 1: Conversion result stored

A/D Conversion Result Upper Register 2A

ADREG2AH
(0xFFFF_F305)

	7	6	5	4	3	2	1	0
Bit symbol	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
Read/Write	R							
After Reset	Undefined							
Function	Stores upper 8 bits of A/D conversion result							

A/D conversion result lower register 3B

ADREG3BL
(0xFFFF_F306)

	7	6	5	4	3	2	1	0
Bit symbol	ADR31	ADR30	—	—	—	—	OVR3	ADR3RF
Read/Write	R		—	—	—	—	R	R
After Reset	Undefined		—	—	—	—	0	0
Function	Stores lower 2 bits of A/D conversion result						Overrun flag 0: No overrun occurred 1: Overrun occurred	A/D conversion result store flag 1: Conversion result stored

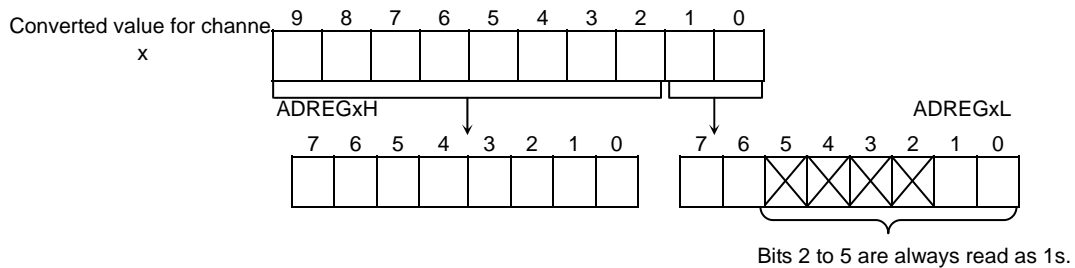### A/D conversion result upper register 3B

ADREG3BH
(0xFFFF_F307)

	7	6	5	4	3	2	1	0
Bit symbol	ADR39	ADR38	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
Read/Write	R							
After Reset	Undefined							
Function	Stores upper 8 bits of A/D conversion result							
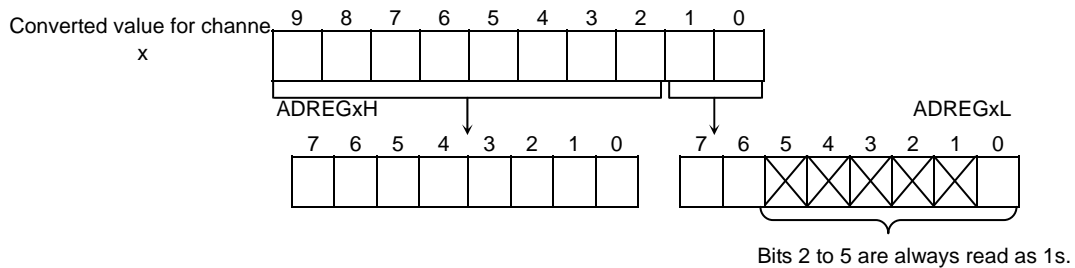
Converted value for channel x

ADREGxH   ADREGxL

Bits 2 to 5 are always read as 1s.

Note1: Bit 0 is the A/D conversion result store flag ADRxRF. This bit is set to 1 when an A/D converted value is stored in the register pair. This bit is cleared to 0 when the lower register (ADREGxL) is read.

Note2: Bit 1 is the overrun flag OVRx. This bit is set to 1 when the next conversion result is written before both conversion result registers (ADREGxH and ADREGxL) have been read. Reading the flag clears the bit.

Figure 3.13.2 A/D Converter Registers (7/12)

A/D Conversion Result Lower Register 4C

ADREG4CL (0xFFFF_F308)		7	6	5	4	3	2	1	0
	Bit symbol	ADR41	ADR40	—	—	—	—	OVR4	ADR4RF
	Read/Write	R		—	—	—	—	R	R
	After Reset	Undefined		—	—	—	—	0	0
	Function	Stores lower 2 bits of A/D conversion result						Overrun flag 0: No overrun occurred 1: Overrun occurred	A/D conversion result store flag 1: Conversion result stored

A/D Conversion Result Upper Register 4C

ADREG4CH (0xFFFF_F309)		7	6	5	4	3	2	1	0
	Bit symbol	ADR49	ADR48	ADR47	ADR46	ADR45	ADR44	ADR43	ADR42
	Read/Write	R							
	After Reset	Undefined							
	Function	Stores upper 8 bits of A/D conversion result							

A/D Conversion Result Lower Register 19

ADREG5DL (0xFFFF_F30A)		7	6	5	4	3	2	1	0
	Bit symbol	ADR51	ADR50	—	—	—	—	OVR5	ADR5RF
	Read/Write	R		—	—	—	—	R	R
	After Reset	Undefined		—	—	—	—	0	0
	Function	Stores lower 2 bits of A/D conversion result						Overrun flag 0: No overrun occurred 1: Overrun occurred	A/D conversion result store flag 1: Conversion result stored

A/D Conversion Result Upper Register 19

ADREG5DH (0xFFFF_F30B)		7	6	5	4	3	2	1	0
	Bit symbol	ADR59	ADR58	ADR57	ADR56	ADR55	ADR54	ADR53	ADR52
	Read/Write	R							
	After Reset	Undefined							
	Function	Stores upper 8 bits of A/D conversion result							

Converted value for channel x

ADREGxH   ADREGxL

Bits 2 to 5 are always read as 1s.

Note1: Bit 0 is the A/D conversion result store flag ADRxRF. This bit is set to 1 when an A/D converted value is stored in the register pair. This bit is cleared to 0 when the lower register (ADREGxL) is read.

Note2: Bit 1 is the overrun flag OVRx. This bit is set to 1 when the next conversion result is written before both conversion result registers (ADREGxH and ADREGxL) have been read. Reading the flag clears the bit.

Figure 3.13.2 A/D Converter Registers (8/12)

A/D Conversion Result Lower Register 6E

ADREG6EL (0xFFFF_F30C)		7	6	5	4	3	2	1	0
	Bit symbol	ADR61	ADR60	—	—	—	—	OVR6	ADR6RF
	Read/Write	R		—	—	—	—	R	R
	After Reset	Undefined		—	—	—	—	0	0
	Function	Stores lower 2 bits of A/D conversion result						Overrun flag 0: No overrun occurred 1: Overrun occurred	A/D conversion result store flag 1: Conversion result stored

A/D Conversion Result Upper Register 6E

ADREG6EH (0xFFFF_F30D)		7	6	5	4	3	2	1	0
	Bit symbol	ADR69	ADR68	ADR67	ADR66	ADR65	ADR64	ADR63	ADR62
	Read/Write	R							
	After Reset	Undefined							
	Function	Stores upper 8 bits of A/D conversion result							

A/D Conversion Result Lower Register 7F

ADREG7FL (0xFFFF_F30E)		7	6	5	4	3	2	1	0
	Bit symbol	ADR71	ADR70	—	—	—	—	OVR7	ADR7RF
	Read/Write	R		—	—	—	—	R	R
	After Reset	Undefined		—	—	—	—	0	0
	Function	Stores lower 2 bits of A/D conversion result						Overrun flag 0: No overrun occurred 1: Overrun occurred	A/D conversion result store flag 1: Conversion result stored

A/D Conversion Result Upper Register 7F

ADREG7FH (0xFFFF_F30F)		7	6	5	4	3	2	1	0
	Bit symbol	ADR79	ADR78	ADR77	ADR76	ADR75	ADR74	ADR73	ADR72
	Read/Write	R							
	After Reset	Undefined							
	Function	Stores upper 8 bits of A/D conversion result							

Converted value for channel x

ADREGxH   ADREGxL

Bits 2 to 5 are always read as 1s.

Note1: Bit 0 is the A/D conversion result store flag ADRxRF. This bit is set to 1 when an A/D converted value is stored in the register pair. This bit is cleared to 0 when the lower register (ADREGxL) is read.

Note2: Bit 1 is the overrun flag OVRx. This bit is set to 1 when the next conversion result is written before both conversion result registers (ADREGxH and ADREGxL) have been read. Reading the flag clears the bit.

Figure 3.13.2 A/D Converter Registers (9/12)

A/D Conversion Result Lower Register SP

		7	6	5	4	3	2	1	0
ADREGSPL (0xFFFF_F310)	Bit symbol	ADRSP1	ADRSP0	—	—	—	—	OVRSP	ADRSPRF
	Read/Write	R		—	—	—	—	R	R
	After Reset	Undefined		—	—	—	—	0	0
	Function	Stores lower 2 bits of A/D conversion result						Overrun flag 0: No overrun occurred 1: Overrun occurred	A/D conversion result store flag 1: Conversion result stored

A/D Conversion Result Upper Register SP

		7	6	5	4	3	2	1	0
ADREGSPH (0xFFFF_F311)	Bit symbol	ADRSP9	ADRSP8	ADRSP7	ADRSP6	ADRSP5	ADRSP4	ADRSP3	ADRSP2
	Read/Write	R							
	After Reset	Undefined							
	Function	Stores upper 8 bits of A/D conversion result							
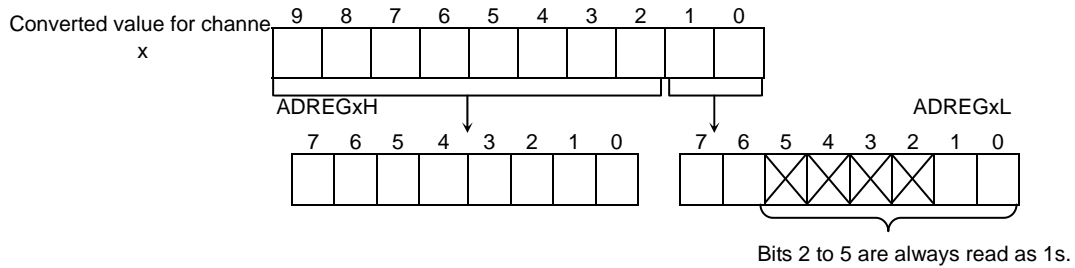
Bits 2 to 5 are always read as 1s.

Note1: Bit 0 is the A/D conversion result store flag ADRxRF. This bit is set to 1 when an A/D converted value is stored in the register pair. This bit is cleared to 0 when the lower register (ADREGxL) is read.

Note2: Bit 1 is the overrun flag OVRx. This bit is set to 1 when the next conversion result is written before both conversion result registers (ADREGxH and ADREGxL) have been read. Reading the flag clears the bit.

Figure 3.13.2 A/D Converter Registers (10/12)

A/D Conversion Result Lower Register

		7	6	5	4	3	2	1	0
ADREGSPL (0xFFFF_F314)	Bit symbol	ADR21	ADR20	—	—	—	—	—	—
	Read/Write	R/W		—	—	—	—	R	R
	After Reset	Undefined		—	—	—	—	0	0
	Function	Stores lower 2 bits of A/D conversion result							

A/D Conversion Result Compare Upper Register

		7	6	5	4	3	2	1	0
ADREGSPH (0xFFFF_F315)	Bit symbol	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
	Read/Write	R/W							
	After Reset	0							
	Function	Stores upper 8 bits of A/D conversion result							

Note: When setting or modifying a value in these registers, first disable the A/D monitor function by setting ADMOD3<ADOBSV> to 0.

Figure 3.13.2 A/D Converter Registers (11/12)

Stores lower 2 bits of A/D conversion result

		7	6	5	4	3	2	1	0
ADCCLK (0xFFFF_F31F)	Bit symbol	—	—	—	—	—	ADCCK2	ADCCK1	ADCCK0
	Read/Write	—	—	—	—	—	R/W	R/W	R/W
	After Reset	—	—	—	—	—	0	0	0
	Function						Selects prescaler output for A/D converter 000: fadc 001: Divided by 2 010: Divided by 4 011: Divided by 8 1XX: Divided by 16		

Note 1: A/D conversion is executed using the clock selected by the above register. However, if the accuracy of conversion needs to be guaranteed, be sure to choose a conversion clock frequency such that the conversion period is at least 1 μs, that is, 2 MHz or less in terms of A/D conversion clock frequency.

Note 2: Do not change the conversion clock frequency while A/D conversion is in progress. After conversion is completed, wait at least 2 ADCLK cycles before changing the clock frequency.
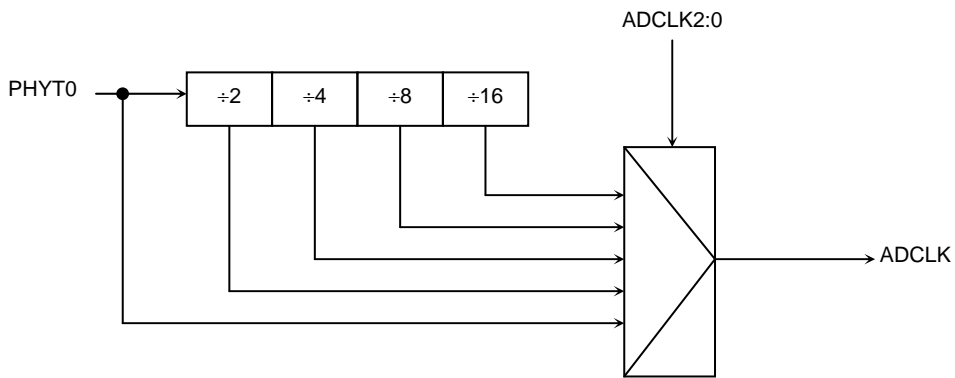
Figure 3.13.2 A/D Converter Registers (12/12)

3.13.2 Functional description

(1) Analog reference voltage

Apply the High-level analog reference voltage to the VREFH pin and the Low-level analog reference voltage to the VREFL pin. VREF is automatically turned on when A/D conversion is started and turned off when conversion is completed, thus preventing IREF from flowing unnecessarily.

(2) Selecting the analog input channel

The procedure for selecting the analog input channel varies with the A/D converter operating mode.

(2-1) Analog reference voltage

- Using a fixed analog input channel (ADMOD0<SCAN> = 0)

Choose one of the pins AN0 to AN15 as the analog input channel by setting ADMOD1<ADCH3:ADCH0> to the appropriate value.

- Scanning analog input channels (ADMOD0<SCAN> = 1)

Choose one of the 24 available scan modes by setting ADMOD1<ADCH3:ADCH0> and ADSCN to the appropriate values.

(2-2) Highest-priority A/D conversion

Choose one of the pins AN0 to AN15 as the analog input channel by setting ADMOD2<HPADCH3:HPADCH0> to the appropriate value.

After a reset, channel-fixed input on the AN0 pin is selected since ADMOD0<SCAN> and ADMOD1<ADCH3:ADCH0> are initialized to 0 and 0000, respectively. The pins other than that used as the analog input channel can be used as ordinary input port pins.

If highest-priority A/D conversion is activated during normal A/D conversion, highest-priority A/D conversion is performed upon the completion of the current conversion cycle. Normal A/D conversion is resumed upon the completion of highest-priority A/D conversion.

Example: When highest-priority A/D conversion for AN15
(ADMOD2<HPADCH3:HPADCH0> = 1111) is activated during repeated scan conversion for channels AN0 to AN3 (ADMOD0<REPEAT:SCAN> = 10 and ADMOD1<ADCH3:ADCH0> = 0011)

Highest-priority A/D Conversion Activated							
Channel Converted	Ch0	Ch1	Ch2	Ch15	Ch3	Ch0	Ch1
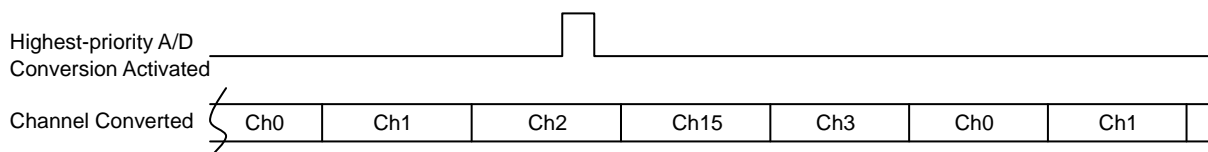
(3) Starting A/D conversion

A/D conversion is classified into normal A/D conversion and highest-priority A/D conversion. Normal A/D conversion can be initiated programmatically by setting ADMOD0<ADS> to 1. Highest-priority A/D conversion can be initiated programmatically by setting ADMOD2<HPADCE> to 1. Normal A/D conversion is performed in one of the four operating modes as specified with ADMOD0<2:1>. Highest-priority A/D conversion is only performed in channel-fixed single conversion mode. A/D conversion can also be activated by a hardware start source, which is specified with ADMOD4<ADHS> for normal A/D conversion and ADMOD4<HADHS> for highest-priority A/D conversion. When the ADHS or HADHS bit is set to 0, A/D conversion is triggered by a falling edge on the $\overline{\text{ADTRG}}$ pin. When the bit is set to 1, normal A/D conversion is triggered by INTTA0 from 8-bit timer 0 and highest-priority A/D conversion is triggered by INTTA1 from 8-bit timer 1. A/D conversion can still be started programmatically if hardware start is enabled.

When normal A/D conversion starts, the normal A/D conversion busy flag (ADMOD0<ADBF>) is set to 1, indicating that normal A/D conversion is in progress. When highest-priority A/D conversion starts, the highest-priority A/D conversion busy flag (ADMOD2<ADBFHP>) is set to 1, indicating that highest-priority A/D conversion is in progress, with the normal A/D conversion busy flag and end flag (EOCFN) holding the values they had before highest-priority A/D conversion starts.

Normal A/D conversion can be restarted by setting ADMOD0<ADS> to 1 during normal A/D conversion. Restarting normal A/D conversion cancels the current conversion. However, if all necessary sampling operations have been completed for the current conversion, the conversion result is stored before A/D conversion is restarted.

If the start of normal A/D conversion using a hardware resource is enabled, normal A/D conversion is restarted when the start condition for the resource is satisfied during normal A/D conversion. The current conversion is cancelled when normal A/D conversion is restarted. However, if all necessary sampling operations have been completed for the current conversion, the conversion result is stored before A/D conversion is restarted.

If ADMOD2<HPADCE> is set to 1 during normal A/D conversion, the result of the current conversion is stored in the conversion result registers, after which highest-priority A/D conversion starts, that is, A/D conversion for the channel specified with ADMOD2<3:0> (channel-fixed single conversion) starts. Once the result of highest-priority conversion is stored in ADREGSP, normal A/D conversion is resumed following the last conversion for which the result was stored.

If the start of highest-priority A/D conversion using a hardware resource is enabled and the start condition for the resource is satisfied during normal A/D conversion, the result of the current conversion is stored in the conversion result registers, after which highest-priority A/D conversion starts, that is, A/D conversion for the channel specified with ADMOD2<3:0> (channel-fixed single conversion) starts. Once the result of highest-priority conversion is stored in ADREGSP, normal A/D conversion is resumed following the last conversion for which the result was stored.

Highest-priority A/D conversion is not restarted even if ADMOD2<HPADCE> is set to 1 during highest-priority A/D conversion.

(4) A/D conversion modes and the A/D conversion completed interrupt

The following four A/D conversion modes are available for normal A/D conversion, as specified with the settings of ADMOD0<2:1>. Highest-priority A/D conversion always operates in channel-fixed single conversion mode regardless of the settings of ADMOD0<2:1>.

- Channel-fixed single conversion mode
- Channel scan single conversion mode
- Channel-fixed repeated conversion mode
- Channel scan repeated conversion mode

(4-1) Normal A/D conversion

Use ADMOD0<REPEAT:SCAN> to select the A/D conversion mode. When A/D conversion has been started, ADMOD0<ADBFN> is set to 1. When the specified A/D conversion has been completed, an A/D conversion completed interrupt (INTAD) is generated and ADMOD0<EOCF> is set to 1, indicating that A/D conversion has been completed. If REPEAT = 0, ADBFN is cleared to 0 simultaneously when EOCF is set to 1. If REPEAT = 1, however, conversion continues with ADBFN held to be 1.

(a) Channel-fixed single conversion mode

Channel-fixed single conversion mode is selected by setting ADMOD0<REPEAT:SCAN> to 00.

In this mode, conversion is performed once for a single selected channel. After the conversion has been completed, ADMOD0<EOCF> will be set to 1 and ADMOD0<ADBF> cleared to 0, thereby generating an INTAD interrupt request. EOCF can be cleared to 0 by reading it.

(b) Channel scan single conversion mode

Channel scan single conversion mode is selected by setting ADMOD0<REPEAT:SCAN> to 01.

In this mode, conversion is performed once for each selected channel which is scanned. After the scan conversion has been completed, ADMOD0<EOCF> will be set to 1 and ADMOD0<ADBF> cleared to 0, thereby generating an INTAD interrupt request. EOCF can be cleared to 0 by reading it.

(c) Channel-fixed repeated conversion mode

Channel-fixed repeated conversion mode is selected by setting ADMOD0<REPEAT:SCAN> to 10.

In this mode, conversion is performed repeatedly for a single selected channel. After the conversion has been completed, ADMOD0<EOCF> will be set to 1. ADMOD0<ADBF>, however, remains at 1 and is not cleared to 0. The timing at which an INTAD interrupt request will be generated depends on the settings of ADMOD0<ITM1:ITM0>, which also determine the timing at which EOCF is set. EOCF can be cleared to 0 by reading it.

If ADMOD0<ITM1:ITM0> = 00, an interrupt request will be generated for each A/D conversion session completed. In that case, the conversion result is always stored in ADREG08. EOCF becomes 1 once the conversion result has been stored.

If ADMOD0<ITM1:ITM0> = 01, an interrupt request will be generated for every fourth A/D conversion session completed. In that case, the conversion results are stored sequentially in registers ADREG08 through ADREG3B. EOCF becomes 1 once the conversion result has been stored in ADREG3B. The next conversion result will be stored in ADREG08 again. EOCF can be cleared to 0 by reading it.

If ADMOD0<ITM1:ITM0> = 10, an interrupt request will be generated for every eighth A/D conversion session completed. In that case, the conversion results are stored sequentially in registers ADREG08 through ADREG7F. EOCF becomes 1 once the conversion result has been stored in ADREG7F. The next conversion result will be stored in ADREG08 again. EOCF can be cleared to 0 by reading it.

The setting of ADMOD0<RI> determines the repeat interval for repeated conversion mode. If ITM1:ITM0 = 00, this bit controls the interval between a single conversion being completed and the next conversion being started. If ITM1:ITM0 = 01, the bit controls the interval between four conversions being completed and the next conversion being started. If ITM1:ITM0 = 10, the bit controls the interval between eight conversions being completed and the next conversion being started.

(d) Channel scan repeated conversion mode

Channel scan repeated conversion mode is selected by setting ADMOD0<REPEAT:SCAN> to 11.

In this mode, conversion is performed repeatedly for selected scanned channels. Each time one scan conversion operation has been completed, ADMOD0<EOCF> will be set to 1, thereby generating an INTAD interrupt request. ADMOD0<ADBF> remains at 1 and is not cleared to 0. EOCF can be cleared to 0 by reading it.

To stop operation in repeat conversion mode (mode (c) or (d)), write a 0 to ADMOD0<REPEAT>. Repeat conversion mode will then be terminated and ADMOD0<ADBF> cleared to 0 as soon as the conversion currently in progress has been completed.

If ADMOD1<I2AD> = 0 and the device enters a standby state (IDLE, SLEEP or STOP mode), the A/D converter will immediately stop operating, even if A/D conversion is in progress. After the device has exited the standby state, if the A/D converter is operating in repeat conversion mode (mode (c) or (d)), A/D conversion will start again from the beginning (the register settings remain the same, status information is initialized and operation is restarted from the beginning); however, if the A/D converter is operating in single conversion mode (mode (a) or (b)), it will not restart conversion operation (it will remain stopped).

(4-2) Highest-priority A/D conversion

Highest-priority A/D conversion is only performed in channel-fixed single conversion mode, regardless of the settings of ADMOD0<REPEAT:SCAN>. When the start condition is satisfied, conversion for the channel specified with ADMOD2<HPADCH3:HPADCH0> is performed once. After the conversion has been completed, a highest-priority A/D conversion completed interrupt will be generated, ADMOD2<EOCFHP> set to 1 and <ADBFHP> cleared to 0. The EOCFHP flag can be cleared to 0 by reading it.

Table 3.13.1 Relationship Among A/D Conversion Modes, Interrupt Generation Timing and Flag Operation
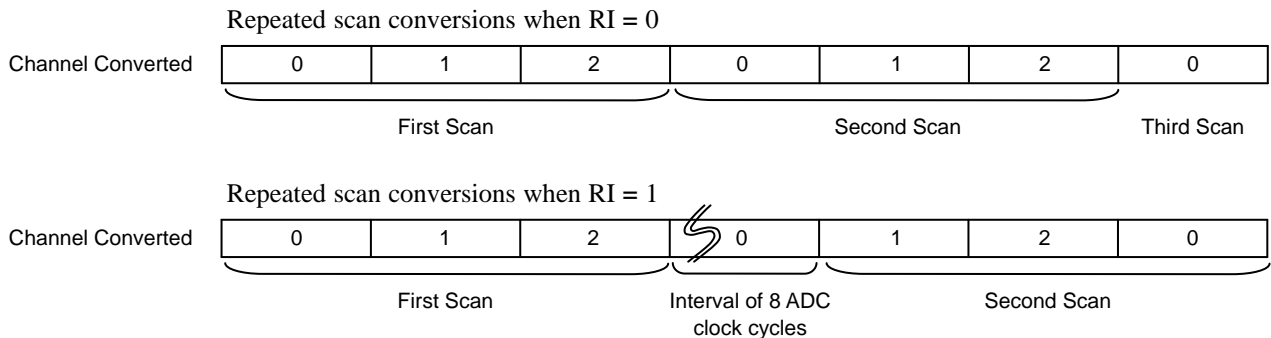
Conversion Mode	Interrupt Generation Timing	EOCF Setting Timing (*1)	ADBF (After Interrupt is Generated)	ADMOD0		
				ITM1:0	REPEAT	SCAN
Channel-fixed single conversion	After conversion has been completed	After conversion has been completed	0	—	0	0
Channel-fixed repeated conversion	Every time one conversion has been completed	Every time one conversion has been completed	1 (*2)	00	1	0
	Every time four conversions have been completed	Every time four conversions have been completed	1 (*2)	01		
	Every time eight conversions have been completed	Every time eight conversions have been completed	1 (*2)	10		
Channel scan single conversion	After scan conversion has been completed	After scan conversion has been completed	0	—	0	1
Channel scan repeated conversion	Every time one scan conversion has been completed	Every time one scan conversion has been completed	1 (*2)	—	1	1

(Note*1) EOCF is cleared when it is read.

(Note*2) If repeat intervals are used with RI set to 1, ADBF indicates 0 during interval periods.

ADMOD0<RI> can be used to control the time between one scan conversion being completed and the next scan conversion being started (repeat interval). This bit is only effective when REPEAT = 1.

Example: When repeated scan for channels AN0 to AN2 is set

Repeated scan conversions when RI = 0

Channel Converted	0	1	2	0	1	2	0

First Scan   Second Scan   Third Scan

Repeated scan conversions when RI = 1

Channel Converted	0	1	2	0	1	2	0

First Scan   Interval of 8 ADC clock cycles   Second Scan

Note: If the start condition for highest-priority A/D conversion is satisfied during an interval period, highest-priority A/D conversion is started immediately. Since the interval counter continues running during highest-priority A/D conversion, the next scan will start when both of the following conditions are satisfied: an overflow of the interval counter and the completion of highest-priority A/D conversion.

(5) Highest-priority conversion mode

Highest-priority A/D conversion can be performed by interrupting normal A/D conversion. Highest-priority A/D conversion can be started either programmatically by setting ADMOD2<HPADCE> to 1 or by using a hardware resource as specified with ADMOD4<7:6>. If highest-priority A/D conversion is started during normal A/D conversion, the converter first stores the result of the current conversion to the appropriate result register pair, and then performs a single conversion for the channel specified with ADMOD2<3:0>. The result of that conversion is stored in ADREGSP, at which point a highest-priority A/D conversion interrupt is generated. Then, normal A/D conversion is resumed following the last conversion for which the result was stored. Any condition that triggers highest-priority A/D conversion is ignored while highest-priority A/D conversion is in progress.

For example, suppose channel scan repeated conversion is being performed for AN0 to AN8. If HPADCE is set to 1 during conversion for AN3, the converter will wait for the conversion for AN3

to complete and then perform conversion for the channel specified with HPADC3:HPADC0. After storing the result of that conversion in ADREGSP, the converter resumes channel scan repeated conversion from AN4.

(6) A/D monitor function

Setting ADMOD3<ADOBSV> to 1 enables the A/D monitor function, which generates an A/D monitor interrupt if the value of the conversion result register specified with REGS<3:0> is greater or less (as specified with ADOBIC) than the value of the compare register. This comparison is performed each time the result is stored in the specified conversion result register, and an interrupt is generated if the condition is satisfied. Since the conversion result register used for the A/D monitor function is usually not read in the program, its overrun flag (OVRn) and conversion result store flag (ADRnRF) are always set. Therefore, do not use those flags of the conversion result register used for the A/D monitor function.

(7) A/D conversion time

Two clock pulses are required for a single A/D conversion. The A/D conversion clock frequency can be selected from among prescaler outputs PHYT0, PHYT1, PHYT2, PHYT4 and PHYT8. To guarantee the accuracy of the conversion, the A/D conversion time must be at least 1 μs, that is, 2 MHz or less in terms of A/D conversion clock frequency. The following figure and tables show example settings:

Example: Repeated scan conversion for channels 0 to 2 (ADC clock frequency = 1 MHz)
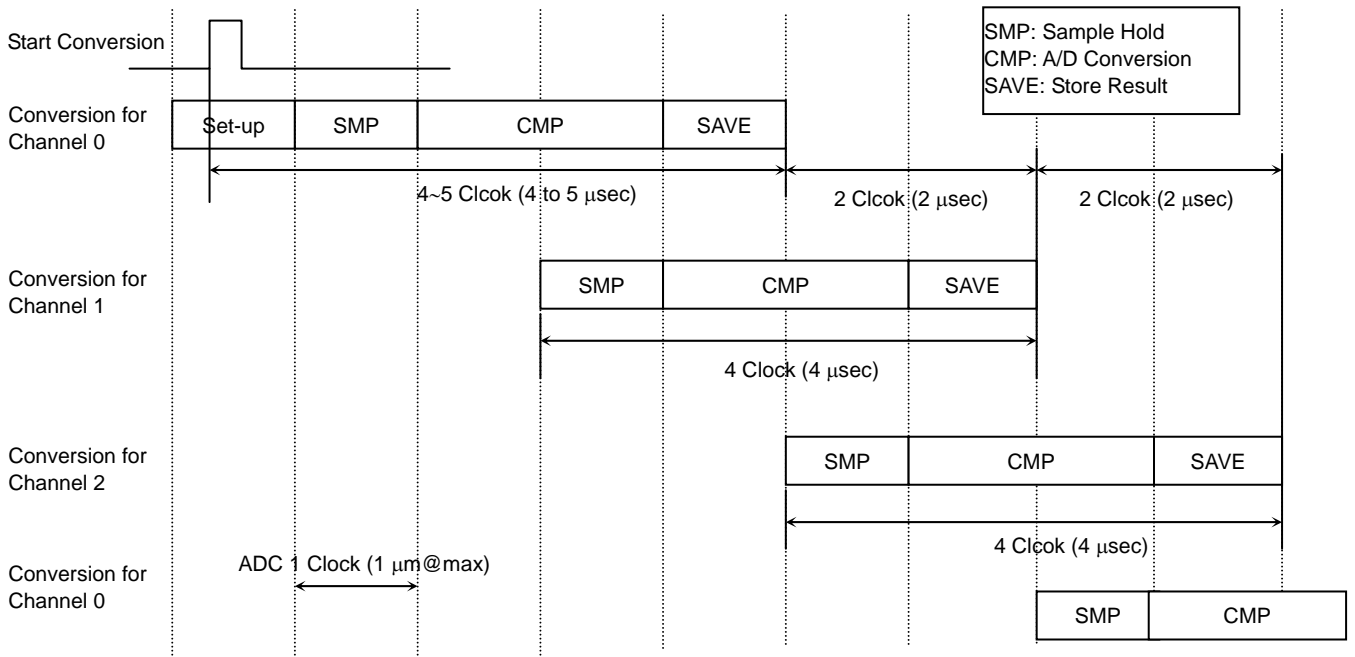
Table 3.13.2 Example A/D Conversion Settings (1)

@f = 32 MHz

Peripheral Clock Select <FPSEL>	Clock Gear <GEAR1:0>	Prescaler Clock Source <PRCK1:0>	A/D Conversion Time				
			φT0	φT1	φT2	φT4	φT8
0 (fgear)	00 (fc)	00 (fperiph/4)	Invalid setting	Invalid setting	1μs	2μs	4μs
		01 (fperiph/2)	Invalid setting	Invalid setting	Invalid setting	1μs	2μs
		10 (fperiph)	Invalid setting	Invalid setting	Invalid setting	Invalid setting	1μs
	01 (fc/2)	00 (fperiph/4)	Invalid setting	1μs	2μs	4μs	8μs
		01 (fperiph/2)	Invalid setting	Invalid setting	1μs	2μs	4μs
		10 (fperiph)	Invalid setting	Invalid setting	Invalid setting	1μs	2μs
	10 (fc/4)	00 (fperiph/4)	1μs	2μs	4μs	8μs	16μs
		01 (fperiph/2)	Invalid setting	1μs	2μs	4μs	8μs
		10 (fperiph)	Invalid setting	Invalid setting	1μs	2μs	4μs
	11 (fc/8)	00 (fperiph/4)	2μs	4μs	8μs	16μs	32μs
		01 (fperiph/2)	1μs	2μs	4μs	8μs	16μs
		10 (fperiph)	Invalid setting	1μs	2μs	4μs	8μs
1 (fc)	00 (fc)	00 (fperiph/4)	Invalid setting	Invalid setting	1μs	2μs	4μs
		01 (fperiph/2)	Invalid setting	Invalid setting	Invalid setting	1μs	2μs
		10 (fperiph)	Invalid setting	Invalid setting	Invalid setting	Invalid setting	1μs
	01 (fc/2)	00 (fperiph/4)	Invalid setting	1μs	2μs	4μs	8μs
		01 (fperiph/2)	Invalid setting	Invalid setting	1μs	2μs	4μs
		10 (fperiph)	Invalid setting	Invalid setting	Invalid setting	1μs	2μs
	10 (fc/4)	00 (fperiph/4)	1μs	2μs	4μs	8μs	16μs
		01 (fperiph/2)	Invalid setting	1μs	2μs	4μs	8μs
		10 (fperiph)	Invalid setting	Invalid setting	1μs	2μs	4μs
	11 (fc/8)	00 (fperiph/4)	2μs	4μs	8μs	16μs	32μs
		01 (fperiph/2)	1μs	2μs	4μs	8μs	16μs
		10 (fperiph)	Invalid setting	1μs	2μs	4μs	8μs

Table 3.13.3 Example A/D Conversion Settings (2)

@f = 32 MHz

PHYT0	Conversion Clock				
	f_{adc}	$f_{adc}/2$	$f_{adc}/4$	$f_{adc}/8$	$f_{adc}/16$
16MHz	Invalid setting	Invalid setting	Invalid setting	Invalid setting	2 µsec
12 MHz	Invalid setting	Invalid setting	Invalid setting	Invalid setting	2.8µsec
10 MHz	Invalid setting	Invalid setting	Invalid setting	Invalid setting	3.2µsec
8 MHz	Invalid setting	Invalid setting	Invalid setting	2 µsec	4 µsec
4 MHz	Invalid setting	Invalid setting	2 µsec	4 µsec	8 µsec
2MHz	Invalid setting	2 µsec	4 µsec	8 µsec	16 µsec

Note: The maximum conversion speed, that is, the minimum conversion time this A/D converter can achieve is 2 µs.
         However, 4 µs is required before the first conversion result can be retrieved from the conversion result register
         (or a maximum of 5 µs is required depending on the conversion start request timing because of the interface
         between the system clock and A/D conversion clock). Subsequently, conversion results can be obtained every
         2 µs. Therefore, in single conversion mode or highest-priority conversion mode, A/D conversion requires 4 or 5
         times the conversion time shown in the above table. In repeated conversion or scan mode, only the first
         conversion requires 4 or 5 times the table value (a maximum of 4 to 5 µs) but subsequent conversions are
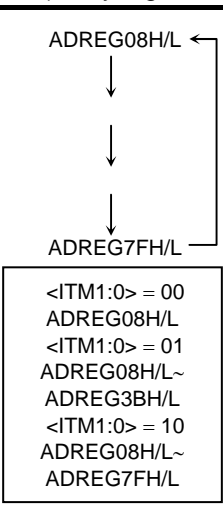   performed within the time shown in the table (a maximum of 2 µs).

(8) Storing and reading out A/D conversion results

A/D conversion results are stored in A/D conversion result upper/lower registers (ADREG08H/L to ADREG7FH/L).

In channel-fixed repeated conversion mode, A/D conversion results are sequentially stored in ADREG08H/L to ADREG7FH/L. However, if ITM1 and ITM0 specify that an interrupt be generated every time conversion has been completed, conversion results will be stored in ADREG08H/L only. If ITM1 and ITM0 specify that an interrupt be generated every fourth time conversion has been completed, conversion results will be sequentially stored in ADREG08H/L to ADREG3BH/L.

Table 3.13.3 shows the relationship between analog input channels and A/D conversion result registers.

Table 3.13.3 Relationship Between Analog Input Channels and A/D Conversion Result Registers

Analog Input Channel (Port A)	A/D Conversion Result Register	
	Other than channel-fixed repeated conversion mode	Channel-fixed repeated conversion mode (every eighth time)
AN0	ADREG08H/L	ADREG08H/L ←
AN1	ADREG19H/L	↓
AN2	ADREG2AH/L	
AN3	ADREG3BH/L	↓
AN4	ADREG4CH/L	
AN5	ADREG5DH/L	↓
AN6	ADREG6EH/L	
AN7	ADREG7FH/L	ADREG7FH/L ─
AN8	ADREG08H/L	<ITM1:0> = 00 ADREG08H/L
AN9	ADREG19H/L	<ITM1:0> = 01 ADREG08H/L~
AN10	ADREG2AH/L	ADREG3BH/L
AN11	ADREG3BH/L	<ITM1:0> = 10 ADREG08H/L~
AN12	ADREG4CH/L	ADREG7FH/L
AN13	ADREG5DH/L	
AM14	ADREG6EH/L	
AM15	ADREG7FH/L	

In highest-priority A/D conversion mode, conversion results are always stored in ADREGSPH/L.

(9) Data polling

To process the results of A/D conversion by means of data polling rather than using an interrupt, poll ADMOD0<EOCF>. If this flag is set, the appropriate A/D conversion result register pair contains the conversion result. Check the flag and then, if it is set, read the A/D conversion result registers. To detect an overrun, first read the upper register and then the lower register. The conversion result is valid if OVRn =0 and ADRnRF = 1 in the lower register.

3.14 Digital/Analog Converter

This section describes the D/A converter the TMP1942 contains.

3.14.1 Features

- Three 10-bit D/A converter channels.

- Each channel contains a full-range buffer amplifier.

- Each channel can be placed in standby state using control registers.

3.14.2 Operation

When the OP and REFON bits of the control register DACCNTn are set to 1s, writing output code and the VALID bit to the output register pair DAREGnL/DAREGnH causes the voltage corresponding to the output code to appear on the DAOUTn output pin. The value in the output registers will be reflected in DAOUT only if the VALID bit is set. Therefore, when updating the code, set the VALID bit if 10-bit data has been updated in DAREGnH first and then DAREGnL. Once the VALID bit has been set to 1, the value stored in DAREGnL/H is fetched into the D/A converter as 10-bit data, which will be recognized as code. Setting DACCNTn<OP> to 0 places the DAOUTn output pin into the high-impedance state. Setting DACCNTn<REFON> to 0 enables reduction of current consumption by decreasing Iref.
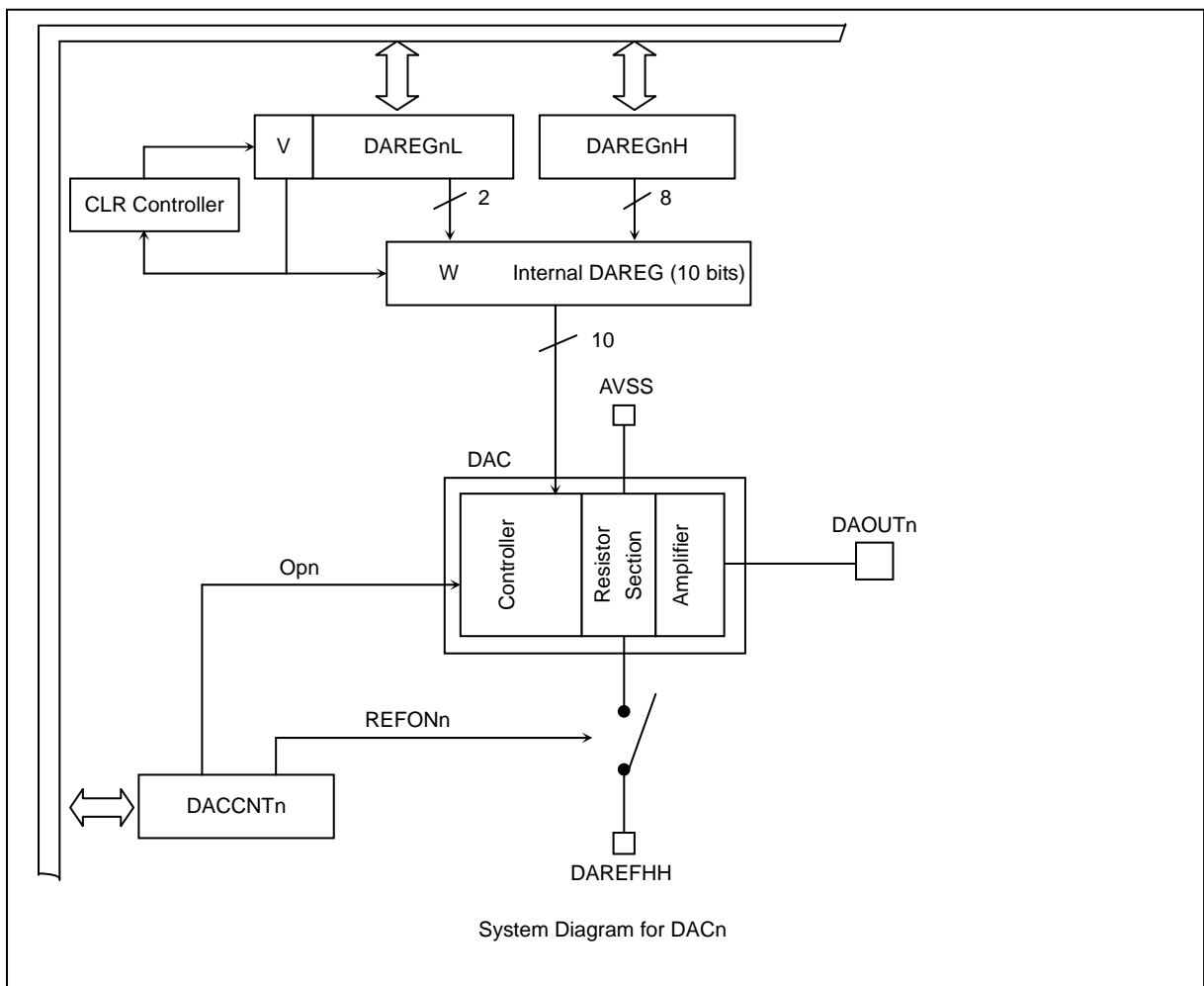

Figure 3.14.1 D/A Converter Block Diagram

DACCNT0 Register

(0xFFFF_F342)		7	6	5	4	3	2	1	0
	Bit symbol	—	—	—	—	—	—	REFON0	OP0
	Read/Write	—	—	—	—	—	—	R/W	R/W
	After Reset	—	—	—	—	—	—	0	0
	Function							0: Ref off 1: Ref on	0: Output high-impedance 1: Output

DACCNT1 Register

(0xFFFF_F346)		7	6	5	4	3	2	1	0
	Bit symbol	—	—	—	—	—	—	REFON1	OP1
	Read/Write	—	—	—	—	—	—	R/W	R/W
	After Reset	—	—	—	—	—	—	0	0
	Function							0: Ref off 1: Ref on	0: Output high-impedance 1: Output

DACCNT2 Register

(0xFFFF_F34A)		7	6	5	4	3	2	1	0
	Bit symbol	—	—	—	—	—	—	REFON2	OP2
	Read/Write	—	—	—	—	—	—	R/W	R/W
	After Reset	—	—	—	—	—	—	0	0
	Function							0: Ref off 1: Ref on	0: Output high-impedance 1: Output

Output Register DAREG0L

(0xFFFF_F340)		7	6	5	4	3	2	1	0
	Bit symbol	DAC01	DAC00	—	—	—	—	—	VALID
	Read/Write	R/W	R/W	R/W	R/W	—	—	—	W
	After Reset	0	0	0	0	—	—	—	0
	Function			Must always be set to 0.	Must always be set to 0.				0: Don't care 1: Output code valid

Output Register DAREG0H

(0xFFFF_F341)		7	6	5	4	3	2	1	0
	Bit symbol	DAC09	DAC08	DAC07	DAC06	DAC05	DAC04	DAC03	DAC02
	Read/Write				R/W				
	After Reset	0	0	0	0	0	0	0	0
	Function								

Note: When writing data to DAREG0, first write DAREG0H and then DAREG0L, using byte accesses.

Output Register DAREG1L

	7	6	5	4	3	2	1	0
Bit symbol	DAC1	DAC0	—	—	—	—	—	VALID
Read/Write	R/W	R/W	R/W	R/W	—	—	—	W
After Reset	0	0	0	0	—	—	—	0
Function			Must always be set to 0.	Must always be set to 0.			0: Ref off 1: Ref on	0: Don't care 1: Output code valid

(0xFFFF_F344)

Output Register DAREG1H

	7	6	5	4	3	2	1	0
Bit symbol	DAC9	DAC8	DAC7	DAC6	DAC5	DAC4	DAC3	DAC2
Read/Write	R/W							
After Reset	0	0	0	0	0	0	0	0
Function								

(0xFFFF_F345)

Output Register DAREG2L

	7	6	5	4	3	2	1	0
Bit symbol	DAC1	DAC0	—	—	—	—	—	VALID
Read/Write	R/W	R/W	R/W	R/W	—	—	—	W
After Reset	0	0	0	0	—	—	—	0
Function			Must always be set to 0.	Must always be set to 0.				0: Don't care 1: Output code valid

(0xFFFF_F348)

Output Register DAREG2H

	7	6	5	4	3	2	1	0
Bit symbol	DAC9	DAC8	DAC7	DAC6	DAC5	DAC4	DAC3	DAC2
Read/Write	R/W							
After Reset	0	0	0	0	0	0	0	0
Function								

(0xFFFF_349)

Note: When writing data to DAREG1 and DAREG0, first write DAREGnH and then DAREGnL, using byte accesses.

3.15 Key on Wake-up Circuit

3.15.1 Overview

- 14 inputs, KEY0 to KEYD, can be used to terminate STOP/SLEEP mode or as an external interrupt. However, all 14 inputs must be set collectively (in the CG block). Whether individual pins are used or not used can be specified separately (KWUPSTn).

- A single interrupt source is available.

- Rising edge, falling edge, High level, or Low level detection can be selected for individual inputs (KWUPSTn).

- The interrupt source is cleared by KWUPCLR in the interrupt handling routine.

- Key input pins have pull-up resistors which can be enabled or disabled by setting bit 0 (PE) of KWUPCNT. Bit 1 (DPE) specifies whether the pull-up resistors are dynamic or static. Pull-up resistors cannot be set individually.

3.15.2 Key on wake-up operation

The TMP1942 has 14 key input pins (KEY0 to KEYD). The KWUPEN bit of the IMCGB1 register in the CG specifies whether the key inputs are used to terminate standby mode or as an ordinary interrupt. Setting the bit to 1 causes all of KEY0 to KEYD to be used to terminate standby mode. Use KWUPSTn<KEYnEN> to specify whether to use each key input and KWUPSTn<KEYn1: KEYn0> to specify the active condition for each key input. The key on wake-up circuit detects key inputs and reports the result of detection to the CG IMCGB1 register using an active High signal. Therefore, set the detection level to High level by setting IMCGB1<EMCG51:EMCG50> to 01. Since the result of detection in the CG is also reported to the interrupt controller (INTC) as an active High signal, set the corresponding interrupt to High level-detected in the INTC. Setting IMCGB1<KWUPEN> to 0 (default) causes all of KEY0 to KEYD to be used as ordinary interrupts. In that case, set the detection level to High level in the INTC but the CG need not be set. Also use KWUPSTn to specify whether each key input is used and its active condition. In the interrupt handling routine, write 1010 to KWUPCLR to clear all key interrupts.

> Note: If two or more key inputs are detected at different times, the second key input is cleared simultaneously with the first key input if the second key input is detected before the key interrupt clearing sequence in the interrupt handling routine for the first key input. If the second key input is detected after the clearing sequence for the first key input, a key interrupt will be generated again.

3.15.3 Pull-up function

Each key input has a pull-up resistor. Setting KWUPCNT<PE> to 1 results in all of KEY0 to KEYD being pulled up. However, any key inputs which have been specified not to be used with KWUPSTn<KEYnEN> will not be pulled up regardless of the setting of this bit.

Setting KWUPCNT<DPE> to 1 selects dynamic pull-up mode, where the key inputs are pulled up only during given periods at a frequency specified with T1S1:T1S0 and T2S1:T2S0. In this mode, current consumed by the key inputs can be reduced. When DPE is set to 0, the key inputs are always pulled up.

Note1: Procedures for using key inputs in static pull-up mode

A) When setting key inputs first after powering up the device
1) Set KWUPCNT (PE = 1, DPE = 0).
2) Set the KWUPSTn<KEYnEN> corresponding to the key inputs to be used to 1.
3) Wait until the pull-up resistors are disabled.
4) Set the active conditions using the KWUPSTn corresponding to the key inputs to be used.
5) Clear the interrupt request using KWUPCLR.
6) Set the CG and INTC (refer to Section 3.4, "Interrupts" for details).

B) When modifying the active condition for a key input during operation
1) Disable key interrupts in the INTC (IMC1<18:16> = 000).
2) Modify the active condition for the key input using the corresponding KWUPSTn.
3) Clear the interrupt request using KWUPCLR.
4) Enable key interrupts in the INTC (set IMC1<18:16> to an appropriate level).

C) When enabling a key input during operation
1) Disable key interrupts in the INTC (IMC1<18:16> = 000).
2) Set the KWUPSTn<KEYnEN> corresponding to the key input to be used to 1.
3) Wait until the pull-up resistors are disabled.
4) Set the active condition using the KWUPSTn corresponding to the key input to be used.
5) Clear the interrupt request using KWUPCLR.
6) Enable key interrupts in the INTC (set IMC1<18:16> to an appropriate level).

Note2: Procedures for using key inputs in dynamic pull-up mode

A) When setting key inputs first after powering up the device
1) Set KWUPCNT (PE = 1, DPE = 0, TnSn = desired time).
2) Set the active conditions using the KWUPSTn corresponding to the key inputs to be used.
3) Clear the interrupt request using KWUPCLR.
4) Set the KWUPSTn<KEYnEN> corresponding to the key inputs to be used to 1.
5) Set the CG and INTC (refer to Section 3.4, "Interrupts" for details).

B) When modifying the active condition for a key input during operation
1) Disable key interrupts in the INTC (IMC1<18:16> = 000).
2) Modify the active condition for the key input using the corresponding KWUPSTn.
3) Clear the interrupt request using KWUPCLR.
4) Enable key interrupts in the INTC (set IMC1<18:16> to an appropriate level).

C) When enabling a key input during operation
1) Disable key interrupts in the INTC (IMC1<18:16> = 000).
2) Set the active condition using the KWUPSTn corresponding to the key input to be used.
3) Clear the interrupt request using KWUPCLR.
4) Set the KWUPSTn<KEYnEN> corresponding to the key input to be used to 1.
5) Enable key interrupts in the INTC (set IMC1<18:16> to an appropriate level).

Note3: Procedures for using key inputs without pull-up resistors

    A) When setting key inputs first after powering up the device

        1) Set KWUPCNT (PE = 0, DPE = 0).

        2) Set the active conditions using the KWUPSTn corresponding to the key inputs to be used.

        3) Clear the interrupt request using KWUPCLR.

        4) Set the KWUPSTn<KEYnEN> corresponding to the key inputs to be used to 1.

        5) Set the CG and INTC (refer to Section 3.4, "Interrupts" for details).

    B) When modifying the active condition for a key input during operation

        1) Disable key interrupts in the INTC (IMC1<18:16> = 000).

        2) Modify the active condition for the key input using the corresponding KWUPSTn.

        3) Clear the interrupt request using KWUPCLR.

        4) Enable key interrupts in the INTC (set IMC1<18:16> to an appropriate level).

    C) When enabling a key input during operation

        1) Disable key interrupts in the INTC (IMC1<18:16> = 000).

        2) Set the active condition using the KWUPSTn corresponding to the key input to be used.

        3) Clear the interrupt request using KWUPCLR.

        4) Set the KWUPSTn<KEYnEN> corresponding to the key input to be used to 1.

        5) Enable key interrupts in the INTC (set IMC1<18:16> to an appropriate level).

> Note: Ensure that fs is operating before attempting to enable dynamic pull-up by setting DPE to 1.
> If DPE is set to 1 when fs is not operating, key inputs cannot be detected.

Key on wake-up control register KWUPCNT

	7	6	5	4	3	2	1	0
Bit symbol	—	—	T2S1	T2S0	T1S1	T1SO	DPE	KYPE
Read/Write	R/W	—	R/W					
After Reset	0	—	—	—	—	—	0	0
Function	Must always be set to 0.		Dynamic pull-up interval 00: 128/fs 10: 512 /fs 01: 256/fs 11: 1024/fs		Dynamic pull-up period 00: 4/fs 10: 16/fs 01: 8/fs 11: 32/fs		0: Static pull-up 1: Dynamic pull-up	0: Disable pull-up 1: Enable pull-up

(0xFFFF_F371)

The following illustrates operation in dynamic pull-up mode:
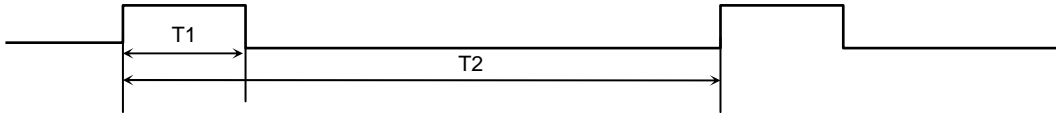
Key inputs are pulled up only during the T1 periods as specified with T1S1: T1S0.

00: 4/fs (125 μs @ fs=32kHz)
01: 8/fs (250 μs @ fs=32kHz)
10: 16/fs (500 μs @ fs=32kHz)
11: 32/fs (1 ms @ fs=32kHz)

Dynamic pull-up is repeated at intervals of T2 as specified with T2S1:T2S0.

00: 128/fs (4 ms @ fs=32kHz)
01: 256/fs (8 m @ fs=32kHz s)
10: 512/fs (16 ms @ fs=32kHz)
11: 1024/fs (32 ms @ fs=32kHz)

3.15.4   Detecting Key Inputs and Detection Timing

1)   When pull-up resistors are disabled with PE set to 0

KWUPSTn<KEYn1:0> can be used to specify a High level, Low level, rising edge or falling edge as the active condition for key inputs. The active condition for key inputs is constantly monitored.

2)   In static pull-up mode with PE set to 1 and DPE set to 0

KWUPSTn<KEYn1:0> can be used to specify a High level, Low level, rising edge or falling edge as the active condition for key inputs. The active condition for key inputs is constantly monitored.

3)   In dynamic pull-up mode with PE set to 1 and DPE set to 1

The active condition for each key input (interrupt) is detected one fs clock cycle before the T1 period ends. Only edge detection is supported. Therefore, key input must be asserted for at least a period of T2. In this case, do not set the active condition to a level. There is a delay of up to T2 before detection. The following figure shows an example when the active condition is a falling edge.

		7	6	5	4	3	2	1	0
KWUPST0	Bit symbol	—	—	KEY01	KEY00	—	—	—	KEY0EN
(0xFFFF_F360)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets KEY0 active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					KEY0 interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
KWUPST1	Bit symbol	—	—	KEY11	KEY10	—	—	—	KEY1EN
(0xFFFF_F361)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets KEY1 active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					KEY1 interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
KWUPST2	Bit symbol	—	—	KEY21	KEY20	—	—	—	KEY2EN
(0xFFFF_F362)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets KEY2 active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					KEY2 interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
KWUPST3	Bit symbol	—	—	KEY31	KEY30	—	—	—	KEY3EN
(0xFFFF_F363)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets KEY3 active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					KEY3 interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
KWUPST4	Bit symbol	—	—	KEY41	KEY40	—	—	—	KEY4EN
(0xFFFF_F364)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets KEY4 active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					KEY4 interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
KWUPST5	Bit symbol	—	—	KEY51	KEY50	—	—	—	KEY5EN
(0xFFFF_F365)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets KEY5 active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					KEY5 interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
KWUPST6	Bit symbol	—	—	KEY61	KEY60	—	—	—	KEY6EN
(0xFFFF_F366)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets KEY6 active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					KEY6 interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
KWUPST7	Bit symbol	—	—	KEY71	KEY70	—	—	—	KEY7EN
(0xFFFF_F367)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets KEY7 active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					KEY7 interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
KWUPST8	Bit symbol	—	—	KEY81	KEY80	—	—	—	KEY8EN
(0xFFFF_F368)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets KEY8 active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					KEY8 interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
KWUPST9	Bit symbol	—	—	KEY91	KEY90	—	—	—	KEY9EN
(0xFFFF_F369)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets KEY9 active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					KEY9 interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
KWUPSTA	Bit symbol	—	—	KEYA1	KEYA0	—	—	—	KEYAEN
(0xFFFF_F36A)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets KEYA active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					KEYA interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
KWUPSTB	Bit symbol	—	—	KEYB1	KEYB0	—	—	—	KEYBEN
(0xFFFF_F36B)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets KEYB active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					KEYB interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
KWUPSTC	Bit symbol	—	—	KEYC1	KEYC0	—	—	—	KEYCEN
(0xFFFF_F36C)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets KEYC active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					KEYC interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
KWUPSTD	Bit symbol	—	—	KEYD1	KEYD0	—	—	—	KEYDEN
(0xFFFF_F36D)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets KEYD active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					KEYD interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
KWUPCLR	Bit symbol	—	—	—	—	KEYCLR3	KEYCLR2	KEYCLR1	KEYCLR0
(0xFFFF_F370)	Read/Write	—	—			W			
	After Reset	—	—	—	—	—	—	—	—
	Function					Writing 1010 clears all key sources.			

3.16 INTB, INTC, INTD, INTE

The TMP1942 supports extended interrupts INTB, INTC, INTD and INTE. These four interrupts are internally ORed and the result is input to the CG and INTC. Therefore, they represent a single interrupt source. You can determine which interrupt has actually occurred by checking the corresponding bits of INTFLG. These flags are cleared when read.

- Can be used to terminate STOP/SLEEP mode (wake-up) or as an external interrupt. When used for wake-up, all four interrupts must be set collectively (in the CG block). Whether individual pins are used or not used can be specified separately (INTnST).

- A single interrupt source is available (INTBCDE).

- Rising edge, falling edge, High level, or Low level detection can be selected for individual inputs (INTnST).

- The interrupt source is cleared by reading INTFLG in the interrupt handling routine.

- Which interrupt has occurred can be determined using the INTFLG register.

		7	6	5	4	3	2	1	0
INTBST	Bit symbol	—	—	INTB1	INTB0	—	—	—	INTBEN
(0xFFFF_F380)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets INTB active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					INTB interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
INTCST	Bit symbol	—	—	INTC1	INTC0	—	—	—	INTCEN
(0xFFFF_F381)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets INTC active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					INTC interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
INTDST	Bit symbol	—	—	INTD1	INTD0	—	—	—	INTDEN
(0xFFFF_F382)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets INTD active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					INTD interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
INTEST	Bit symbol	—	—	INTE1	INTE0	—	—	—	INTEEN
(0xFFFF_F383)	Read/Write	—	—	R/W		—	—	—	R/W
	After Reset	—	—	1	0	—	—	—	0
	Function			Sets INTE active condition 00: Low level 01: High level 10: Falling edge 11: Rising edge					INTE interrupt input 0: Disable 1: Enable

		7	6	5	4	3	2	1	0
INTFLG	Bit symbol	—	—	—	—	INTES	INTDS	INTCS	INTBS
(0xFFFF_F384)	Read/Write	—	—	—	—	R			
	After Reset	—	—	—	—	0	0	0	0
	Function					0: Interrupt not generated 1: Interrupt generated	0: Interrupt not generated 1: Interrupt generated	0: Interrupt not generated 1: Interrupt generated	0: Interrupt not generated 1: Interrupt generated

Note: Setting procedures

A) When setting INT inputs first after powering up the device

1) Set the active conditions using the INTnST corresponding to the interrupt inputs to be used.

2) Clear the interrupt request by reading INTFLG.

3) Set the INTnST<INTnEN> corresponding to the interrupt inputs to be used to 1.

4) Set the CG and INTC (refer to Section 3.4, "Interrupts" for details).

B) When modifying the active condition for an INT input during operation

1) Disable INTBCD interrupts in the INTC (IMC1<26:24> = 000).

2) Modify the active condition for the INT input using the corresponding INTnST.

3) Clear the interrupt request by reading INTFLG.

4) Enable INTBCD interrupts in the INTC (set IMC1<26:24> to an appropriate level).

C) When enabling an INT input during operation

1) Disable INTBCD interrupts in the INTC (IMC1<26:24> = 000).

2) Set the active condition using the INTnST corresponding to the interrupt input to be used.

3) Clear the interrupt request by reading INTFLG.

4) Set the INTnST<INTnEN> corresponding to the interrupt input to be used to 1.

5) Enable INTBCD interrupts in the INTC (set IMC1<26:24> to an appropriate level).

3.17 ROM Correction Function

This section describes the ROM correction function the TMP1942 supports.

The TMP1942, however, only supports the registers used for the ROM correction function. When debugging the ROM correction function, therefore, use the ROM correction circuit only to replace the contents of the registers and check subsequent operation by rewriting data in the appropriate flash memory areas. The mask ROM version of the product supports the full ROM correction function.

3.17.1 Features

- Can replace data at four locations: 8 words for each.

- When the PC value or the address generated by the DMAC matches the address stored in an address register (including 5 low-order "don't care" bits), the data from the corresponding ROM correction data register located in RAM will be used in place of the ROM data at that address.

- ROM correction is automatically enabled by setting an address in an address register.

- If it is necessary to correct more than eight words in ROM, for example, when modifying a program, place a instruction for jumping to a RAM address in the data register in RAM so that you can correct data within RAM.

3.17.2 Operation

By setting the physical address of a ROM area (including a projected area) in the address register ADDREGn, you can substitute the data from the data register in RAM corresponding to the ADDREGn for the ROM data at that address. Setting an address in ADDREGn automatically enables the ROM correction function. Upon a reset, the entire ROM correction function is disabled. Therefore, to perform ROM correction in the initial routine after the reset process completes, set an address in an appropriate address register. The ROM correction function is enabled for the address register(s) for which an address is set, so that ROM data will be replaced if the address matches with the value of the PC (when the CPU holds bus control) or if it matches with the source or destination address generated by the DMAC (when the DMAC holds bus control). For example, setting addresses in ADDREG0 and ADDREG3 enables ROM correction for the corresponding areas, so that an address match will constantly be monitored for these address registers and if the addresses match, ROM data will be replaced. In that case, ROM correction will not be performed for ADDREG2 and ADDREG4. Although the address registers have bits 31:5, an address match is detected only for A<18:5> to simplify the circuit. Internally, the ROMCS signal which indicates the ROM area and the match detection from the ROM correction circuit are logically ANDed. ROM correction addresses can only be specified on 8-word boundaries (that is, A0 to A4 are 0). This means data is replaced in 32-byte units. To replace only part of 32 bytes, write the same data for the addresses for which no replacement is required.

The following table shows the relationship between ADDREGn and the RAM areas:

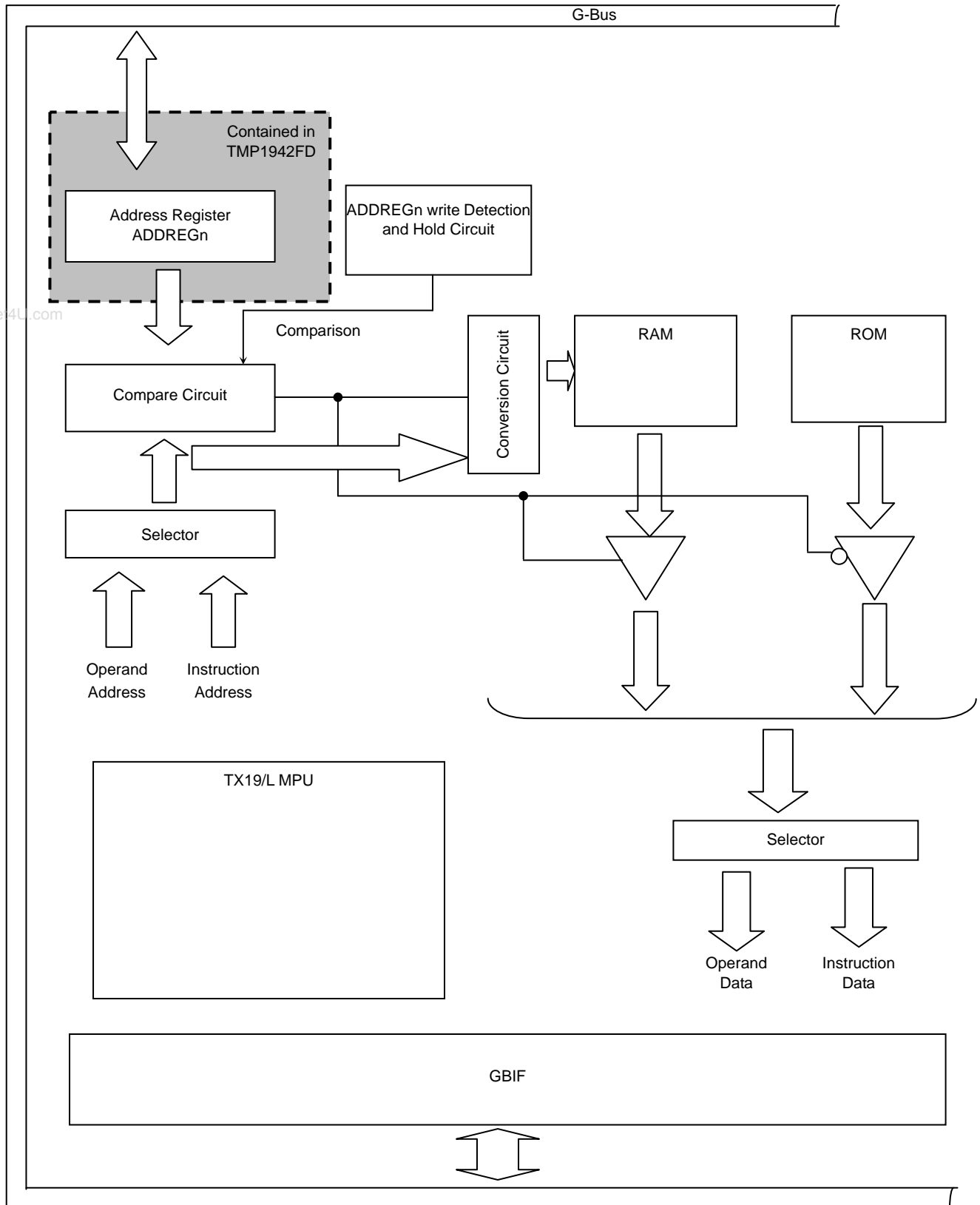| Address Register	Corresponding RAM Area
ADDREG0	0xFFFF_BF80 ~ 0xFFFF_BF9F
ADDREG1	0xFFFF_BFA0 ~ 0xFFFF_BFBF
ADDREG2	0xFFFF_BFC0 ~ 0xFFFF_BFDF
ADDREG3	0xFFFF_BFE0 ~ 0xFFFF_BFFF

Figure 3.17.1 ROM Correction System Diagram

3.17.3 Registers

(1) Address registers

ADDREG0 (0xFFFF_E540)		7	6	5	4	3	2	1	0
	Bit symbol	ADD07	ADD06	ADD05	—	—	—	—	—
	Read/Write	R/W			—				
	After Reset	0	0	0	—	—		—	—
	Function								
		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	0	0	0	0	0	0	0
	Function								
		23	22	21	20	19	18	17	16
	Bit symbol	ADD023	ADD022	ADD021	Add020	ADD019	ADD018	ADD017	ADD016
	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	ADD031	ADD030	ADD029	ADD028	ADD027	ADD026	ADD025	ADD024
	Read/Write	R/W							
	After Reset	0	0	0	0	0	0	0	0
	Function								

ADDREG1 (0xFFFF_E544)		7	6	5	4	3	2	1	0
	Bit symbol	ADD07	ADD06	ADD05	—	—	—	—	—
	Read/Write	R/W			—				
	After Reset	0	0	0	—	—	—	—	—
	Function								
		15	14	13	12	11	10	9	8
	Bit symbol	ADD015	ADD014	ADD13	ADD012	ADD011	ADD010	ADD09	ADD08
	Read/Write	R/W							
	After Reset	0	0	0	0	0	0	0	0
	Function								
		23	22	21	20	19	18	17	16
	Bit symbol	ADD023	ADD022	ADD021	Add020	ADD019	ADD018	ADD017	ADD016
	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	ADD031	ADD030	ADD029	ADD028	ADD027	ADD026	ADD025	ADD024
	Read/Write	R/W							
	After Reset	0	0	0	0	0	0	0	0
	Function								

ADDREG2 (0xFFFF_E548)		7	6	5	4	3	2	1	0
	Bit symbol	ADD07	ADD06	ADD05	—	—	—	—	—
	Read/Write		R/W				—		
	After Reset	0	0	0	—	—	—	—	—
	Function								
		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	0	0	0	0	0	0	0
	Function								
		23	22	21	20	19	18	17	16
	Bit symbol	ADD023	ADD022	ADD021	Add020	ADD019	ADD018	ADD017	ADD016
	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	ADD031	ADD030	ADD029	ADD028	ADD027	ADD026	ADD025	ADD024
	Read/Write				R/W				
	After Reset	0	0	0	0	0	0	0	0
	Function								

ADDREG1 (0xFFFF_E54C)		7	6	5	4	3	2	1	0
	Bit symbol	ADD07	ADD06	ADD05	—	—	—	—	—
	Read/Write		R/W				—		
	After Reset	0	0	0	—	—	—	—	—
	Function								
		15	14	13	12	11	10	9	8
	Bit symbol	ADD015	ADD014	ADD13	ADD012	ADD011	ADD010	ADD09	ADD08
	Read/Write				R/W				
	After Reset	0	0	0	0	0	0	0	0
	Function								
		23	22	21	20	19	18	17	16
	Bit symbol	ADD023	ADD022	ADD021	Add020	ADD019	ADD018	ADD017	ADD016
	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	ADD031	ADD030	ADD029	ADD028	ADD027	ADD026	ADD025	ADD024
	Read/Write				R/W				
	After Reset	0	0	0	0	0	0	0	0
	Function								

Note: DMAC transfer to an address register is not supported. However, DMAC transfer to the substitution data areas allocated in RAM is supported. The ROM correction function is valid for both CPU access and DMAC access.

3.18 Timer for Real-Time Clock

The TMP1942 contains a timer for real-time clock.

Using the 32.768-kHz low-frequency clock, the timer implements a time-keeping function by generating interrupts every 0.0625 s, 0.125 s, 0.25 s or 0.50 s.

The timer for real-time clock can operate in any mode in which low-frequency oscillation is enabled. In addition, a real-time clock interrupt can be used to release the device from a standby mode (other than STOP mode). When using a real-time clock interrupt (INTRTC), set the IMCGB3 register in the CG block appropriately.

3.18.1 Configuration

Figure 3.18.1 shows a block diagram of the timer for real-time clock.
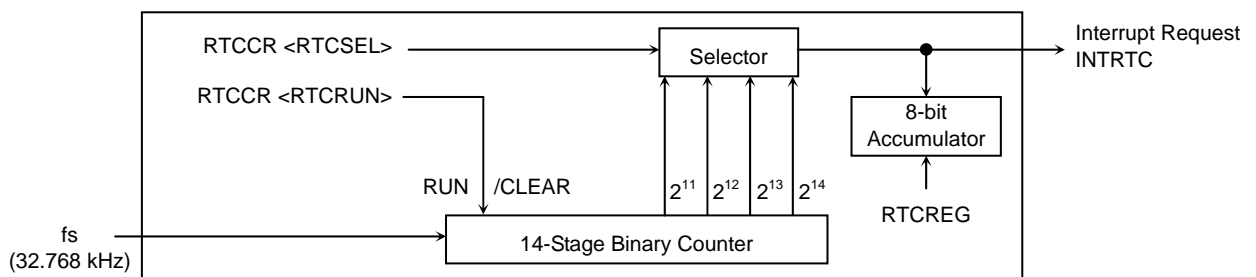
Figure 3.18.1 Block Diagram of Timer for Real-Time Clock

The timer for real-time clock can be controlled using the timer for real-time clock control register (RTCCR). Figure 3.18.2 shows the functions of this register.

		7	6	5	4	3	2	1	0
RTCCR	Bit symbol	—	—	—	—	RTCRCLR	RTCSEL1	RTCSEL0	RTCRUN
(0xFFFF_F0A0)	Read/Write	—	—	—	—	R/W	R/W		R/W
	After Reset	0	—	—	—	0	0	0	0
	Function	Must always be set to 0.				Clears accumulator 0: Clear 1: Don't care	00: 2^{14}/fs 01: 2^{13}/fs 10: 2^{12}/fs 11: 2^{11}/fs		0: Stop and clear 1: Count

Interrupt generation cycle
(fs = 32.768 kHz)

00	0.50 s
01	0.25 s
10	0.125 s
11	0.0625 s

Figure 3.18.2 Timer for Real-Time Clock Control Register

The timer for real-time clock has an accumulator which, when set, holds a cumulative count of the timer for real-time clock interrupts which have been generated. If, for example, an interrupt generation cycle of 0.5 second is selected, this register can hold a cumulative count for up to 127.5 seconds.

Accumulator

	7	6	5	4	3	2	1	0
Bit symbol	RUI7	RUI6	RUI5	RUI4	RUI3	RUI2	RUI1	RUI0
Read/Write	R							
After Reset	0	0	0	0	0	0	0	0
Function	Accumulate count value							

RTCREG (0xFFFF_F0A4)

Figure 3.18.3 Accumulator of Timer for Real-Time Clock

Each time an INTRTC interrupt is generated, the accumulator is incremented after one cycle of the fs clock. The accumulator must be read in SLOW mode.

An instruction for clearing the accumulator is not accepted within one fs clock cycle after the generation of an INTRTC interrupt. To clear the accumulator, execute two clear accumulator instructions in SLOW mode.
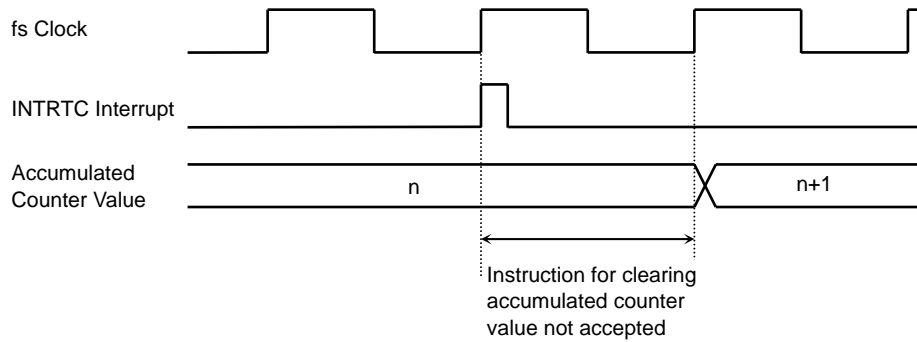
Example 1: Clearing the accumulator

```
             7 6 5 4 3 2 1 0
SYSCR1   ←   x x 1 — — x — —    Select SLOW mode.
RTCCR    ←   0 x x x 0 — — 1  ⎤
RTCCR    ←   0 x x x 0 — — 1  ⎦ Clear accumulator twice.
SYSCR1   ←   x x 0 — — x — —    Restore NORMAL mode.
```

Example 2: Setting up a timer for real-time clock interrupt

Initial settings

```
             7 6 5 4 3 2 1 0
IMCGB3   ←   0 0 1 1 0 0 0 1
IMCEHL   ←   0 0 0 1 0 X X X    Set interrupt level.
EICRCG   ←   0 0 0 0 0 1 1 1    Clear interrupt request for CG block.
INTCLR   ←   0 0 1 1 1 0 1 0    Clear interrupt request for INTC block.
RTCCR    ←   0 0 0 0 1 X X 1    Start timer counting.
```

INTRTC interrupt

```
             7 6 5 4 3 2 1 0
EICRCG   ←   0 0 0 0 0 1 1 1    Clear interrupt request for CG block.
INTCLR   ←   0 0 1 1 1 0 1 0    Clear interrupt request for INTC block.
Processing
End of interrupt
```

Note 1: X = Don't care

Note 2: To disable interrupts, set IMCEHL before setting IMCGB3.

3.19 Watchdog Timer (Runaway Detection Timer)

The TMP1942 contains a watchdog timer for the purpose of runaway detection.

When the CPU starts operating erratically (runaway) due to noise or other causes, the watchdog timer (WDT) detects this runaway condition to re-establish a normal condition. Upon detecting a runaway condition, the watchdog timer notifies the CPU by generating a non-maskable interrupt.

Also, output from the watchdog timer can be transmitted to a reset input (internal to the chip) in order to forcibly reset the device.

3.19.1 Configuration

Figure 3.19.1 shows a block diagram of the watchdog timer.
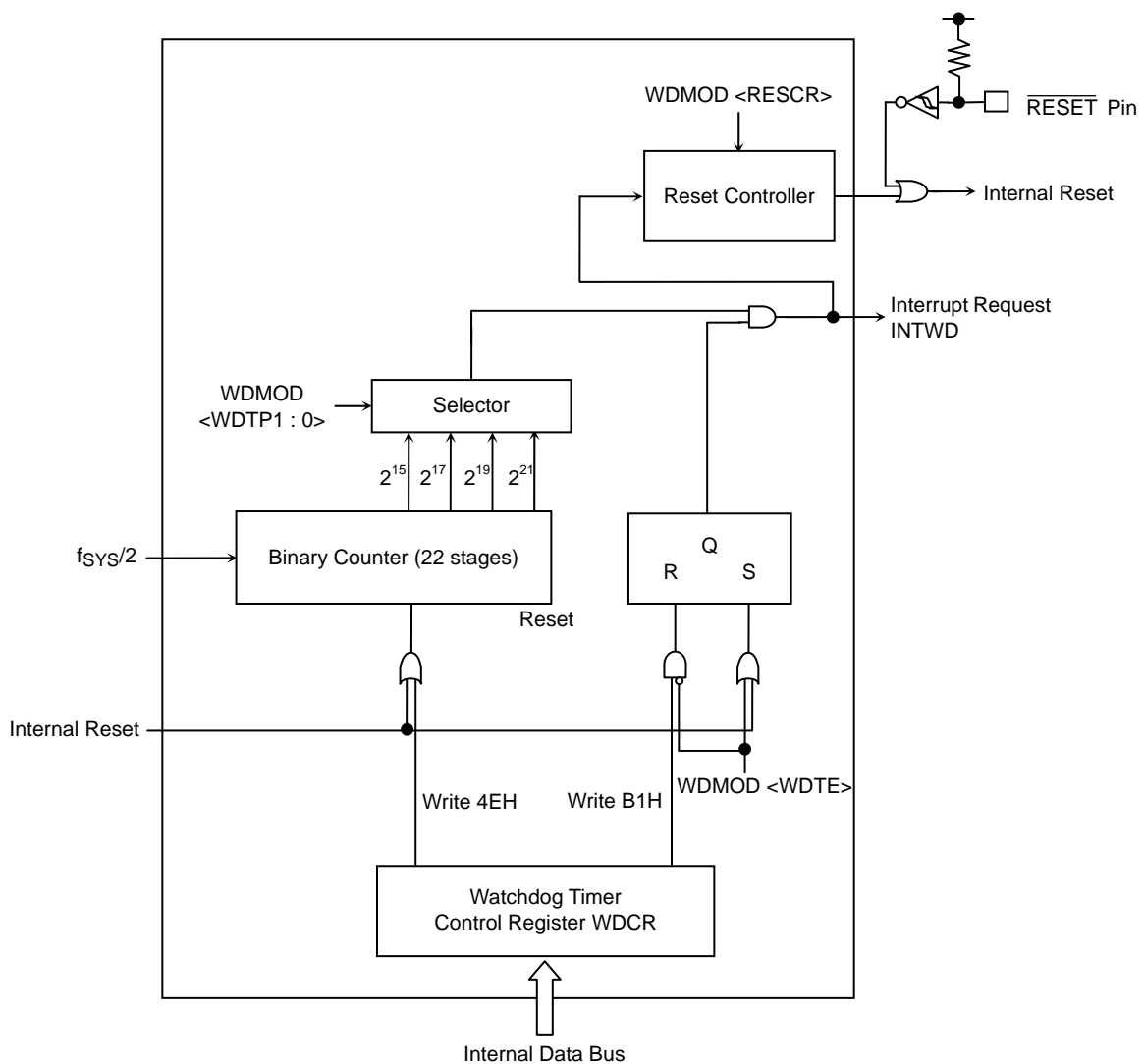
Figure 3.19.1 Watchdog Timer Block Diagram

The watchdog timer consists of a 22-stage binary counter clocked by the system clock fsys/2. Four binary counter outputs are available: 2^{15}, 2^{17}, 2^{19} and 2^{21}. Any one of these counter outputs can be selected using WDMOD<WDTP1:WDTP0>, so that when the selected counter output overflows, a watchdog timer interrupt will be generated, as shown in Figure 3.19.2.
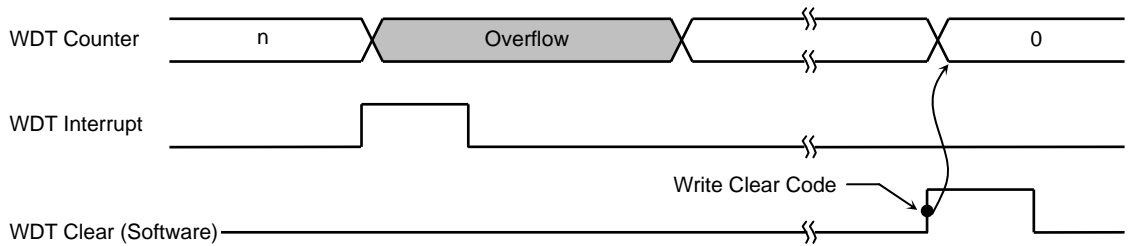

Figure 3.19.2 Normal Mode

Also, it is possible to reset the chip itself when the counter output overflows. In this case, the chip is reset for a period of 22 to 29 states as shown in Figure 3.19.3. When the chip is reset in this way, the watchdog timer is clocked by a clock of fsys, instead of by the afore-mentioned input clock fsys/2. The fsys clock is derived by dividing the high-speed oscillator's clock fC by a clock gear of 8.

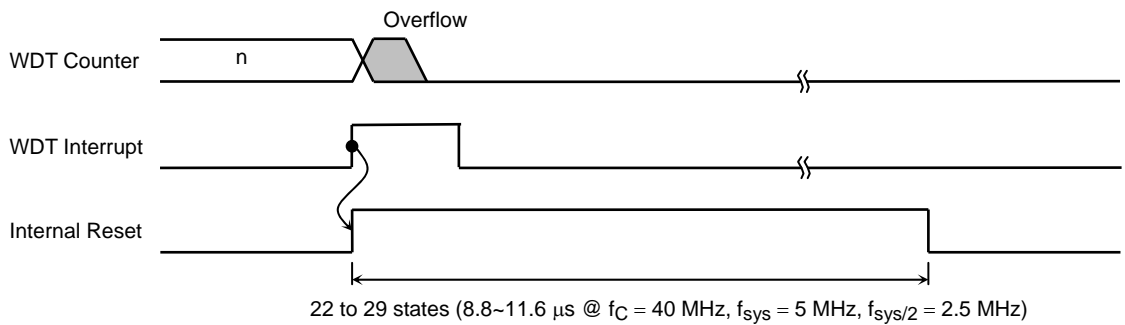22 to 29 states (8.8~11.6 μs @ f_C = 40 MHz, f_{sys} = 5 MHz, $f_{sys/2}$ = 2.5 MHz)

Figure 3.19.3 Reset Mode

Note: Even when the chip is reset by the watchdog timer, the $\overline{\text{PLLOFF}}$ pin is sampled. Thus, the $\overline{\text{PLLOFF}}$ pin must be held at a constant logic level, either High or Low.

3.19.2 Control Registers

The watchdog timer (WDT) can be controlled using two control registers (WDMOD and WDCR).

(1) Watchdog timer mode register (WDMOD)

a. Setting the watchdog timer detection time (WDTP1:WDTP0)

These two bits are used to set the watchdog timer interrupt detection time necessary for detecting a runaway condition. Upon a reset, WDMOD<WDTP1:WDTP0> are initialized to 00. Figure 3.19.4 shows watchdog timer detection times.

b. Enabling/disabling the watchdog timer (WDTE)

Upon a reset, WDMOD<WDTE> is initialized to 1, enabling the watchdog timer.

To disable the watchdog timer, set this bit to 0 and, at the same time, write the disable code (B1H) to the WDCR register. This dual setting ensures that the watchdog timer cannot easily be disabled by a runaway condition.

To re-enable the watchdog timer after it has been disabled, simply set the WDMOD<WDTE> bit to 1.

c. Connecting the watchdog timer output to reset (RESCR)

This bit is used to specify whether or not the CPU itself will be reset upon the detection of a runaway condition. Upon a reset, WDMOD<RESCR> is initialized to 0. When WDMOD<RESCR> = 0, the CPU will not be reset by the watchdog timer output.

(2) Watchdog timer control register (WDCR)

This register controls the watchdog timer by disabling the watchdog timer function and clearing the binary counter.

• Disabling the watchdog timer

The watchdog timer can be disabled by setting WDMOD<WDTE> to 0 and then writing the disable code (B1H) to the WDCR register.

```
WDMOD    ← 0 – – – – – – –    Clear WDTE to 0.
WDCR     ← 1 0 1 1 0 0 0 1    Write disable code (B1H).
```

• Enabling the watchdog timer

Set WDMOD<WDTE> to 1.

• Clearing the binary counter

Writing the clear code (4EH) to the WDCR register clears the binary counter and restarts counting.

```
WDCR     ← 0 1 0 0 1 1 1 0    Write clear code (4EH).
```

Note: Writing the disable code (B1H) causes the binary counter to be cleared.

	7	6	5	4	3	2	1	0
Bit symbol	WDTE	WDTP1	WDTP0	—	—	I2WDT	RESCR	—
Read/Write	R/W	R/W		—	—	R/W		R/W
After Reset	1	0	0	—	—	0	0	0
Function	Controls WDT 1: Enable	Selects WDT detection time 00: $2^{16}/f_{SYS}$ 01: $2^{18}/f_{SYS}$ 10: $2^{20}/f_{SYS}$ 11: $2^{22}/f_{SYS}$				IDLE 0: Idle 1: Operate	1: Transmits WDT output to chip's internal reset pin	Must always be set to 0.

WDMOD (0xFFFF_F090)

Controls watchdog timer output

0	—
1	Transmit WDT output to reset

Watchdog timer detection time   @ fc = 32 MHz, fs = 32.768 kHz

SYSCR1 System Clock Selection <SYSCK>	SYSCR1 Clock Gear Value <GEAR1:0>	Watchdog Timer Detection Time			
		WDMOD<WDTP1, 0>			
		00	01	10	11
1 (fs)	xxx	2.0 s	8.0 s	32.0 s	128.0 s
0 (fc)	00 (fc)	2.048 ms	8.192 ms	32.768 ms	131.072 ms
	01 (fc/2)	4.096 ms	16.384 ms	65.536 ms	262.144 ms
	10 (fc/4)	8.192 ms	32.768 ms	131.072 ms	524.288 ms
	11 (fc/8)	16.384 ms	65.536 ms	262.144 ms	1048.576 ms

Disables/enables watchdog timer

0	Disable
1	Enable

Figure 3.19.4 Watchdog Timer Mode Register

	7	6	5	4	3	2	1	0
Bit symbol	—							
Read/Write	W							
After Reset	—							
Function	B1H: WDT disable code 4EH: WDT clear code							

WDCR (0xFFFF_F091)

Disables or clears WDT

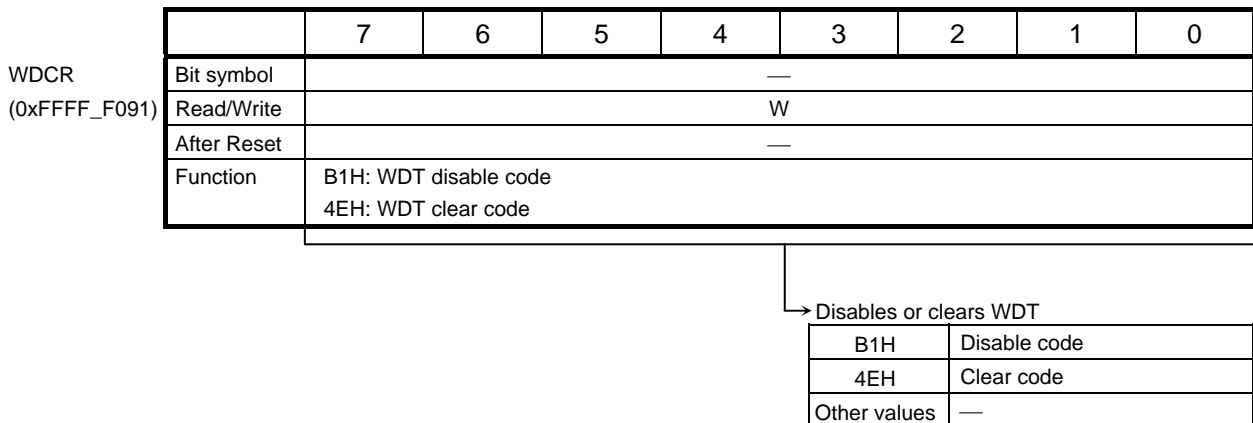| B1H	Disable code
4EH	Clear code
Other values	—

Figure 3.19.5 Watchdog Timer Control Register

3.19.3 Functional Description

After the detection time which has been set in WDMOD<WDT1:WDT0>, the watchdog timer generates an interrupt (INTWDT). The binary counter for the watchdog timer must be cleared to zero by software before an INTWDT interrupt can occur. If runaway occurs in the CPU due to noise or other causes, and prevents the CPU from executing the instruction to clear the binary counter, the binary counter will overflow and generate an INTWDT interrupt. This interrupt notifies the CPU that it has gone out of control, so that the CPU can restore itself to a normal condition by executing a program to correct the runaway condition. Also, output from the watchdog timer can be transmitted to the reset pin or other pins of peripheral devices to address the CPU runway condition.

The watchdog timer will start operating as soon as the device has completed its reset sequence.

In SLEEP and STOP modes, the watchdog timer is reset and remains idle. If the bus is free ($\overline{\text{BUSAK}}$ = Low), it will continue to count. In IDLE mode, the WDMOD<I2WDT> setting determines whether the watchdog timer is on or off. Before placing the device into IDLE mode, set WDMOD<I2WDT> as required.

Examples:

1) Clearing the binary counter

```
            7 6 5 4 3 2 1 0
WDCR    ← 0 1 0 0 1 1 1 0       Write clear code (4EH).
```

2) Setting the watchdog timer detection time to $2^{18}/f_{sys}$

```
            7 6 5 4 3 2 1 0
WDMOD   ← 1 0 1 – – – – –
```

3) Disabling the watchdog timer

```
            7 6 5 4 3 2 1 0
WDMOD   ← 0 – – – – – – –       Clear WDTE to 0.
WDCR    ← 1 0 1 1 0 0 0 1       Write disable code (B1H).
```

4 Electrical Characteristics

4.1 Absolute Maximum Ratings

The letter x in equations presented in this chapter represents the cycle period of the fsys clock selected through the programming of the SYSCR1.SYSCK bit. The fsys clock may be derived from either the high-speed or low-speed crystal oscillator. The programming of the clock gear function also affects the fsys frequency. All relevant values in this chapter are calculated with the high-speed (fc) system clock (SYSCR1.SYSCK=0) and a clock gear factor of 1/fc (SYSCR1.GEAR[1:0]=00).

Parameter		Symbol	Rating	Unit
Supply voltage		V_{CC3}	$-0.5\sim4.0$	V
		V_{CC5}	$-0.5\sim6.0$	V
Input voltage		V_{IN3}	$-0.5\sim V_{CC3}+0.5$	V
		V_{IN5}(Note)	$-0.5\sim V_{CC5}+0.5$	V
Low-level output current		V_{AIN}	$-0.5\sim AVCC+0.5$	V
Analog input		VAREFH	$-0.5\sim AVCC+0.5$	V
		DAREFH	$-0.5\sim DAVCC+0.5$	V
Low-level output current	Per pin	I_{OL}	5	mA
	Total	ΣI_{OL}	80	
High-level output current	Per pin	I_{OH}	-5	
	Total	ΣI_{OH}	-80	
Power dissipation (Ta = 85°C)		PD	600	mW
Soldering temperature (10 s)		T_{SOLDER}	260	°C
Storage temperature		T_{STG}	$-65\sim150$	°C
Operating temperature		T_{OPR}	$-40\sim85$	°C

$V_{CC3}=DV_{CC3}=AV_{CC}=DAV_{CC}=CV_{CC}$, $V_{CC5}=DV_{CC51}=DV_{CC52}$

$V_{SS}=DV_{SS}=AV_{SS}=DAV_{SS}=CV_{SS}$

Note : PortC , PortF

Note: 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 maximum rating value is exceeded with respect to current, voltage, power dissipation, 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.

4.2 DC Electrical Characteristics (1/4)

Ta=-40~85°C

Parameter	Symbol	Condition		Min	Typ (Note 1)	Max	Unit
Supply voltage DAVCC=AVCC =CVCC=DVCC3 DAVSS=AVSS =CVSS= 0V	DVCC3	PLLON (INTLV="H")	fosc = 5~8MHz fsys = 2.5~32MHz fs = 30~34kHz	3.0		3.6	V
			fosc = 5~7MHz fsys = .5~28MHz fs = 30~34kHz	2.7			
		PLLOFF (Crystal)	(INTLV="H") fosc = 10~20MHz fsys = 1~20MHz fs = 30~34kHz (INTLV="L") fosc = 10~16MHz fsys = 1~16MHz fs = 30~34kHz	2.7			
		PLLOFF (External clock)	fosc = 20~32MHz fsys = 1.25~16MHz fs = 30~34kHz <DFOSC> = "0"	3.0			
			fosc = 10~16MHz fsys = 1~16MHz fs = 30~34kHz	2.7			
	DVCC5* (Note 2)	fsys = 1~32MHz fs = 30~34kHz		4.5		5.25	
Low-level input voltage	P00~P17(AD0~15)	V_{IL}	DVCC3≥2.7V DVCC5≥4.5V	− 0.3		0.6	V
	P20~PB7 ,PD0~PE7	V_{IL1}				0.3DVCC3	
	PLLOFF ,BW0 ,BW1, RSTPUP , RESET , NMI	V_{IL2}				0.2 DVCC3	
	PC0~PC7 ,PF0~PF6	V_{IL3}				0.3 DVCC5	
	X1	V_{IL4}				0.2 DVCC3	
High-level input voltage	P00~P17(AD0~15)	V_{IH}		2.0		DVCC3 + 0.3	
	P20~PB7 ,PD0~PE7	V_{IH1}		0.7DVCC3			
	PLLOFF ,BW0 ,BW1 , RSTPUP , RESET , NMI	V_{IH2}		0.8DVCC3			
	PC0~PC7 ,PF0~PF6	V_{IH3}		0.7DVCC5		DVCC5 + 0.3	
	X1	V_{IH4}		0.8DVCC3		DVCC3 + 0.3	
Low-level output voltage		V_{OL}	I_{OL} = 1.6mA DVCC3≥2.7V DVCC5≥4.5V			0.45	V
High-level output voltage		V_{OH1}	I_{OH} = − 400 µA DVCC3≥2.7V	2.4			V
		V_{OH2} (Note3)	DVCC5≥2.7V	2.4			
			DVCC5≥4.5V	4.2			

Note 1: V_{CC3} = 3.3 V, V_{CC5} = 5.0 V, Ta = 25°C, unless otherwise noted.
Note 2: DVCC5*:DVCC51,DVCC52
    DVCC5*can be used also as 2.7V≤DVCC5*≤3.6V.
Note 3: PortC,PortF

4.3 DC Electrical Characteristics (2/4)

Ta=-40~85°C

Parameter	Symbol	Condition	Min	Typ (Note)	Max	Unit
Input leakage current	I_{LI}	$0.0 \leq V_{IN} \leq D\ VCCn$ (n=3,5)		0.02	± 5	μA
Output leakage current	I_{LO}	$0.2 \leq V_{IN} \leq DVCCn - 0.2$ (n=3,5)		0.05	± 10	
Power-down voltage (STOP mode, RAM backup)	V_{STOP1}	$V_{IL2} = 0.2DVCC3$ $V_{IH2} = 0.8DVCC3$	2.2		3.6	V
	V_{STOP2}	$V_{IL2} = 0.2DVCC5$ $V_{IH2} = 0.8DVCC5$	V_{STOP1}		5.25	
Pull-up resistor at Reset	RRST	$V_{CC} = 3.3V \pm 0.3V$	100		550	kΩ
Programmable pull-up resistor P32~P37,P40~P43 KEY0~KEYD	PKH1	$DVCC3 = 3.3V \pm 0.3V$	30	45	100	kΩ
	PKH2	$DVCC5 = 4.5V~5.25V$	30	55	100	
Pin capacitance (except power/ground pins)	C_{IO}	fc = 1MHz			10	pF

Note: V_{CC3} = 3.3 V, V_{CC5} = 5.0 V, Ta = 25°C, unless otherwise noted.

4.4 DC Electrical Characteristics (3/4)

(1) TMP1942CYUE

DVCC3=3.3V±0.3V , DVCC51= DVCC52 = 3.3V±0.3V ,Ta=-40~85°C

Parameter	Symbol	Condition	Min	Typ (Note1)	Max	Unit
NORMAL (Note2) Gear=1/1	I_{CC}	f_{SYS} = 32MHz (f_{OSC} = 8MHz , PLLON) INTLV="H"		70	90	mA
IDLE(Doze)				22	34	
IDLE(Halt)				20	30	
NORMAL (Note2) Gear=1/1		f_{SYS} = 16MHz (f_{OSC} = 16MHz, PLLOFF) INTLV="L"		44	58	mA
IDLE(Doze)				11	15	
IDLE(Halt)				10	13	
SLOW		fs = 32.768kHz		50	120	μA
SLEEP		fs = 32.768kHz		8	60	μA
STOP		DVCC3 = 2.7~3.6V DVCC5 = 2.7~3.6V		1	50	μA

Note1: V_{CC3} = 3.3 V, V_{CC5} = 5.0 V, Ta = 25°C, unless otherwise noted.

Note2: The measurement conditions of Icc NORMAL

      CPU:Dhrystone(Ver.2.1)(There is external memory access)

      8bit Timer:500kHz/50%Output×3ch,50kHz/50%Output×3ch

      16 bit Timer:500kHz/50%Output×3ch,50kHz/50%Output×3ch,2ms Interval Timer×6ch,2-phase

           pulse input counter×2ch

      SIO:UART(11.5kbps)×1ch,I/O interface mode (50kHz)×4ch

      ADC:Fixed channel, Continuous conversion

      DAC:Output(0x200)×3ch

Note3: The measurement conditions of Icc SLOW,Icc SLEEP

   CPU:Equivalent to NORMAL mode

      Timer for Real-Time clock, 2-pulse input counter,

      Dynamic pull-up mode (16ms cycle, 250us sampling)

Note4: The supply current flowing through the DVCC3, DVCC5, CVCC, AVCC and DAVCC pins is include in the digital
      supply current parameter (ICC).

Note5: The supply current flowing through the A/D and D/A converter is include in the reference current parameter
      (ICC Normal).

(2) TMP1942CZUE/XB

DVCC3=3.3V±0.3V , DVCC51= DVCC52 = 3.3V±0.3V ,Ta=-40~85°C

Parameter	Symbol	Condition	Min.	Typ. (Note1)	Max.	Unit
NORMAL (Note2) Gear=1/1	I_{CC}	$f_{sys} = 32MHz$ ($f_{osc} = 8MHz$, PLLON) INTLV="H"		70	90	mA
IDLE(Doze)				22	34	
IDLE(Halt)				20	30	
NORMAL (Note2) Gear=1/1		$f_{sys} = 16MHz$ ($f_{osc} = 16MHz$, PLLOFF) INTLV="L"		40	58	mA
IDLE(Doze)				11	15	
IDLE(Halt)				10	13	
SLOW		$fs = 32.768kHz$		50	120	μA
SLEEP		$fs = 32.768kHz$		8	60	μA
STOP		$DVCC3 = 2.7~3.6V$ $DVCC5 = 2.7~3.6V$		1	50	μA

Note1: V_{CC3} = 3.3 V, V_{CC5} = 5.0 V, Ta = 25°C, unless otherwise noted.
Note2: The measurement conditions of Icc NORMAL
      CPU:Dhrystone(Ver.2.1)(There is external memory access)
      8bit Timer:500kHz/50%Output×3ch,50kHz/50%Output×3ch
      16 bit Timer:500kHz/50%Output×3ch,50kHz/50%Output×3ch,2ms Interval Timer×6ch,2-phase
           pulse input counter×2ch
      SIO:UART(11.5kbps)×1ch,I/O interface mode (50kHz)×4ch
      ADC:Fixed channel, Continuous conversion
      DAC:Output(0x200)×3ch
Note3: The measurement conditions of Icc SLOW,Icc SLEEP
  CPU:Equivalent to NORMAL mode
      Timer for Real-Time clock, 2-pulse input counter,
      Dynamic pull-up mode (16ms cycle, 250us sampling)
Note4: The supply current flowing through the DVCC3, DVCC5, CVCC, AVCC and DAVCC pins is include in the digital
      supply current parameter (ICC).
Note5: The supply current flowing through the A/D and D/A converter is include in the reference current parameter
      (ICC Normal).

4.5 DC Electrical Characteristics   (4/4)

(1) TMP1942CYUE

DVCC3=3.3V±0.3V , DVCC51= DVCC52 = 5.0V±0.25V ,Ta=-40~85°C

Parameter	Symbol	Condition	Min	Typ (Note1)	Max	Unit
NORMAL Gear=1/1	I_{CC}	$f_{sys} = 32MHz$ ($f_{osc} = 8MHz$, PLLON) INTLV="H"		70	90	mA
		$f_{sys} = 16MHz$ ($f_{osc} = 16MHz$, PLLOFF) INTLV="L"		44	58	mA
SLOW		fs = 32.768kHz		50	120	μA
SLEEP		fs = 32.768kHz		8	60	
STOP		DVCC3 = 2.7~3.6V DVCC5 = 4.75~5.25V		1	50	μA

Note1: Note1: V_{CC3} = 3.3 V, V_{CC5} = 5.0 V, Ta = 25°C, unless otherwise noted.

Note2: The measurement conditions of Icc NORMAL: Please refer to 4.4 DC Electrical Characteristics (3/4) Note2 and Note3.

Note3: An electroc current to use in CVCC, AVCC and DAVCC is inncluded in DVCC3(ICC).

Note4: An electroc current to use in DVCC51 and DVCC52 is inncluded in DVCC5(ICC).

(2) TMP1942CZUE/XB

DVCC3=3.3V±0.3V , DVCC51= DVCC52 = 5.0V±0.25V ,Ta=-40~85°C

Parameter	Symbol	Condition	Min	Typ (Note1)	Max	Unit
NORMAL Gear=1/1	I_{CC}	$f_{sys} = 32MHz$ ($f_{osc} = 8MHz$, PLLON) INTLV="H"		70	90	mA
		$f_{sys} = 16MHz$ ($f_{osc} = 16MHz$, PLLOFF) INTLV="L"		44	58	mA
SLOW		fs = 32.768kHz		50	120	μA
SLEEP		fs = 32.768kHz		8	60	
STOP		DVCC3 = 2.7~3.6V DVCC5 = 4.75~5.25V		1	50	μA

Note1: Note1: V_{CC3} = 3.3 V, V_{CC5} = 5.0 V, Ta = 25°C, unless otherwise noted.

Note2: The measurement conditions of Icc NORMAL: Please refer to 4.4 DC Electrical Characteristics (3/4) Note2 and Note3.

Note3: An electroc current to use in CVCC, AVCC and DAVCC is inncluded in DVCC3(ICC).

Note4: An electroc current to use in DVCC51 and DVCC52 is inncluded in DVCC5(ICC).

4.6 10bit A/D Converter Electrical Characteristics

(1) TMP1942CYUE

Ta=-40~85°C

Parameter		Symbol	Condition	Min	Typ	Max	Unit
Reference (+)		VREFH		2.7		3.6	V
				AVCC− 0.3	AVCC	AVCC+ 0.3	
Reference (−)		VREFL		AVSS	AVSS	AVSS + 0.2	V
Analog input		VAIN		VREFL		VREFH	V
Reference current	Conversion	IREF	DVCC3 = AVCC = VREFH = 3.3V ± 0.3V DVSS = AVSS = VREFL		2	2.5	mA
	No conversion		DVCC3 = AVCC = VREFH = 2.7~3.6V DVSS = AVSS = VREFL		± 0.02	± 5	μA
Analog input capacitance		—				20	pF
Analog input impedance		—				5	kΩ
INL error		—	DVCC3 = AVCC = VREFH = 3.3V ± 0.3V DVSS = AVSS = VREFL AIN resistance<5Ω AIN load capacitance<50pF AVCC load capacitance≥10μF VREFH load capacitance≥10μF Conversion time≥2μs (Scan mode) Conversion time≥4μs (Single mode)			±2. 5	LSB
DNL error		—				± 2	LSB
Offset error		—				± 4	LSB
Gain error		—				± 4	LSB

Note1: 1LSB = (VREFH − VREFL) / 1024[V]

Note2: The A/D converter must be stopped when operating the TMP1942 with the low-speed clock (fs).

Note3: The supply current flowing through the AVCC pin is included in the digital supply current parameter (ICC).

(2) TMP1942CZUE/XB

Ta=-40~85°C

Parameter		Symbol	Condition	Min	Typ	Max	Unit
Reference (+)		VREFH		2.7		3.6	V
				AVCC− 0.3	AVCC	AVCC+ 0.3	
Reference(−)		VREFL		AVSS	AVSS	AVSS + 0.2	V
Analog input		VAIN		VREFL		VREFH	V
Reference current	Conversion	IREF	DVCC3 = AVCC = VREFH = 3.3V ± 0.3V DVSS = AVSS = VREFL		2.2	2.85	mA
	No conversion		DVCC3 = AVCC = VREFH = 2.7~3.6V DVSS = AVSS = VREFL		± 0.02	± 5	μA
Analog input capacitance		—				20	pF
Analog input impedance		—				5	kΩ
INL error		—	DVCC3 = AVCC = VREFH = 3.3V ± 0.3V DVSS = AVSS = VREFL AIN resistance<5Ω AIN load capacitance<50pF AVCC load capacitance≥10μF VREFH load capacitance≥10μF Conversion time≥2μs (Scan mode) Conversion time≥4μs (Single mode)			±2. 5	LSB
DNL error		—				± 2	LSB
Offset error		—				± 4	LSB
Gain error		—				± 4	LSB

Note1: 1LSB = (VREFH − VREFL) / 1024[V]

Note2: The A/D converter must be stopped when operating the TMP1942 with the low-speed clock (fs).

Note3: The supply current flowing through the AVCC pin is included in the digital supply current parameter (ICC).

4.7 10bit D/A Converter Electrical Characteristics

Ta=-40~85°C

Parameter		Symbol	Condition	Min	Typ	Max	Unit
Reference (+)		DAREFH		2.7		3.6	V
				DAVCC − 0.3	DAVCC	DAVCC+0.3	V
Reference current	<DAREFON> = 1	IDREF	DVCC3 = DAVCC = DAREFH = 3.3V ± 0.3V DVSS = DAVSS		0.6	1	mA
	<DAREFON> = 0		DVCC3 = DAVCC = DAREFH = 2.7~3.6V DVSS = DAVSS		± 0.02	± 5	μA
Output current		IDAOUT	DVCC3 = DAVCC = DAREFH = 2.7~3.6V DVSS = DAVSS	±1	±1.5		mA
Outpu voltage range		DAOUT	DVCC3 = DAVCC = DAREFH = 2.7~3.6V DVSS = DAVSS	DAVSS+0.3		DAVCC-0.3	V
Gain error		─	DVCC3 = DAVCC = DAREFH = 3.3V ± 0.3V DVSS = DAVSS		± 1	± 3	LSB

Note1: 1LSB = (DAREFH − DAVSS) / 1024[V]

Note2: The D/A converter must be stopped when operating the TMP1942 with the low-speed clock (fs).

Note3: The supply current flowing through the DAVCC pin is included in the digital supply current parameter (ICC).

Note4: IDREF electoric current value is an electoric current value when I moved three D/A converter.

4.8 AC Electrical Characteristics

(1) $V_{CC} = 3.0 \sim 3.6$ V, Ta = $0 \sim 70°C$, ALE = 0.5 clock cycle

(recommended when t_{SYS} is 50 ns or longer)

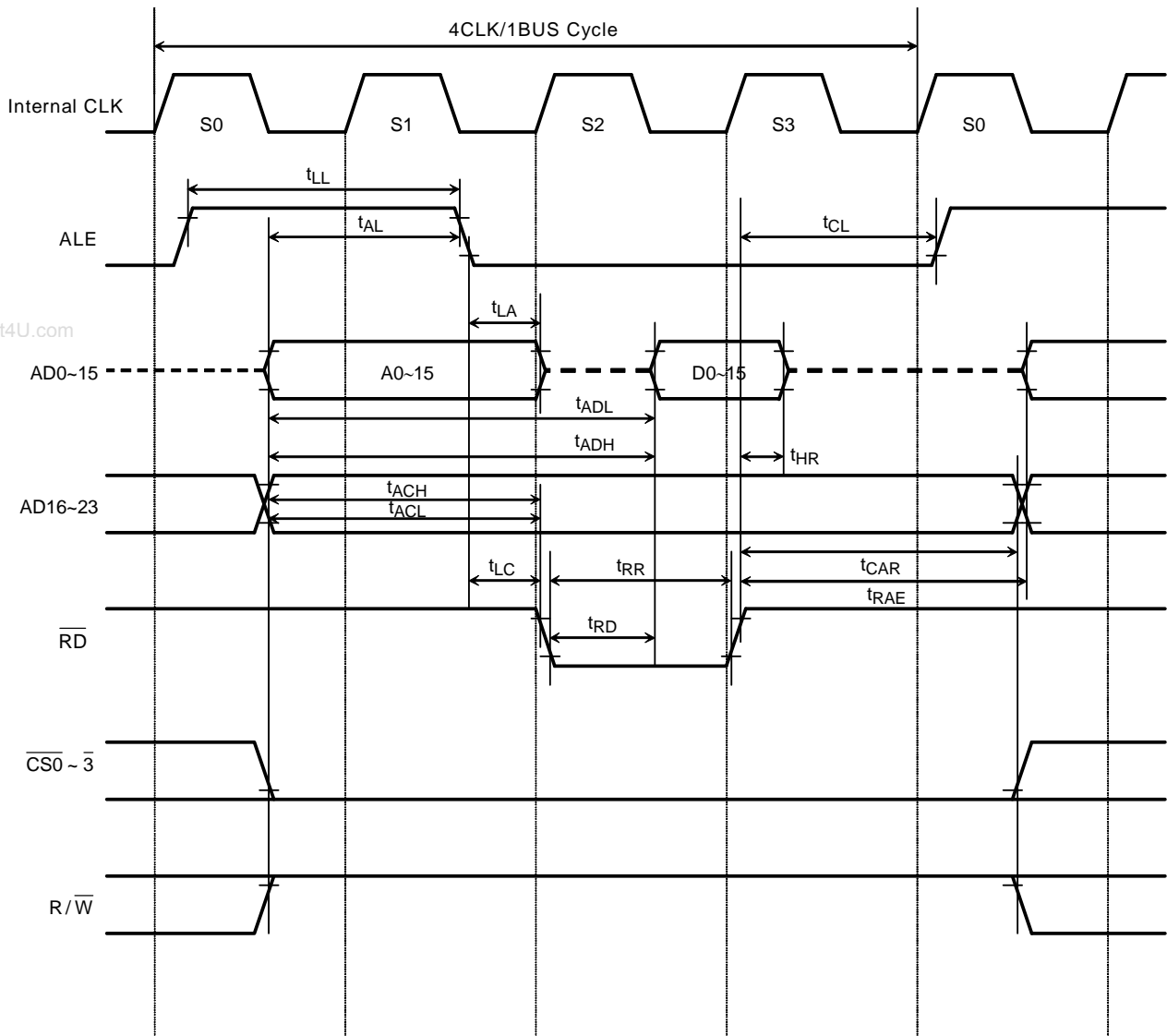| No.	Parameter	Symbol	Equation		20 MHz(fsys)*		Unit
			Min	Max	Min	Max	
1	System clock period (x)	t_{SYS}	31.25	33333	50		ns
2	A0–A15 valid to ALE low	t_{AL}	0.4x – 12		8		ns
3	A0–A15 hold after ALE low	t_{LA}	0.4x – 8		12		ns
4	ALE pulse width high	t_{LL}	0.4x – 6		14		ns
5	ALE low to $\overline{RD}$ or $\overline{WR}$ asserted	t_{LC}	0.4x –8		12		ns
6	$\overline{RD}$ or $\overline{WR}$ negated to ALE high	t_{CL}	x – 15		35		ns
7	A0–A15 valid to $\overline{RD}$ or $\overline{WR}$ asserted	t_{ACL}	x – 20		30		ns
8	A0–A23 valid to $\overline{RD}$ or $\overline{WR}$ asserted	t_{ACH}	x – 20		30		ns
9	A0–A23 hold after $\overline{RD}$ or $\overline{WR}$ negated	t_{CAR}	x – 15		35		ns
10	A0–A15 valid to D0–D15 data in	t_{ADL}		x (2 + W) – 42		58	ns
11	A0–A23 hold after $\overline{RD}$ or $\overline{WR}$ negated	t_{ADH}		x (2 + W) – 42		58	ns
12	$\overline{RD}$ asserted to D0–D15 data in	t_{RD}		x (1 + W) – 28		22	ns
13	$\overline{RD}$ width low	t_{RR}	x (1 + W) – 10		40		ns
14	D0–D15 hold after $\overline{RD}$ negated	t_{HR}	0		0		ns
15	$\overline{RD}$ negated to next A0–A15 output	t_{RAE}	x – 15		35		ns
16	$\overline{WR}$ width low	t_{WW}	x (1 + W) – 10		40		ns
17	D0–D15 valid to $\overline{WR}$ negated	t_{DW}	x (1 + W) – 18		32		ns
18	D0–D15 hold after $\overline{WR}$ negated	t_{WD}	x – 15		35		ns
19	A0–A23 valid to $\overline{WAIT}$ input	t_{AWH}		1.5x – 30		45	ns
20	A0–A15 valid to $\overline{WAIT}$ input	t_{AWL}		1.5x – 30		45	ns
21	$\overline{WAIT}$ hold after $\overline{RD}$ or $\overline{WR}$ asserted	t_{CW}	(0.5 + N – 1) x +2	(0.5 + N) x – 17	27	58	ns

*WAIT = 0

AC measurement conditions:
- Output levels: High = 2.4 V, Low = 0.45 V, CL = 30 pF
- Input levels: High = 2 V, Low = 0.6 V

W: Number of wait-state cycles inserted (0 to 7 for programmed wait insertion)
N : Value of N for (1 + N) wait insertion

(2)$V_{CC} = 3.0 \sim 3.6$ V, Ta $=\ 0 \sim 70°C$, ALE $= 1.5$ clock cycles

No.	Parameter	Symbol	Equation		32 MHz(fsys)*		Unit
			Min	Max	Min	Max	
1	System clock period (x)	t_{SYS}	31.25	33333			ns
2	A0–A15 valid to ALE low	t_{AL}	1.4x – 12		31		ns
3	A0–A15 hold after ALE low	t_{LA}	0.4x – 8		4		ns
4	ALE pulse width high	t_{LL}	1.4x – 6		37		ns
5	ALE low to $\overline{RD}$ or $\overline{WR}$ asserted	t_{LC}	0.4x – 8		4		ns
6	$\overline{RD}$ or $\overline{WR}$ negated to ALE high	t_{CL}	x – 15		16		ns
7	A0–A15 valid to $\overline{RD}$ or $\overline{WR}$ asserted	t_{ACL}	2x – 20		42		ns
8	A0–A23 valid to $\overline{RD}$ or $\overline{WR}$ asserted	t_{ACH}	2x – 20		42		ns
9	A0–A23 hold after $\overline{RD}$ or $\overline{WR}$ negated	t_{CA}	x – 15		16		ns
10	A0–A15 valid to D0–D15 data in	t_{ADL}		x (3 + W) – 42		51	ns
11	A0–A23 valid to D0–D15 data in	t_{ADH}		x (3 + W) – 42		51	ns
12	$\overline{RD}$ asserted to D0–D15 data in	t_{RD}		x (1 + W) – 28		3	ns
13	$\overline{RD}$ width low	t_{RR}	x (1 + W) – 10		21		ns
14	D0–D15 hold after $\overline{RD}$ negated	t_{HR}	0		0		ns
15	$\overline{RD}$ negated to next A0–A15 output	t_{RAE}	x – 15		16		ns
16	$\overline{WR}$ width low	t_{WW}	x (1 + W) – 10		21		ns
17	D0–D15 valid to $\overline{WR}$ negated	t_{DW}	x (1 + W) – 18		13		ns
18	D0–D15 hold after $\overline{WR}$ negated	t_{WD}	x – 15		16		ns
19	A0–A23 valid to $\overline{WAIT}$ input	t_{AWH}		2.5x – 30		48	ns
20	A0–A15 valid to $\overline{WAIT}$ input	t_{AWL}		2.5x – 30		48	ns
21	$\overline{WAIT}$ hold after $\overline{RD}$ or $\overline{WR}$ asserted	t_{CW}	(0.5 + N – 1) x + 2	(0.5 + N) x – 17	18	29	ns

*WAIT = 0

AC measurement conditions:
- Output levels: High = 2.4 V, Low = 0.45 V, CL = 30 pF
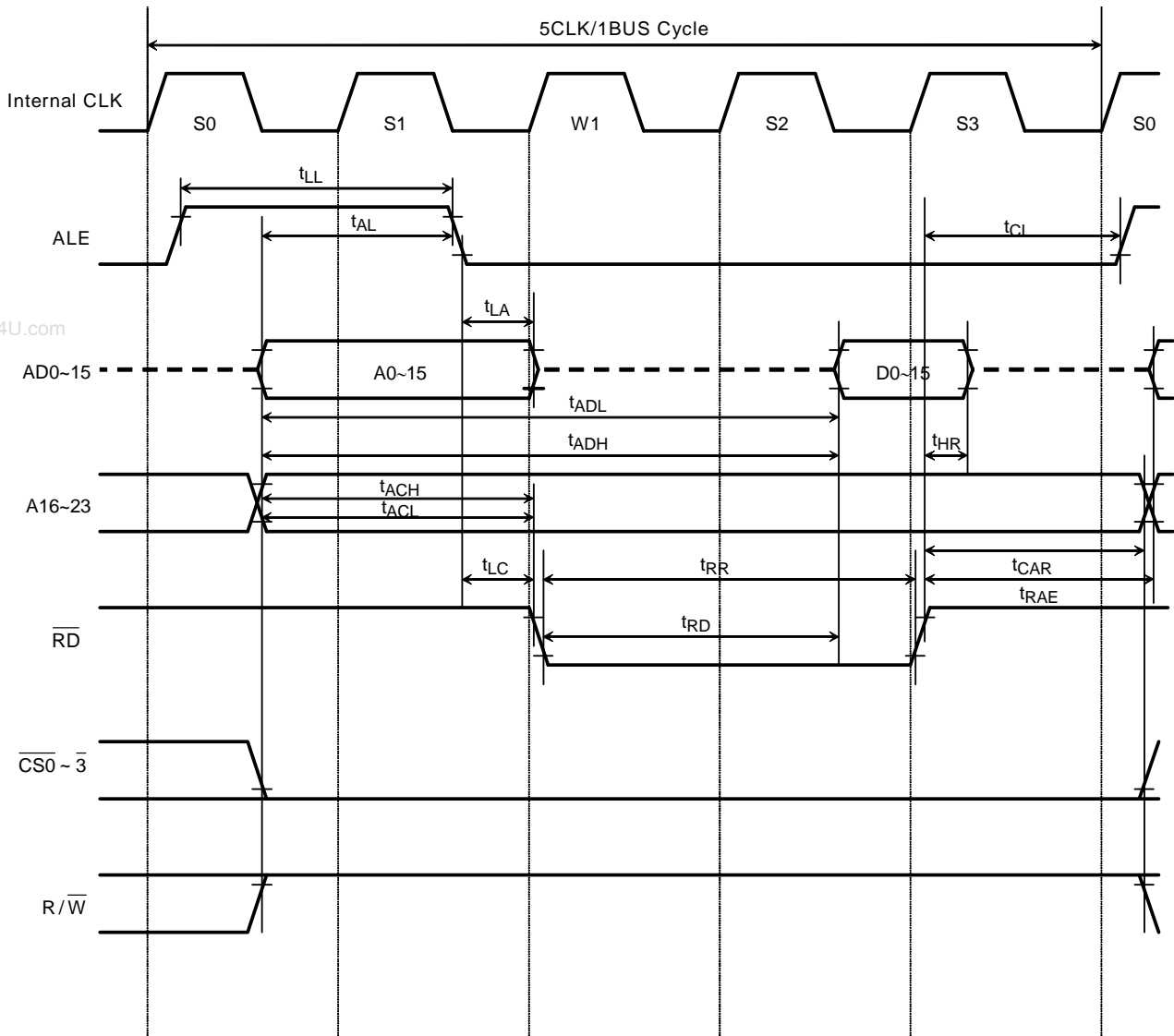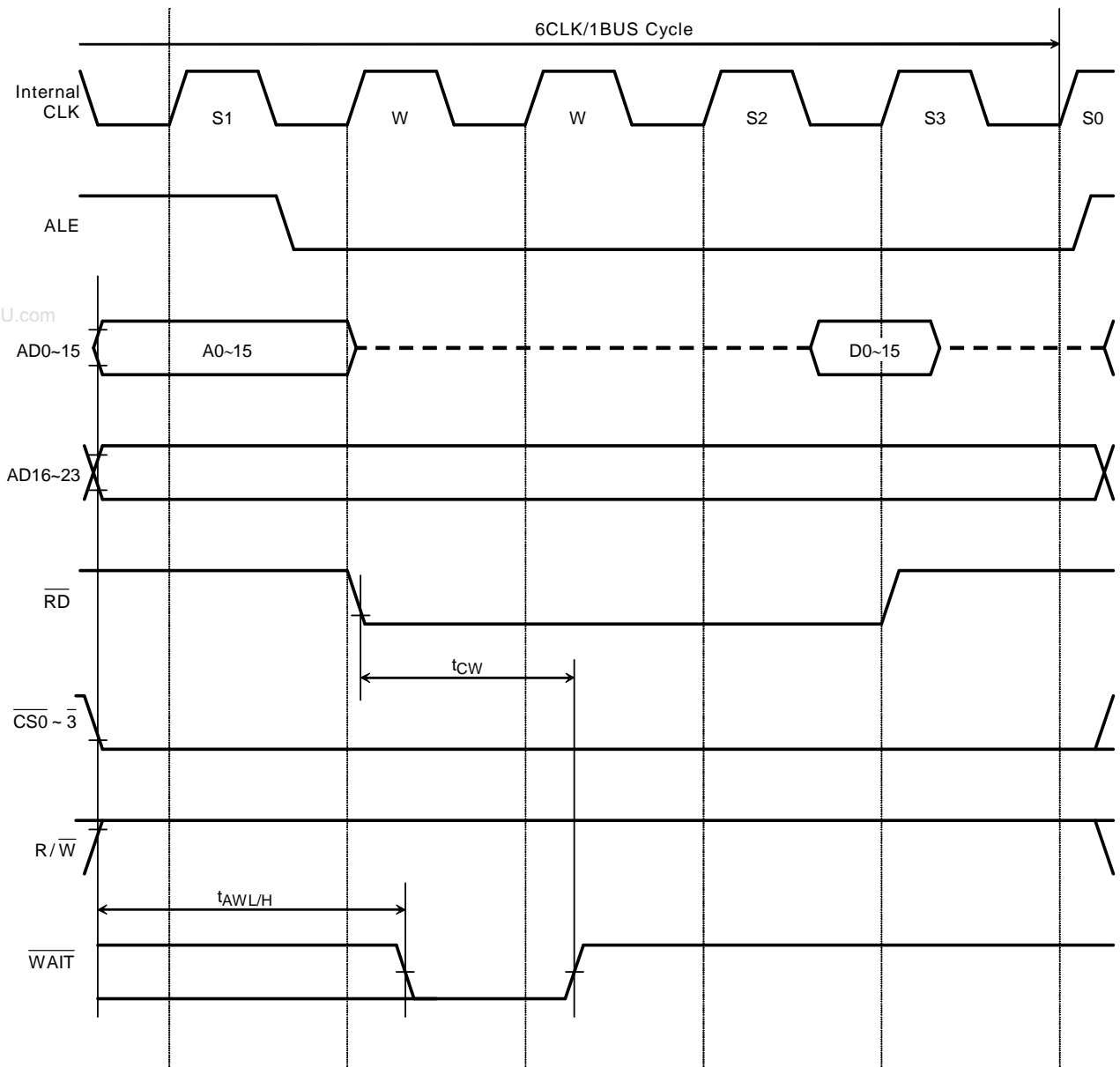- Input levels: High = 2 V, Low = 0.6 V

W: Number of wait-state cycles inserted (0 to 7 for programmed wait insertion)
N : Value of N for (1 + N) wait insertion

(1) Read Cycle Timing (ALE = 1.5,No-Wait)
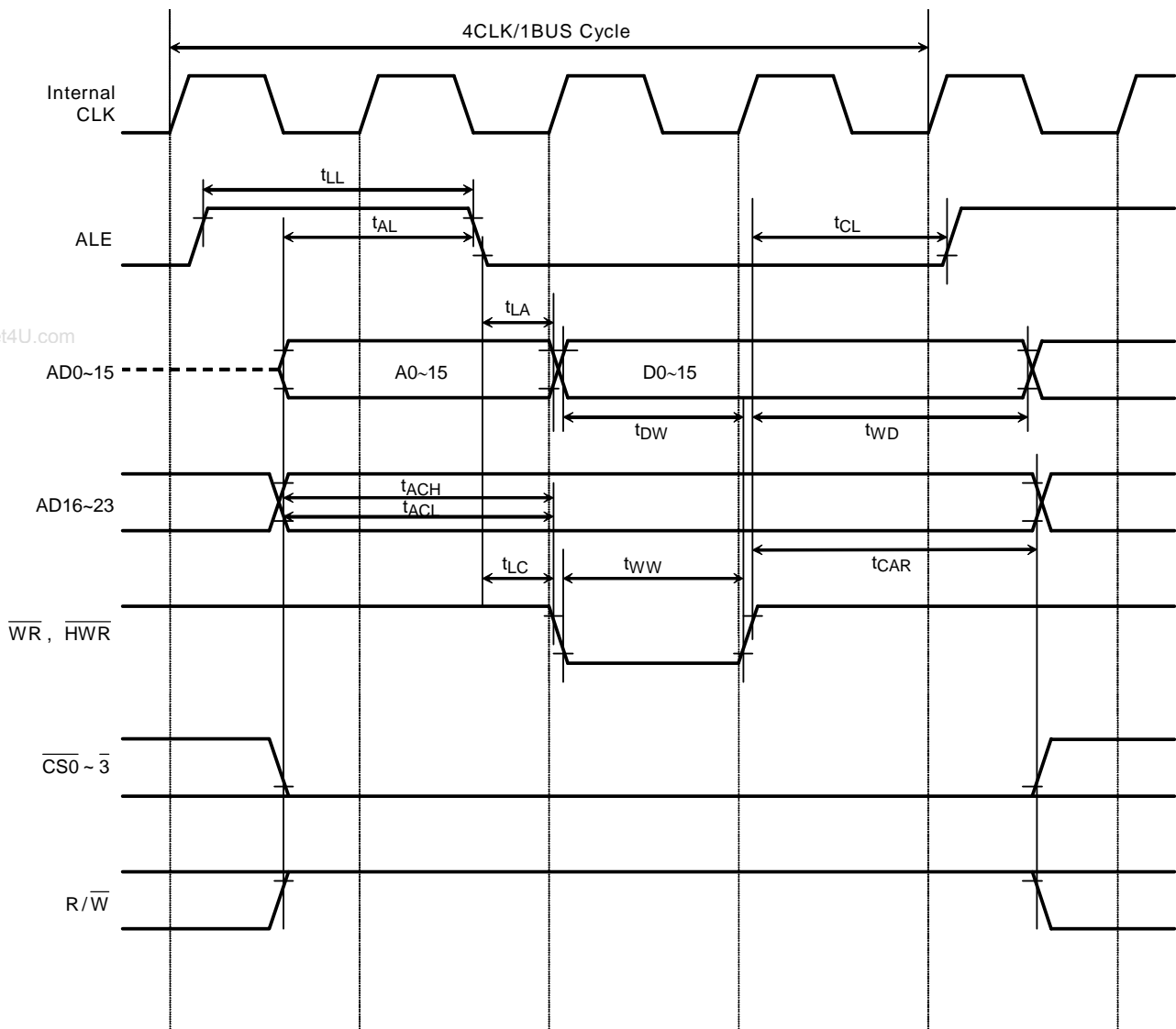
(2) Read Cycle Timing (ALE = 1.5,1-Wait (Internal wait))

5CLK/1BUS Cycle

Internal CLK

S0 S1 W1 S2 S3 S0

ALE

t_{LL}

t_{AL}

t_{CI}

t_{LA}

AD0~15 A0~15 D0~15

t_{ADL}

t_{ADH}

t_{HR}

A16~23

t_{ACH}

t_{ACL}

t_{LC} t_{RR} t_{CAR}

t_{RAE}

$\overline{RD}$

t_{RD}

$\overline{CS0} \sim \overline{3}$

R / $\overline{W}$

(3) Read Cycle Timing (ALE = 1.5.2-Wait (External N = 1))

Note: If t_{AWH} and/or t_{AWL} cannot be satisified, a bus cycle must be initiated with the $\overline{\text{WAIT}}$ pin asserted.

(4)   Write Cycle Timing (ALE = 1.5,No-Wait)

SIO Timing

(1) I/O Interface Mode

In the tables 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,SIO1,SIO3,SIO4)

Parameter	Symbol	Equation		20 MHz		32 MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK period	t_{SCY}	16x		800		500		ns
TxD data to SCLK rise or fall	t_{OSS}	$(t_{SCY}/2) - 5x - 23$		127		72		ns
TxD data hold after SCLK rise or fall*	t_{OHS}	$(t_{SCY}/2) + 3x$		550		343		ns
RxD data valid to SCLK rise or fall*	t_{SRD}	$2x + 8$		108		70		ns
RxD data hold after SCLK rise or fall*	t_{HSR}	0		0		0		ns

SIO5(DVCC51=2.7V~3.6V or 4.5V~5.25V)

Parameter	Symbol	Equation		20 MHz		32 MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK period	t_{SCY}	16x		800		500		ns
TxD data to SCLK rise or fall	t_{OSS}	$(t_{SCY}/2) - 5x - 23$		127		72		ns
TxD data hold after SCLK rise or fall*	t_{OHS}	$(t_{SCY}/2) + 3x$		550		343		ns
RxD data valid to SCLK rise or fall*	t_{SRD}	$2x + 8$		108		70		ns
RxD data hold after SCK rise	t_{HSR}	0		0		0		ns

Note *: SCLK rise or fall: Measured relative to the programmed active edge of SCLK.

2. SCLK Output Mode（SIO0,SIO1,SIO3,SIO4）

Parameter	Symbol	Equation		20 MHz		32 MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK period (programmable)	t_{SCY}	16x		800		500		ns
TxD data to SCLK rise	t_{OSS}	$(t_{SCY}/2) - 15$		385		235		ns
TxD data hold after SCLK rise	t_{OHS}	$(t_{SCY}/2) - 15$		385		235		ns
RxD data valid to SCK rise	t_{SRD}	$x + 23$		73		54		ns
RxD data hold after SCK rise	t_{HSR}	0		0		0		ns

(SIO5 DVCC51=2.7V~3.6V or 4.5V~5.25V)

Parameter	Symbol	Equation		20 MHz		32 MHz		Unit
		Min	Max	Min	Max	Min	Max	
SCLK period (programmable)	t_{SCY}	16x		800		500		ns
TxD data to SCLK rise	t_{OSS}	$(t_{SCY}/2) - 15$		385		235		ns
TxD data hold after SCLK rise	t_{OHS}	$(t_{SCY}/2) - 15$		385		235		ns
RxD data valid to SCK rise	t_{SRD}	$x + 23$		73		54		ns
RxD data hold after SCK rise	t_{HSR}	0		0		0		ns

4.9 SBI Timing

(1)   I2C mode

In the table below, the letters x and T represent the fsys and φT0 cycle periods, respectively. The letter n denotes the value of n programmed into the SCK[2:0] (SCL output frequency select) field in the SBI0CR1.

Parameter	Symbol	Equation		Standard Mode fsys = 8 MHz, n = 4		Fast Mode fsys = 32 MHz, n = 4		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
Low period of the SCL clock (Note 1)	t_{LOW}			4.7		1.3		μs
SCL clock high width	t_{HIGH}			4.0		0.6		μs
Setup time for a repeated START condition	$t_{SU;STA}$	Software (Note 5)		4.7		0.6		μs
Data hold time(Input)(Note3,4)	$t_{HD;DAT}$			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}	Software (Note 5)		4.7		1.3		μs

Note1: SCL clock low width (output) is calculated with $(2 (n - 1) + 4)T$.

                Standard mode: 6 μsec @ Typ (fsys = 8 MHz, n = 4)

                Fast mode: 1.5μsec @ Typ (fsys = 32 MHz, n = 4)

Note2: SCL clock high width (output) is caluculated with $(2 (n - 1))T$.

                Standard mode: 4 μsec @ Typ (fsys = 8 MHz, n = 4)

                Fast mode: 1μsec @ Typ (fsys = 32 MHz, n = 4)

Note3: The output data hold time is equal to 12X.

Note4: 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 fall edge of SCL. However,TMP1942CY/CZ 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.

Note5: Software-dependent.
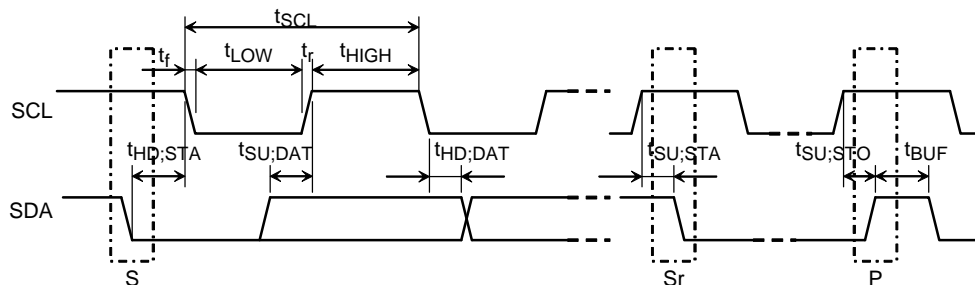
S: START condition
Sr: Repeated START condition
P; STOP condition

Note6: To operate the SBI in I2C Fast mode, the fysy frequency must be no less than 20 MHz. To operate the SBI in I2C Standard mode, the fysy fewquency must be no less than 4 MHz.

(2) Clock-Synchronous 8-Bit SIO Mode

In the tables below, the letters x and T represent the fsys and φT0 cycle periods, respectively. The letter n denotes the value of n programmed into the SCK[2:0] (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（DVCC51=2.7V~3.6V or 4.5V~5.25V）

Parameter	Symbol	Equation		32 MHz		Unit
		Min	Max	Min	Max	
SCK period	t_{SCY}	16x		500		ns
SO data to SCK rise	t_{OSS}	$(t_{SCY}/2) - (6x + 30)$		34		ns
SO data hold after SCK rise	t_{OHS}	$(t_{SCY}/2) + 4x$		374		ns
SI data valid to SCK rise	t_{SRD}	0		0		ns
SI data hold after SCK rise	t_{HSR}	4x + 10		134		ns

4. SCK Output Mode（DVCC51=2.7V~3.6V or 4.5V~5.25V）

Parameter	Symbol	Equation		32 MHz		Unit
		Min	Max	Min	Max	
SCK period (programmable)	t_{SCY}	$2^n/T$		1000		ns
SO data to SCK rise	t_{OSS}	$(t_{SCY}/2) - 20$		480		ns
SO data hold after SCK rise	t_{OHS}	$(t_{SCY}/2) - 20$		480		ns
SI data valid to SCK rise	t_{SRD}	2x + 30		92		ns
SI data hold after SCK rise	t_{HSR}	0		0		ns
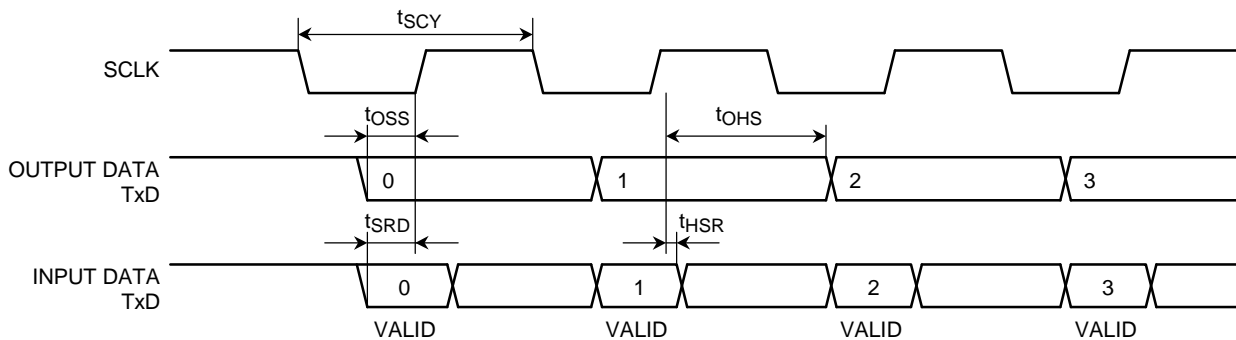
4.10　Event Counters

In the table below, the letter x represents the fsys cycle period.

Parameter	Symbol	Equation		32 MHz		Unit
		Min	Max	Min	Max	
Clock low pulse width	t_{VCKL}	2X + 100		163		ns
Clock high pulse width	t_{VCKH}	2X + 100		163		ns

4.11　Timer Capture

In the table below, the letter x represents the fsys cycle period

Parameter	Symbol	Equation		32MHz		Unit
		Min	Max	Min	Max	
Low pulse width	t_{CPL}	2X + 100		163		ns
High pulse width	t_{CPH}	2X + 100		163		ns

4.12　General Interrupts (INT0 to INTA)

In the table below, the letter x represents the fsys cycle period

Parameter	Symbol	Equation		32 MHz		Unit
		Min	Max	Min	Max	
Low pulse width for INT0–INTA	t_{INTAL}	X + 100		132		ns
High pulse width for INT0–INTA	t_{INTAH}	X + 100		132		ns

4.13　$\overline{NMI}$ and STOP/SLEEP Wake-up Interrupts

Parameter	Symbol	Equation		32 MHz		Unit
		Min	Max	Min	Max	
Low pulse width for $\overline{NMI}$ and INT0–INT4	t_{INTBL}	100		100		ns
High pulse width for INT0–INT4	t_{INTBH}	100		100		ns

4.14　SCOUT pin

Parameter	Symbol	Equation		32 MHz		Unit
		Min	Max	Min	Max	
SCOUT high pulse width	t_{SCH}	0.5T − 5		10.6		ns
SCOUT low pulse width	t_{SCL}	0.5T − 5		10.6		ns

Note: In the above table, the letter T represents the cycle period of the SCOUT output clock.
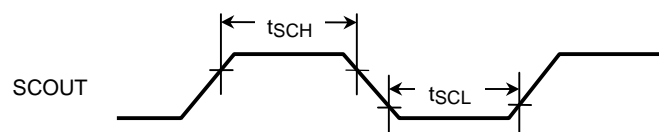
4.15 Bus Request and Bus Acknowledge Signals
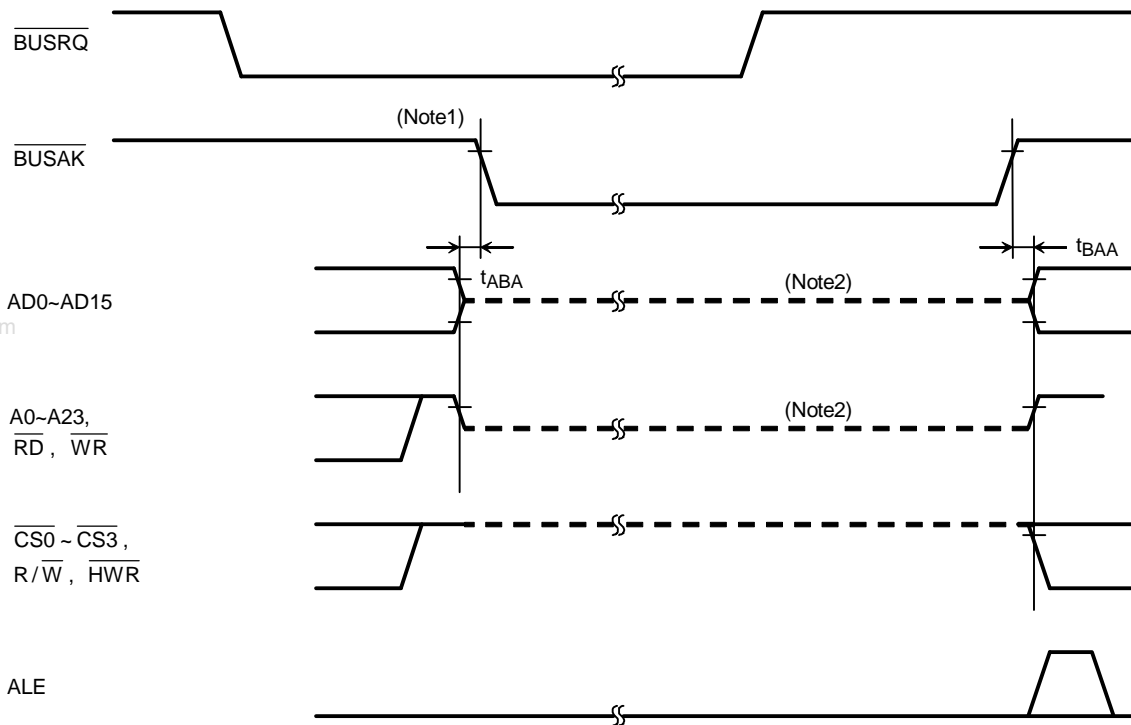
Parameter	Symbol	Equation		32 MHz		Unit
		Min	Max	Min	Max	
Bus float to $\overline{BUSAK}$ asserted	t_{ABA}	0	80	0	80	ns
Bus float after $\overline{BUSAK}$ negated	t_{BAA}	0	80	0	80	ns

Note 1: If the current bus cycle has not terminated due to wait-state insertion, the TMP1941AF does not respond to $\overline{BUSRQ}$ until the wait state ends.

Note 2: This broken lines indicate 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. The equipment manufacturer may maintain the bus at a predefined state by means of off-chip resistors, but he or she should design, considering the time (determined by the CR constant) it takes for a signal to reach a desired state. The on-chip, integrated programmable pullup/pulldown resistors remain active, depending on internal signal states.

4.16 KWUP

Pull-up Register Inactive

Parameter	Symbol	Equation		32 MHz		Unit
		Min	Max	Min	Max	
Low pulse width for KEY0~D	tky$_{TBL}$	100		100		ns
High pulse width for KEY0~D	tky$_{TBH}$	100		100		ns

Static Pull-up

Parameter	Symbol	Equation		32 MHz		Unit
		Min	Max	Min	Max	
Low pulse width for KEY0~D	tky$_{TBL}$	100		100		ns

Dynamic Pull-up

Parameter	Symbol	Equation		32 MHz		Unit
		Min	Max	Min	Max	
Low pulse width for KEY0~D	tky$_{TBL}$	T2+100		T2+100		ns

T2: Dynamic pull-up frequency

4.17 2-phase input pulse counter mode

Parameter	Symbol	Equation		32 MHz		Unit
		Min	Max	Min	Max	
2-phase input pulse cycle	Tdcyc	8Y		250		μs
2-phase input set up	Tabs	Y+20		31.27		μs
2-phase input hold	Tabh	Y+20		31.27		μs

Y:Sampling clock(fs or fsys/2)
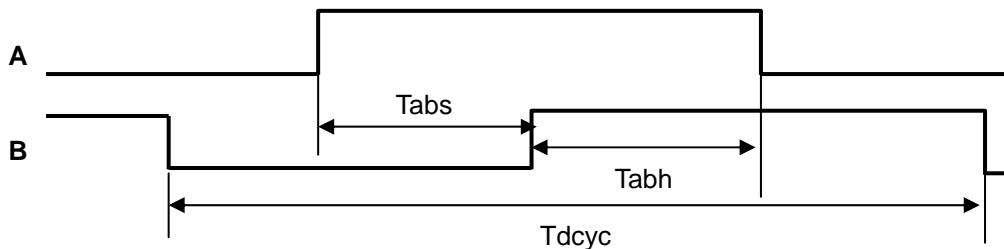
4.18 ADTRG Input

Parameter	Symbol	Equation		32 MHz		Unit
		Min	Max	Min	Max	
ADTRG low level pulse width	tad$_L$	fsysy/2+20		51.25		ns
ADTRG high level pulse interval	Tadh	fsysy/2+20		51.25		ns

5. I/O Register Summary

The internal I/O registers occupy 8-kbyte addresses from FFFFE000H through FFFFFFFFH.

(1) I/O Ports

(2) Watchdog Timer (WDT)

(3) Real-Time Clock (RTC)

(4) 8-Bit Timer

(5) 16-Bit Timer

(6) UART/Serial I/O 0/1 (UART/SIO)

(7) I2CBUS/Serial I/O (I2C/SIO)

(8) UART/Serial I/O 3/4/5 (UART/SIO)

(9) 10-Bit A/D Converter (ADC)

(10) 10-Bit D/A Converter (DAC)

(11) Key On Waik up (KWUP)

(12) Interrupt Controller (INTC)

(13) DMA Controller (DMAC)

(14) Chip Select (CS)/Wait Controller

(15) Clock Generator (CG)

(16) FLASH

(17) ROM correction

Table Organization

Mnemonic	Register Name	Address	7	6			1	0	
									→ Bit Symbol
									→ Read/Write
									→ Reset Value
									→ Function

Access

R/W ： Read/Write. The user can read and write the register bit.

R ： Read omly.

W ： Write only.

W* ： The user can read and write the register bit, but a read always returns a value of 1.

[1] I/O PORT

Address	Mnemonic
FFFFF000H	P0
1H	P1
2H	P0CR
3H	
4H	P1CR
5H	P1FC
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFF010H	
1H	
2H	P2
3H	
4H	P2CR
5H	P2FC
6H	
7H	
8H	P3
9H	
AH	P3CR
BH	P3FC
CH	
DH	
EH	P4
FH	

Address	Mnemonic
FFFFF020H	P4CR
1H	P4FC
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFF040H	P5
1H	P6
2H	
3H	P5FC
4H	
5H	P6FC
6H	
7H	
8H	
9H	
AH	
BH	
CH	P9
DH	
EH	P9CR
FH	P9FC

Address	Mnemonic
FFFFF050H	PA
1H	PB
2H	PACR
3H	PAFC
4H	PBCR
5H	PBFC
6H	
7H	
8H	PC
9H	PD
AH	PCCR
BH	PCFC
CH	PDCR
DH	PDFC1
EH	PDFC2
FH	PDODE

Address	Mnemonic
FFFFF060H	PE
1H	PF
2H	PECR
3H	PEFC
4H	PFCR
5H	PFFC
6H	PEODE
7H	PFODE
8H	
9H	
AH	
BH	
CH	Reserved
DH	Reserved
EH	Reserved
FH	Reserved

[2] WDT

Address	Mnemonic
FFFFF090H	WDMOD
1H	WDCR
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[3] RTC

Address	Mnemonic
FFFFF0A0H	RTCCR
1H	
2H	
3H	
4H	RTCREG
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[4] 8 bit Timers

Address	Mnemonic	Address	Mnemonic	Address	Mnemonic
FFFFF100H	TA01RUN	FFFFF110H	TA45RUN	FFFFF120H	TA89RUN
1H		1H		1H	
2H	TA0REG	2H	TA4REG	2H	TA8REG
3H	TA1REG	3H	TA5REG	3H	TA9REG
4H	TA01MOD	4H	TA45MOD	4H	TA89MOD
5H	TA1FFCR	5H	TA5FFCR	5H	TA9FFCR
6H		6H		6H	
7H		7H		7H	
8H	TA23RUN	8H	TA67RUN	8H	TAABRUN
9H		9H		9H	
AH	TA2REG	AH	TA6REG	AH	TAAREG
BH	TA3REG	BH	TA7REG	BH	TABREG
CH	TA23MOD	CH	TA67MOD	CH	TAABMOD
DH	TA3FFCR	DH	TA7FFCR	DH	TABFFCR
EH		EH		EH	
FH		FH		FH	

[5] 16 bit Timers

Address	Mnemonic	Address	Mnemonic	Address	Mnemonic	Address	Mnemonic
FFFFF140H	TB0RUN	FFFFF150H	TB1RUN	FFFFF160H	TB2RUN	FFFFF170H	TB3RUN
1H		1H		1H		1H	
2H	TB0MOD	2H	TB1MOD	2H	TB2MOD	2H	TB3MOD
3H	TB0FFCR	3H	TB1FFCR	3H	TB2FFCR	3H	TB3FFCR
4H	TB0ST	4H	TB1ST	4H	TB2ST	4H	TB3ST
5H		5H		5H		5H	
6H		6H		6H		6H	
7H		7H		7H		7H	
8H	TB0RG0L	8H	TB1RG0L	8H	TB2RG0L	8H	TB3RG0L
9H	TB0RG0H	9H	TB1RG0H	9H	TB2RG0H	9H	TB3RG0H
AH	TB0RG1L	AH	TB1RG1L	AH	TB2RG1L	AH	TB3RG1L
BH	TB0RG1H	BH	TB1RG1H	BH	TB2RG1H	BH	TB3RG1H
CH	TB0CP0L	CH	TB1CP0L	CH	TB2CP0L	CH	TB3CP0L
DH	TB0CP0H	DH	TB1CP0H	DH	TB2CP0H	DH	TB3CP0H
EH	TB0CP1L	EH	TB1CP1L	EH	TB2CP1L	EH	TB3CP1L
FH	TB0CP1H	FH	TB1CP1H	FH	TB2CP1H	FH	TB3CP1H

Address	Mnemonic	Address	Mnemonic	Address	Mnemonic	Address	Mnemonic
FFFFF180H	TB4RUN	FFFFF190H	TB5RUN	FFFFF1A0H	TB6RUN	FFFFF1B0H	TB7RUN
1H		1H		1H		1H	
2H	TB4MOD	2H	TB5MOD	2H	TB6MOD	2H	TB7MOD
3H	TB4FFCR	3H	TB5FFCR	3H	TB6FFCR	3H	TB7FFCR
4H	TB4ST	4H	TB5ST	4H	TB6ST	4H	TB7ST
5H		5H		5H		5H	
6H		6H		6H		6H	
7H		7H		7H		7H	
8H	TB4RG0L	8H	TB5RG0L	8H	TB6RG0L	8H	TB7RG0L
9H	TB4RG0H	9H	TB5RG0H	9H	TB6RG0H	9H	TB7RG0H
AH	TB4RG1L	AH	TB5RG1L	AH	TB6RG1L	AH	TB7RG1L
BH	TB4RG1H	BH	TB5RG1H	BH	TB6RG1H	BH	TB7RG1H
CH	TB4CP0L	CH	TB5CP0L	CH	TB6CP0L	CH	TB7CP0L
DH	TB4CP0H	DH	TB5CP0H	DH	TB6CP0H	DH	TB7CP0H
EH	TB4CP1L	EH	TB5CP1L	EH	TB6CP1L	EH	TB7CP1L
FH	TB4CP1H	FH	TB5CP1H	FH	TB6CP1H	FH	TB7CP1H

Address	Mnemonic
FFFFF1C0H	TB8RUN
1H	
2H	TB8MOD
3H	Reserved
4H	TB8ST
5H	
6H	
7H	
8H	TB8RG0L
9H	TB8RG0H
AH	TB8RG1L
BH	TB8RG1H
CH	TB8CP0L
DH	TB8CP0H
EH	TB8CP1L
FH	TB8CP1H

Address	Mnemonic
FFFFF1D0H	TB9RUN
1H	
2H	TB9MOD
3H	Reserved
4H	TB9ST
5H	
6H	
7H	
8H	TB9RG0L
9H	TB9RG0H
AH	TB9RG1L
BH	TB9RG1H
CH	TB9CP0L
DH	TB9CP0H
EH	TB9CP1L
FH	TB9CP1H

Address	Mnemonic
FFFFF1E0H	TBARUN
1H	
2H	TBAMOD
3H	Reserved
4H	TBAST
5H	
6H	
7H	
8H	TBARG0L
9H	TBARG0H
AH	TBARG1L
BH	TBARG1H
CH	TBACP0L
DH	TBACP0H
EH	TBACP1L
FH	TBACP1H

Address	Mnemonic
FFFFF1F0H	TBBRUN
1H	
2H	TBBMOD
3H	Reserved
4H	TBBST
5H	
6H	
7H	
8H	TBBRG0L
9H	TBBRG0H
AH	TBBRG1L
BH	TBBRG1H
CH	TBBCP0L
DH	TBBCP0H
EH	TBBCP1L
FH	TBBCP1H

Address	Mnemonic
FFFFF200H	TBCRUN
1H	
2H	TBCMOD
3H	Reserved
4H	TBCST
5H	
6H	
7H	
8H	TBCRG0L
9H	TBCRG0H
AH	TBCRG1L
BH	TBCRG1H
CH	TBCCP0L
DH	TBCCP0H
EH	TBCCP1L
FH	TBCCP1H

Address	Mnemonic
FFFFF210H	TBDRUN
1H	
2H	TBDMOD
3H	Reserved
4H	TBDST
5H	
6H	
7H	
8H	TBDRG0L
9H	TBDRG0H
AH	TBDRG1L
BH	TBDRG1H
CH	TBDCP0L
DH	TBDCP0H
EH	TBDCP1L
FH	TBDCP1H

[6] UART/SIO 0/1

Address	Mnemonic
FFFFF230H	SC0BUF
1H	SC0CR
2H	SC0MOD0
3H	BR0CR
4H	BR0ADD
5H	SC0MOD1
6H	SC0MOD2
7H	
8H	SC1BUF
9H	SC1CR
AH	SC1MOD0
BH	BR1CR
CH	BR1ADD
DH	SC1MOD1
EH	SC1MOD2
FH	

[7] I2CBUS/SIO

Address	Mnemonic
FFFFF240H	SBI0CR1
1H	SBI0DBR
2H	I2C0AR
3H	SBI0CR2/SR
4H	SBI0BR0
5H	(SBI0BR1)
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[8] UART/SIO 3/4

Address	Mnemonic
FFFFF280H	SC3BUF
1H	SC3CR
2H	SC3MOD0
3H	BR3CR
4H	BR3ADD
5H	SC3MOD1
6H	SC3MOD2
7H	
8H	SC4BUF
9H	SC4CR
AH	SC4MOD0
BH	BR4CR
CH	BR4ADD
DH	SC4MOD1
EH	SC4MOD2
FH	

UART/SIO 5

Address	Mnemonic
FFFFF290H	SC5BUF
1H	SC5CR
2H	SC5MOD0
3H	BR5CR
4H	BR5ADD
5H	SC5MOD1
6H	SC5MOD2
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[9] 10 bitADC

Address	Mnemonic
FFFFF300H	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

Address	Mnemonic
FFFFF310H	ADREGSPL
1H	ADREGSPH
2H	
3H	
4H	ADCOML
5H	ADCOMH
6H	
7H	
8H	ADMOD0
9H	ADMOD1
AH	ADMOD2
BH	ADMOD3
CH	ADMOD4
DH	
EH	
FH	ADCLK

[10] 10BIT DAC

Address	Mnemonic
FFFFF340H	DAREG0L
1H	DAREG0H
2H	DACCNT0
3H	
4H	DAREG1L
5H	DAREG1H
6H	DACCNT1
7H	
8H	DAREG2L
9H	DAREG2H
AH	DACCNT2
BH	
CH	
DH	
EH	
FH	

[11] KWUP

Address	Mnemonic
FFFFF360H	KWUPST0
1H	KWUPST1
2H	KWUPST2
3H	KWUPST3
4H	KWUPST4
5H	KWUPST5
6H	KWUPST6
7H	KWUPST7
8H	KWUPST8
9H	KWUPST9
AH	KWUPSTA
BH	KWUPSTB
CH	KWUPSTC
DH	KWUPSTD
EH	
FH	

Address	Mnemonic
FFFFF370H	KWUPCLR
1H	KWUPCNT
2H	Reserved
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[12] INTBCDE

Address	Mnemonic
FFFFF 380H	INTBST
1H	INTCST
2H	INTDST
3H	INTEST
4H	INTFLG
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic

[13] INTC

Address	Mnemonic
FFFFE000H	IMC0
1H	
2H	
3H	
4H	IMC1
5H	
6H	
7H	
8H	IMC2
9H	
AH	
BH	
CH	IMC3
DH	
EH	
FH	

Address	Mnemonic
FFFFE010H	IMC4
1H	
2H	
3H	
4H	IMC5
5H	
6H	
7H	
8H	IMC6
9H	
AH	
BH	
CH	IMC7
DH	
EH	
FH	

Address	Mnemonic
FFFFE020H	IMC8
1H	
2H	
3H	
4H	IMC9
5H	
6H	
7H	
8H	IMCA
9H	
AH	
BH	
CH	IMCB
DH	
EH	
FH	

Address	Mnemonic
FFFFE030H	IMCC
1H	
2H	
3H	
4H	IMCD
5H	
6H	
7H	
8H	IMCE
9H	
AH	
BH	
CH	IMCF
DH	
EH	
FH	

Address	Mnemonic
FFFFE040H	IVR
1H	
2H	IVR
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFE050H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFE060H	INTCLR
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFE070H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[14] DAMC

Address	Mnemonic
FFFFE200H	CCR0
1H	
2H	
3H	
4H	CSR0
5H	
6H	
7H	
8H	SAR0
9H	
AH	
BH	
CH	DAR0
DH	
EH	
FH	

Address	Mnemonic
FFFFE210H	BCR0
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	DTCR0
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFE220H	CCR1
1H	
2H	
3H	
4H	CSR1
5H	
6H	
7H	
8H	SAR1
9H	
AH	
BH	
CH	DAR1
DH	
EH	
FH	

Address	Mnemonic
FFFFE230H	BCR1
1H	
2H	
3H	
4H	NCR1
5H	
6H	
7H	
8H	DTCR1
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFE240H	CCR2
1H	
2H	
3H	
4H	CSR2
5H	
6H	
7H	
8H	SAR2
9H	
AH	
BH	
CH	DAR2
DH	
EH	
FH	

Address	Mnemonic
FFFFE250H	BCR2
1H	
2H	
3H	
4H	NCR2
5H	
6H	
7H	
8H	DTCR2
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFE260H	CCR3
1H	
2H	
3H	
4H	CSR3
5H	
6H	
7H	
8H	SAR3
9H	
AH	
BH	
CH	DAR3
DH	
EH	
FH	

Address	Mnemonic
FFFFE270H	BCR3
1H	
2H	
3H	
4H	NCR3
5H	
6H	
7H	
8H	DTCR3
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFE280H	DCR
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	DHR
DH	
EH	
FH	

Address	Mnemonic
FFFFE290H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFE2A0H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

Address	Mnemonic
FFFFE2B0H	
1H	
2H	
3H	
4H	
5H	
6H	
7H	
8H	
9H	
AH	
BH	
CH	
DH	
EH	
FH	

[15] CS/WAIT Controller

Address	Mnemonic	Address	Mnemonic	Address	Mnemonic	Address	Mnemonic
FFFFE400H	BMA0	FFFFE410H		FFFFE480H	B01CS	FFFFE490H	
1H		1H		1H		1H	
2H		2H		2H		2H	
3H		3H		3H		3H	
4H	BMA1	4H		4H	B23CS	4H	
5H		5H		5H		5H	
6H		6H		6H		6H	
7H		7H		7H		7H	
8H	BMA2	8H		8H	BEXCS	8H	
9H		9H		9H		9H	
AH		AH		AH		AH	
BH		BH		BH		BH	
CH	BMA3	CH		CH		CH	
DH		DH		DH		DH	
EH		EH		EH		EH	
FH		FH		FH		FH	

[16] CG

Address	Mnemonic	Address	Mnemonic	Address	Mnemonic	Address	Mnemonic
FFFFEE00H	SYSCR0	FFFFEE10H	IMCGA0	FFFFEE20H	EICRCG	FFFFEE40H	Reserved
1H	SYSCR1	1H		1H	Reserved	1H	Reserved
2H	SYSCR2	2H		2H	Reserved	2H	Reserved
3H	SYSCR3	3H		3H	Reserved	3H	Reserved
4H	ADCCK	4H	IMCGB0	4H		4H	Reserved
5H		5H		5H		5H	Reserved
6H		6H		6H		6H	Reserved
7H		7H		7H		7H	Reserved
8H		8H	Reserved	8H		8H	Reserved
9H		9H	Reserved	9H		9H	Reserved
AH		AH	Reserved	AH		AH	Reserved
BH		BH	Reserved	BH		BH	Reserved
CH		CH		CH		CH	
DH		DH		DH		DH	
EH		EH		EH		EH	
FH		FH		FH		FH	

[17] FLASH(FLASH only./Access to FLASH is not possible with DMA.)

Address	Mnemonic	Address	Mnemonic
FFFFE510H	SEQMOD	FFFFE520H	FLCS
1H		1H	
2H		2H	
3H		3H	
4H	SEQCNT	4H	
5H		5H	
6H		6H	
7H		7H	
8H		8H	
9H		9H	
AH		AH	
BH		BH	
CH		CH	
DH		DH	
EH		EH	
FH		FH	

[18] ROM correction(Access to FLASH is not possible with DMA.)

Address	Mnemonic
FFFFE540H	ADDREG0
1H	
2H	
3H	
4H	ADDREG1
5H	
6H	
7H	
8H	ADDREG2
9H	
AH	
BH	
CH	ADDREG3
DH	
EH	
FH	

6. JTAG Interface

The TMP1942FDXB/CYXB processor provides a boundary-scan interface that is compatible with Joint Test Action Group (JTAG) specifications, using the industry-standard JTAG protocol (IEEE Standard 1149.1/D6).

This chapter describes that interface, including descriptions of boundary scanning, the pins and signals used by the interface, and the Test Access Port (TAP).

6.1 What Boundary Scanning Is

With the evolution of ever-denser integrated circuits (ICs), surface-mounted devices, double-sided component mounting on printed-circuit boards (PCBs), and buried vias, in-circuit tests that depend upon making physical contact with internal board and chip connections have become more and more difficult to use. The greater complexity of ICs has also meant that tests to fully exercise these chips have become much larger and more difficult to write.

One solution to this difficulty has been the development of *boundary-scan* circuits. A boundary-scan circuit is a series of shift register cells placed between each pin and the internal circuitry of the IC to which the pin is connected, as shown in Figure 6.1.1. Normally, these boundary-scan cells are bypassed; when the IC enters test mode, however, the scan cells can be directed by the test program to pass data along the shift register path and perform various diagnostic tests. To accomplish this, the tests use the four signals described in the next section: **TDI**, **TDO**, **TMS**, **TCK, and** $\overline{\text{TRST}}$.
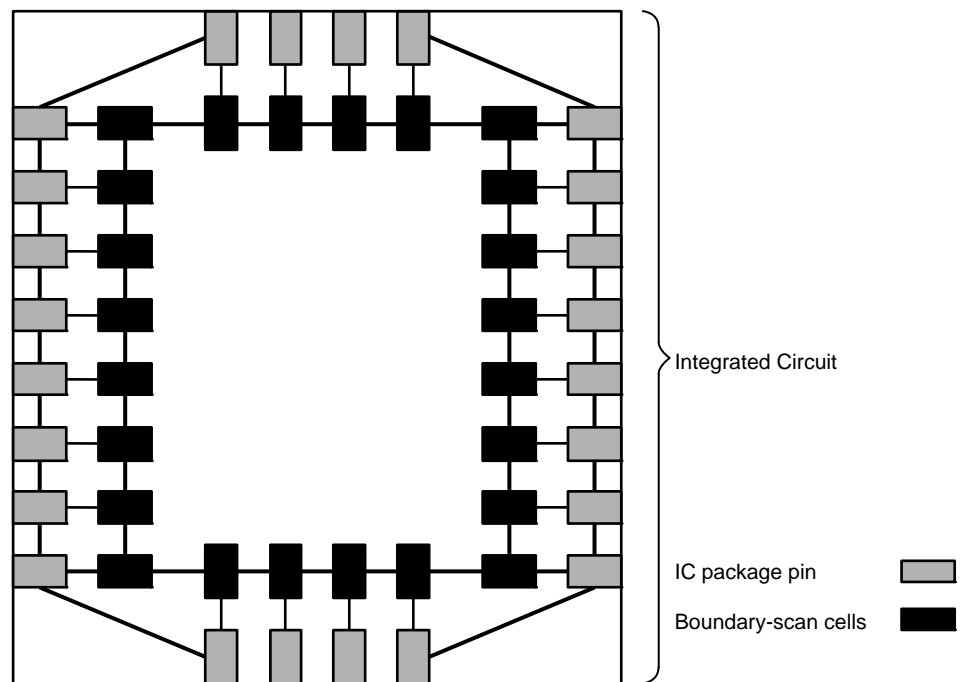
Figure 6.1.1 JTAG Boundary-scan Cells

6.2 Signal Summary

The JTAG interface signals are listed below and shown in Figure 6.2.1.

- TDI   JTAG serial data in

- TDO   JTAG serial data out

- TMS   JTAG test mode select

- TCK   JTAG serial clock input

- $\overline{\text{TRST}}$ JTAG test reset input
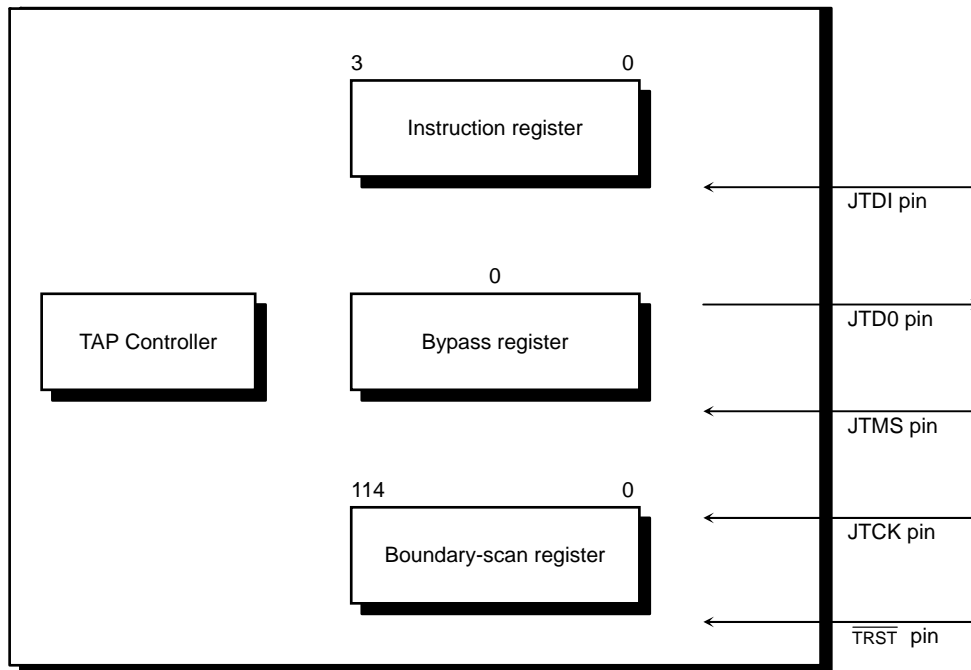
Figure 6.2.1 JTAG Interface Signals and Registers

The JTAG boundary-scan mechanism (referred to in this chapter as JTAG mechanism) allows testing of the connections between the processor, the printed circuit board to which it is attached, and the other components on the circuit board.

The JTAG mechanism does not provide any capability for testing the processor itself.

6.3 JTAG Controller and Registers

The processor contains the following JTAG controller and registers:

- Instruction register

- Boundary-scan register

- Bypass register

- ID Code register

- Test Access Port (TAP) controller

The processor executes the standard JTAG EXTEST operation associated with External Test functionality testing.

The basic operation of JTAG is for the TAP controller state machine to monitor the JTMS input signal. When it occurs, the TAP controller determines the test functionality to be implemented. This includes either loading the JTAG instruction register (IR), or beginning a serial data scan through a data register (DR), listed in Table 6.3.1. As the data is scanned in, the state of the JTMS pin signals each new data word, and indicates the end of the data stream. The data register to be selected is determined by the contents of the Instruction register.

6.3.1 Instruction Register

The JTAG Instruction register includes eight shift register-based cells; this register is used to select the test to be performed and/or the test data register to be accessed. As listed in Table 6.3.1, this encoding selects either the Boundary-scan register or the Bypass register or Device Identification register.

Table 6.3.1 JTAG Instruction Register Bit Encoding

Instruction Code (MSB → LSB)	Instruction	Selected Data Register
0000	EXTEST	Boundary Scan Register
0001	SAMPLE/PRELOAD	Boundary Scan Register
0010 to 1110	Reserved	Reserved
1111	BYPASS	Bypass register

Figure 66.3.1 shows the format of the Instruction register

3	2	1	0
MSB			LSB
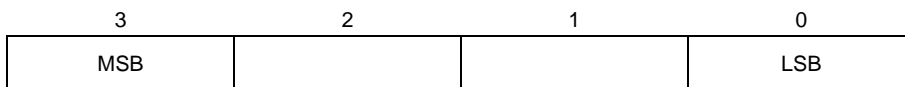
Figure 66.3.1 Instruction Register

The instruction code is shifted out to the Instruction register from the LSB.

Figure 6.3.2 Instruction Register Shift Direction

6.3.2 Bypass Register

The Bypass register is 1 bit wide. When the TAP controller is in the Shift-DR (Bypass) state, the data on the TDI pin is shifted into the Bypass register, and the Bypass register output shifts to the TDO output pin.

In essence, the Bypass register is a short-circuit which allows bypassing of board-level devices, in the serial boundary-scan chain, which are not required for a specific test. The logical location of the Bypass register in the boundary-scan chain is shown in Figure6.3.3. Use of the Bypass register speeds up access to boundary-scan registers in those ICs that remain active in the board-level test datapath.
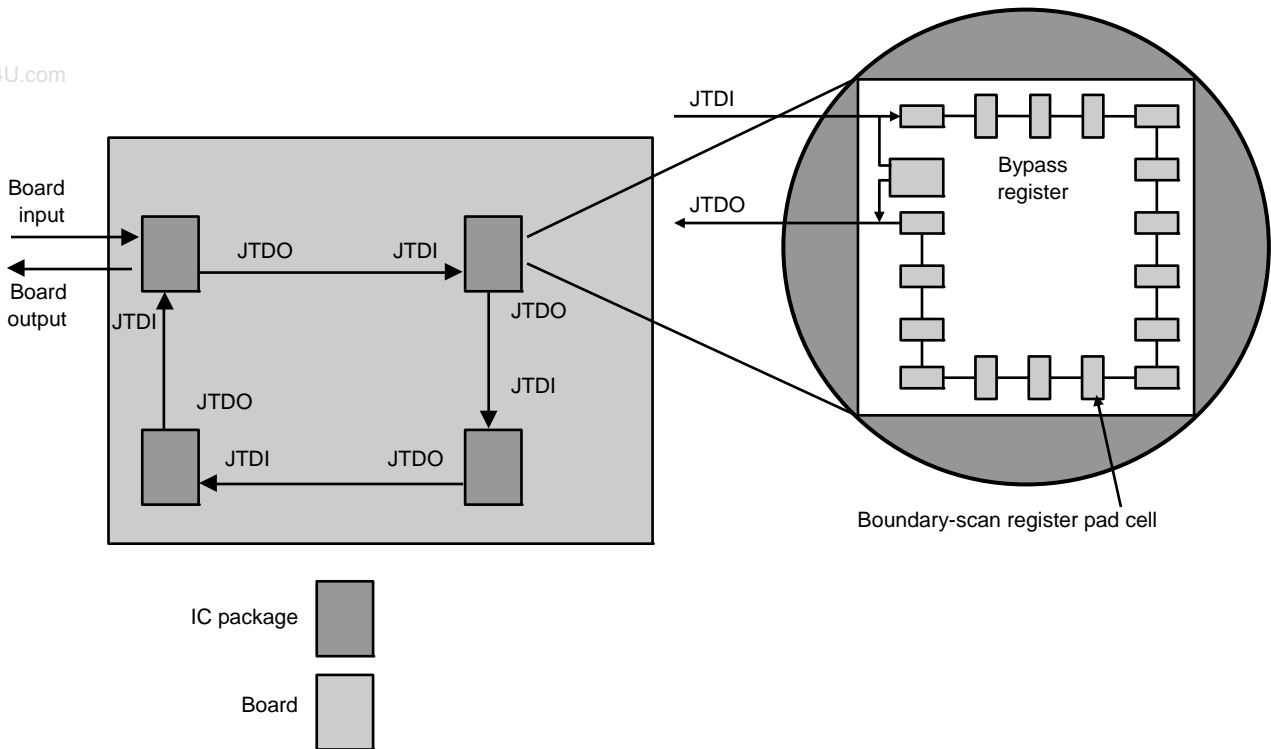
Figure 6.3.3 Bypass Register Operation

6.3.3 Boundary-Scan Register

The Boundary Scan register includes all of the inputs and outputs of the TMP1942 processor, except some analog output and control signals. The pins of the TMP1942 chip can be configured to drive any arbitrary pattern by scanning into the Boundary Scan register from the Shift-DR state. Incoming data to the processor is examined by shifting while in the Capture-DR state with the Boundary Scan register enabled.

The Boundary-scan register is a single, 115-bit-wide, shift register-based path containing cells connected to all input and output pads on the TMP1942 processor.

The TDI input is loaded to the LSB of the Boundary Scan register. The MSB of the Boundary Scan register is retrieved from the JTDO output.

6.3.4 Test Access Port (TAP)

The Test Access Port (TAP) consists of the five signal pins: $\overline{\text{TRST}}$, **TDI**, **TDO**, **TMS**, and **TCK**. Serial test data and instructions are communicated over these five signal pins, along with control of the test to be executed.

As Figure shows, data is serially scanned into one of the three registers (Instruction register, Bypass register, or the Boundary-scan register) from the **TDI** pin, or it is scanned from one of these three registers onto the **TDO** pin.

The **TMS** input controls the state transitions of the main TAP controller state machine.

The **TCK** input is a dedicated test clock that allows serial JTAG data to be shifted synchronously, independent of any chip-specific or system clocks.
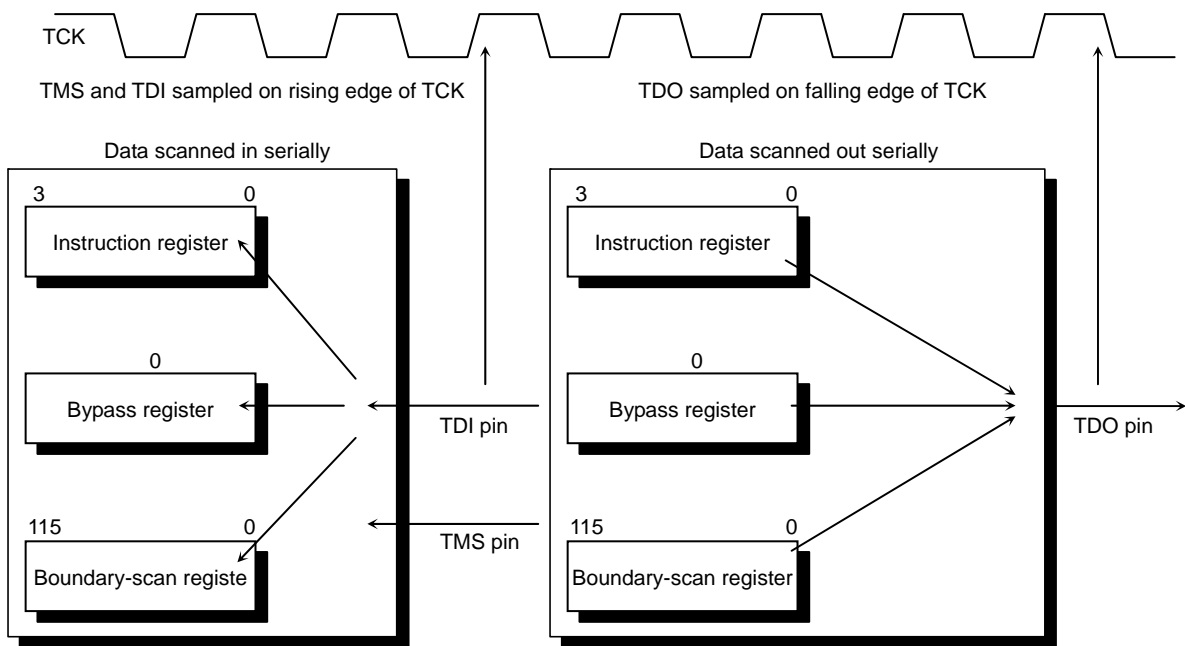
Figure 6.3.4 JTAG Test Access Port

Data on the **TDI** and **TMS** pins is sampled on the rising edge of the **TCK** input clock signal. Data on the **TDO** pin changes on the falling edge of the **TCK** clock signal.

6.3.5 TAP Controller

The processor implements the 16-state TAP controller as defined in the IEEE JTAC specification.

6.3.6 Controller Reset

The TAP controller state machine can be put into Reset state the following:

- assertion of the $\overline{\text{TRST}}$ signal (Low) resets the TAP controller.

- keeping the **TMS** input signal asserted through five consecutive rising edges of **TCK** input.

In either case, keeping **TMS** asserted maintains the Reset state.

6.3.7 TAP Controller

The state transition diagram of the TAP controller is shown in Figure6.3.5. Each arrow between states is labeled with a 1 or 0, indicating the logic value of TMS that must be set up before the rising edge of TCK to cause the transition.
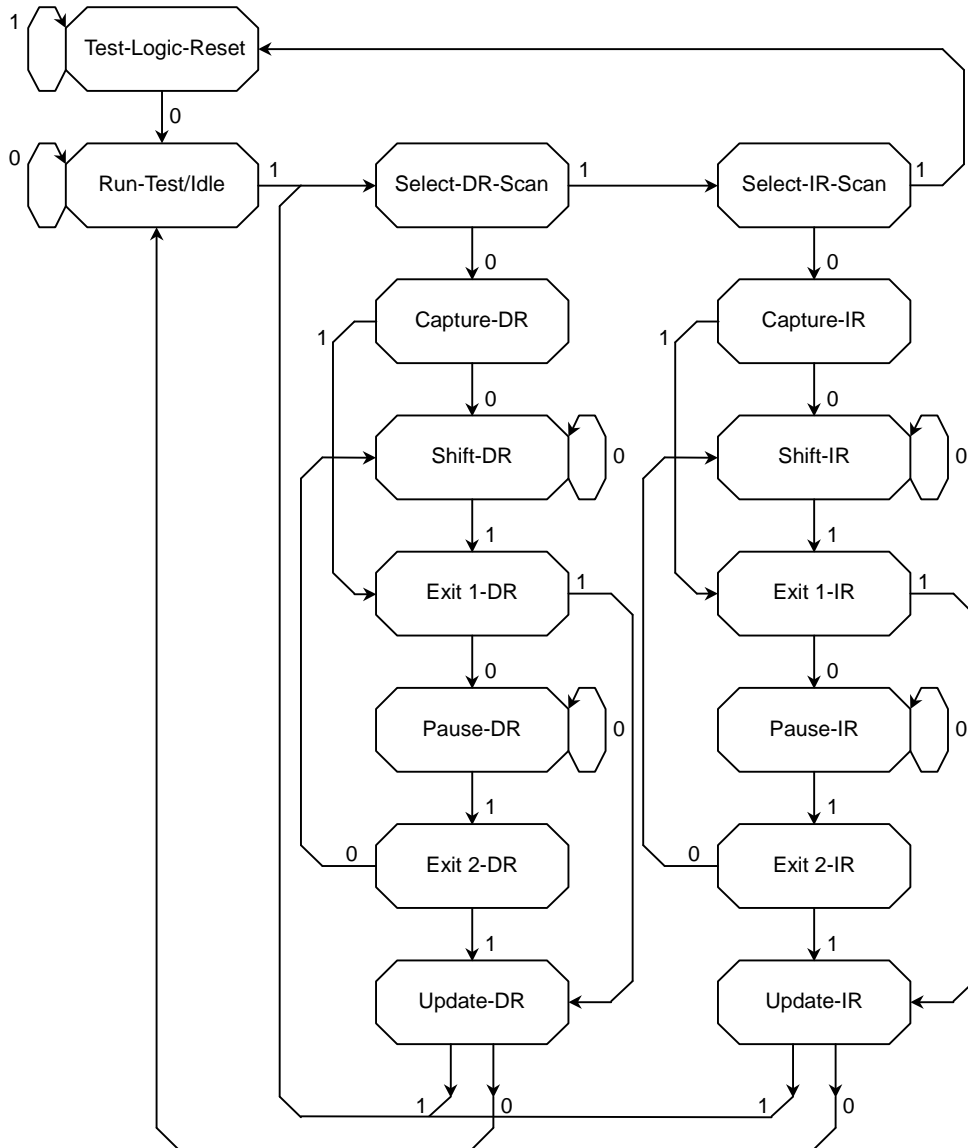
Figure 6.3.5 TAP Controller State Diagram

The following paragraphs describe each of the controller states. The left vertical column in Figure6.3.5 is the data column, and the right vertical column is the instruction column. The data column and instruction column reference data register (DR) and instruction register (IR), respectively.

- Test-Logic-Reset
  When the TAP controller is in the Reset state, the Device Identification register is selected as default. The three most significant bits of the Boundary-scan register are cleared to 0, disabling the outputs.

  The controller remains in this state while TMS is high. If TMS is held low while the controller is in this state, then the controller moves to the Run-Test/Idle state.

- Run-Test/Idle
  In the Run-Test/Idle state, the IC is put in a test mode only when certain instructions such as a built-in self test (BIST) instruction are present. For instructions that do not cause any activities in this state, all test data registers selected by the current instruction retain their previous states.

  The controller remains in this state while TMS is held low. When TMS is high, the controller moves to the Select-DR-Scan state.

- Select-DR-Scan
  This is a temporary controller state. Here, the IC does not execute any specific functions.

  If TMS is held low when the controller is in this state, then the controller moves to the Capture-DR state. If TMS is held high, the controller moves to the Select-IR-Scan state in the instruction column.

- Select-IR-Scan
  This is a temporary controller state. Here, the IC does not execute any specific functions.

  If TMS is held low when the controller is in this state, then the controller moves to the Capture-IR state. If TMS is held high, the controller returns to the Test-Logic-Reset state.

- Capture-DR
  In this controller state, if the test data register selected by the current instruction on the rising edge of TCK has parallel inputs, then data can be parallel-loaded into the shift portion of the data register. If the test data register does not have parallel inputs, or if data need not be loaded into the selected data register, then the data register retains its previous state.

  If TMS is held low while the controller is in this state, the controller moves to the Shift-DR state. If TMS is held high, the controller moves to the Exit1-DR state.

- Shift-DR
  In this controller state, the test data register connected between TDI and TDO shifts data one stage forward towards its serial output.

  When the controller is in this state, then it remains in the Shift-DR state if TMS is held low, or moves to the Exit1-DR state if TMS is held high.

- Exit 1-DR
This is a temporary controller state.

If TMS is held low when the controller is in this state, the controller moves to the Pause-DR state. If TMS is held high, the controller moves to the Update-DR state.

- Pause-DR
This state allows the shifting of the data register selected by the instruction register to be temporarily suspended. Both the instruction register and the data register retain their current states.

When the controller is in this state, then it remains in the Pause-DR state if TMS is held low, or moves to the Exit2-DR state if TMS is held high.

- Exit 2-DR
This is a temporary controller state.

When the controller is in this state, then it returns to the Shift-DR state if TMS is held low, or moves on to the Update-DR state if TMS is held high.

- Update-DR
In this state, data is latched, on the falling edge of TCK, onto the parallel outputs of the data registers from the shift register path. The data held at the parallel output does not change while data is shifted in the associated shift register path.

When the controller is in this state, it moves to either the Run-Test/Idle state if TMS is held low, or the Select-DR-Scan state if TMS is held high.

- Capture-IR
In this state, data is parallel-loaded into the instruction register. The two least significant bits are assigned the values "01". The higher-order bits of the instruction register can receive any design specific values. The Capture-IR state is used for testing the instruction register. Faults in the instruction register, if any exist, may be detected by shifting out the data loaded in it.

When the controller is in this state, it moves to either the Shift-IR state if TMS is low, or the Exit1-IR state if TMS is high.

- Shift-IR
In this state, the instruction register is connected between TDI and TDO and shifts the captured data toward its serial output on the rising edge of TCK.

When the controller is in this state, it remains in the Shift-IR state if TMS is low, or moves to the Exit1-IR state if TMS is high.

- Exit 1-IR
  This is a temporary controller state.

  When the controller is in this state, it moves to either the Pause-IR state if TMS is held low, or the Update-IR state if TMS is held high.

- Pause-IR
  This state allows the shifting of the instruction register to be temporarily suspended. Both the instruction register and the data register retain their current states.

  When the controller is in this state, it remains in the Pause-IR state if TMS is held low, or moves to the Exit2-IR state if TMS is held high.

- Exit 2-IR
  This is a temporary controller state.

  When the controller is in this state, it moves to either the Shift-IR state if TMS is held low, or the Update-IR state if TMS is held high.

- Update-IR
  This state allows the instruction previously shifted into the instruction register to be output in parallel on the rising edge of TCK. Then it becomes the current instruction, setting a new operational mode.

  When the controller is in this state, it moves to either the Run-Test/Idle state if TMS is low, or the Select-DR-Scan state if TMS is high.

Table 6.3.2 shows the boundary scan order of the processor signals.

Table 6.3.2 TMP1942 JTAG Boundary-Scan Ordering

[TDI]	1:P50	2: P51	3:P52	4: P53	5:P54	6:P55
7: P56	8: P57	9: P60	10: P61	11: P 62	12: P63	13: P64
14: P65	15: P66	16: P67	17: P00	18:P01	19: P02	20: P03
21: P04	22: P05	23:P06	24: P07	25: P10	26: P11	27: P12
28: P13	29: P14	30: P15	31: P16	32: P17	33: P20	34: P21
35: P22	36: P23	37: P24	38: P25	39: P26	40:P27	41: ALE
42: BW1	43: P30	44: P31	45: P32	46: P33	47: P34	48: P35
49: P36	50: P37	51:P40	52:P41	53:P42	54:P43	55:P44
56: P90	57: P91	58: P92	59: P93	60: P94	61: P95	62: P96
63: P97	64: PA0	65: PA1	66: PA2	67: PA3	68: PA4	69: PA5
70: PA6	71: PA7	72:RSTPUP	73: PC0	74: PC1	75: PC2	76:PC3
77:PC4	78:PC5	79: PC6	80:PC7	81:PF0	82: PF1	83: PF2
84: PF3	85: PF4	86: PF5	87: PF6	88:TEST1	89:RESET	90:PD6
91:PD7	92:NMI	93:BW0	94:PB0	95:PB1	96:PB2	97:PB3
98:PB4	99:PB5	100:PB6	101:PB7	102:PD0	103:PD1	104:PD2
105:PD3	106:PD4	107:PD5	108:PE0	109:PE1	110:PE2	111:PE3
112:PE4	113:PE5	114:PE6	115:PE7	[TDO]:		

6.4 Instructions for JTAG

This section defines the instructions supplied and the operations that occur in response to those instructions.

6.4.1 The EXTEST Instruction

This instruction is used for external interconnect test, and targets the boundary scan register between TDI and TDO. The EXTEST instruction permits BSR cells at output pins to shift out test patterns in the Update-DR state and those at input pins to capture test results in the Capture-DR state.

Typically, before EXTEST is executed, the initialization pattern is first shifted into the boundary scan register using the SAMPLE/PRELOAD instruction. In the Update-DR state, the boundary scan register loaded with the initialization pattern causes known data to be driven immediately from the IC onto its external interconnects. This eliminates the possibility of bus conflicts damaging the IC outputs. The flow of data through the boundary scan register while the EXTEST instruction is selected is shown in Figure 6.4.1, which follows:
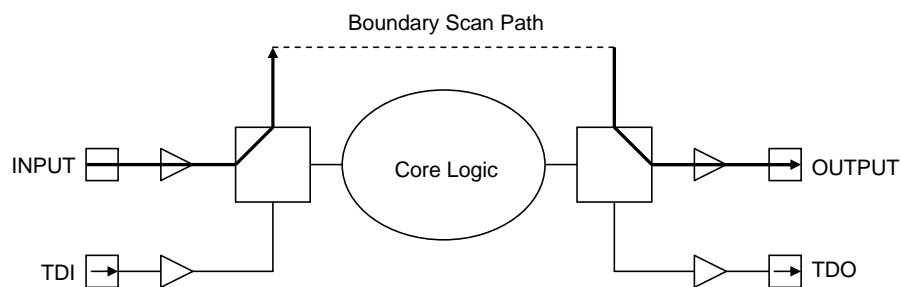
Figure 6.4.1 Test Data Flow While the EXTEST Instruction is Selected

The following steps describe the basic test algorithm of an external interconnect test.

1. Initialize the TAP controller to the Test-Logic-Reset state.

2. Load the instruction register with SAMPLE/PRELOAD. This causes the boundary scan register to be connected between TDI and TDO.

3. Initialize the boundary scan register by shifting in determinate data.

4. Then, load the initial test data into the boundary scan register.

5. Load the instruction register with EXTEST.

6. Capture the data applied to the input pin into the boundary scan register.

7. Shift out the captured data while simultaneously shifting in the next test pattern.

8. Read out the data in the boundary scan register onto the output pin.

Steps 6 to 8 are repeated for each test pattern.

6.4.2 The SAMPLE/PRELOAD Instruction

This instruction targets the boundary scan register between TDI and TDO. As the instruction's name implies, two functions are performed through use of the SAMPLE/ PRELOAD instruction.

- SAMPLE allows the input and output pads of an IC to be monitored. While it does so, it does not disconnect the system logic from the IC pins. The SAMPLE function occurs in the Capture-DR controller state. An example application of SAMPLE is to take a snapshot of the activity of the IC's I/O pins so as to verify the interaction between ICs during normal functional operation. The flow of data for the SAMPLE phase of the SAMPLE/PRELOAD instruction is shown in Figure 6.4.2.
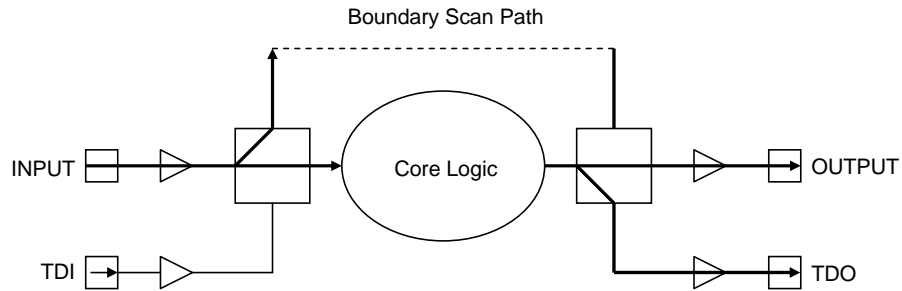
Figure 6.4.2 Test Data Flow While SAMPLE is Selected

- PRELOAD allows the boundary scan register to be initialized before another instruction is selected. For example, prior to selection of the EXTEST instruction, initialization data is shifted into the boundary scan register using PRELOAD as described in the previous subsection. PRELOAD permits shifting of the boundary scan register without interfering with the normal operation of the system logic. The flow of data for the PRELOAD phase of the SAMPLE/PRELOAD instruction is shown in Figure 6.4.3.
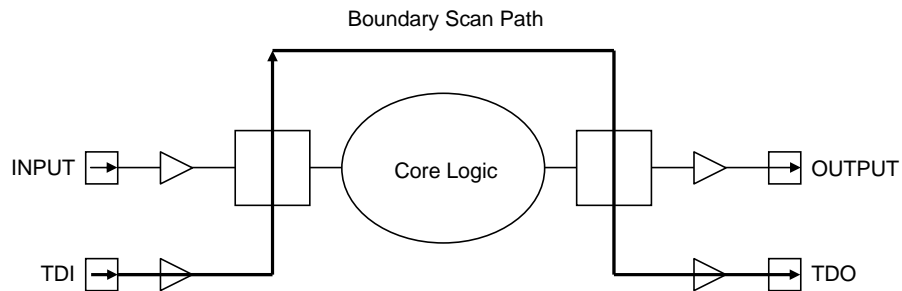
Figure 6.4.3 Test Data Flow While PRELOAD is Selected

6.4.3 The BYPASS Instruction

This instruction targets the bypass register between JTDI and JTDO. The bypass register provides a minimum length serial path through the IC (or between JTDI and JTDO) when the IC is not required for the current test. The BYPASS instruction does not cause interference to the normal operation of the on-chip system logic. The flow of data through the bypass register while the BYPASS instruction is selected is shown in Figure 6.4.4.
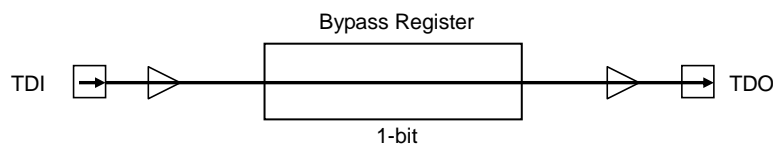
Figure 6.4.4 Test Data Flow While the Bypass Instruction is Selected

6.5 Note

This section describes details of JTAG boundary-scan operation that are specific to the processor.

- The DAOUT0, 1, 2, **X2**, and **X1** signal pads do not support JTAG.

- Reset for JTAG

(1) JTAG circuit is initialized by $\overline{\text{TRST}}$ assertion. And then deassert $\overline{\text{TRST}}$.

(2) At input to **TMS** = 1 and asserted for more 5 TCK cycles.

7   I/O Port Equivalent-Circuit Diagrams

- How to read circuit diagrams

  The circuit diagrams in this chapter are drawn using the same gate symbols as for the 74HCxx Series standard CMOS logic ICs.

  The signal named STOP has a unique function. This signal goes active-high if the CPU sets the HALT bit when the STBY[1:0] field in the SYSCR2 register is programmed to 01 (i.e., STOP mode) and the Drive Enable (FRVE) bit in the same register is cleared. If the DRVE bit is set, the STOP signal remains inactive (at logic 0).

- The input protection circuit has a resistor in the range of several tens to several hundreds of ohms.

■ P0(D0 to D7 / AD0 to AD7), P1(D8 to D15 / AD8 to AD15, A8 to A15), P2(A16 to A23, A0 to A7), P92 to P97, PA0 to PA6, PB0 to PB6, PC0 to PC5, PC7, PD0, PD1, PD4, PE1, PE4, PE6, PE7, PF3, PF6
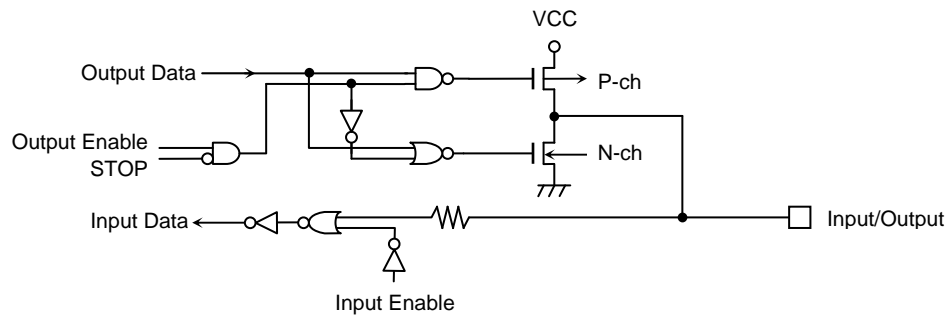
■ P30($\overline{RD}$), P31($\overline{WR}$), DCLK, PCST3 to PCST0, SDAO / TPC, TDO
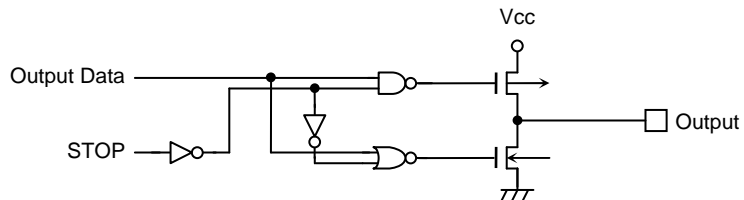
■ P32 to P36, P40 to P43

■ P5 (AN0 to AN7)

Analog Input
Channel Select

Analog Input

Input Data

Input Enable

Input

■ P6 (AN8 to AN15)

Vcc

Programmable
Pull-up Resistor

Analog Input
Channel Select

Analog Input

Pull-up Control

Input Data

Input Enable

Input
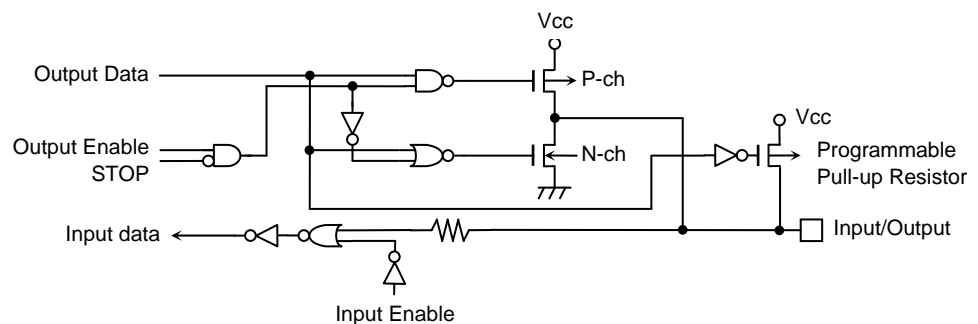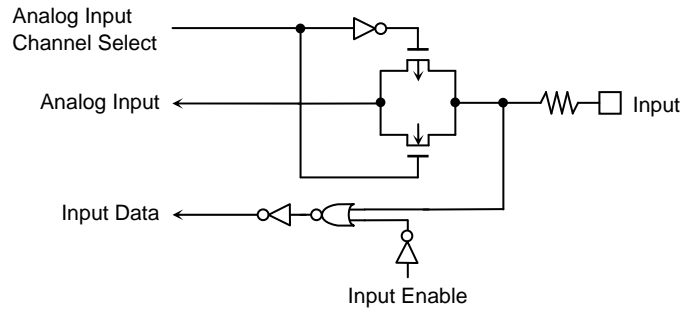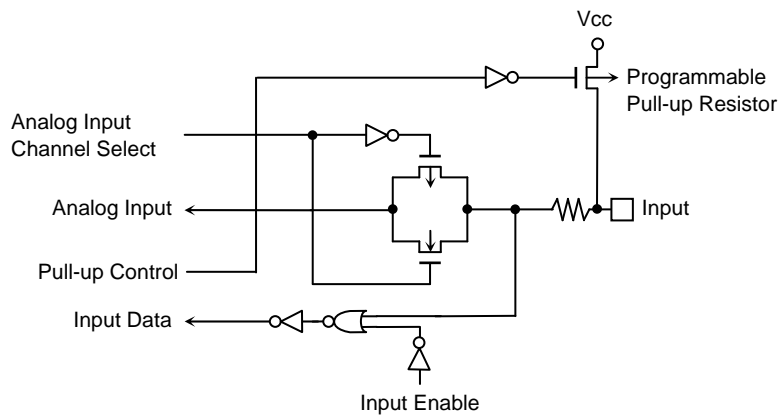
■ PD2, PD3, PD5, PE0, PE2, PE3, PE5, PF0, PF2, PF4, PF5

Vcc

Output Data

Open-Drain
Output Enable

Output Enable
STOP

Input Data

Input Enable

P-ch

N-ch

Input/Output

■ P90, P91, PA7, PB7, PC6, PF1

Vcc

Vcc

Output Data

Output Enable
STOP

Pull-up Control

Input Data

Input Enable

P-ch

N-ch

Programmable
Pull-up Resistor

Input/Output
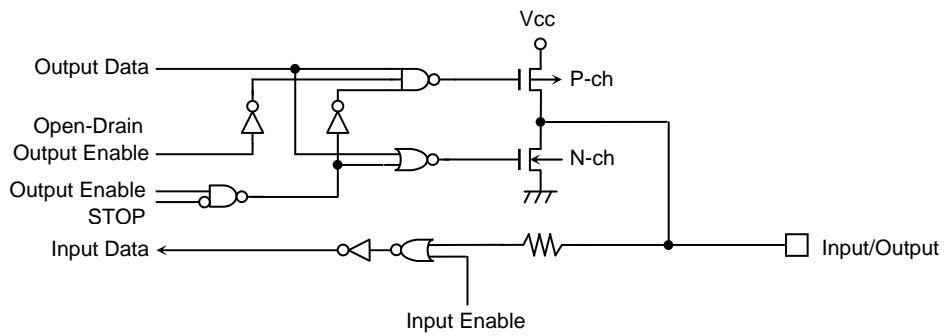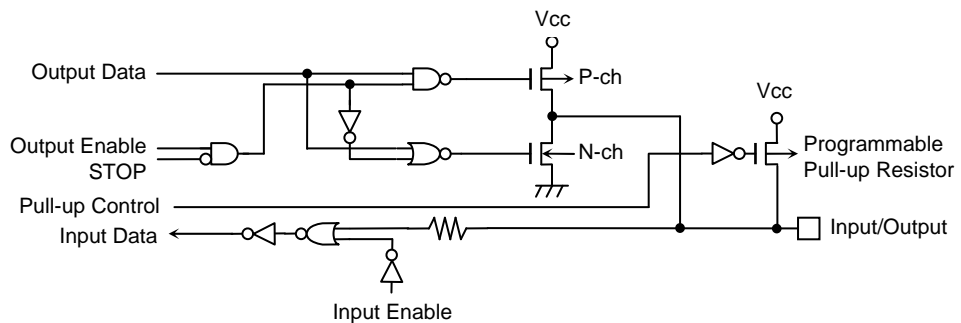
■ PD6 (XT1), PD7 (XT2)
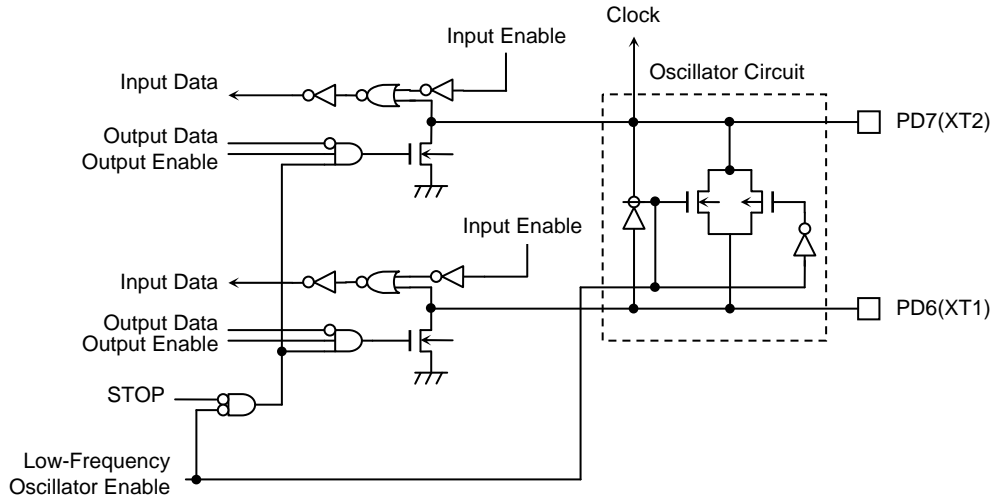
■ $\overline{\text{NMI}}$, BW0 to BW1, $\overline{\text{PLLOFF}}$, RSTPUP
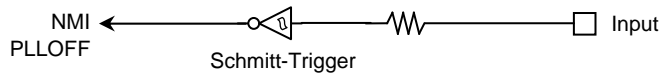
■ $\overline{\text{RESET}}$
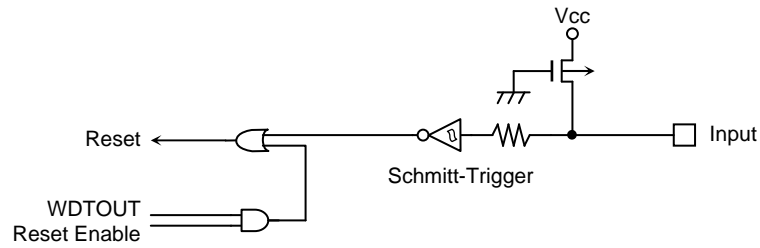
■ $\overline{\text{DRESET}}$, $\overline{\text{DBGE}}$, SDI/$\overline{\text{DINT}}$, TCK, TMS, TDI
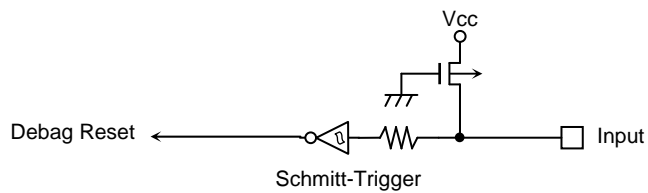
■ $\overline{\text{TRST}}$
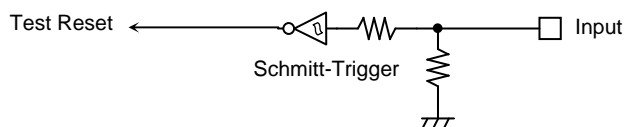
■ X1, X2
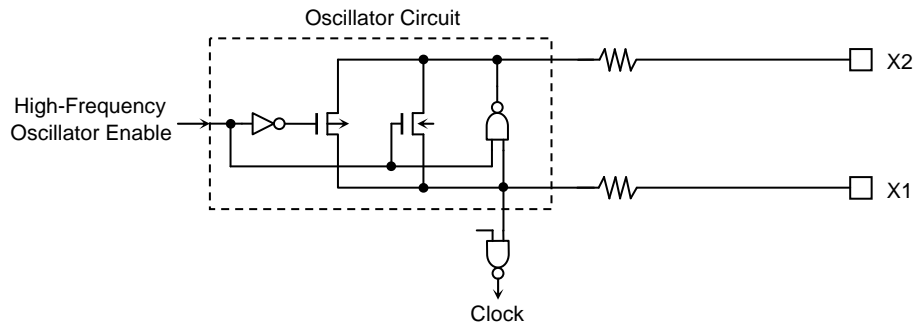
■ VREFH, VREFL
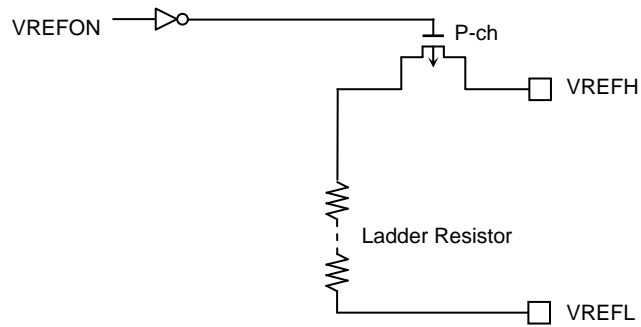
8. Notations, Precautions and Restrictions

8.1 Notations and Terms

(1) I/O register fields are often referred to as <register_mnemonic>.<field_name> for the interest of brevity. For example, TRUN.T0RUN means the T0RUN bit in the TRUN register.

(2) fc, fs, fsys, state

fosc: Clock supplied from the X1 and X2 pins

fpll: Clock generated by the on-chip PLL

fc: Clock selected by the $\overline{\text{PLLOFF}}$ pin

fs: Clock supplied from the XT1 and XT2 pins

fgear: Clock selected by the SYSCR1.GEAR[1:0] bits

fsys: Clock selected by the SYSCR1.SYSCK bit

The fsys cycle is referred to as a state.

In addition, the clock selected by the SYSCR1.FPSEL bit and the prescaler clock source selected by the SYSCR0.PRCK[1:0] bits are referred to as fperiph and φT0 respectively.

8.2 Precautions and Restrictions

(1) Processor Revision Identifier

The Process Revision Identifier (PRId) register in the TX19 core of the TMP1942 contains 0x0000_2C91.

(2) BW0 to BW1 Pins

The BW0 and BW1 pins must be connected to the DVCC pin to ensure that their signal levels do not fluctuate during chip operation.

(3) Oscillator Warm-Up Counter

If an external crystal is utilized, an interrupt signal programmed to bring the TMP1942 out of STOP mode triggers the on-chip warm-up counter. The system clock is not supplied to the on-chip logic until the warm-up counter expires.

(4) Programmable Pull-up Resistors

When port pins are configured as input ports, the integrated pull-up resistors can be enabled and disabled under software control. The pull-up resistors are not programmable when port pins are configured as output ports.

The relevant port registers are programmed with the data resistor.

(5) External Bus Mastership

The pin states while the bus is granted to an external device are described in Chapter 7, I/O Ports.

(6) Watchdog Timer (WDT)

Upon reset, the WDT is enabled. If the watchdog timer function is not required, it must be disabled after reset. When relevant pins are configured as bus arbitration signals, the I/O peripherals including the WDT can operate during external bus mastership.

(7) A/D Converter (ADC)

The ladder resistor network between the VREFH and VREFL pins can be disconnected under software control. This helps to reduce power dissipation, for example, in STOP mode.

(8) Undefined Bits in I/O Registers

Undefined I/O register bits are read as undefined states. Therefore, software must be coded without relying on the states of any undefined bits.

(9) Notations, Precautions and Restrictions

Overflow Exception #1

Problem:

When an overflow exception is taken, the EPC register might contain an incorrect return address, pointing to the instruction immediately following the one that caused an overflow.

The restart location in the EPC register should be the address of the arithmetic instruction that caused the exception, rather than the following instruction.
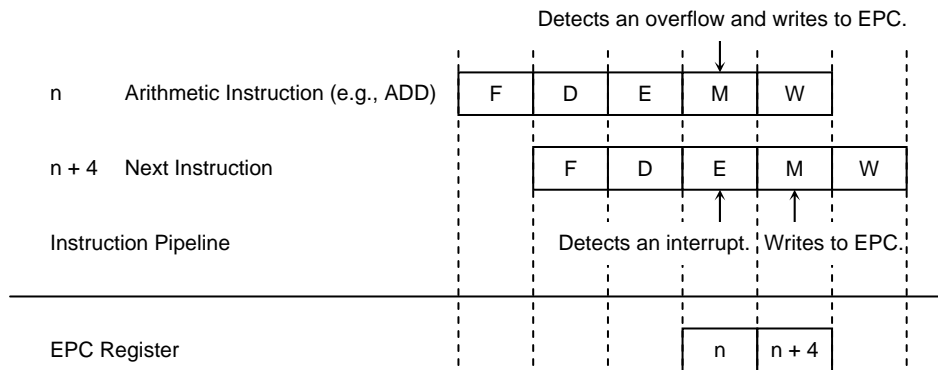
In the above example, the processor writes address n to the EPC register upon detection of an overflow. However, executing the next instruction generates an interrupt at the same time, causing the processor to rewrite the EPC register with address n+4 in the next cycle.

- **Problem-Causing Situation:**
  A) Software uses the ADD, ADDI or SUB instruction in the 32-bit ISA.
  B) The ADD, ADDI or SUB instruction causes an overflow.
  C) Another exception is requested simultaneously with the overflow.

This problem occurs when all of these conditions are true.

Workarounds:

- Before returning from the overflow exception handler, determine whether the instruction pointed to by the EPC register caused an overflow.

- Make sure that two arithmetic instructions will not appear consecutively.

- Disable interrupts prior to arithmetic instructions.
  You should always use one of these workarounds to avoid this problem.

Note: Toshiba's compiler uses no instructions that could cause an overflow. Therefore, since condition c) above never becomes true, this problem does not occur.

Overflow Exception #2

Problem:

If an overflow exception caused a jump to the exception handler and the first instruction in that exception handler caused another exception, the EPC register should point to the address of the first instruction in the exception handler. However, the EPC register might contain the address that caused the overflow exception.

- **Problem-Causing Situation:**

  When, with the instruction pipeline full, an overflow exception was taken at the following sequence of instructions and then the first instruction in the overflow exception handler causes another exception

  ADD, ADDI or SUB   <= # Instruction that causes an overflow

  Jump or branch instruction <= # Instruction with a delay slot

  Delay slot

  > Note: Toshiba's compiler uses no instructions that could cause an overflow. Therefore, this problem does not occur.

Workaround:

Don't place a jump or branch instruction immediately following an instruction that could cause an overflow (ADD, ADDI or SUB).

LWL and LWR Instructions

Problem:

The LWL or LWR instruction might provide incorrect results.

- **Problem-Causing Situation #1:**

  a. The destination of a load instruction (LB, LBU, LH, LHU, LW, LWL or LWR) is identical to that of the LWL or LWR instruction.

  b. The instruction pipeline is full. (The load instruction and the LWL or LWR instruction will be executed consecutively.)

  c. The DMAC is programmed for data cache snooping. Once the load instruction is executed, the DMAC initiates a DMA transaction. After it has been serviced, the LWLor LWR instruction is executed.

  This problem occurs when all of these conditions are true.

- **Problem-Causing Situation #2:**

  a. The destination of a load instruction (LB, LBU, LH, LHU, LW, LWL or LWR) is identical to that of the LWL or LWR instruction.

  b. The Doze or Halt bit in the Config register is set to 1 immediately before the load instruction.

  c. The instruction pipeline is full. (The load instruction and the LWL or LWR instruction will be executed consecutively.)

  d. After the load instruction is executed, the processor is put in the STOP, SLEEP or IDLE mode.

  e. After an interrupt signaling brings the processor out of the STOP, SLEEP or IDLE mode, the LWL or LWR instruction is executed.

Note:	This applies to the case in which an interrupt signaling does not generate an interrupt upon exit from STOP, SLEEP or IDLE mode. In other words, either the IEc bit in the Status register is cleared (interrupts disabled), or if the IEc bit is set, the priority level of the incoming interrupt signaling is lower than the mask level programmed in the CMask field in the Status register. (Exit from STOP, SLEEP or IDLE mode can be accomplished even with such settings.)

  This problem occurs when all of these conditions are true.

Workarounds:

To use the LWL or LWR instruction,

1) Place a NOP between a load instruction and the LWL or LWR instruction, or

2) Disable the data cache snooping of the DMAC before the LWL or LWR instruction is executed. Also, don't put the processor in STOP, SLEEP or IDLE mode before the LWL or LWR instruction is executed.

Overflow Exception When a DSU Probe Is Used

Problem:

It looks as if an overflow exception caused a jump to the reset and nonmaskable exception vector address (0xBFC0_0000).

- **Problem-Causing Situation:**

    When an overflow exception occurs, with the processor connected to a DSU probe

Note: Toshiba's compiler uses no instructions that could cause an overflow. Therefore, this problem does not occur.

Workaround:

Don't place a jump or branch instruction immediately following an instruction that could cause an overflow (ADD, ADDI or SUB).

IDLE (Doze) Mode

Problem:

A deadlock might occur when returning to normal operating mode from IDLE (Doze) mode.

- **Problem-Causing Situation:**

    When the DMAC initiates a DMA transaction with snooping enabled after the Doze bit in the Config register is set and before the CPU clock stops.

Workaround:

If snooping is enabled, stop the DMAC before putting the processor in IDLE (Doze) mode.