CMOS 16-Bit Microcontrollers

TMP95CU54AF

1. Outline and Features

The TMP95CU54A is a high-speed 16-bit microcontroller designed for the control of various mid- to large-scale equipment.

The TMP95CU54A comes in a 100-pin flat package.

Listed below are the features of the TMP95CU54A.

(1) High-speed 16-bit CPU (900/H CPU)
- Instruction mnemonics are upward-compatible with the TLCS-90/900
- 16 Mbytes of linear address space
- General-purpose registers and register banks
- 16-bit multiplication and division instructions; bit transfer and arithmetic instructions
- Micro DMA : Four-channels (667 ns/2 bytes at 24 MHz)

(2) Minimum instruction execution time : 167 ns (at 24 MHz)

(3) Built-in RAM : 3 Kbytes
Built-in ROM : 96 Kbytes

(4) External memory expansion
- Expandable up to 16 Mbytes (shared program/data area)
- External data bus width select pin (AM8/$\overline{16}$)
- Can simultaneously support 8/16-bit width external data bus
  … Dynamic data bus sizing

(5) 8-bit timers : 8 channels
- With event counter function : 2 channels

(6) 16-bit timer/event counter : 2 channels

RESTRICTIONS ON PRODUCT USE 030619EBP

- The information contained herein is subject to change without notice.
- The information contained herein is presented only as a guide for the applications of our products. No responsibility is assumed by TOSHIBA for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patent or patent rights of TOSHIBA or others.
- TOSHIBA is continually working to improve the quality and reliability of its products. Nevertheless, semiconductor devices in general can malfunction or fail due to their inherent electrical sensitivity and vulnerability to physical stress. It is the responsibility of the buyer, when utilizing TOSHIBA products, to comply with the standards of safety in making a safe design for the entire system, and to avoid situations in which a malfunction or failure of such TOSHIBA products could cause loss of human life, bodily injury or damage to property.
  In developing your designs, please ensure that TOSHIBA products are used within specified operating ranges as set forth in the most recent TOSHIBA products specifications. Also, please keep in mind the precautions and conditions set forth in the "Handling Guide for Semiconductor Devices," or "TOSHIBA Semiconductor Reliability Handbook" etc..
- The TOSHIBA products listed in this document are intended for usage in general electronics applications (computer, personal equipment, office equipment, measuring equipment, industrial robotics, domestic appliances, etc.). These TOSHIBA products are neither intended nor warranted for usage in equipment that requires extraordinarily high quality and/or reliability or a malfunctionor failure of which may cause loss of human life or bodily injury ("Unintended Usage"). Unintended Usage include atomic energy control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, combustion control instruments, medical instruments, all types of safety devices, etc.. Unintended Usage of TOSHIBA products listed in this document shall be made at the customer's own risk.
- The products described in this document are subject to the foreign exchange and foreign trade laws.
- TOSHIBA products should not be embedded to the downstream products which are prohibited to be produced and sold, under any law and regulations.
- For a discussion of how the reliability of microcontrollers can be predicted, please refer to Section 1.3 of the chapter entitled Quality and Reliability Assurance/Handling Precautions.

(7) General-purpose serial interface : 2 channels

(8) Serial Expansion Interface : 1 channel

(9) CAN Controller : 1 channel

(10) 10-bit AD converter : 8 channels

(11) Watchdog timer

(12) Bus width/wait controller : 4 blocks

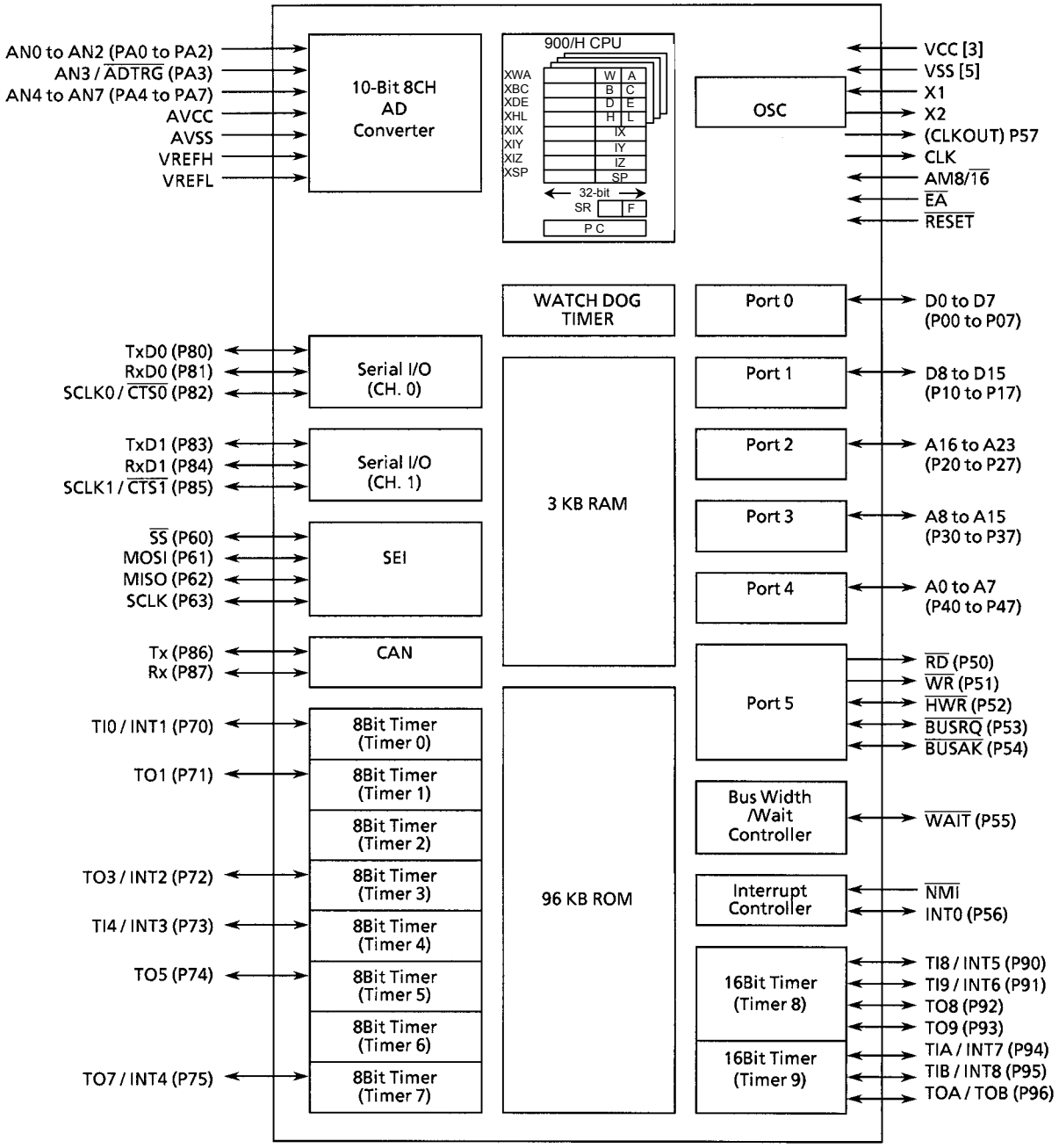
(13) Interrupts : 49 interrupts
- 9 CPU interrupts : Software interrupt instruction and illegal instruction
- 30 internal interrupts : Seven selectable priority levels
- 10 external interrupts : Seven selectable priority levels

(14) Input/output ports : 81 pins

(15) Standby mode
- Four HALT modes : RUN, IDLE2, IDLE1, STOP

(16) Operating voltage
- V_{CC} = 4.5 to 5.5 V

(17) Package
- P-LQFP100-1414-0.50D



Note: After a reset, functions in parentheses ( ) are selected for the shared pins.

Figure 1.1 TMP95CU54A Block Diagram

2. Pin Assignment and Pin Functions

This section shows the TMP95CU54A pin assignment, and the names and an outline of the functions of the input/output pins.

2.1 Pin Assignment Diagram

Figure 2.1.1 is the pin assignment diagram for the TMP95CU54A.

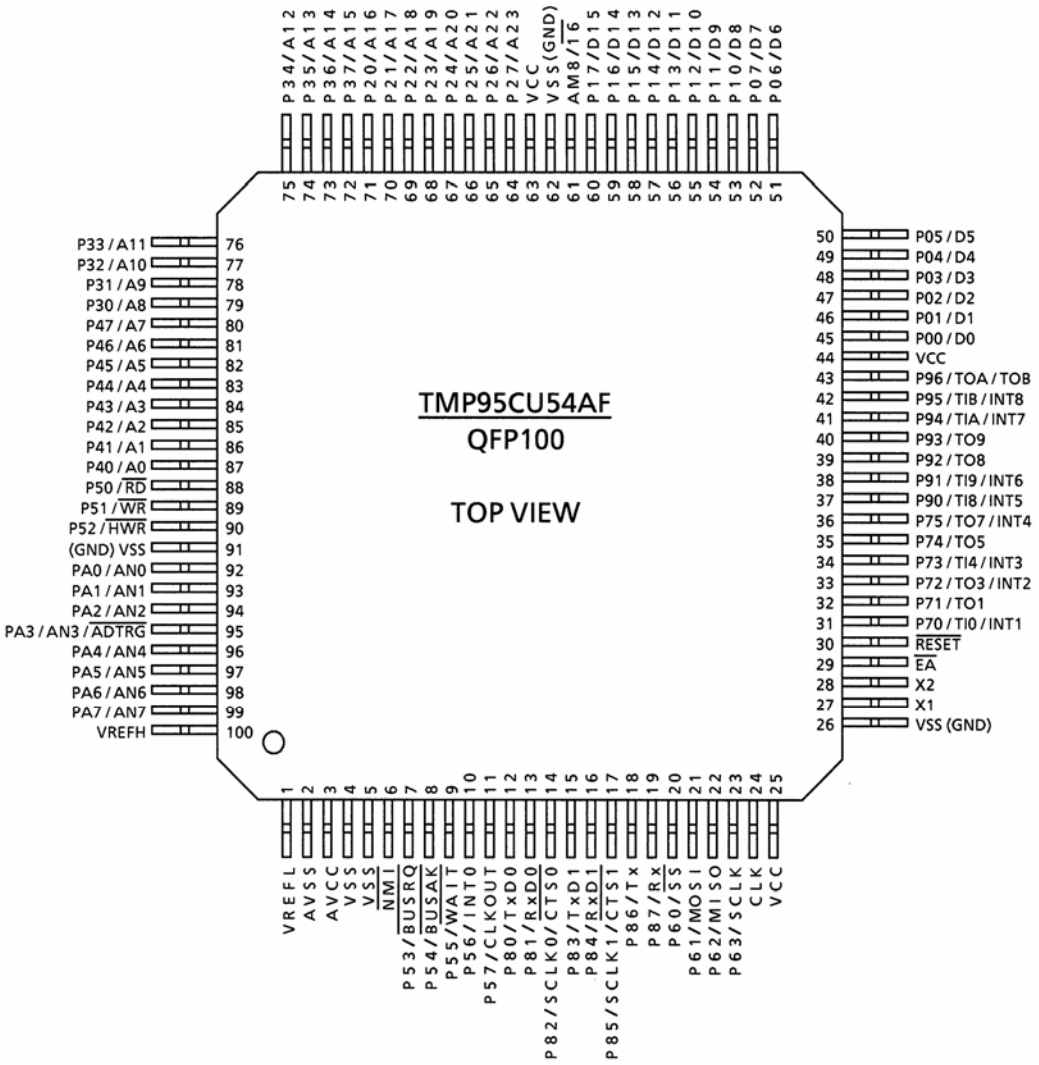


Figure 2.1.1 Pin Assignment Diagram (100-Pin LQFP)

2.2 Pin Names and Functions

Table 2.2.1 shows the names and functions of the input/output pins.

Table 2.2.1 Pin Names and Functions (1/4)

Pin Name	Number of Pins	Input/Output	Function
P00 to P07	8	Input/output	Port 0: I/O port. Input or output specifiable in units of bits
/ D0 to D7		Input/output	Data: Data bus 0 to 7
P10 to P17	8	Input/output	Port 1: I/O port. Input or output specifiable in units of bits
/ D8 to D15		Input/output	Data: Data bus 8 to 15
P20 to P27	8	Input/output	Port 2: I/O port. Input or output specifiable in units of bits
/ A16 to A23		Output	Address: Address bus 16 to 23
P30	1	Input/output	Port 30: I/O port (with built-in pull-up resistor during input mode.)
/ A8		Output	Address: Address bus 8
P31 to P37	7	Input/output	Port 31 to 37: I/O port. Input or output specifiable in units of bits
/ A9 to A15		Output	Address: Address bus 9 to 15
P40 to P47	8	Input/output	Port 4: I/O port. Input or output specifiable in units of bits
/ A0 to A7		Output	Address: Address bus 0 to 7
P50	1	Output	Port 50: Output-only port
/ $\overline{RD}$		Output	Read: Outputs strobe signal to read external memory (setting P5 <P50> = 0 and P5FC <P50F> = 1 outputs strobe signal at all read timings)
P51	1	Output	Port 51: Output-only port.
/ $\overline{WR}$		Output	Write: Outputs strobe signal to write data on pins D0 to D7
P52	1	Input/output	Port 52: I/O port (with built-in pull-up resistor)
/ $\overline{HWR}$		Output	Upper write: Outputs strobe signal to write data on pins D8 to D15
P53	1	Input/output	Port 53: I/O port (with built-in pull-up resistor)
/ $\overline{BUSRQ}$		Input	Bus request: Input pin to request external bus release
P54	1	Input/output	Port 54: I/O port (with built-in pull-up resistor)
/ $\overline{BUSAK}$		Output	Bus acknowledge: Output pin to acknowledge that CPU received $\overline{BUSRQ}$ and released external bus.
P55	1	Input/output	Port 55: I/O port (with built-in pull up resistor)
/ $\overline{WAIT}$		Input	Wait: Bus wait request pin for CPU (Effective when 1 WAIT + N mode, or 0 + N WAIT mode. Set using bus width/wait control register.)
P56	1	Input/output	Port 56: I/O port (with built-in pull-up resistor)
/ INT0		Input	Interrupt request pin 0: Interrupt request pin with programmable level/rising edge.

Table 2.2.1 Pin Names and Functions (2/4)

Pin Name	Number of Pins	Input/Output	Function
P57	1	Output	Port 57: Output-only port (with built-in pull-up resistor)
/ CLKOUT		Output	CLKOUT output: Outputs external clock divided by 6. Pulled up during reset.
P60	1	Input/output	Port 60: I/O port
/ $\overline{SS}$		Input	SEI slave select input
P61	1	Input/output	Port 61: I/O port
/ MOSI		Input/output	SEI master output, slave input
P62	1	Input/output	Port 62: I/O port
/ MISO		Input/output	SEI master input, slave output
P63	1	Input/output	Port 63: I/O port
/ SCLK		Input/output	SEI clock input/output
P70	1	Input/output	Port 70: I/O port
/ TI0		Input	Timer input 0: Input pin for timer 0
/ INT1		Input	Interrupt request pin 1: Rising-edge interrupt request pin ↑
P71	1	Input/output	Port 71: I/O port.
/ TO1		Output	Timer output 1: Output pin for timer 0 or 1
P72	1	Input/output	Port 72: I/O port
/ TO3		Output	Timer output 3: Output pin for timer 2 or 3
/ INT2		Input	Interrupt request pin 2: Rising-edge interrupt request pin ↑
P73	1	Input/output	Port 73: I/O port
/ TI4		Input	Timer input 4: Input pin for timer 4
/ INT3		Input	Interrupt request pin 3: Rising-edge interrupt request pin ↑
P74	1	Input/output	Port 74: I/O port
/ TO5		Output	Timer output 5: Output pin for timer 4 or 5
P75	1	Input/output	Port 75: I/O port
/ TO7		Output	Timer output 7: Output pin for timer 6 or 7
/ INT4		Input	Interrupt request pin 4: Rising-edge interrupt request pin ↑
P80	1	Input/output	Port 80: I/O port (with built-in pull-up resistor)
/ TxD0		Output	Serial transmission data 0
P81	1	Input/output	Port 81: I/O port (with built-in pull-up resistor)
/ RxD0		Input	Serial receive data 0
P82	1	Input/output	Port 82: I/O port (with built-in pull-up resistor)
/ SCLK0		Input/output	Serial clock input/output 0
/ $\overline{CTS0}$		Input	Serial data ready to send 0 (Clear-to-send)

Table 2.2.1 Pin Names and Functions (3/4)

Pin Name	Number of Pins	Input/Output	Function
P83	1	Input/output	Port 83: I/O port (with built-in pull-up resistor)
/ TxD1		Output	Serial transmission data 1
P84	1	Input/output	Port 84: I/O port (with built-in pull-up resistor)
/ RxD1		Input	Serial receive data 1
P85	1	Input/output	Port 85: I/O port (with built-in pull-up resistor)
/ SCLK1		Input/output	Serial clock input/output 1
/ $\overline{\text{CTS1}}$		Input	Serial data ready to send 1 (Clear-to-send)
P86	1	Input/output	Port 86: I/O port (with built-in pull-up resistor)
/ Tx		Output	CAN transmission data
P87	1	Input/output	Port 87: I/O port (with built-in pull-up resistor)
/ Rx		Input	CAN receive data
P90	1	Input/output	Port 90: I/O port
/ TI8		Input	Timer input 8: Input pin for timer 8
/ INT5		Input	Interrupt request pin 5: Interrupt request pin with programmable rising/falling edge ↑ ↓
P91	1	Input/output	Port 91: I/O port
/ TI9		Input	Timer input 9: Input pin for timer 8
/ INT6		Input	Interrupt request pin 6: Rising edge interrupt request pin ↑
P92	1	Input/output	Port 92: I/O port
/ TO8		Output	Timer output 8: Output pin for timer 8
P93	1	Input/output	Port 93: I/O port
/ TO9		Output	Timer output 9: Output pin for timer 8
P94	1	Input/output	Port 94: I/O port
/ TIA		Input	Timer input A: Input pin for timer 9
/ INT7		Input	Interrupt request pin 7: Interrupt request pin with programmable rising/falling edge ↑ ↓
P95	1	Input/output	Port 95: I/O port
/ TIB		Input	Timer input B: Input pin for timer 9
/ INT8		Input	Interrupt request pin 8: Rising edge interrupt request pin ↑
P96	1	Input/output	Port 96: I/O port
/ TOA		Output	Timer output A: Output pin for timer 9
/ TOB		Output	Timer output B: Output pin for timer 9
PA0 to PA2	3	Input	Port A0 to A2: Input-only port
/ AN0 to AN2		Input	Analog input 0 to 2: AD converter input pins
PA3	1	Input	Port A3: Input-only port
/ AN3		Input	Analog input 3: AD converter input pin
/ $\overline{\text{ADTRG}}$		Input	External start trigger

Table 2.2.1 Pin Names and Functions (4/4)

Pin Name	Number of Pins	Input/Output	Function
PA4 to PA7	4	Input	Port A4 to A7: Input-only port
/ AN4 to AN7		Input	Analog input 4 to 7: AD converter input pins
$\overline{NMI}$	1	Input	Non-maskable interrupt request pin: Interrupt request pin with programmable falling edge or both falling and rising edge
CLK	1	Output	Clock output: Outputs external clock divided by 4. Pulled up during reset.
$\overline{EA}$	1	Input	External access: Connect to VCC.
AM8 / $\overline{16}$	1	Input	Address mode: External data bus width select pin Connect this pin to VCC. Data bus width at external access can be set by bus width/wait control register.
$\overline{RESET}$	1	Input	Reset: Initializes TMP95CU54A (with built-in pull-up resistor)
VREFH	1	Input	Reference voltage input pin for AD converter (high)
VREFL	1	Input	Reference voltage input pin for AD converter (low)
AVCC	1		Power supply pin for AD converter: Connect to power supply
AVSS	1		GND pin for AD converter: Connect to GND
X1 / X2	2	Input/output	Oscillator connecting pin
VCC	3		Power supply pin: Connect all VCC pins to power supply
VSS	5		GND pin: Connect all VSS pins to GND (0 V)

Note: Disconnect the pull-up resistors from pins other than $\overline{RESET}$ pin by software.
P30 is pulled-up during reset and input mode.
P57 and CLK pin are pulled-up only during reset.

3. Operation

The following is a bock-by-block description of the functions and basic operation of the TMP95CU54A.

Notes and restrictions for each block are outlined in "7, Use Precautions and Restrictions" on page 276 of this manual.

3.1 CPU

TMP95CU54A incorporates a high-performance 16-bit CPU (900/H-CPU). For CPU operation, see the section dealing with the TLCS-900/H CPU.

The following describes the unique functions of the CPU used in the TMP95CU54A; these functions are not covered in the TLCS-900/H CPU section.

3.1.1 Reset

When resetting the TMP95CU54A microcontroller, ensure that the power supply voltage is within the operating voltage range, and that the internal high-frequency oscillator has stabilized. Then hold the $\overline{\text{RESET}}$ input to low level for at least 10 system clocks (ten states: 0.83 μs at 24 MHz).

When the reset is accepted, the CPU:

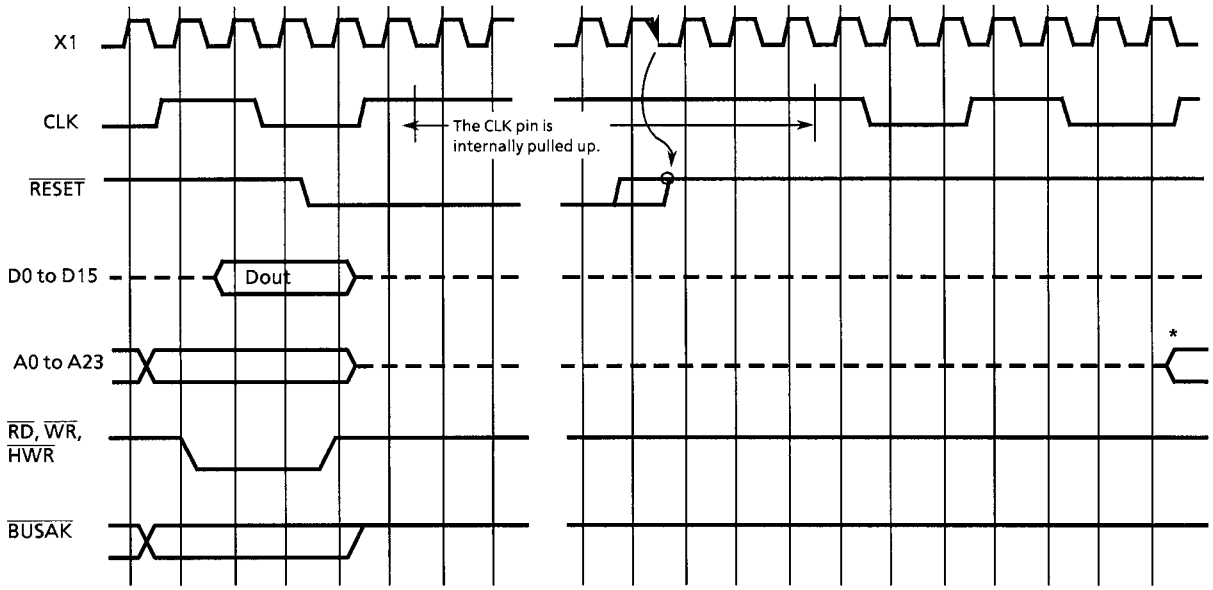
- Sets as follows the program counter (PC) in accordance with the reset vector stored at address FFFF00H - FFFF02H:
  PC (7:0) ← value at FFFF00H address
  PC (15:8) ← value at FFFF01H address
  PC (23:16) ← value at FFFF02H address
- Sets the stack pointer (XSP) to 100H.
- Sets bits <IFF2:0> of the status register (SR) to 111 (sets the interrupt level mask register to level 7).
- Sets the <MAX> bit of the status register to 1 (MAX mode).
  (Note: As this product does not support a MIN mode, do not write 0 to <MAX>.)
- Clears bits <RFP2:0> of the status register to 000 (sets the register bank to 0).

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

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

- Initializes the internal I/O registers.
- Sets the port pins, including the pins that also act as internal I/O, to general-purpose input or output port mode.
- Pulls up the CLK pin to high level.
  (Note: During reset, do not reduce the external voltage level as this can cause malfunction.)
- Pulls up the P30 pin to high level.
  (Note: During reset, do not reduce the external voltage level as this can cause malfunction.)

Figure 3.1.1 shows an example of the basic timing of the reset operation.



* After confirmation that $\overline{RESET}$ = high, A0 to A23 are output at the X1 rising edge of the 10th or 12th clock.

Figure 3.1.1 TMP95CU54A Reset Timing Example

3.1.2 External data bus width selection (AM8/$\overline{16}$ Pin)

Connect the input pin to VCC. After a reset, this pin accesses ROM by the internal 16-bit bus.

The data bus width for an external access depends on the setting in the <B0BUS>, <B1BUS>, <B2BUS>, <B3BUS> or <BEXBUS> bit of the bus width/wait control registers. To access the 16-bit bus, set port 1 to D8 to D15.

3.2 Memory Map

Figure 3.2.1 shows the memory map and the access widths for the CPU addressing modes.

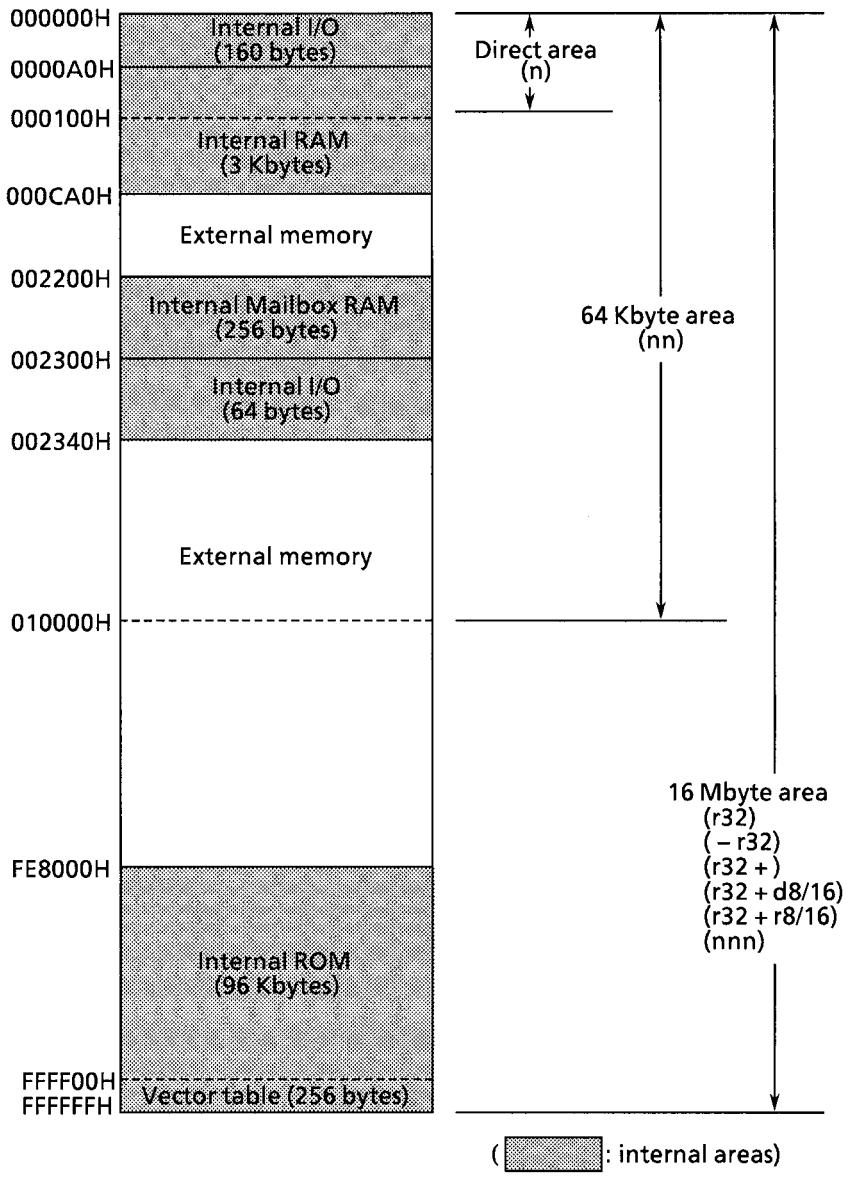


Figure 3.2.1 TMP95CU54A Memory Map

3.3 Interrupts

Interrupts are controlled by the CPU interrupt mask register <IFF2:0> (bits 14 to 12 of the status register) and by the built-in interrupt controller.

The TMP95CU54A has a total of 49 interrupts divided into the following five types:

Interrupts generated by CPU : 9
- Software interrupts : 8
- Illegal instruction : 1

Internal interrupts : 30
- Internal I/O interrupts : 26
- Micro DMA transfer end interrupts : 4

External interrupts : 10
- Interrupts from external pins ($\overline{NMI}$, INT0 to INT8)

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

One of seven (variable) priority levels can be assigned to each maskable interrupt. The priority level of non-maskable interrupts is fixed at 7, the highest level.

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

The CPU compares the priority level of the interrupt with the value of the CPU interrupt mask register <IFF2:0>. If the priority level of the interrupt is higher than the value of the interrupt mask register, the CPU accepts the interrupt. However, software interrupts and illegal instruction interrupts generated by the CPU are processed without comparison with the <IFF2:0> value.

The interrupt mask register <IFF2:0> value can be updated using the value of the EI instruction (executing EI num sets the content of <IFF2:0> to num). For example, specifying EI 3 enables the acceptance of maskable interrupts whose priority level set in the interrupt controller is 3 or higher, and enables the acceptance of non-maskable interrupts. However, if EI or EI 0 is specified, maskable interrupts with a priority level of 1 or higher and non-maskable interrupts are accepted (operationally identical to “EI 1”).

Operationally, the DI instruction (<IFF2:0> is 7) is identical to the EI 7 instruction, but as the priority level of maskable interrupts is 0 to 6, the DI instruction is used to disable maskable interrupts. The EI instruction is valid immediately after execution begins. (With the TLCS-90, the EI instruction is valid after execution of the instruction following the EI instruction.)

In addition to the general-purpose interrupt processing mode described above, the TLCS-900/H interrupts also have a micro DMA processing mode.

Because the CPU transfers data (byte transfer, word transfer, or 4-byte transfer) automatically in micro DMA mode, this mode can be used for speeding up interrupt processing, such as transferring data to I/O.

The TMP95CU54A also has a micro DMA soft start function for requesting micro DMA processing by software rather than by interrupt.

Figure 3.3.1 shows the overall interrupt processing flow.

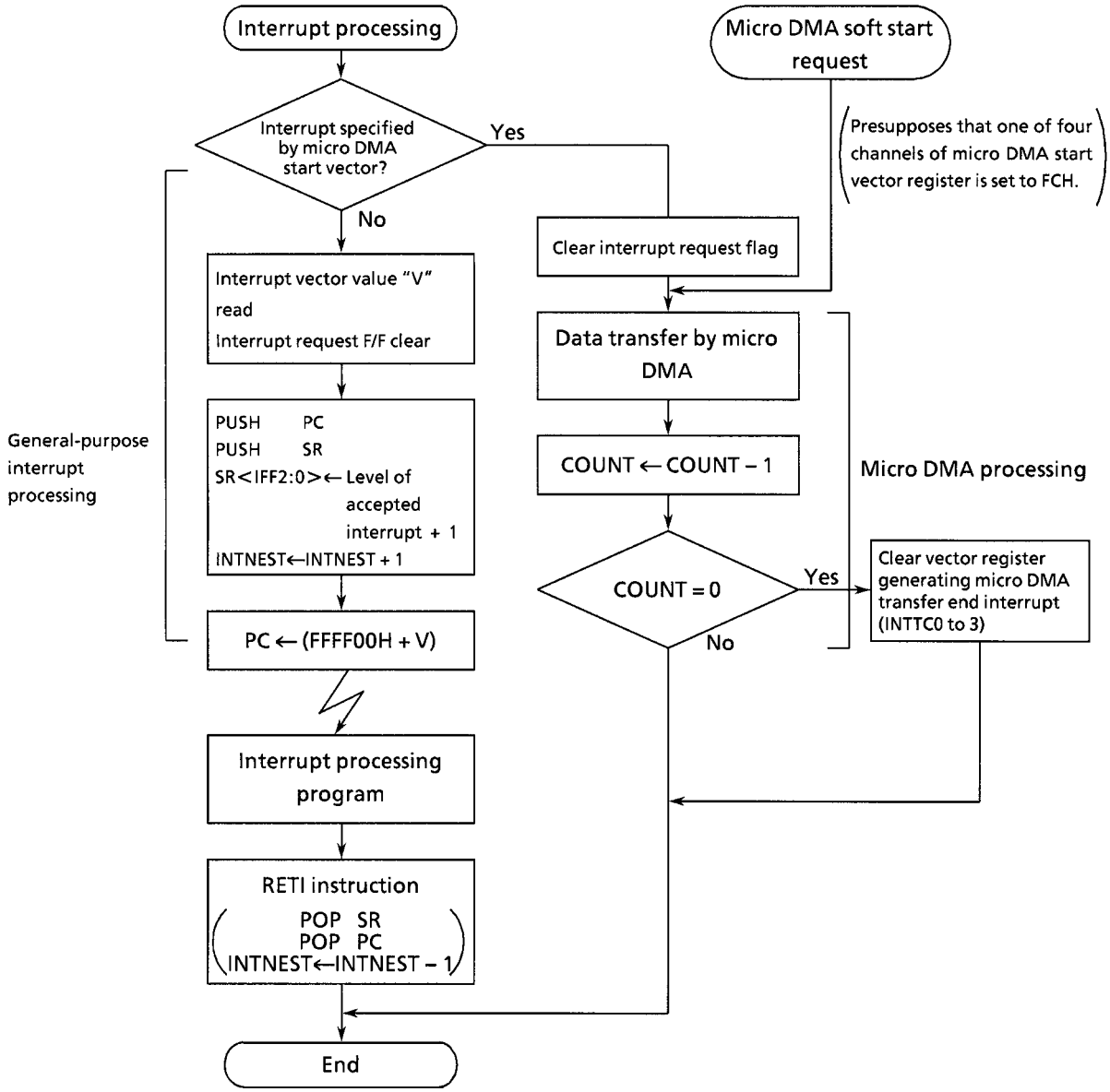


Figure 3.3.1 Interrupt and Micro DMA Processing Flow

3.3.1 General-purpose interrupt processing

When the CPU accepts an interrupt, the CPU performs the following processing. However, in the case of software interrupts and illegal instruction interrupts generated by the CPU, the CPU skips [1] and [3] and executes steps [2], [4], and [5].

[1] The CPU reads the interrupt vector from the interrupt controller. If there are simultaneous interrupts set to the same level, the interrupt controller generates an interrupt vector in accordance with the default priority and clears the interrupt request.
(The default priority is already fixed for each interrupt: the smaller the vector value, the higher the priority level.)

[2] The CPU saves the contents of the program counter (PC) and status register (SR) to the stack area (indicated by XSP).

[3] The CPU sets the value of the CPU's interrupt mask register <IFF2:0> to the received interrupt level incremented by 1. However, if the incremented value level is 7 or higher, the CPU just sets the register to 7.

[4] The CPU increments interrupt nesting counter INTNEST by 1.

[5] The CPU jumps to the address indicated by the data at address FFFF00H + interrupt vector, and starts the interrupt processing routine.

Table 3.3.1 shows the times for the above processing.

Table 3.3.1 Interrupt Processing Times for Bus Widths

Stack Area Bus Width (Bits)	Interrupt Vector Area Bus Width	Number of Interrupt Processing Execution States	Interrupt Processing Time (μs) @ fc = 24 MHz
8	8	28	2.33
	16	24	2.00
16	8	22	1.83
	16	18	1.50

When the CPU has completed the interrupt processing, use the RETI instruction to return to the main routine. This instruction restores the contents of the program counter and status register from the stack, and decrements interrupt nesting counter INTNEST by 1.

Non-maskable interrupts cannot be disabled by program. Maskable interrupts can be enabled or disabled by program. The program can set a priority level for every interrupt source. (Setting the priority level to 0 (or 7) disables the interrupt request.)

If a request is received for an interrupt with a higher priority level than that set in the CPU interrupt mask register <IFF2:0>, the CPU accepts the interrupt. Set the CPU interrupt mask register <IFF2:0> to the received interrupt priority level incremented by 1.

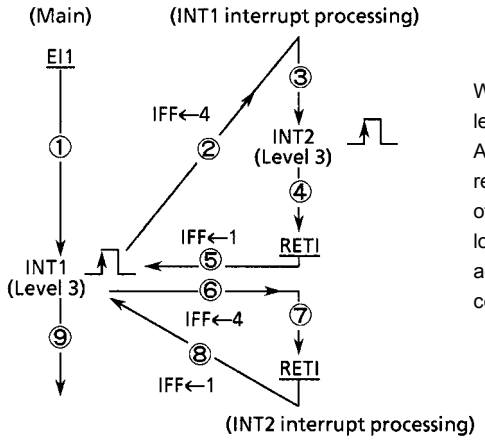
If, during interrupt processing, an interrupt is generated with a higher level than the interrupt being currently processed, or if, during non-maskable interrupt processing, a non-maskable interrupt request is generated from another source, the CPU suspends the current processing routine and accepts the later interrupt. Then, after the CPU has finished processing the later interrupt, the CPU returns to the interrupt it previously suspended and resumes processing.

If the CPU receives a request for another interrupt while already performing processing steps [1] to [5], the second interrupt is sampled immediately after execution of the start instruction for its interrupt processing routine. Specifying DI as the start instruction disables maskable interrupt nesting. (Note: In the 900 and 900/L, sampling is performed before execution of the start instruction.)

After a reset, the interrupt mask register <IFF2:0> is initialized to 111, thus disabling maskable interrupts.

The following steps (1) through (5) show the interrupt processing flow.

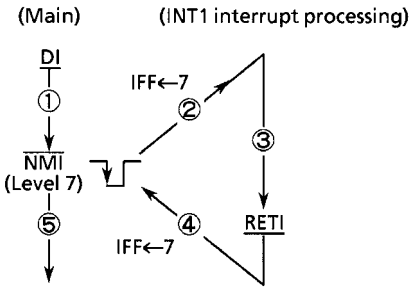
(1) Maskable interrupts



When the CPU accepts an interrupt, it sets IFF to the priority level of the interrupt incremented by 1.
Accordingly, if during interrupt processing an interrupt request is received with the same or a lower priority than that of the interrupt being processed, because this priority level is lower than the IFF value, the second interrupt cannot be accepted until the processing of the prior interrupt is complete.

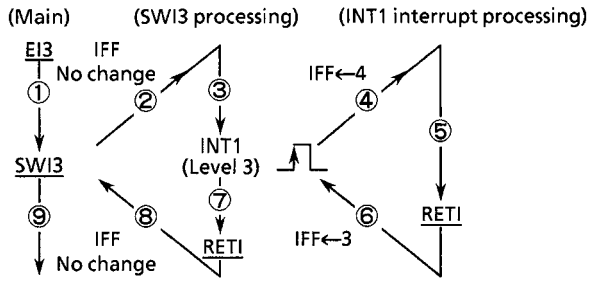
Note: __ (underline): Instruction
①, ②, : Execution flow
IFF: Interrupt mask register

(2) Non-maskable interrupts (NMI, INTWD)



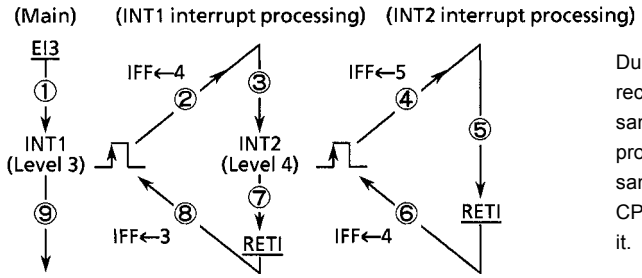
When the DI instruction is executed (IFF is 7), only non-maskable interrupts can be received (because the priority level of non-maskable interrupts is fixed to 7.)
When the EI instruction is executed, the CPU sets IFF to 7 upon acceptance of an NMI or INTWD interrupt.

(3) Non-maskable interrupts (Software interrupts, illegal instruction interrupts)



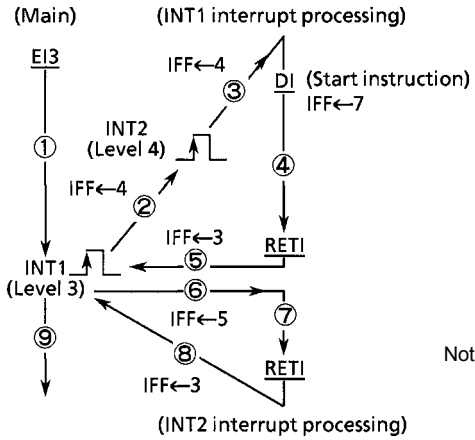
When the DI instruction is executed (IFF is 7), the CPU can accept interrupts. However, unlike with NMI or INTWD interrupts, IFF does not change upon acceptance of an interrupt. Therefore, during processing of a software interrupt, if a request is received for an interrupt with a priority the same or higher than the IFF value, the interrupt is nested.

(4) Interrupt nesting



During interrupt processing, if a request is received for an interrupt with a priority the same or higher than the interrupt being processed (the interrupt priority level is the same as or higher than the IFF value), the CPU receives the second interrupt and nests it.

(5) Interrupt sampling (Maskable interrupt nesting disabled)



If, after the time the CPU has accepted an interrupt but before the CPU begins processing it, the CPU receives a request for another interrupt with a higher priority, the second interrupt is nested after execution of the start instruction for processing of the interrupt accepted first.
Accordingly, issuing the DI instruction as the start instruction disables nesting of maskable interrupts.

Note: __ (underline): Instruction.
①, ②,: Execution flow
IFF: Interrupt mask register

Table 3.3.2 shows the TMP95CU54A interrupt vectors and micro DMA start vectors. With the TMP95CU54A, FFFF00H to FFFFFFH (256 bytes) is allocated to the interrupt vector area.

Table 3.3.2 TMP95CU54A Interrupt Vectors and Micro DMA Start Vectors

Default priority	Type	Interrupt source and source of micro DMA request	Vector value V	Vector reference address	Micro DMA start vector
1	Non-maskable	Reset or [SWI0] instruction	0000H	FFFF00H	–
2		[SWI1] instruction	0004H	FFFF04H	–
3		Illegal instruction or [SWI2] instruction	0008H	FFFF08H	–
4		[SWI3] instruction	000CH	FFFF0CH	–
5		[SWI4] instruction	0010H	FFFF10H	–
6		[SWI5] instruction	0014H	FFFF14H	–
7		[SWI6] instruction	0018H	FFFF18H	–
8		[SWI7] instruction	001CH	FFFF1CH	–
9		NMI: $\overline{\text{NMI}}$ pin input	0020H	FFFF20H	–
10		INTWD: Watchdog timer	0024H	FFFF24H	–
–	–	Micro DMA (Note)	–	–	–
11	Maskable	INT0: INT0 pin input	0028H	FFFF28H	28H
12		INT1: INT1 pin input	002CH	FFFF2CH	2CH
13		INT2: INT2 pin input	0030H	FFFF30H	30H
14		INT3: INT3 pin input	0034H	FFFF34H	34H
15		INT4: INT4 pin input	0038H	FFFF38H	38H
16		INT5: INT5 pin input	003CH	FFFF3CH	3CH
17		INT6: INT6 pin input	0040H	FFFF40H	40H
18		INT7: INT7 pin input	0044H	FFFF44H	44H
19		INT8: INT8 pin input	0048H	FFFF48H	48H
20		INTT0: 8-bit timer 0	004CH	FFFF4CH	4CH
21		INTT1: 8-bit timer 1	0050H	FFFF50H	50H
22		INTT2: 8-bit timer 2	0054H	FFFF54H	54H
23		INTT3: 8-bit timer 3	0058H	FFFF58H	58H
24		INTT4: 8-bit timer 4	005CH	FFFF5CH	5CH
25		INTT5: 8-bit timer 5	0060H	FFFF60H	60H
26		INTT6: 8-bit timer 6	0064H	FFFF64H	64H
27		INTT7: 8-bit timer 7	0068H	FFFF68H	68H
28		INTTR8: 16-bit timer 8 (TREG8)	006CH	FFFF6CH	6CH
29		INTTR9: 16-bit timer 8 (TREG9)	0070H	FFFF70H	70H
30		INTTRA: 16-bit timer 9 (TREGA)	0074H	FFFF74H	74H
31		INTTRB: 16-bit timer 9 (TREGB)	0078H	FFFF78H	78H
32		INTTO8: 16-bit timer 8 (Overflow)	007CH	FFFF7CH	7CH
33		INTTO9: 16-bit timer 9 (Overflow)	0080H	FFFF80H	80H
34		INTRX0: Serial receive (Channel 0)	0084H	FFFF84H	84H
35		INTTX0: Serial transmission (Channel 0)	0088H	FFFF88H	88H
36		INTRX1: Serial receive (Channel 1)	008CH	FFFF8CH	8CH
37		INTTX1: Serial transmission (Channel 1)	0090H	FFFF90H	90H
38		INTCR: CAN receive	0094H	FFFF94H	–
39		INTCT: CAN transmission	0098H	FFFF98H	–
40		INTCG: CAN global	009CH	FFFF9CH	–
41		INTSE0: SEI0 (SEF/WCOL/SOVF)	00A0H	FFFFA0H	–
42		INTAD: AD conversion end	00A4H	FFFFA4H	A4H
43		INTTC0: Micro DMA end (Channel 0)	00A8H	FFFFA8H	–
44		INTTC1: Micro DMA end (Channel 1)	00ACH	FFFFACH	–
45		INTTC2: Micro DMA end (Channel 2)	00B0H	FFFFB0H	–
46		INTTC3: Micro DMA end (Channel 3)	00B4H	FFFFB4H	–
47		INTSE1: SEI1 (TSRC)	00B8H	FFFFB8H	B8H
48		INTSE2: SEI2 (TSTC)	00BCH	FFFFBCH	BCH
–		(Reserved)	00C0H	FFFFC0H	–
to		to	to	to	to
–		(Reserved)	00FCH	FFFFFCH	–
–	–	Micro DMA soft start request	–	–	FCH

Note: Micro DMA default priority
If an interrupt request is generated by a source specified by micro DMA, the interrupt has the highest priority of the maskable interrupts (irrespective of the default priority allocated to all channels).

Setting reset vectors and interrupt vectors

[1] Reset vector

FFFF00H	PC (7:0)
FFFF01H	PC (15:8)
FFFF02H	PC (23:16)
FFFF03H	XX

XX: Don't care

[2] Interrupt vectors (Other than reset vector)

Vector reference address + 0	PC (7:0)
+ 1	PC (15:8)
+ 2	PC (23:16)
+ 3	XX

XX: Don't care

(Setting example)

Where the reset vector is defined as FF0000H, the NMI vector as FF9ABCH, and the INT1 vector as FF3456H

```
ORG     0FF0000H
LD      A, B
        ⋮
ORG     0FF9ABCH
LD      B, C
        ⋮
ORG     0FF3456H
LD      C, A
        ⋮
ORG     0FFFF00H
DL      0FF0000H        ; reset vector = FF0000H

ORG     0FFFF20H
DL      0FF9ABCH        ; NMI vector = FF9ABCH

ORG     0FFFF2CH
DL      0FF3456H        ; INT1 vector = FF3456H
```

Reference:

ORG and DL are assembler directives

ORG: For location counter control

DL: To define (32-bit) long word data

3.3.2 Micro DMA processing

In addition to general-purpose interrupt processing, the TMP95CU54A supports a micro DMA function. Interrupt requests set by the micro DMA perform micro DMA processing at the highest priority level of maskable interrupts (level 6), regardless of the priority level of the particular interrupt source.

Because the micro DMA function is implemented with the cooperative operation of the CPU, when the CPU is put into stand-by state by a HALT instruction, micro DMA requirements will be ignored (pending).

(1) Micro DMA operation

When an interrupt request is generated by an interrupt source specified by the micro DMA start vector register, the micro DMA triggers a micro DMA request to the CPU at interrupt priority level 6 and starts processing the request. The four micro DMA channels allow micro DMA processing to be set for up to four types of interrupts at any one time.

When micro DMA is accepted, the interrupt request flip-flop assigned to that channel is cleared. The data are automatically transferred from the transfer source address to the transfer destination address set in the control register, and the transfer counter is decremented by 1. If the decremented counter reads other than 0, DMA processing ends with no change in the value of the micro DMA start vector register. If the decremented reading is 0, the micro DMA transfer end interrupt (INTTC0 to 3) passes from the CPU to the interrupt controller. In addition, the micro DMA start vector register is cleared to 0, the next micro DMA is disabled, and micro DMA processing is complete.

If a micro DMA request is set for more than one channel at a time, the priority is not based on the interrupt priority level but on the channel number: the smaller the channel number the higher the priority. (Channel 0 (high) --> channel 3 (low)).

If an interrupt request is triggered for the interrupt source in use during the interval between the clearing of the micro DMA start vector and the next setting, general-purpose interrupt processing is executed at the interrupt level set. Therefore, when using the interrupt only for starting the micro DMA (not using the interrupt as a general-purpose interrupt), first set the interrupt level to 0 (interrupt requests disabled).

When using micro DMA and general-purpose interrupts together as described above, first set the level of the interrupt used to start micro DMA processing lower than all the other interrupt levels. In this case, the cause of a general interrupt is limited to the edge interrupt.

Example: When using external interrupt INT0 to 3 to start micro DMA0 to 3, set:
External interrupt INT0 to 3 interrupt level ..... “1”
Level of other interrupts ..... “2” to “6”

As with other maskable interrupts, the priority of the micro DMA transfer end interrupt is determined by the interrupt level and the default priority.

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

Three micro DMA transfer modes are supported: 1-byte transfer, 2-byte (one word) transfer, and 4-byte transfer. After a transfer in any mode, the transfer source/destination addresses are incremented, decremented, or remain unchanged. This simplifies the transfer of data from I/O to memory, from memory to I/O, and from I/O to I/O. For details of the transfer modes, see 3.3.2 (4) Transfer Mode Register.

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

Micro DMA processing can be started by the 30 interrupts (INT0 to INTTX1, INTAD, INTSE1, INTSE2) shown in the micro DMA start vectors of Table 3.3.2 and by the micro DMA soft start, making a total of 31 interrupts.

Figure 3.3.2 shows the micro DMA cycle in transfer destination address INC mode (the same as for other modes, with the exception of COUNTER mode).

[1] Word transfer (the conditions for this cycle are based on an external 16-bit bus, 0 waits, transfer source/transfer destination addresses both even-numbered values)

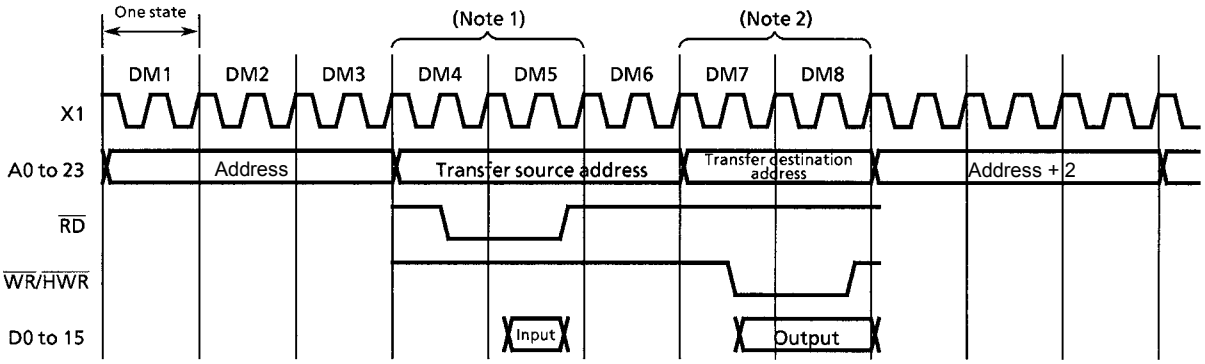


Figure 3.3.2 Timing of Micro DMA Cycle (1/3)

States 1 to 3: Instruction fetch cycle (gets next address code).
If three or more instruction codes are inserted in the instruction queue buffer, this cycle becomes a dummy cycle.
States 4 to 5: Micro DMA read cycle
State 6 : Dummy cycle (the address bus remains as in state 5)
States 7 to 8: Micro DMA write cycle

Note 1: If the source address area is an 8-bit bus, it is incremented by two states.
Note 2: If the destination address area is an 8-bit bus, it is incremented by two states.

[2] Word transfer (the conditions for this cycle are based on a 16-bit external bus, 0 waits, transfer source/transfer destination addresses both odd-numbered values)

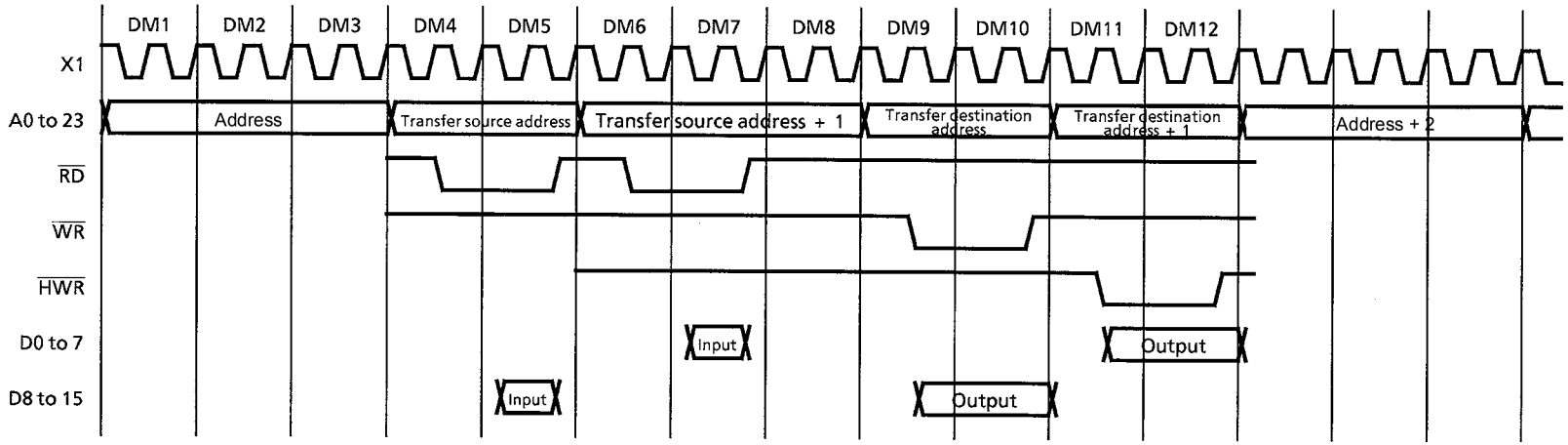


Figure 3.3.2 Timing of Micro DMA Cycle (2/3)

[3] 4-byte transfer (the conditions for this cycle are based on a 16-bit external bus, 0 waits, transfer source/transfer destination addresses both even-numbered values)

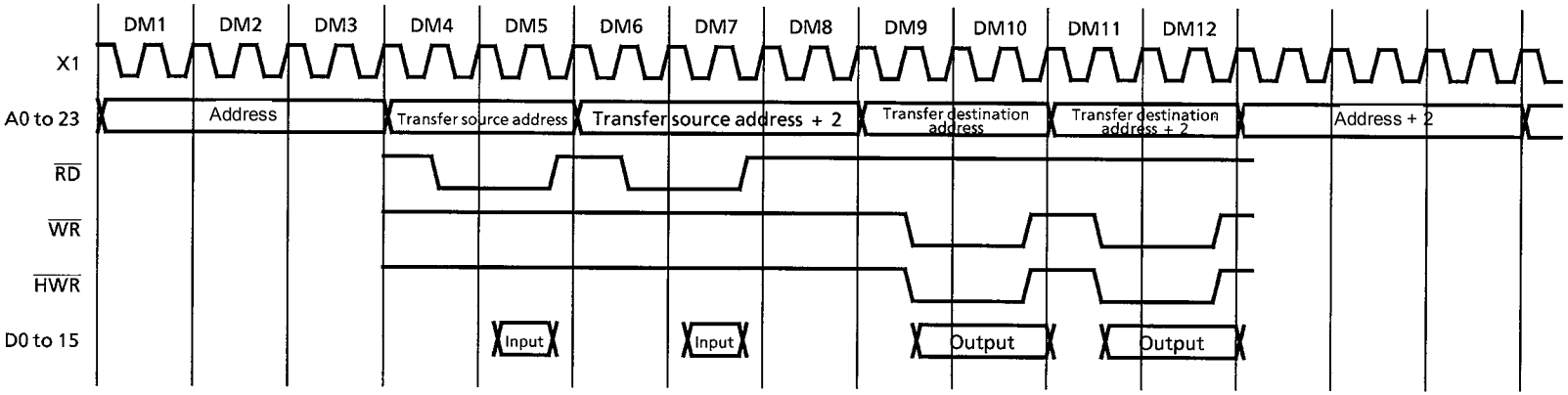


Figure 3.3.2 Timing of Micro DMA Cycle (3/3)

(2) Micro DMA soft start function

In addition to starting micro DMA by interrupt, the TMP95CU54A supports a micro DMA soft start function. This starts micro DMA by generating a cycle to write to the soft DMA control register.

To code a soft start, write micro DMA start vector FCH to micro DMA start vector register DMA0V to 3V (at memory addresses 5AH, 5BH, 5CH, and 5DH).

Then, write any data to soft DMA control register SDMACR0 to 3 (at memory addresses 6AH, 6BH, 6CH, and 6DH). (The value of the data has no effect on the operation of the soft start.) This starts micro DMA of the applicable channel once. Then, whenever data are written again to the soft DMA control register, as long as the micro DMA transfer counter register values are other than 0, a soft start can be continuously triggered (without rewriting the micro DMA start vector).

Setting the micro DMA start vector is a prerequisite for generating a micro DMA software start. (The software start request is a one-shot request and not saved. Therefore, even if a cycle which writes to the soft DMA control register is generated, unless the micro DMA start vector is already set, a soft start cannot be generated.)

(3) Structure of micro DMA-only registers

Figure 3.3.3 shows the micro DMA-only registers. These registers are incorporated in the CPU. (See 3.2.5, Control Registers in Chapter 3, TLCS-900/H CPU.) To set the registers use the LDC instruction.

Set the transfer source address in the transfer source address register; the transfer destination address, in the transfer destination address register. These address registers use only the lower 24 bits. They support a 16M-byte address space.

Use the transfer counter register to set the number of times micro DMA is performed between 1 and 65536.

For details on setting the transfer mode register, see 3.3.2 (4), Transfer Mode Register.

Only the LDC cr, r instruction can load data into the micro DMA-only registers.

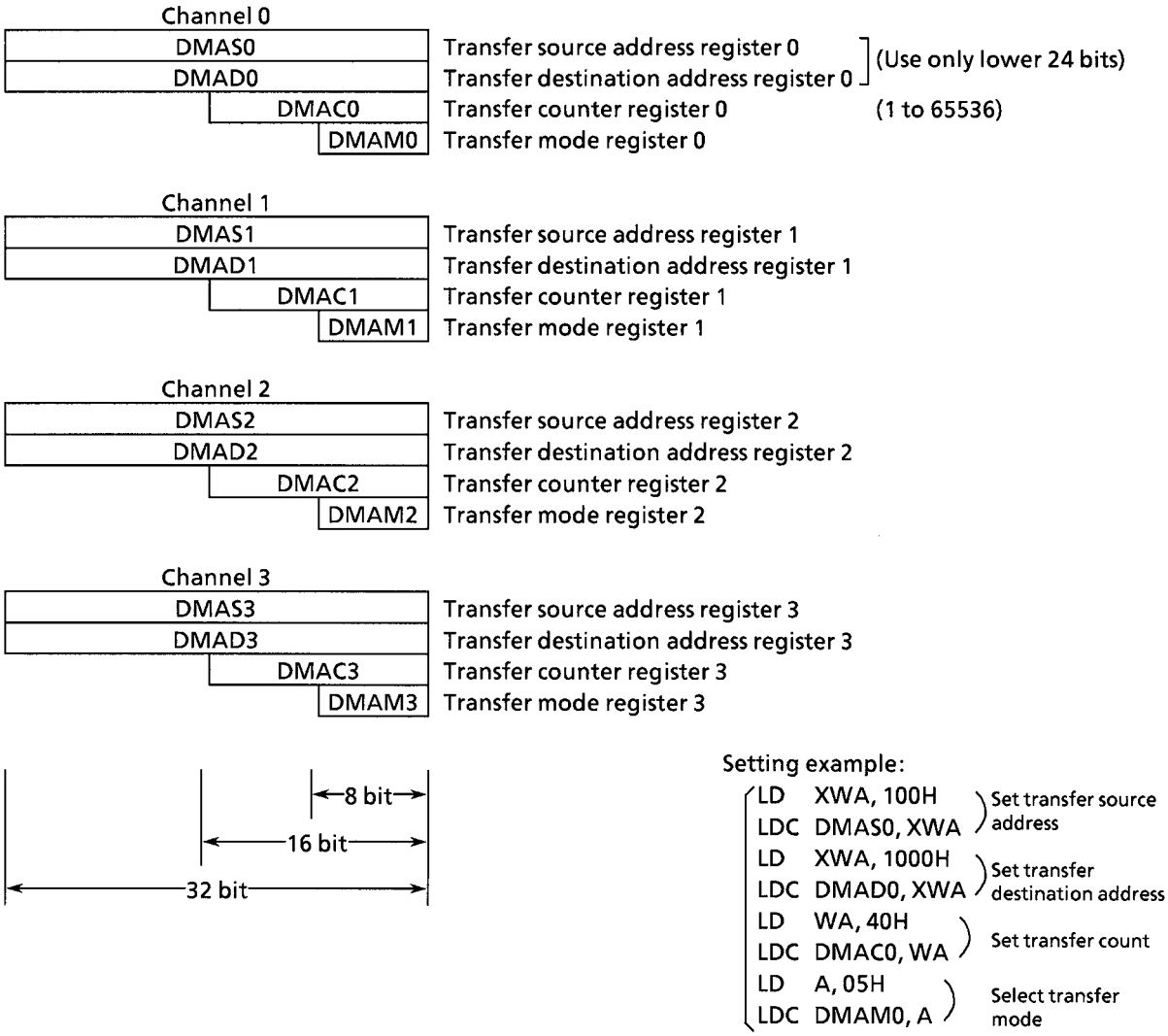


Figure 3.3.3 Micro DMA-Only Registers

(4) Transfer mode register

To set micro DMA transfer mode, use transfer mode register DMAM0 to 3. Table 3.3.3 shows the settings for each mode and the numbers of execution states.

Table 3.3.3 Micro DMA Transfer Mode

DMAM0 to 3	8-bit	
	0 0 0	Mode

Note: When setting a value in this register, write 0 to the upper three bits.

			Number of Transfer Bytes	Mode Description	Number of Execution States (*)	Minimum Execution Time @ fc = 24 MHz
000 (Fixed)	000	00	Byte transfer	Transfer destination address INC mode ..... For I/O to memory (DMADn +) ← (DMASn) DMACn ← DMACn – 1 If DMACn = 0, then INTTCn generated	8 states	667 ns
		01	Word transfer			
		10	4-byte transfer		12 states	1000 ns
	001	00	Byte transfer	Transfer destination address DEC mode ..... For I/O to memory (DMADn –) ← (DMASn) DMACn←DMACn – 1 If DMACn = 0, then INTTCn generated	8 states	667 ns
		01	Word transfer			
		10	4-byte transfer		12 states	1000 ns
	010	00	Byte transfer	Transfer source address INC mode ..... For memory to I/O (DMADn) ← (DMASn +) DMACn←DMACn – 1 If DMACn = 0, then INTTCn generated	8 states	667 ns
		01	Word transfer			
		10	4-byte transfer		12 states	1000 ns
	011	00	Byte transfer	Transfer source address DEC mode ..... For memory to I/O (DMADn) ← (DMASn –) DMACn←DMACn – 1 If DMACn = 0, then INTTCn generated	8 states	667 ns
		01	Word transfer			
		10	4-byte transfer		12 states	1000 ns
	100	00	Byte transfer	Address fixed mode ..... For I/O to I/O (DMADn) ← (DMASn) DMACn←DMACn – 1 If DMACn = 0, then INTTCn generated	8 states	667 ns
		01	Word transfer			
		10	4-byte transfer		12 states	1000 ns
	101	00	Counter mode ..... For counting number of times interrupts generated DMASn ← DMASn + 1 DMACn ← DMACn – 1 If DMACn = 0, then INTTCn generated		5 states	417 ns

* For external 16-bit bus, 0 waits, word/4-byte transfer mode, transfer source/transfer destination addresses both have even-numbered values.

Note: n: Corresponding micro DMA channels 0 to 3
DMADn + / DMASn + : Post increment (increments register value after transfer)
DMADn -/ DMASn-: Post decrement (decrements register value after transfer)
The I/Os in the table mean fixed addresses; memory means incremented and decremented addresses.
Do not use undefined code, that is, codes other than those listed above for the transfer mode register.

3.3.3 Interrupt Controller Control

Figure 3.3.4 is a block diagram of the interrupt controller circuit. The left-hand side of this diagram shows the interrupt controller. The right-hand side shows the CPU interrupt request signal circuit and CPU halt release circuit. (For details on halt modes, see 3.4, Standby Function.)

The interrupt controller has a total of 40 interrupt channels, consisting of NMI, INTWD, INT0 to 8, INTT0 to 7, INTTR8 to O9, INTRX0 to TX1, INTCR to G, INTSE0 to 2, INTAD, and INTTC0 to 3.

Each interrupt channel supports:

- Interrupt request flag (40 channels)
- Interrupt priority setting register (38 channels (NMI and INTWD excluded)).

In addition, there are also four channels of start vector registers for performing micro DMA processing.

(1) Interrupt request flags

The function of the interrupt request flag is to indicate the generation of an interrupt request. Apart from NMI and INTWD, each channel has a clear bit <IxxC> for clearing the interrupt requests (see Figure 3.3.5, Interrupt Priority Setting Registers). Reading clear bit <IxxC> reads the state of the interrupt request flag and indicates whether an interrupt request is generated or not.

The interrupt request flags are zero-cleared by the following operations:

[1] A reset (clears all interrupt request flags)

[2] When the CPU accepts an interrupt and reads the vector of the accepted interrupt channel

[3] When the CPU accepts the micro DMA request of the specified channel

[4] When 0 is written to clear bit <IxxC> of the interrupt priority setting register

Note: [2], [3], and [4] operations do not include INT0 level mode or INTRX0, 1.

In addition, flags are also cleared by the following operations.

Table 3.3.4 Other Flag Clearing Operations

Interrupt source \ Flag clearing source		Other operations that clear interrupt flags
INT0	Edge mode	Switching to level mode
	Level mode	Change in pin input after interrupt is generated (high level → low level)
INTRX0, 1		Reading serial channel receive buffer

Before clearing an interrupt request by writing 0 to the clear bit or by performing a Table 3.3.4 operation to clear the interrupt request flag, first execute the DI instruction.

(INT0 interrupt cautions)

Note the following cautions when using the INT0 interrupt in level mode.

In level mode, the INT0 pin input must be held continuously at high level until the interrupt response sequence is completed. Likewise, when releasing the halt in this mode, the INT0 pin must be held continuously at high level until the halt is released.

When using INT0 level mode, be sure that a low level is not input as a result of noise as this can cause malfunction.

When switching the INT0 pin operation mode from level to edge mode, first disable the INT0 interrupt as follows. (In level mode, an accepted interrupt request must be cleared.)

Setting example:

```
DI                          ; disable interrupt
LD (IIMC), XX0XXX0XB        ; switch from level to edge
LD (INTE0AD), XXXX0nnnB     ; clear interrupt request flag and set INT0
                              interrupt level to n
EI                          ; enable interrupt
```

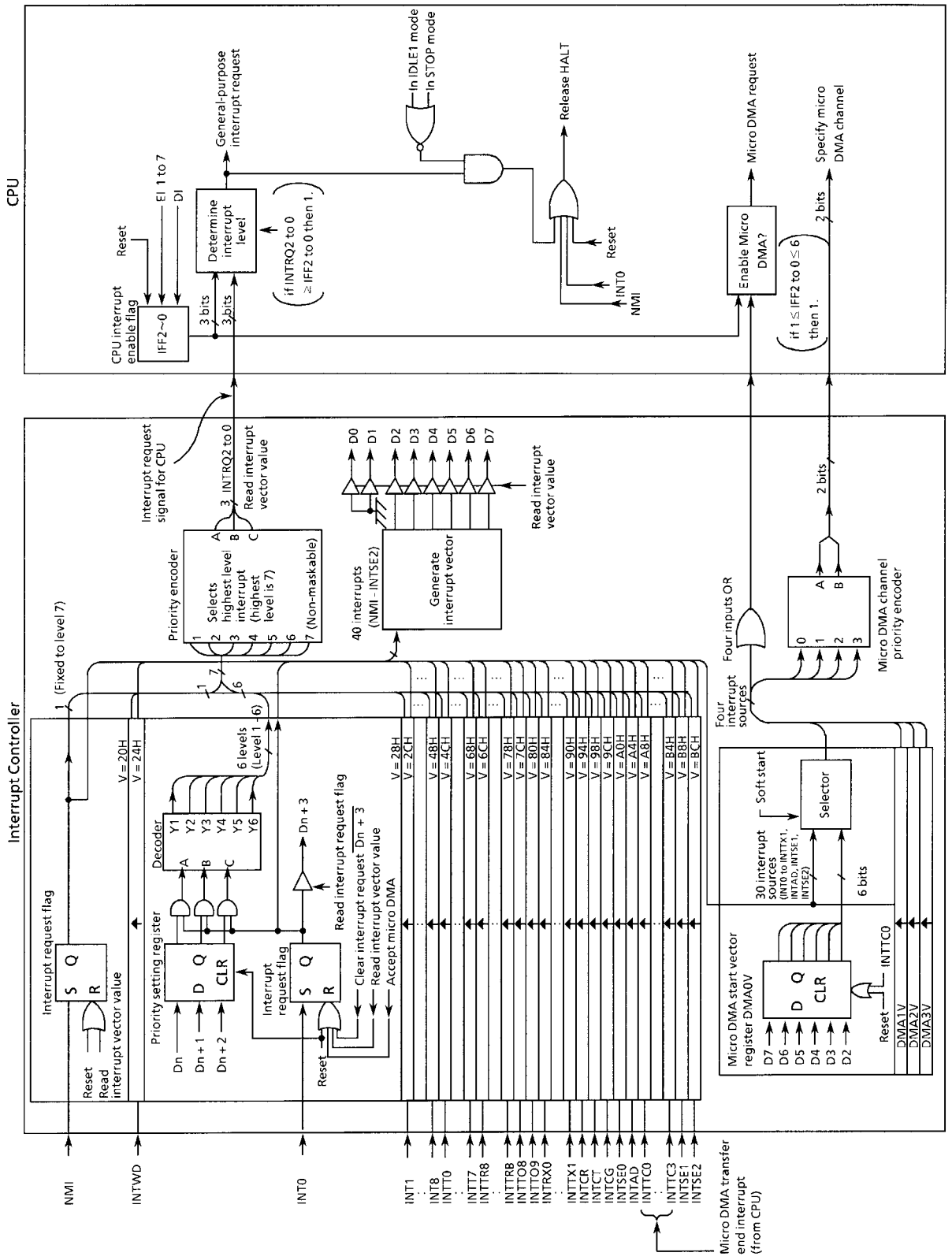


Figure 3.3.4 Block Diagram of Interrupt Controller

(2) Interrupt priority setting register

Figure 3.3.5 shows the interrupt priority setting registers. Each of the 38 interrupt channels (INT0 to AD, INTTC0 to 3, INTSE1, INTSE2) has an interrupt request level setting bit <IxxM2:0>. An interrupt request is generated at six interrupt levels (levels 1 through 6). Setting the priority level to 0 (or 7) disables the corresponding interrupt request. The priority level for non-maskable interrupts ($\overline{NMI}$ pin input) is fixed to 7. If two or more interrupts with the same level occur simultaneously, the interrupts are accepted in accordance with the default priority.

Symbol	Address	7	6	5	4	3	2	1	0	
INTE0AD	70H	INTAD				INT0				←Interrupt source
		IADC	IADM2	IADM1	IADM0	I0C	I0M2	I0M1	I0M0	←bit Symbol
		R/W	W			R/W (Note1)	W			←Read/Write
		0	0	0	0	0	0	0	0	←After reset
INTE12	71H	INT2				INT1				
		I2C	I2M2	I2M1	I2M0	I1C	I1M2	I1M1	I1M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTE34	72H	INT4				INT3				
		I4C	I4M2	I4M1	I4M0	I3C	I3M2	I3M1	I3M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTE56	73H	INT6				INT5				
		I6C	I6M2	I6M1	I6M0	I5C	I5M2	I5M1	I5M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTE78	74H	INT8				INT7				
		I8C	I8M2	I8M1	I8M0	I7C	I7M2	I7M1	I7M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTET01	75H	INTT1 (Timer 1)				INTT0 (Timer 0)				
		IT1C	IT1M2	IT1M1	IT1M0	IT0C	IT0M2	IT0M1	IT0M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTET23	76H	INTT3 (Timer 3)				INTT2 (Timer 2)				
		IT3C	IT3M2	IT3M1	IT3M0	IT2C	IT2M2	IT2M1	IT2M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTET45	77H	INTT5 (Timer 5)				INTT4 (Timer 4)				
		IT5C	IT5M2	IT5M1	IT5M0	IT4C	IT4M2	IT4M1	IT4M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	

(Do not use read-modify-write instructions.)

Note1: In INT0 level mode, writing 0 to <I0C> does not clear the interrupt request flag.

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

IxxC	Function (Read)	Function (Write)
0	No interrupt request	Clears interrupt request flag
1	Interrupt request	----- Don't care -----

Figure 3.3.5 Interrupt Priority Setting Registers (1/2)

Symbol	Address	7	6	5	4	3	2	1	0	
INTET67	78H	INTT7 (Timer 7)				INTT6 (Timer 6)				←Interrupt source
		IT7C	IT7M2	IT7M1	IT7M0	IT6C	IT6M2	IT6M1	IT6M0	←bit Symbol
		R/W	W			R/W	W			←Read/Write
		0	0	0	0	0	0	0	0	←After reset
INTET89	79H	INTTR9 (TREG9)				INTTR8 (TREG8)				
		IT9C	IT9M2	IT9M1	IT9M0	IT8C	IT8M2	IT8M1	IT8M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTETAB	7AH	INTTRB (TREGB)				INTTRA (TREGA)				
		ITBC	ITBM2	ITBM1	ITBM0	ITAC	ITAM2	ITAM1	ITAM0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTEOV	7BH	INTTO9				INTTO8				
		ITO9C	ITO9M2	ITO9M1	ITO9M0	ITO8C	ITO8M2	ITO8M1	ITO8M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTES0	7CH	INTTX0				INTRX0				
		ITX0C	ITX0M2	ITX0M1	ITX0M0	IRX0C	IRX0M2	IRX0M1	IRX0M0	
		R/W	W			R (Note2)	W			
		0	0	0	0	0	0	0	0	
INTES1	7DH	INTTX1				INTRX1				
		ITX1C	ITX1M2	ITX1M1	ITX1M0	IRX1C	IRX1M2	IRX1M1	IRX1M0	
		R/W	W			R (Note2)	W			
		0	0	0	0	0	0	0	0	
INTEC01	7EH	INTCT				INTCR				
		IC1C	IC1M2	IC1M1	IC1M0	IC0C	IC0M2	IC0M1	IC0M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTEC2S0	7FH	INTSE0				INTCG				
		ISE0C	ISE0M2	ISE0M1	ISE0M0	IC2C	IC2M2	IC2M1	IC2M0	
		R/W (Note3)	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTETC01	80H	INTTC1				INTTC0				
		ITC1C	ITC1M2	ITC1M1	ITC1M0	ITC0C	ITC0M2	ITC0M1	ITC0M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTETC23	81H	INTTC3				INTTC2				
		ITC3C	ITC3M2	ITC3M1	ITC3M0	ITC2C	ITC2M2	ITC2M1	ITC2M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTESE12	82H	INTSE2				INTSE1				
		ISE2C	ISE2M2	ISE2M1	ISE2M0	ISE1C	ISE1M2	ISE1M1	ISE1M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	

(Do not use read-modify-write instructions.)

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

IxxC	Function (Read)	Function (Write)
0	No interrupt request	Clears interrupt request flag
1	Interrupt request	----- Don't care -----

Note2: As <IRX0C> and <IRX1C> are read-only registers, writing 0 to them does not clear the interrupt request flag.

Note3: Please clear <ISE0C> by writing 0 after clearing the status flag <SEF> or <MODF> of SEI.

Figure 3.3.5 Interrupt Priority Setting Registers (2/2)

From among simultaneous interrupts, the interrupt controller selects the interrupt request with the highest level and sends its vector address to the CPU.

Then, the CPU compares the priority level of the interrupt request with the value of the interrupt mask register <IFF2:0> in the status register. If the priority level of the interrupt request is higher than the value of the interrupt mask register, the CPU accepts the interrupt. When the CPU side interrupt mask register <IFF2:0> is set to the priority level of the received interrupt incremented by 1, subsequent interrupt requests are only accepted if their level is equal to or greater than the incremented value.

(3) Micro DMA start vector

The interrupt controller has four channels of micro DMA start vector registers. Writing the micro DMA start vector value (Table 3.3.2) for each interrupt source to these registers makes the applicable interrupt request into a micro DMA request. But first set values in the registers for micro DMA parameters (DMAS, DMAD, DMAC, DMAM). Figure 3.3.6 shows the micro DMA start vector registers.

The function of the micro DMA start vector registers is to select the interrupt to use with micro DMA processing. The micro DMA start source is assigned to the interrupt source whose micro DMA start vector matches the vector value set in the micro DMA start vector register.

When the value of the micro DMA transfer counter is set to 0 after micro DMA processing, the CPU generates a micro DMA transfer end interrupt (INTTC0 to 3) corresponding to the micro DMA start vector register. When the micro DMA start vector register is cleared, the micro DMA startup source is released. Therefore, when continuously performing micro DMA processing, set the start vector value in the micro DMA start vector register again during processing of the micro DMA transfer end interrupt.

When the same vector is set in the micro DMA start vector registers of multiple channels, the lower the channel number the higher the priority.

The channel with the lowest number is executed until the micro DMA transfer end interrupt. Unless the micro DMA start vector is set again during the processing of the micro DMA transfer end interrupt, the subsequent micro DMA startup moves to the next smallest channel number. (This operation is called a micro DMA chain.)

Micro DMA0 start vector register

DMA0V (005AH) (Do not use read-modify-write instructions.)

	7	6	5	4	3	2	1	0
bit Symbol	DMA0V7	DMA0V6	DMA0V5	DMA0V4	DMA0V3	DMA0V2		
Read/Write	W							
After reset	0	0	0	0	0	0		
Function	Set startup interrupt source for micro DMA channel 0							

Micro DMA1 start vector register

DMA1V (005BH) (Do not use read-modify-write instructions.)

	7	6	5	4	3	2	1	0
bit Symbol	DMA1V7	DMA1V6	DMA1V5	DMA1V4	DMA1V3	DMA1V2		
Read/Write	W							
After reset	0	0	0	0	0	0		
Function	Set startup interrupt source for micro DMA channel 1							

Micro DMA2 start vector register

DMA2V (005CH) (Do not use read-modify-write instructions.)

	7	6	5	4	3	2	1	0
bit Symbol	DMA2V7	DMA2V6	DMA2V5	DMA2V4	DMA2V3	DMA2V2		
Read/Write	W							
After reset	0	0	0	0	0	0		
Function	Set startup interrupt source for micro DMA channel 2							

Micro DMA3 start vector register

DMA3V (005DH) (Do not use read-modify-write instructions.)

	7	6	5	4	3	2	1	0
bit Symbol	DMA3V7	DMA3V6	DMA3V5	DMA3V4	DMA3V3	DMA3V2		
Read/Write	W							
After reset	0	0	0	0	0	0		
Function	Set startup interrupt source for micro DMA channel 3							

Setting Micro DMA Startup Source

Micro DMA startup source	Value set in micro DMA start vector register	Micro DMA startup source	Value set in micro DMA start vector register
INT 0 interrupt	28H	INTT 7 interrupt	68H
INT 1 interrupt	2CH	INTTR 8 interrupt	6CH
INT 2 interrupt	30H	INTTR 9 interrupt	70H
INT 3 interrupt	34H	INTTR A interrupt	74H
INT 4 interrupt	38H	INTTR B interrupt	78H
INT 5 interrupt	3CH	INTTO 8 interrupt	7CH
INT 6 interrupt	40H	INTTO 9 interrupt	80H
INT 7 interrupt	44H	INTRX 0 interrupt	84H
INT 8 interrupt	48H	INTTX 0 interrupt	88H
INTT 0 interrupt	4CH	INTRX 1 interrupt	8CH
INTT 1 interrupt	50H	INTTX 1 interrupt	90H
INTT 2 interrupt	54H	INTAD interrupt	A4H
INTT 3 interrupt	58H	INTSE1 interrupt	B8H
INTT 4 interrupt	5CH	INTSE2 interrupt	BCH
INTT 5 interrupt	60H	Micro DMA soft start	FCH
INTT 6 interrupt	64H		

Figure 3.3.6 Setting Micro DMA Start Vector Register and Startup Source

(4) External interrupt control

Table 3.3.5 shows the function settings for the external interrupt pins.

TMP95CU54A can select the operating mode for the $\overline{\text{NMI}}$, INT0, INT5, or INT7 pins from among external interrupt functions. (For details on the external interrupt function pulse width, see "4.7 Interrupt Operations".)

Table 3.3.5 Setting Functions on External Interrupt Pins

Interrupt pin	Shared pin	Mode	Setting method
$\overline{\text{NMI}}$	—	Falling edge	IIMC<NMIREE> = 0
		Both falling and rising edges	IIMC<NMIREE> = 1
INT0	P56	Rising edge	IIMC<I0LE> = 0, <I0IE> = 1
		Level	IIMC<I0LE> = 1, <I0IE> = 1
INT1	P70	Rising edge	———
INT2	P72	Rising edge	———
INT3	P73	Rising edge	———
INT4	P75	Rising edge	———
INT5	P90	Rising edge	T8MOD<CAP12M1:0> = 0, 0 or 0, 1 or 1, 1
		Falling edge	T8MOD<CAP12M1:0> = 1, 0
INT6	P91	Rising edge	———
INT7	P94	Rising edge	T9MOD<CAP34M1:0> = 0, 0 or 0, 1 or 1, 1
		Falling edge	T9MOD<CAP34M1:0> = 1, 0
INT8	P95	Rising edge	———

The input mode of the NMI and INT0 interrupts can be controlled by interrupt input mode control register IIMC.

Figure 3.3.7 shows the interrupt input mode control register.

Interrupt Input Mode Control Register

IIMC (0059H)

(Do not use read-modify-write instructions.)

	7	6	5	4	3	2	1	0
bit Symbol			–			I0IE	I0LE	NMIREE
Read/Write			W			W		
After reset			0			0	0	0
Function			Note: Be sure to write 0.			INT0 input control 0: Disabled 1: Enabled	INT0 interrupt control 0: Rising edge 1: High level	NMI interrupt control 0: Falling edge 1: Both edges

Note: The INT0 pin can also be used for the halt release described later.
When using as port 56 in HALT mode, be sure to set <I0IE> to 0 before entering HALT mode.

NMI interrupt control

0	Generates interrupt request on falling edge
1	Generates interrupt request on both edges

INT0 interrupt control

0	Generates interrupt request on rising edge
1	Generates interrupt request on high level

INT0 input control

0	INT0 disabled
1	INT0 enabled

Figure 3.3.7 Interrupt Input Mode Control Register

(5) Caution

When the CPU fetches an instruction to clear the interrupt request flag for the interrupt controller immediately before an interrupt is generated, the CPU may execute the instruction between receiving the interrupt and reading the interrupt vector.

To avoid the above occurring, clear the interrupt request flag by entering the instruction to clear the flag after the DI instruction. When setting an interrupt enable again by EI instruction after the execution of a clearing instruction, execute the EI instruction after the clearing instruction and following the execution of more than one more instruction. If the EI instruction is placed immediately after the clearing instruction, an interrupt could be enabled before interrupt request flags are cleared.

When changing the value of the interrupt mask register<IFF2:0> by execution of a POP SR instruction, disable interrupts by DI instruction before execution of the POP SR instruction.

3.4 Standby Function

(1) HALT modes

When the TMP95CU54A executes a HALT instruction, WDMOD<HALTM1:0> of the watchdog timer mode register can be used to set one of the following HALT modes: RUN, IDLE2, IDLE1, STOP. Figure 3.4.1 shows the watchdog timer mode control register.

Watchdog Timer Mode Control Register

WDMOD (006EH)	7	6	5	4	3	2	1	0
bit Symbol	WDTE	WDTP1	WDTP0	WARM	HALTM1	HALTM0	RESCR	DRVE
Read/Write	R/W							
After reset	1	0	0	0	0	0	0	0
Function	Controls watchdog timer 0: Disable 1: Enable	Selects watchdog timer detection time 00: 2^{16}/fc 01: 2^{18}/fc 10: 2^{20}/fc 11: 2^{22}/fc		Warmup time 0: 2^{14}/fc 1: 2^{16}/fc	Selects HALT mode 00: RUN mode 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode		Controls internal reset at runaway detection 1: Performs internal reset by detecting runaway	Pin control at STOP mode 1: Drives pins even during STOP mode

Controls pin state in STOP mode

0	Input/output off
1	Maintains state prevailing before HALT

Controls internal reset at runaway detection

0	Don't care
1	Executes internal reset on runaway detection

Sets HALT mode

00	RUN mode (CPU only halted)
01	STOP mode (all circuits halted)
10	IDLE1 mode (oscillator only operating)
11	IDLE2 mode (some I/O only operating)

Selects warmup time when returning from STOP mode

0	2^{14}/fc (around 683 μs @ 24 MHz)
1	2^{16}/fc (around 2.7 ms @ 24 MHz)

Selects watchdog timer detection time

00	2^{16}/fc (around 2.7 ms @ 24 MHz)
01	2^{18}/fc (around 10.9 ms @ 24 MHz)
10	2^{20}/fc (around 43.7 ms @ 24 MHz)
11	2^{22}/fc (around 174.8 ms @ 24 MHz)

Controls watchdog timer enable/disable

0	Disable
1	Enable

Figure 3.4.1 Watchdog Timer Mode Control Register

The characteristics of RUN, IDLE2, IDLE1, and STOP modes are as follows:

[1] RUN: In this mode, only the CPU is halted. Power dissipation is almost the same as when the CPU is operating.

[2] IDLE2: Only the internal oscillator and specific internal I/O operate. Power dissipation is around one half that when the CPU is operating.

[3] IDLE1: Only the internal oscillator operates; all other circuits are halted. Power dissipation is one tenth of operating mode dissipation.

[4] STOP: All internal circuits, including the internal oscillator, are halted. In this mode, power dissipation drops considerably.

Table 3.4.1 shows the operation of all blocks in HALT modes.

Table 3.4.1 Blocks and I/O Pin Operation in Halt Modes

Halt mode		RUN	IDLE2	IDLE1	STOP
WDMOD <HALTM1, 0>		00	11	10	01
Operating block	CPU	Halted			
	I/O ports	Maintains state prevailing at HALT instruction execution			See Table 3.4.3
	8-bit timers	Operating		Halted	
	16-bit timers				
	Serial channels				
	Serial expansion				
	CAN controller				
	AD converter				
	Watchdog timer				
Interrupt controller					

(2) Release from HALT mode

Release from HALT mode can trigger an interrupt request or a reset. A combination of the interrupt mask register <IFF2:0> state and the halt mode determine the useable halt release source. (For details, see Table 3.4.2)

- Release by interrupt request

The operation to release HALT mode by using an interrupt request differs according to the interrupt enable state. If the interrupt request level set prior to the execution of the HALT instruction is higher than the interrupt mask register value, after HALT mode is released, interrupt processing is performed by this source, and processing starts from the next instruction following the HALT instruction. If the interrupt request level is lower than the interrupt mask register value, HALT mode is not released. (At a non-maskable interrupt, interrupt processing is performed after HALT mode release irrespective of the mask register value.)

However, in the case of the INT0 interrupt only, HALT mode can be released if the interrupt request level is lower than the interrupt mask register value. In this case the interrupt processing is not performed. Processing always starts from the next instruction following the HALT instruction. (The INT0 interrupt request flag is held at 1.)

Note: Usually, interrupts can release all halts status. However, the interrupts (= NMI, INT0) which can release the HALT mode may not be able to do so if they are input during the period when the CPU is shifting to the HALT mode (for about 3 clocks of fc) with IDLE1 or STOP mode (RUN and IDLE2 are not applicable to this case). (In this case, an interrupt request is kept on hold internally)

If another interrupt is generated after it has shifted completely to HALT mode,

halt status can be released without difficulty. The priority of this interrupt is compared with that of the interrupt kept on hold internally, and the interrupt with the higher priority is handled first followed by the other interrupt.

- Release by reset

All HALT modes can be released by a reset. However, when releasing STOP mode, allow sufficient reset time (at least 3ms) for the oscillator to stabilize.

When releasing HALT mode by a reset, the internal RAM retains the data prevailing immediately prior to entering the HALT mode. However, other settings are initialized.

Table 3.4.2 Halt Release Sources and Halt Release Operation

Interrupt accept state			Interrupt enabled (interrupt request level) ≥ (interrupt mask)				Interrupt disabled (interrupt request level) < (interrupt mask)			
HALT mode			RUN	IDLE2	IDLE1	STOP	RUN	IDLE2	IDLE1	STOP
HALT release source	Interrupt source	NMI	□	□	□	□ *1	–	–	–	–
		INTWD	□	×	×	×	–	–	–	–
		INT0	□	□	□	□ *1	O	O	O	O *1
		INT1 to 8	□	□	×	×	×	×	×	×
		INTT0 to 7	□	□	×	×	×	×	×	×
		INTTR8, 9, A, B	□	□	×	×	×	×	×	×
		INTTO8, 9	□	□	×	×	×	×	×	×
		INTRX0, TX0	□	□	×	×	×	×	×	×
		INTRX1, TX1	□	□	×	×	×	×	×	×
		INTCR, CT, CG	□	□	×	×	×	×	×	×
		INTSE0, 1, 2	□	□	×	×	×	×	×	×
		INTAD	□	×	×	×	×	×	×	×
	RESET		□	□	□	□	□	□	□	□

□: After HALT mode release, the CPU starts interrupt processing (a reset initializes the LSI).

O: After HALT mode release, processing starts from the next instruction following the HALT instruction. (No interrupt processing)

×: Not used for HALT release.

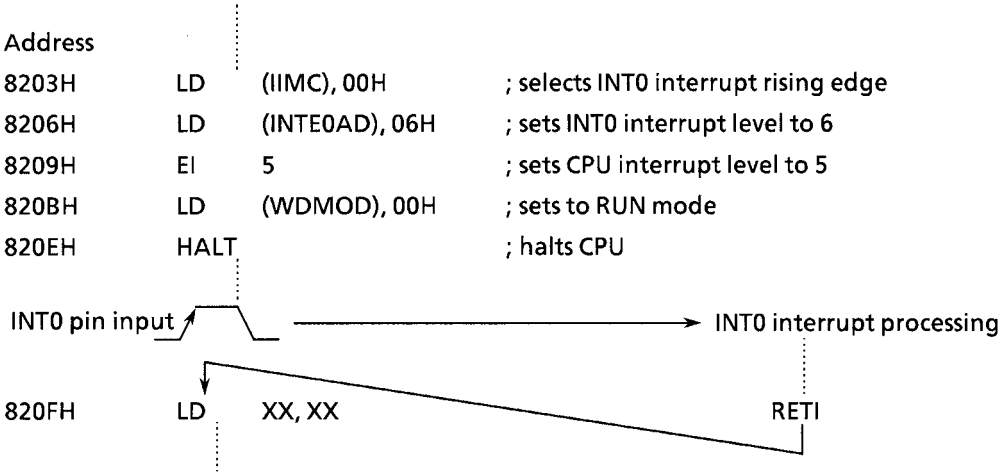
−: As the highest priority level (interrupt request level) for a non-maskable interrupt is fixed to 7, this combination is not available.

*1: Releases HALT after the warmup time has elapsed.

Note: When releasing HALT in an interrupt enabled state by using a level mode INT0 interrupt, maintain high level on pin INT0 until interrupt processing begins. If pin INT0 changes to low level before interrupt processing begins, interrupt processing cannot start correctly.

(Example of release from HALT mode)

Releasing HALT mode using the edge mode INT0 interrupt when the CPU is in RUN mode:



(3) Operation in each mode

[1] RUN mode

In RUN mode, the system clock continues operating even after execution of the HALT instruction. Only the CPU instruction execution operations stop.

In HALT mode, interrupt requests are sampled on the falling edge of the CLK signal.

All the external and internal interrupts can be used for releasing RUN mode. (See Table 3.4.2, Halt Release Sources and Halt Release Operation.)

Figure 3.4.2 shows the timing example for releasing HALT mode using an interrupt.

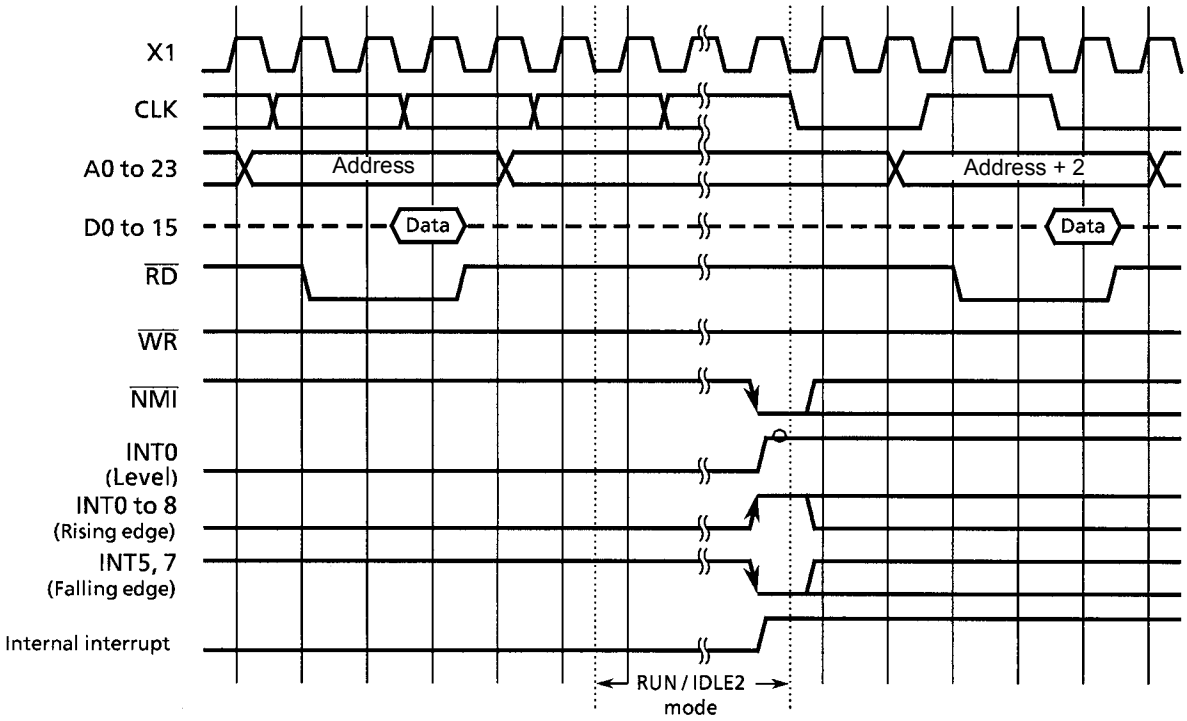


Figure 3.4.2 Example of Timing for Releasing Halt by Interrupt (RUN or IDLE2 Mode)

[2] IDLE2 mode

In IDLE2 mode, the system clock is supplied only to specific internal I/O. CPU instruction execution halts.

In IDLE2 mode, the timing for releasing HALT mode by interrupt is the same as in RUN mode.

External and internal interrupts, apart from INTWD/INTAD, can release IDLE2 mode. (See Table 3.4.2, Halt Release Sources and Halt Release Operation.)

Before entering HALT mode in IDLE2 mode, disable the watchdog timer (to prevent the generation of a watchdog timer interrupt immediately after halt mode release).

[3] IDLE1 mode

In IDLE1 mode, only the internal oscillator operates. The system clock stops. The CLK pin outputs high level.

The interrupt request sampling in HALT mode is asynchronous to the system clock. However, the release (resumption of operation) is synchronous.

Release IDLE1 mode by an external interrupt (NMI, INT0). (See Table 3.4.2, Halt Release Sources and Halt Release Operation.)

Figure 3.4.3 shows the timing example for releasing HALT mode by interrupt.

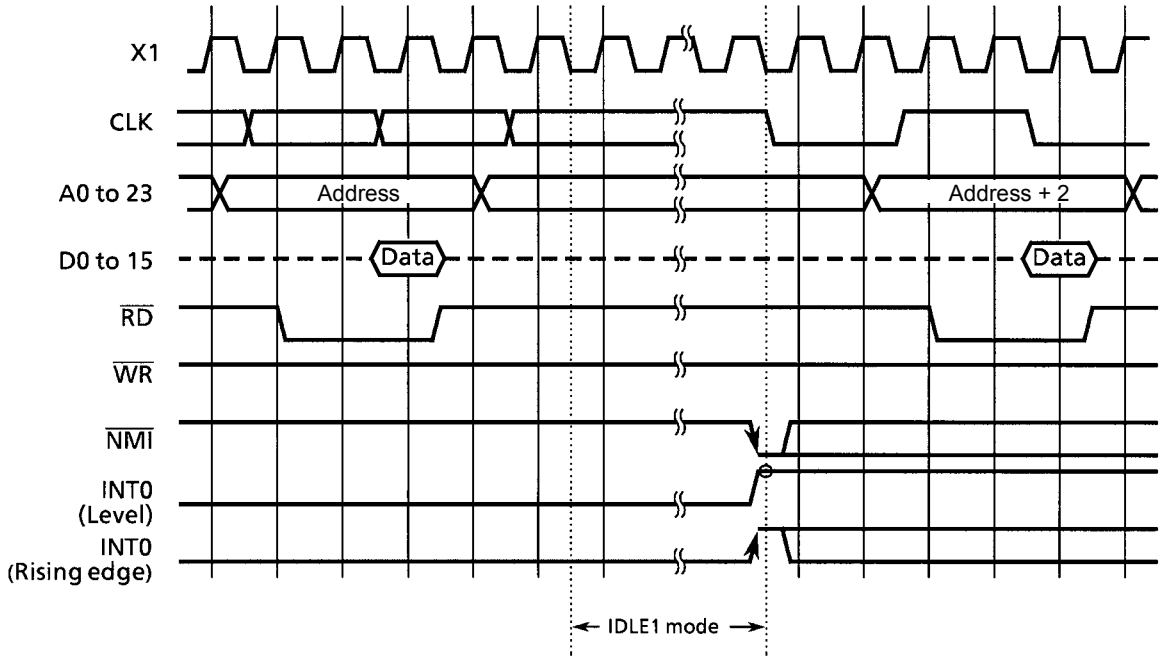


Figure 3.4.3 Example of Timing for Releasing HALT by Interrupt (IDLE1 Mode)

[4] STOP mode

In STOP mode, all internal circuits, including the internal oscillator, are halted. The pin states in STOP mode differ according to the setting of the watchdog timer mode register WDMOD<DRVE>. (For details on the WDMOD<DRVE> settings, see Figure 3.4.1). Table 3.4.3 shows the pin states in STOP mode.

Release STOP mode by an external interrupt (NMI, INT0). When releasing STOP mode, system clock output starts after the elapse of the warmup time (as set in the warmup counter) in order to stabilize the internal oscillator. Set the warmup time in the WDMOD<WARM> register.

Figure 3.4.4 shows an example of the timing for releasing HALT by interrupt.

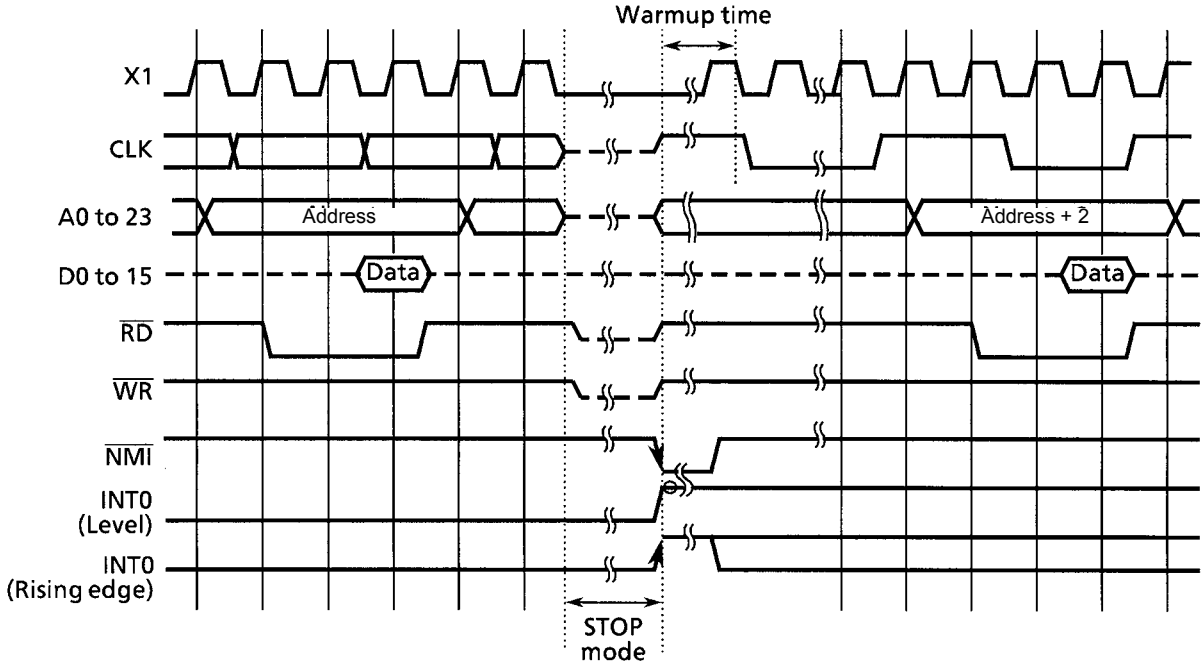


Figure 3.4.4 Example of Timing for Releasing HALT by Interrupt (STOP Mode)

Table 3.4.3 Pin States in Stop Mode

Pin Name	Input/Output	<DRVE> = 0	<DRVE> = 1
P00 to 07	Input mode Output mode Input/output (D0 to D7)	■ ■ −	■ Output −
P10 to 17	Input mode Output mode Input/output (D8 to D15)	■ ■ −	■ Output −
P20 to 27	Input mode Output mode Output (A16 to A23)	■ ■ −	■ Output Output
P30	Input mode Output mode Output (A8)	PU ■ −	PU Output Output
P31 to 37	Input mode Output mode Output (A9 to A15)	■ ■ −	■ Output Output
P40 to 47	Input mode Output mode Output (A0 to A7)	■ ■ −	■ Output Output
P50 ($\overline{RD}$), P51 ($\overline{WR}$)	Output mode Output ($\overline{RD}$, $\overline{WR}$)	■ −	Output High level output
P52 to 55	Input mode Output mode	PU* PU	PU Output
P56 (INT0)	Input mode Output mode Input mode (INT0)	PU PU Input	PU Output Input
P57 (CLKOUT)	Output mode Output (CLKOUT)	PU −	Output High level output
P60 to 63	Input mode Output mode	− −	Input Output
P70 to 75	Input mode Output mode	− −	Input Output
P80, 83, 86	Input mode Output mode	PU* PU*	PU Output
P81, 82, 84, 85, 87	Input mode Output mode	PU* PU	PU Output
P90 to 97	Input mode Output mode	− −	Input Output
PA0 to 7 (AN0 to 7)	Input Input ($\overline{ADTRG}$)	■ −	■ Input
$\overline{NMI}$	Input	Input	Input
CLK	Output	−	High level output
$\overline{RESET}$	Input	Input	Input
$\overline{EA}$	Input	Fixed to High level	Fixed to High level
AM8/$\overline{16}$	Input	Fixed to High level	Fixed to High level
X1	Input	−	−
X2	Output	High level	High level

−: Indicates that input is invalid for an input pin or a pin in input mode. Also, that the pin is set to high impedance for an output pin or a pin in output mode.

Input: The input gate is functioning. To prevent the input pin from floating, fix the input voltage to low or high.

Output: Output state

PU: Programmable pull-up pin. The input gate is functioning. Pins without pull-up set must be fixed to prevent through current.

PU*: Programmable pull-up pin. The input gate is disabled. A through current does not occur even if high impedance is set.

■: The input gate continues to operate if the HALT instruction is executed and the CPU is halted at the port register address value. To prevent a through current in this case, either fix the pin or ensure by software that the situation does not occur. In other cases, input is invalid.

X: Cannot be used.

Note: The port register controls the programmable pull-up. However, if the function is set for a pin shared with an output function (eg, TxD0), the pull-up selection for the pin depends on the output function data. For pins that are shared with input functions, the port register setting alone determines whether or not a pull-up resistor is used.

3.5 Port Functions

TMP95CU54A has a total of 81 bits for input/output ports.

As well as being used as general-purpose I/O ports, port pins are also used for internal CPU and built-in I/O functions. Table 3.5.1 lists port pin functions; Table 3.5.2, pin settings.

Table 3.5.1 Port Pin Functions

Port Name	Pin Name	Number of Pins	Direction	R	Direction Setting Unit	Pin Name for Built-In Function
Port 0	P00 to P07	8	Input/output	–	Bit	D0 to D7
Port 1	P10 to P17	8	Input/output	–	Bit	D8 to D15
Port 2	P20 to P27	8	Input/output	–	Bit	A16 to A23
Port 3	P30	1	Input/output	↑	Bit	A8
	P31 to P37	7	Input/output	–	Bit	A9 to A15
Port 4	P40 to P47	8	Input/output	–	Bit	A0 to A7
Port 5	P50	1	Output	–	(Fixed)	$\overline{RD}$
	P51	1	Output	–	(Fixed)	$\overline{WR}$
	P52	1	Input/output	↑	Bit	$\overline{HWR}$
	P53	1	Input/output	↑	Bit	$\overline{BUSRQ}$
	P54	1	Input/output	↑	Bit	$\overline{BUSAK}$
	P55	1	Input/output	↑	Bit	$\overline{WAIT}$
	P56	1	Input/output	↑	Bit	INT0
	P57	1	Output	↑	(Fixed)	CLKOUT
Port 6	P60	1	Input/output	–	Bit	$\overline{SS}$
	P61	1	Input/output	–	Bit	MOSI
	P62	1	Input/output	–	Bit	MISO
	P63	1	Input/output	–	Bit	SCLK
Port 7	P70	1	Input/output	–	Bit	TI0/INT1
	P71	1	Input/output	–	Bit	TO1
	P72	1	Input/output	–	Bit	TO3/INT2
	P73	1	Input/output	–	Bit	TI3/INT3
	P74	1	Input/output	–	Bit	TO5
	P75	1	Input/output	–	Bit	TO7/INT4
Port 8	P80	1	Input/output	↑	Bit	TxD0
	P81	1	Input/output	↑	Bit	RxD0
	P82	1	Input/output	↑	Bit	SCLK0/$\overline{CTS0}$
	P83	1	Input/output	↑	Bit	TxD1
	P84	1	Input/output	↑	Bit	RxD1
	P85	1	Input/output	↑	Bit	SCLK1/$\overline{CTS1}$
	P86	1	Input/output	↑	Bit	Tx
	P87	1	Input/output	↑	Bit	Rx
Port 9	P90	1	Input/output	–	Bit	TI8/INT5
	P91	1	Input/output	–	Bit	TI9/INT6
	P92	1	Input/output	–	Bit	TO8
	P93	1	Input/output	–	Bit	TO9
	P94	1	Input/output	–	Bit	TIA/INT7
	P95	1	Input/output	–	Bit	TIB/INT8
	P96	1	Input/output	–	Bit	TOA/TOB
Port A	PA0 to PA2	3	Input	–	(Fixed)	AN0 to AN2
	PA3	1	Input	–	(Fixed)	AN3 / $\overline{ADTRG}$
	PA4 to PA7	4	Input	–	(Fixed)	AN4 to AN7

R: ↑ = With programmable pull-up resistor

Note: P30 is pulled-up during reset and input mode.
P57 is pulled-up only during reset.

Table 3.5.2 Port Pin Setting Methods (1/3)

n: Corresponding port no. X: Don't care

Port Name	Pin Name	Function	Port Register Setting		
			Pn	PnCR	PnFC
Port 0	P00 to P07	Input port	X	0	None
		Output port	X	1	
		D0 to D7	X	X	
Port 1	P10 to P17	Input port	X	0	0
		Output port	X	1	0
		D8 to D15	X	0	1
Port 2	P20 to P27	Input port	X	0	X
		Output port	X	1	0
		A16 to A23	X	1	1
Port 3	P30	Input port (with pull-up)	X	0	X
		Output port	X	1	0
		A8	X	1	1
	P31 to P37	Input port	X	0	X
		Output port	X	1	0
		A9 to A15	X	1	1
Port 4	P40 to P47	Input port	X	0	X
		Output port	X	1	0
		A0 to A7	X	1	1
Port 5	P50	Output port	X	None	0
		$\overline{RD}$ output at external access only	1		1
		Always $\overline{RD}$ output	0		1
	P51	Output port	X		0
		$\overline{WR}$ output at external access only	X		1
	P52	Input port (no pull-up)	0	0	0
		Input port (with pull-up)	1	0	0
		Output port	X	1	0
		$\overline{HWR}$ output	X	1	1
	P53	Input port (no pull-up)	0	0	0
		Input port (with pull-up)	1	0	0
		Output port	X	1	X
		$\overline{BUSRQ}$ input (no pull-up)	0	0	1
		$\overline{BUSRQ}$ input (with pull-up)	1	0	1
	P54	Input port (no pull-up)	0	0	0
		Input port (with pull-up)	1	0	0
		Output port	X	1	X
		$\overline{BUSAK}$ output	X	1	1
	P55	Input port/$\overline{WAIT}$ input (no pull-up)	0	0	None
		Input port/$\overline{WAIT}$ input (with pull-up)	1	0	
		Output port	X	1	
	P56	Input port/INT0 input (no pull-up) (Note 1)	0	0	
		Input port/INT0 input (with pull-up) (Note 1)	1	0	
		Output port	X	1	
	P57	Output port	X	None	0
		CLKOUT output	X		1

Note 1: When using pin P56 as an INT0 input, enable interrupt input with interrupt input mode control register IIMC<I0LE>.

Table 3.5.2 Port Pin Setting Methods (2/3)

n: Corresponding port no. X: Don't care

Port Name	Pin Name	Function	Port Register Setting		
			Pn	PnCR	PnFC
Port 6	P60	Input port/$\overline{SS}$ input (slave)	X	0	X
		$\overline{SS}$ input (master mode-fault enable)	X	0	0
		$\overline{SS}$ input (master mode-fault disable)	X	0	1
		Output port	X	1	X
	P61	Input port/MOSI input (slave)	X	0	0
		Output port	X	1	0
		MOSI output (master) (Note 2)	X	1	1
	P62	Input port/MISO input (master)	X	0	0
		Output port	X	1	0
		MISO output (slave) (Note 2)	X	1	1
	P63	Input port/SCLK input (slave)	X	0	0
		Output port	X	1	0
		SCLK output (master) (Note 2)	X	1	1
Port 7	P70	Input port/TI0/INT1 input	X	0	None
		Output port	X	1	
	P71	Input port	X	0	X
		Output port	X	1	0
		TO1 output	X	1	1
	P72	Input port/INT2 input	X	0	X
		Output port	X	1	0
		TO3 output	X	1	1
	P73	Input port/TI4/INT3 input	X	0	None
		Output port	X	1	
	P74	Input port	X	0	X
		Output port	X	1	0
		TO5 output	X	1	1
	P75	Input port/INT4 input	X	0	X
		Output port	X	1	0
		TO7 output	X	1	1
Port 8	P80	Input port (no pull-up)	0	0	0
		Input port (with pull-up)	1	0	0
		Output port	X	1	0
		TxD0 output (Note 2)	X	1	1
	P81	Input port/RxD0 input (no pull-up)	0	0	None
		Input port/RxD0 input (with pull-up)	1	0	
		Output port	X	1	
	P82	Input port/SCLK0/$\overline{CTS0}$ input (no pull-up)	0	0	0
		Input port/SCLK0/$\overline{CTS0}$ input (with pull-up)	1	0	0
		Output port	X	1	0
		SCLK0 output	X	1	1
	P83	Input port (no pull-up)	0	0	0
		Input port (with pull-up)	1	0	0
		Output port	X	1	0
		TxD1 output (Note 2)	X	1	1

Note 2: Open drain enable register ODE<ODE4:0> is used to set the open drain output mode for pins TxD0, TxD1, MOSI, MISO, and SCLK.

Table 3.5.2 Port Pin Setting Methods (3/3)

n: Corresponding port no. X: Don't care

Port Name	Pin Name	Function	Port Register Setting		
			Pn	PnCR	PnFC
Port 8	P84	Input port/ RxD1 input (no pull-up)	0	0	None
		Input port/ RxD1 input (with pull-up)	1	0	
		Output port	X	1	
	P85	Input port/SCLK1/$\overline{\text{CTS1}}$ input (no pull-up)	0	0	0
		Input port/SCLK1/$\overline{\text{CTS1}}$ input (with pull-up)	1	0	0
		Output port	X	1	0
		SCLK1 output	X	1	1
	P86	Input port (no pull-up)	0	0	0
		Input port (with pull-up)	1	0	0
		Output port	X	1	0
		Tx output	X	1	1
	P87	Input port/ Rx input (no pull-up)	0	0	None
		Input port/ Rx input (with pull-up)	1	0	
		Output port	X	1	
Port 9	P90	Input port/TI8/INT5 input	X	0	None
		Output port	X	1	
	P91	Input port/TI9/INT6 input	X	0	
		Output port	X	1	
	P92	Input port	X	0	X
		Output port	X	1	0
		TO8 output	X	1	1
	P93	Input port	X	0	X
		Output port	X	1	0
		TO9 output	X	1	1
	P94	Input port/TIA/INT7 input	X	0	None
		Output port	X	1	
	P95	Input port/TIB/INT8 input	X	0	
		Output port	X	1	
	P96	TOA/TOB output (Note 3)	X	1	1
Port A	PA0 to PA7	Input port	X	None	
		AN0 to AN7 input (Note 4)	X		

Note 3: P9FC<TOS1> is used to switch between the TOA and TOB timer outputs to pin P96.

Note 4: When PA0 to PA7 are used as AD converter input channels, AD mode control register 1 ADMOD1<ADCH2:0> is used to select the channel.

3.5.1 Port 0 (P00 to P07)

Port 0 is an 8-bit general-purpose input/output port with each port bit settable as an input or output.

In addition to functioning as a general-purpose input/output port, port 0 also functions as the data bus (D0 to D7). The port 0 control register P0CR sets the pins as inputs or outputs.

A reset sets all the bits of the P0CR register to 0, and sets all pins to input mode.

When external memory is accessed, the port automatically functions as the data bus (D0 to D7) and all bits of P0CR are cleared to 0.

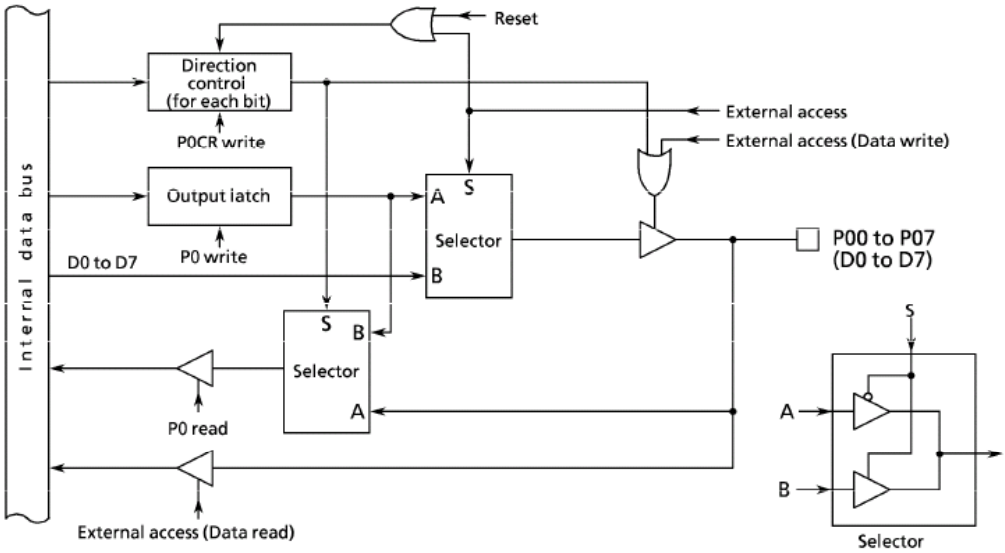


Figure 3.5.1 Port 0 (P00 to P07)

Port 0 Register

		7	6	5	4	3	2	1	0
P0 (0000H)	bit Symbol	P07	P06	P05	P04	P03	P02	P01	P00
	Read/Write	R/W							
	After reset	Input mode (output latch register undefined)							
	Function	Also functions as D7 to D0							

Port 0 Control Register

		7	6	5	4	3	2	1	0
P0CR (0002H)	bit Symbol	P07C	P06C	P05C	P04C	P03C	P02C	P01C	P00C
	Read/Write	W							
Read-modify-write instructions prohibited.	After reset	0	0	0	0	0	0	0	0
	Function	Port 0 input/output settings 0: Input 1: Output							

Note: When functioning as a data bus (D0 to D7), P0CR is cleared to 0.

Figure 3.5.2 Register for Port 0

3.5.2 Port 1 (P10 to P17)

Port 1 is an 8-bit general-purpose input/output port with each port bit settable as an input or output.

In addition to functioning as a general-purpose input/output port, port 1 also functions as a data bus (D8 to D15). The port 1 control register P1CR and function register P1FC set the port 1 functions.

Reset sets all the bits of the P1 output latch register and all bits of the P1CR and P1FC registers to 0, and sets port 1 to input mode.

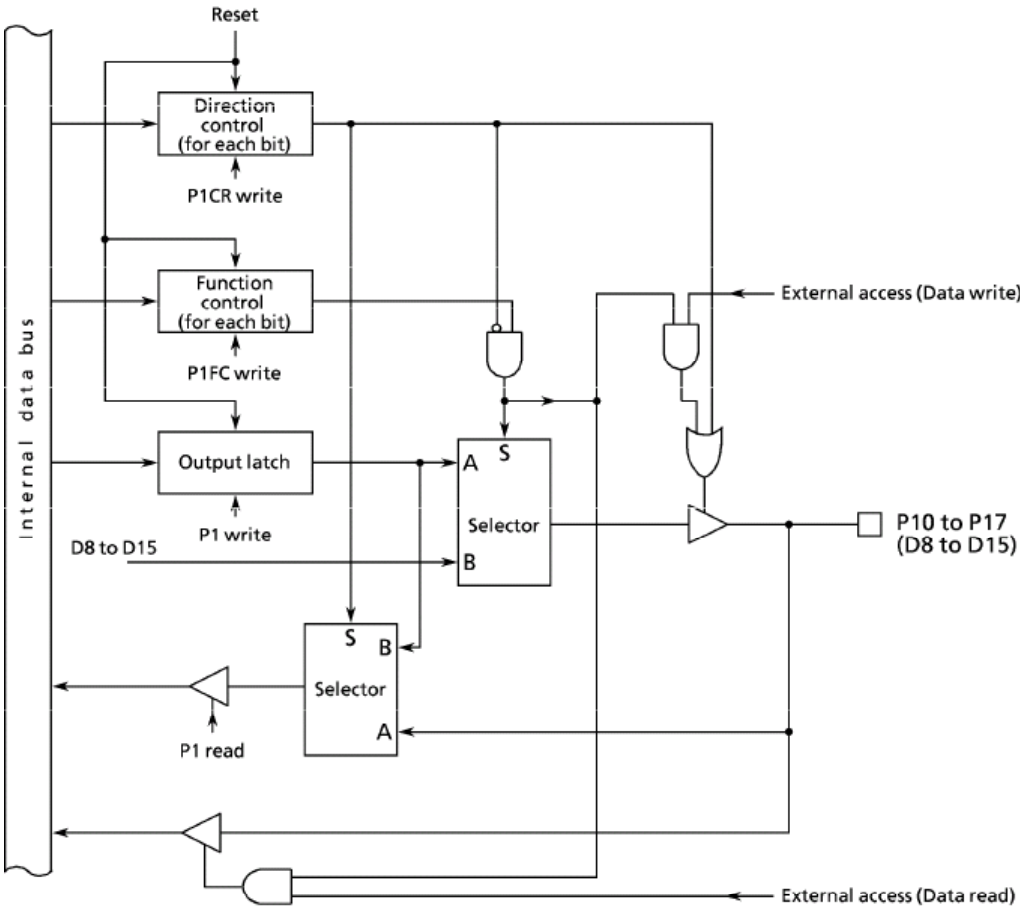


Figure 3.5.3 Port 1 (P10 to P17)

Port 1 Register

P1 (0001H)

	7	6	5	4	3	2	1	0
bit Symbol	P17	P16	P15	P14	P13	P12	P11	P10
Read/Write	R/W							
After reset	Input mode (output latch register cleared to 0)							
Function	Also functions as D15 to D8							

Port 1 Control Register

P1CR (0004H)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Port 1 function settings							

Port 1 Function Register

P1FC (0005H)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol	P17F	P16F	P15F	P14F	P13F	P12F	P11F	P10F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Port 1 function settings							

Port 1 function settings

P1FC<P1xF> / P1CR<P1xC>	0	1
0	Input port	Data bus (D15 to D8)
1	Output port	Do not set

Figure 3.5.4 Register for Port 1

3.5.3 Port 2 (P20 to P27)

Port 2 is an 8-bit general-purpose input/output port with each port bit settable as an input or output.

In addition to functioning as a general-purpose input/output port, port 2 also functions as an address bus (A16 to A23). The port 2 control register P2CR and function register P2FC set the port 2 functions.

Reset sets all the bits of the P2 output latch register and all bits of the P2CR and P2FC registers to 0, setting port 2 to input mode.

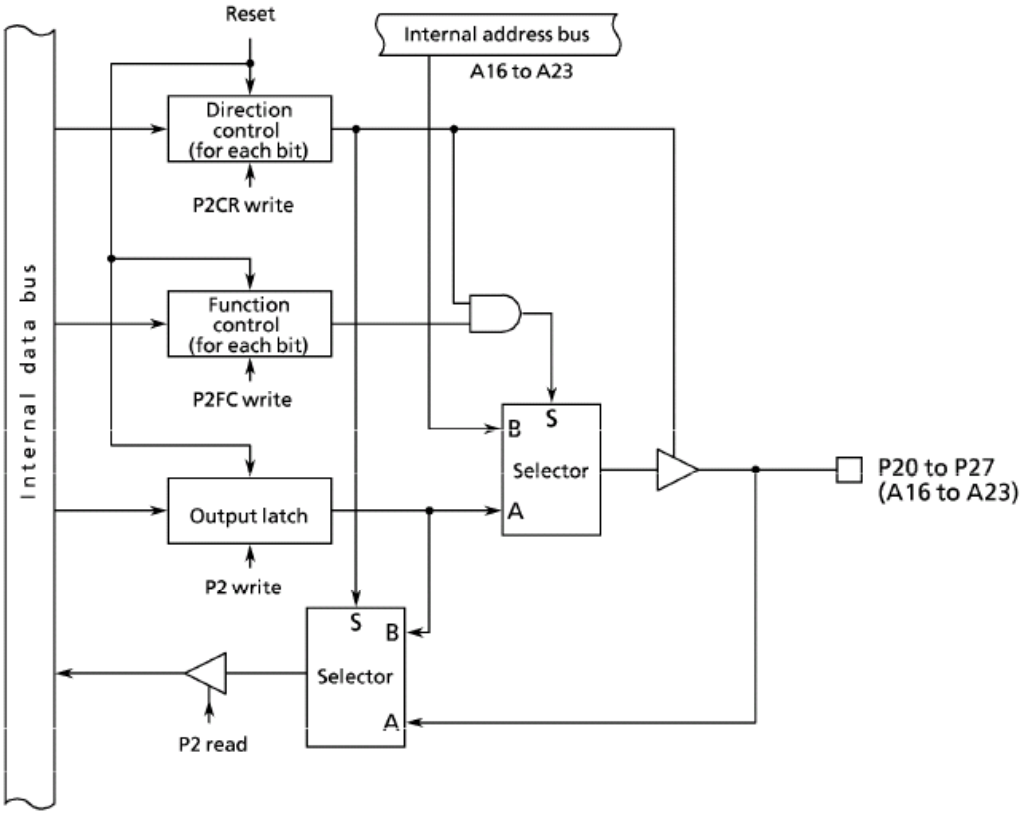


Figure 3.5.5 Port 2 (P20 to P27)

Port 2 Register

P2 (0006H)

	7	6	5	4	3	2	1	0
bit Symbol	P27	P26	P25	P24	P23	P22	P21	P20
Read/Write	R/W							
After reset	Input mode (output latch register set to 0)							
Function	Also functions as A23 to A16							

Port 2 Control Register

P2CR (0008H)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol	P27C	P26C	P25C	P24C	P23C	P22C	P21C	P20C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Port 2 function settings							

Port 2 Function Register

P2FC (0009H)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol	P27F	P26F	P25F	P24F	P23F	P22F	P21F	P20F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Port 2 function settings							

Port 2 function settings

P2CR<P2xC> \ P2FC<P2xF>	0	1
0	Input port	
1	Output port	Address bus (A23 to A16)

Note: When setting the address bus (A23 to A16), first set P2CR, then P2FC.

Figure 3.5.6 Register for Port 2

3.5.4 Port 3 (P30 to P37)

Port 3 is an 8-bit general-purpose input/output port with each port bit settable as an input or output.

In addition to functioning as a general-purpose input/output port, port 3 also functions as an address bus (A8 to A15). The port 3 control register P3CR and function register P3FC set the port 3 functions.

Reset sets all the bits of the P3 output latch register and all bits of the P3CR and P3FC registers to 0. Pin P30 is set to input mode with pull-up; P31 to P37 are set to input mode.

(1) Port 30 (A8)

In addition to being a general-purpose input/output port, port 30 also functions as an address bus (A8).

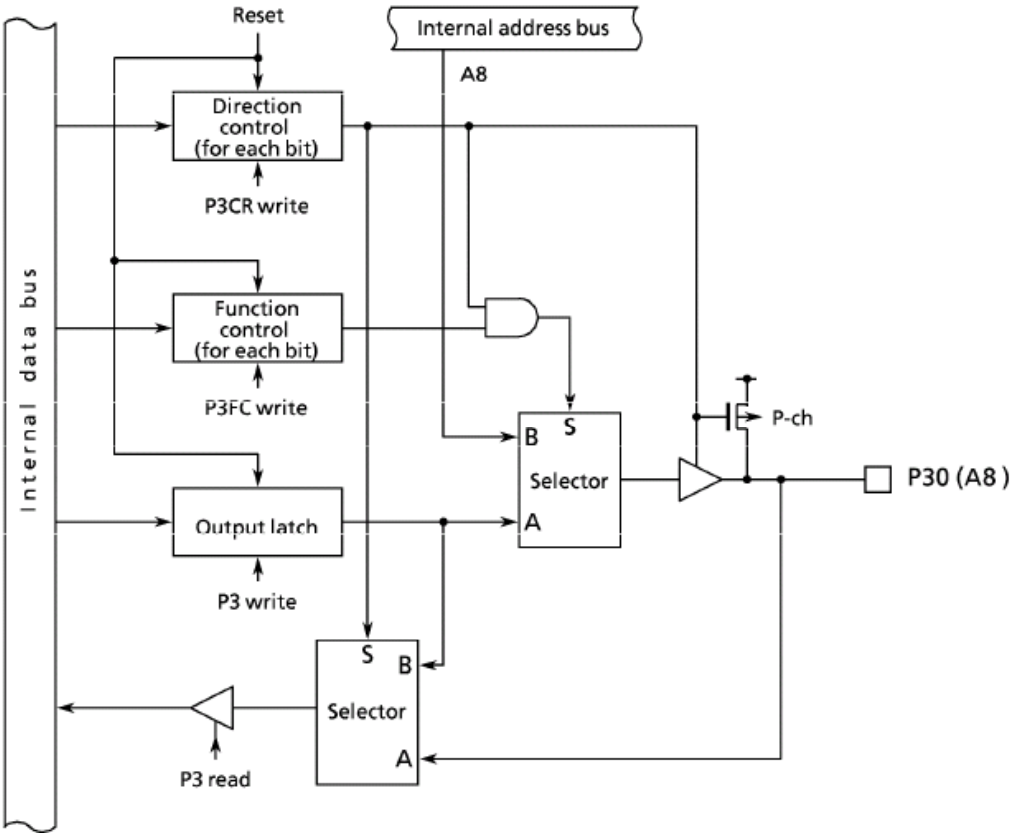


Figure 3.5.7 Port 3 (P30)

(2) Port 31 to 37 (A9 to A15)

In addition to functioning as a general-purpose input/output port, port 31 to 37 also functions as an address bus (A9 to A15).

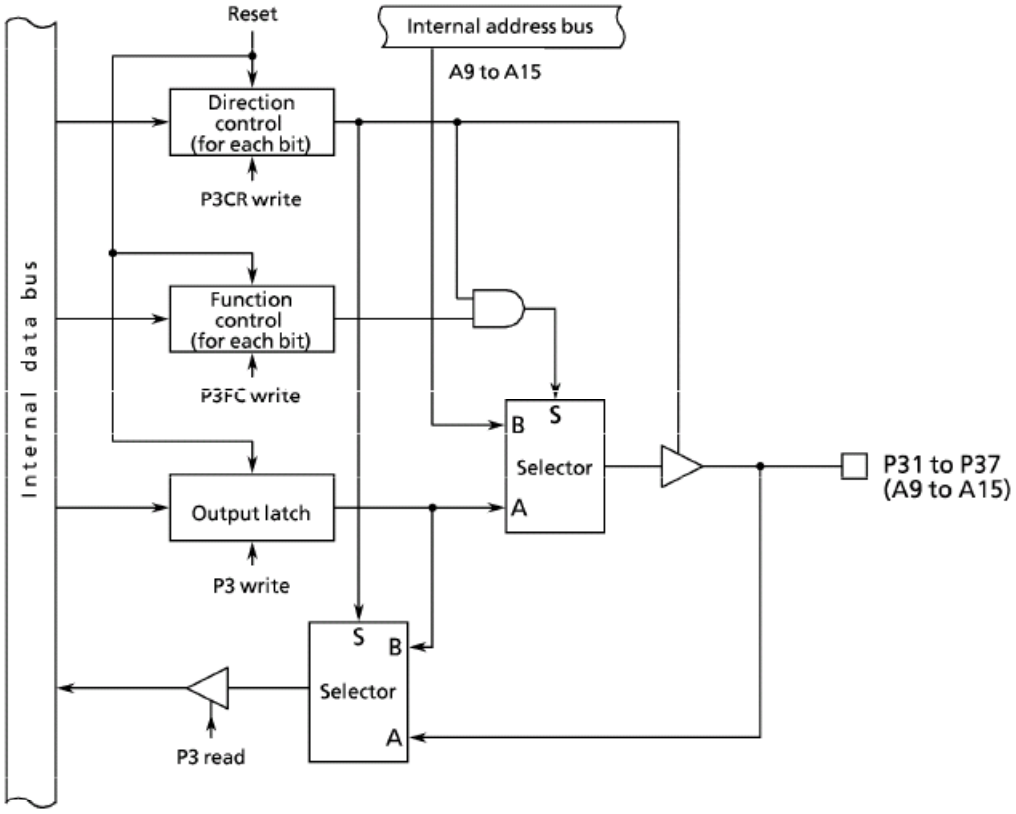


Figure 3.5.8 Port 3 (P31 to P37)

Port 3 Register

P3 (0007H)

	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 latch register cleared to 0)							
Function	Also functions as A15 to A8							

Port 3 Control Register

P3CR (000AH)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol	P37C	P36C	P35C	P34C	P33C	P32C	P31C	P30C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Port 3 function settings							

Port 3 Function Register

P3FC (000BH)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol	P37F	P36F	P35F	P34F	P33F	P32F	P31F	P30F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Port 3 function settings							

Port 3 function settings

P3CR<P3xC> \ P3FC<P3xF>	0	1
0	Input port (P30 is pull-up)	
1	Output port	Address bus (A15 to A8)

Note: When setting the address bus (A15 to A8), first set P3CR, then P3FC.

Figure 3.5.9 Register for Port 3

3.5.5 Port 4 (P40 to P47)

Port 4 is an 8-bit general-purpose input/output port with each port bit settable as an input or output.

In addition to functioning as a general-purpose input/output port, port 4 also functions as an address bus (A0 to A7). The port 4 control register P4CR and function register P4FC set the port 4 functions.

Reset sets all the bits of the P4 output latch register and all bits of the P4CR and P4FC registers to 0, setting port 4 to input mode.

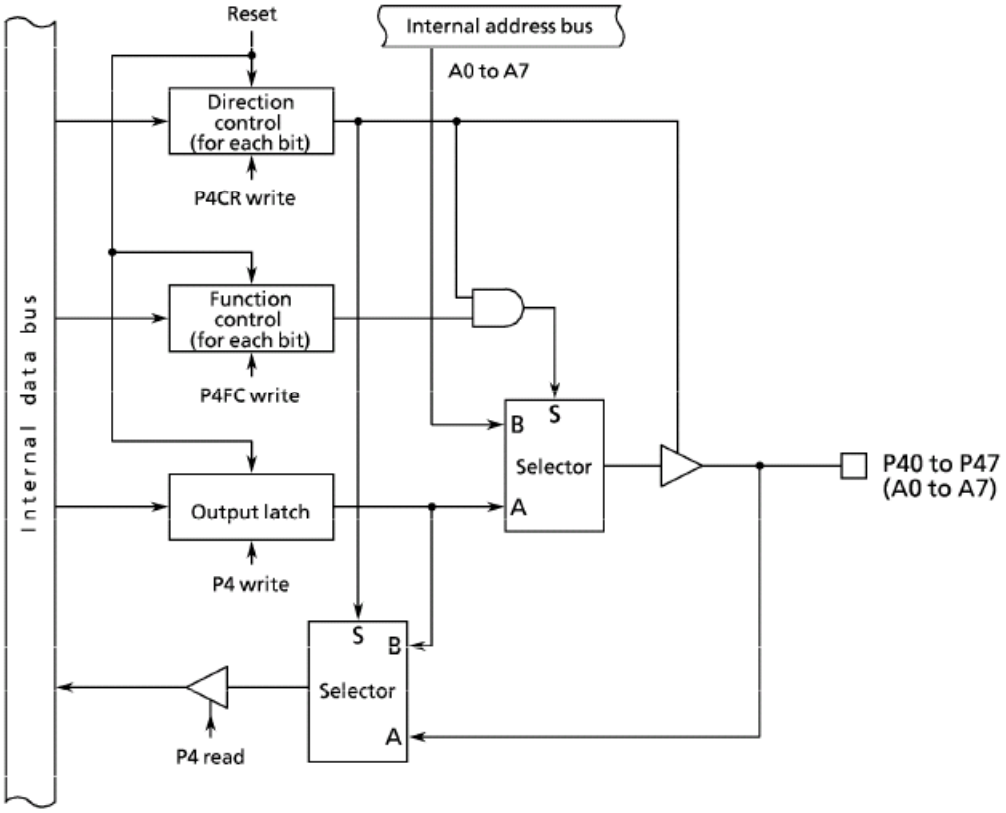


Figure 3.5.10 Port 4 (P40 to P47)

Port 4 Register

P4 (000CH)

	7	6	5	4	3	2	1	0
bit Symbol	P47	P46	P45	P44	P43	P42	P41	P40
Read/Write	R/W							
After reset	Input mode (output latch register cleared to 0)							
Function	Also functions as A7 to A0							

Port 4 Control Register

P4CR (000EH)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol	P47C	P46C	P45C	P44C	P43C	P42C	P41C	P40C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Port 4 function settings							

Port 4 Function Register

P4FC (000FH)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol	P47F	P46F	P45F	P44F	P43F	P42F	P41F	P40F
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Port 4 function settings							

Port 4 function settings

P4CR<P4xC> \ P4FC<P4xF>	0	1
0	Input port	
1	Output port	Address bus (A7 to A0)

Note: When setting the address bus (A7 to A0), first set P4CR, then P4FC.

Figure 3.5.11 Register for Port 4

3.5.6 Port 5 (P50 to P57)

Port 5 is an 8-bit general-purpose input/output port with each port bit settable as an input or output. However, P50, P51 and P57 are output-only ports.

In addition to functioning as a general-purpose input/output port, port 5 also has a CPU control/status signal input/output function, a $\overline{\text{WAIT}}$ input function, an INT0 external interrupt input function, and a CLKOUT output function. The port 5 control register P5CR and function register P5FC set the port 5 functions.

Reset sets all the bits of the P5 output latch register and bit 7 of P5FC to 1 and clears all bits of P5CR (bits 0, 1 and 7 are unused) and bits 0, 1, 2, 3 and 4 of P5FC (bits 5 and 6 are unused) to 0. Pins P50 and P51 output 1 and P52 to P56 are set to input mode with resistors pulled up and P57 output CLKOUT.

When P50 is set as the $\overline{\text{RD}}$ pin (when P5FC<P50F> = 1) when P5<P50> is cleared to 0, the P50 $\overline{\text{RD}}$ signal is output even when an internal address area is accessed, and external PSRAM (pseudo SRAM) can be refreshed. If <P50> is set to 1, the $\overline{\text{RD}}$ signal is output only when an external area is accessed.

(1) Port 50 ($\overline{\text{RD}}$)

In addition to functioning as a general-purpose output-only port, port 50 can also function as the $\overline{\text{RD}}$ pin.

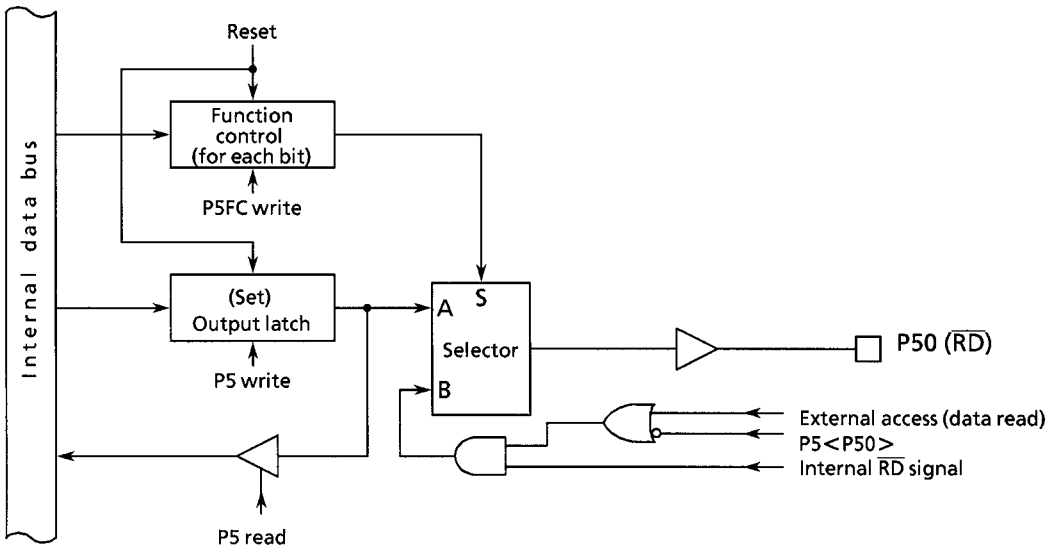


Figure 3.5.12 Port 5 (P50)

(2) Port 51 ($\overline{WR}$)

In addition to functioning as a general-purpose output-only port, port 51 can also function as the $\overline{WR}$ pin.

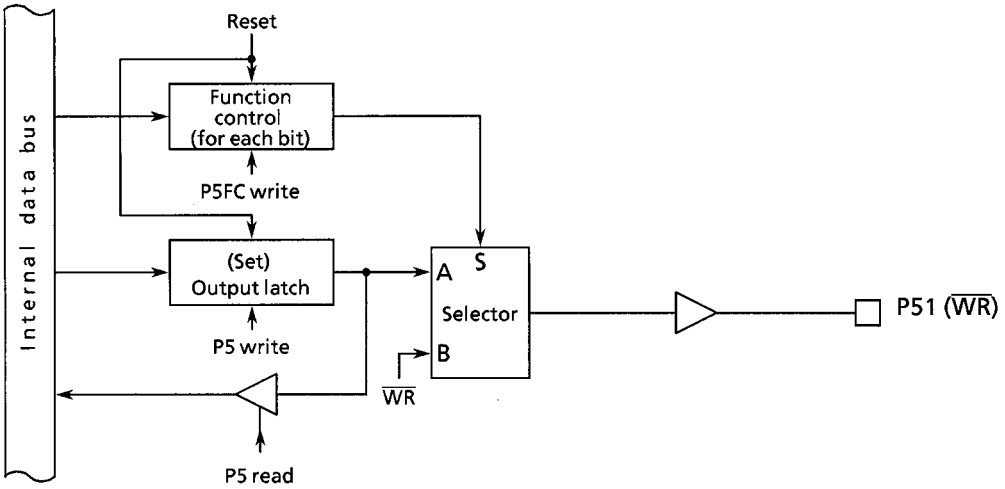


Figure 3.5.13 Port 5 (P51)

(3) Ports 52, 54 ($\overline{HWR}$, $\overline{BUSAK}$)

In addition to being general-purpose input/output ports, port 52 can also function as the $\overline{HWR}$ pin, and port 54 can also function as the $\overline{BUSAK}$ pin.

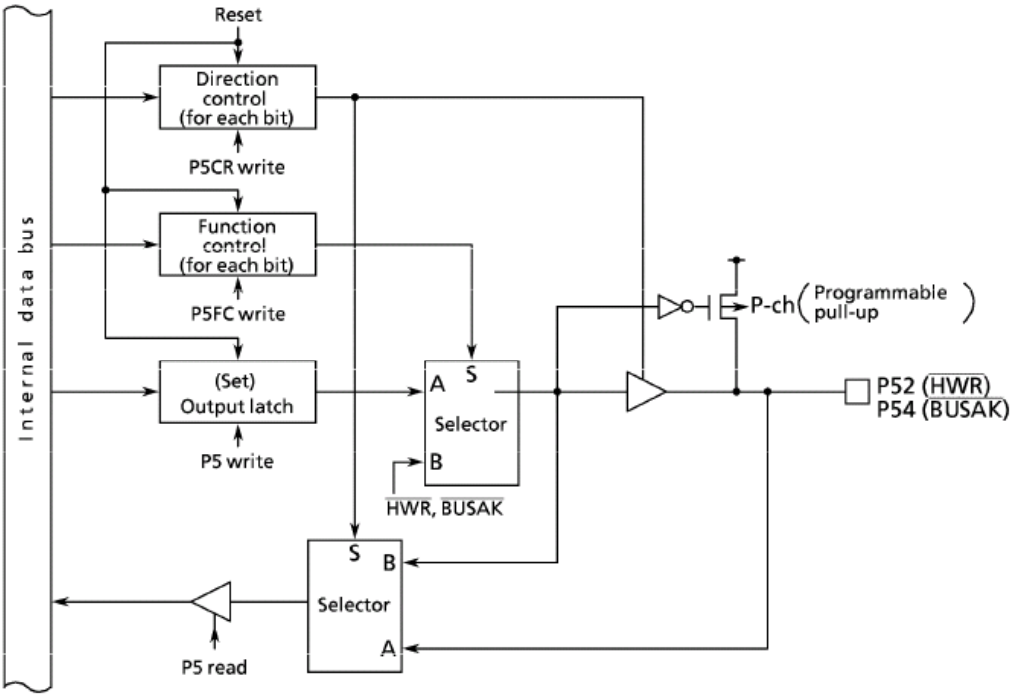


Figure 3.5.14 Port 5 (P52, P54)

(4) Port 53 ($\overline{\text{BUSRQ}}$)

In addition to being a general-purpose input/output port, port 53 also functions as the $\overline{\text{BUSRQ}}$ pin.

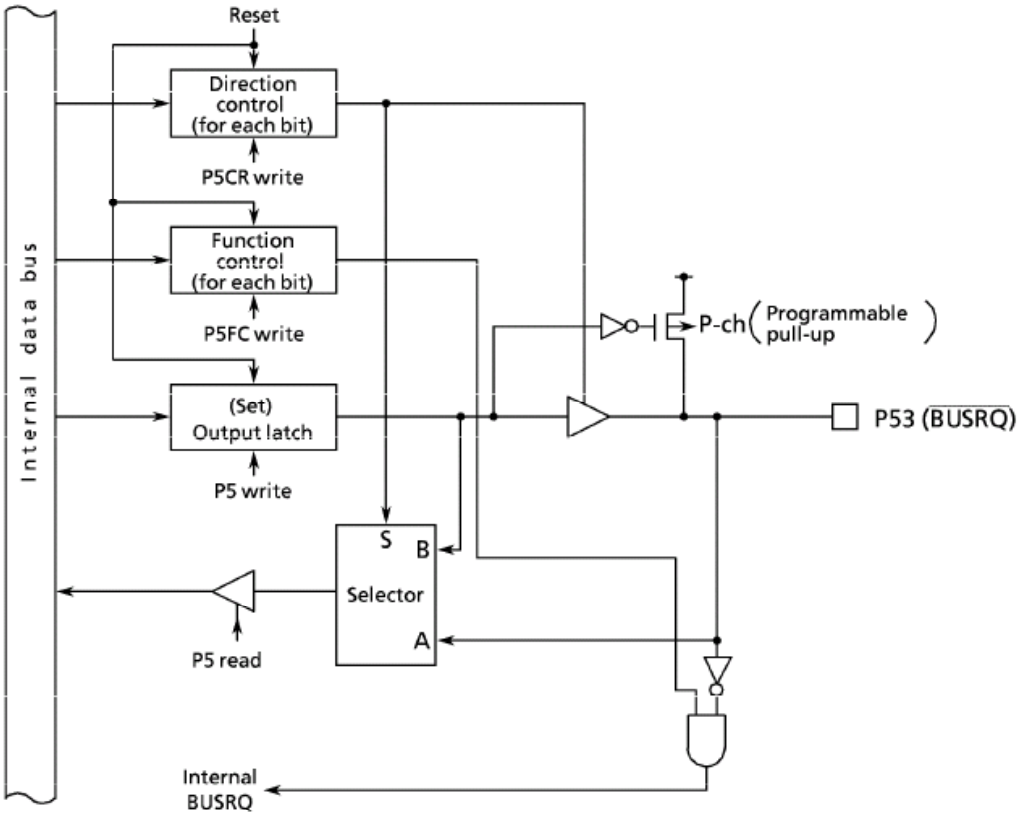


Figure 3.5.15 Port 5 (P53)

(5) Port 55 ($\overline{\text{WAIT}}$)

In addition to being a general-purpose input/output port, port 55 also functions as the $\overline{\text{WAIT}}$ pin.

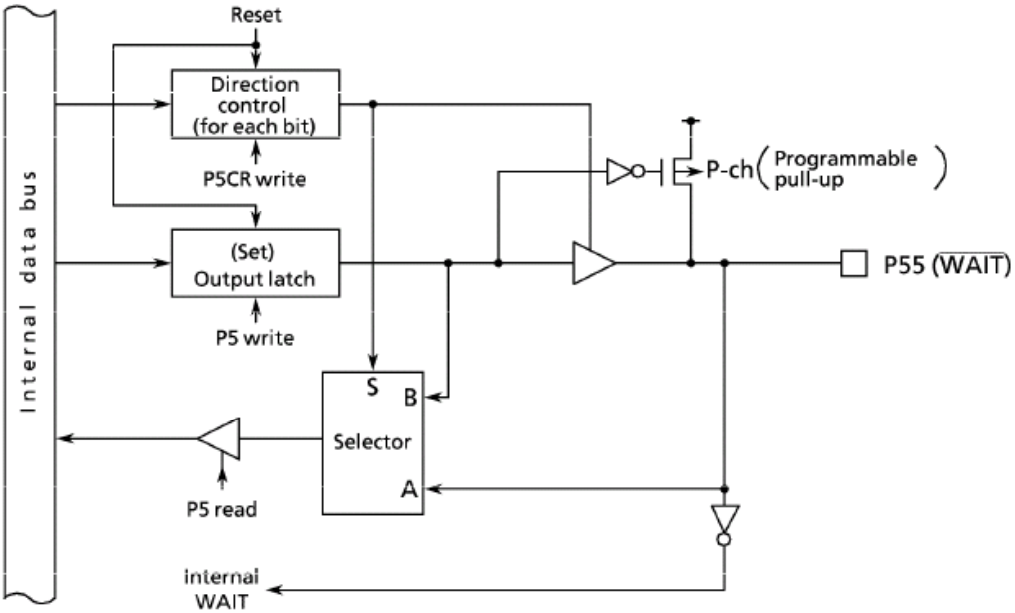


Figure 3.5.16 Port 5 (P55)

(6) Port 56 (INT0)

In addition to being a general-purpose input/output port, port 56 also functions as the external interrupt request input INT0 pin.

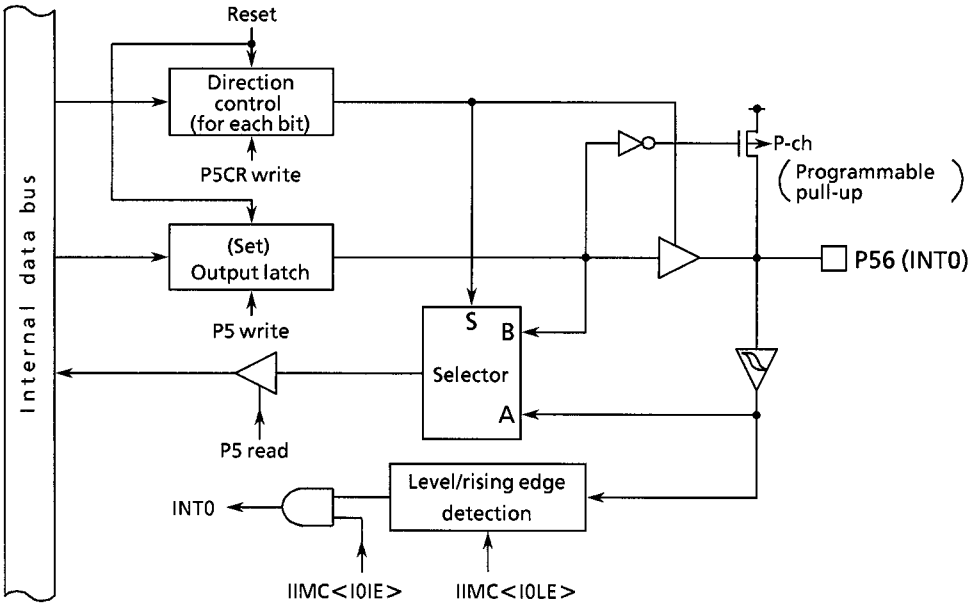
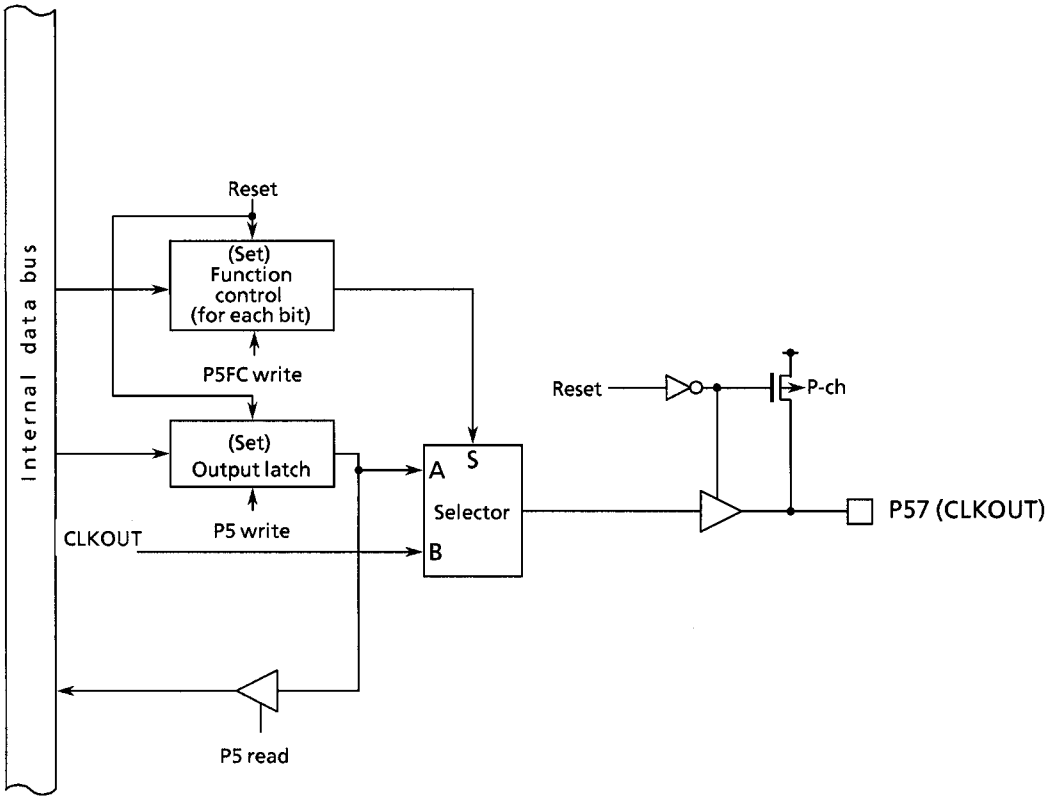


Figure 3.5.17 Port 5 (P56)

(7) Port 57 (CLKOUT)

In addition to being a general-purpose output port, port 57 also functions as the CLKOUT output pin.



Note: During the reset, port 57 is pulled up. After the reset, port 57 functions as the CLKOUT output pin.

Figure 3.5.18 Port 5 (P57)

Port 5 Register

P5 (000DH)

	7	6	5	4	3	2	1	0
bit Symbol	P57	P56	P55	P54	P53	P52	P51	P50
Read/Write	R/W							
After reset	Output only (set to 1)	Input mode (Set to 1 / pulled up)					Output only (set to 1)	
Function	Also functions as CLKOUT	Also functions as INT0	Also functions as $\overline{WAIT}$	Also functions as $\overline{BUSAK}$	Also functions as $\overline{BUSRQ}$	Also functions as $\overline{HWR}$	Also functions as $\overline{WR}$	Also functions as $\overline{RD}$

Note: When port 5 is in input mode, the internal pull-up resistor is controlled by the P5 register. When using port 5 in input mode or in both input and output modes (if just one bit is set to input mode), read-modify-write instructions cannot be executed. The internal pull-up resistor setting may change depending on the state of the input pin.

Port 5 Control Register

P5CR (0010H)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol		P56C	P55C	P54C	P53C	P52C		
Read/Write		W						
After reset		0	0	0	0	0		
Function		Port 56 to 52 input/output settings 0: Input 1: Output						

Port 56 function settings (Note 2)

<P56C> \ <P56>	0	1
0	Input port / INT0 input	Input port / INT0 input (pull-up)
1	Output port	

Port 55 function settings (Note 1)

<P55C> \ <P55>	0	1
0	Input port / $\overline{WAIT}$ input	Input port / $\overline{WAIT}$ input (pull-up)
1	Output port	

Note 1: When using port 55 as the $\overline{WAIT}$ pin, set P5CR<P55C> to 0 and set the bus width/wait control register WAITCx<BxW2:0> to 010 (1 WAIT + N) or 100 (0 + N WAIT).

Note 2: When using port 56 as the INT0 pin, set P5CR<P56C> to 0 and set the interrupt input mode control register IIMC<I0IE> to 1.

Figure 3.5.19 Register for Port 5 (1/2)

Port 5 Function Register

P5FC (0011H)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol	P57F			P54F	P53F	P52F	P51F	P50F
Read/Write	W			W				
After reset	0			0	0	0	0	0
Function	0: Port 1: CLKOUT			0: Port 1: $\overline{BUSAK}$	0: Port 1: $\overline{BUSRQ}$	0: Port 1: $\overline{HWR}$	0: Port 1: $\overline{WR}$	0: Port 1: $\overline{RD}$

Port 53 function settings

<P53C> \ <P53F>	0	1
0	Input port (When <P53> = 0, not pulled up; when <P53> = 1, pulled up)	$\overline{BUSRQ}$ input (When <P53> = 0, not pulled up; when <P53> = 1, pulled up)
1	Output port	

Port 50 function settings

<P50F> \ <P50>	0	1
0	Output port	
1	Normally $\overline{RD}$ output (for PSRAM)	$\overline{RD}$ output only at external access

Port 51 function settings

0	Output port
1	$\overline{WR}$ output only at external access

Port 54 function settings

<P54C> \ <P54F>	0	1
0	Input port (When <P54> = 0, not pulled up; when <P54> = 1, pulled up)	Do not set
1	Output port	$\overline{BUSAK}$ output

Port 52 function settings

<P52C> \ <P52F>	0	1
0	Input port (When <P52> = 0, not pulled up; when <P52> = 1, pulled up)	Do not set
1	Output port	$\overline{HWR}$ output

Port 57 function settings

0	Output port
1	CLKOUT output

Figure 3.5.19 Register for Port 5 (2/2)

3.5.7 Port 6 (P60 to P63)

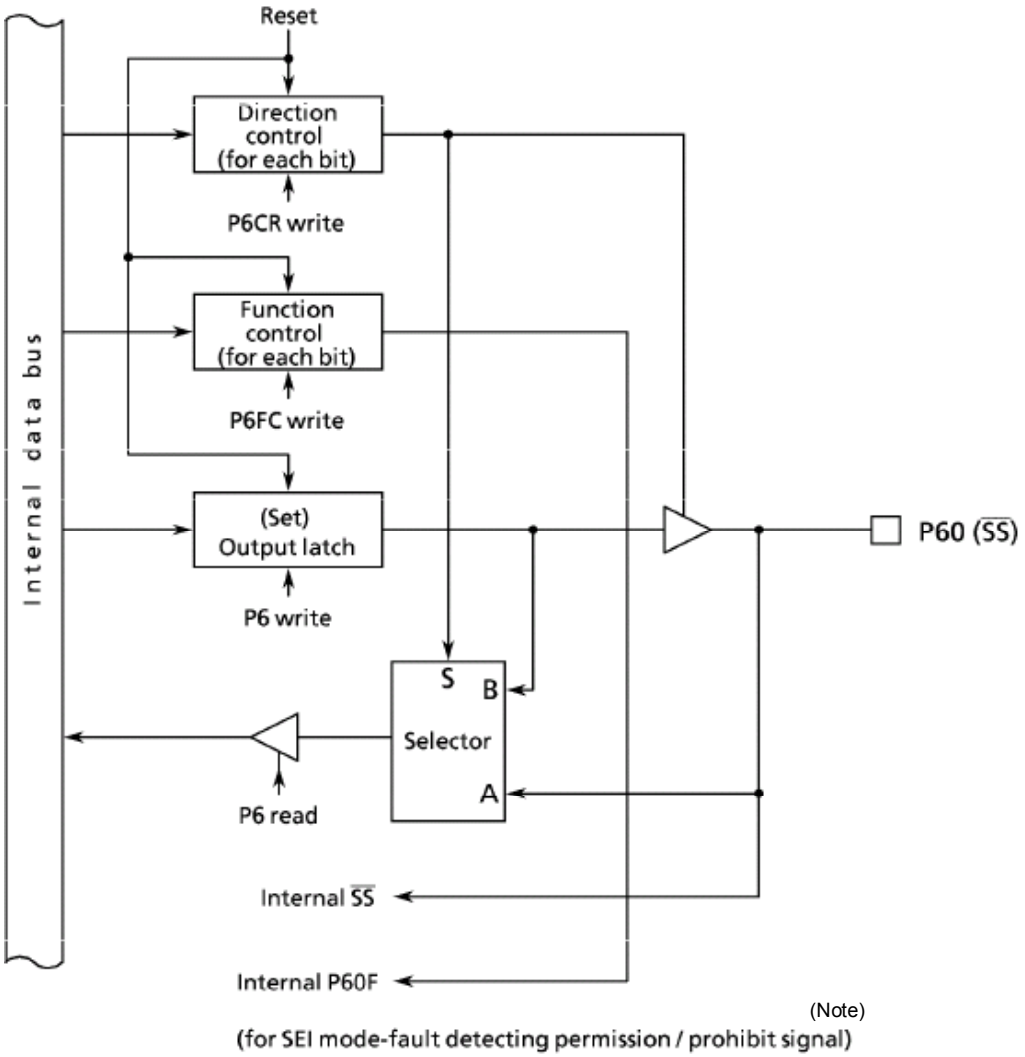
Port 6 is a 4-bit general-purpose input/output port with each bit settable as an input or output.

In addition to functioning as a general-purpose input/output port, port 6 also has a serial expansion interface function ($\overline{SS}$, MOSI, MISO and SCLK). The port 6 control register P6CR and the port 6 function register P6FC set the functions.

Reset sets the P60 to P63 output latch to 1. Reset also clears all bits of the P6CR and P6FC register to 0, setting port 6 to a general-purpose input port.

(1) Port 60 ($\overline{SS}$)

In addition to being a general-purpose input/output port, port 60 also functions as the $\overline{SS}$ pin.



Note: There is no Mode fault detection.

Figure 3.5.20 Port 6 (P60)

(2) Port 61, 62, 63 (MOSI, MISO, SCLK)

In addition to being general-purpose input/output ports, port 61 also functions as the MOSI pin, port 62 also functions as the MISO pin, and port 63 also functions as the SCLK pin.

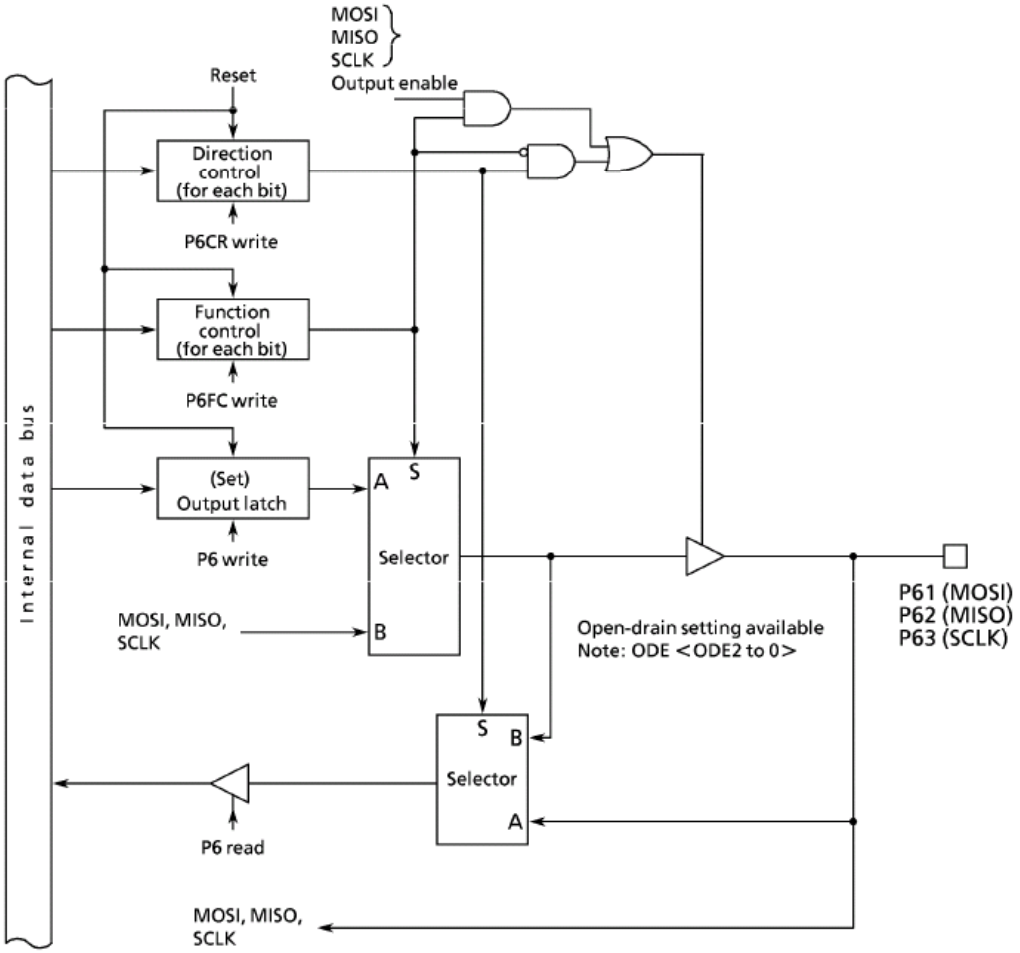


Figure 3.5.21 Port 6 (P61, P62, P63)

Port 6 Register

P6 (0012H)

	7	6	5	4	3	2	1	0
bit Symbol					P63	P62	P61	P60
Read/Write					R/W			
After reset					Input mode (set to 1)			
Function					Also functions as SCLK.	Also functions as MISO.	Also functions as MOSI.	Also functions as $\overline{SS}$.

Port 6 Control Register

P6CR (0014H)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol					P63C	P62C	P61C	P60C
Read/Write					W			
After reset					0	0	0	0
Function					Port 6 input/output settings 0:Input 1:Output			

Port 6 Function Register

P6FC (0015H)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol					P63F	P62F	P61F	P60F
Read/Write					W			
After reset					0	0	0	0
Function					0: Port 1: SCLK	0: Port 1: MISO	0: Port 1: MOSI	mode-fault detect 0: enable 1: disable

Port 63 function settings

<P63F> <P63C>	0	1
0	Input port SCLK (slave)	Do not set
1	Output port	SCLK (master)

Port 62 function settings

<P62F> <P62C>	0	1
0	Input port MISO (master)	Do not set
1	Output port	MISO (Slave)

Port 61 function settings

<P61F> <P61C>	0	1
0	Input port MOSI (slave)	Do not set
1	Output port	MOSI (master)

Port 60 function settings (Note2)

<P60F>	mode fault detect
0	enable (master)
1	disable (master)

Note: When setting pins MOSI, MISO, and SCLK to open drain output, set the open drain enable register ODE <ODE2:0> to 1.
Pins P60/$\overline{SS}$ (master/slave), P61/MOSI (slave), P62/MISO (master), and P63/SCLK (slave) do not have port/function switching registers. Therefore even when the pins are used as input ports, data are still input to SEI as function data.

Note2:There is no Mode fault detection. Set >P60F>, which is the enable/disable bit for Mode fault detection, to "1" to disable the Mode fault detection function.

Figure 3.5.22 Register for Port 6

3.5.8 Port 7 (P70 to P75)

Port 7 is a 6-bit general-purpose input/output port with each port bit settable as an input or output.

In addition to functioning as general-purpose input/output port pins, port 7 pins also function as event count inputs for the 8-bit timer, outputs for the 8-bit timer, and INT1 to 4 inputs for the external interrupt function.

Port 7 control register P7CR and port 7 function register P7FC set the port 7 functions.

Reset clears all bits of the output latch register and P7CR to 0, setting all pins to input mode.

To enable the timer output function, write 1 to the corresponding bits in both P7CR and P7FC.

(1) Port 70, 73 (TI0/INT1, T14/INT3)

In addition to functioning as a general-purpose input/output port, port 70 can also function as the event count input TI0 for timer 0 and as the external interrupt request input INT1.

In addition to functioning as a general-purpose input/output port, port 73 can also function as the event count input TI4 for timer 4 and as the external interrupt request input INT3.

Caution when using INT1 and INT3 interrupts

Input is always enabled for the INT1 and INT3 external interrupt requests.

Caution is required if port 70 or 73 is used as a general-purpose input/output port or a timer event count input while the INT1 and INT3 interrupt functions are in use. This is because rising edges on these input/output signals generate interrupt requests.

Caution when using timer event count inputs TI0 and TI4

Input is always enabled for the timer event count inputs TI0 and TI4.

Caution is required if port 70 or 73 is used as a general-purpose input/output port or an INT1 or INT3 interrupt during event counting based on TI0 or TI4. This is because these input/output signals trigger an event count on the timer.

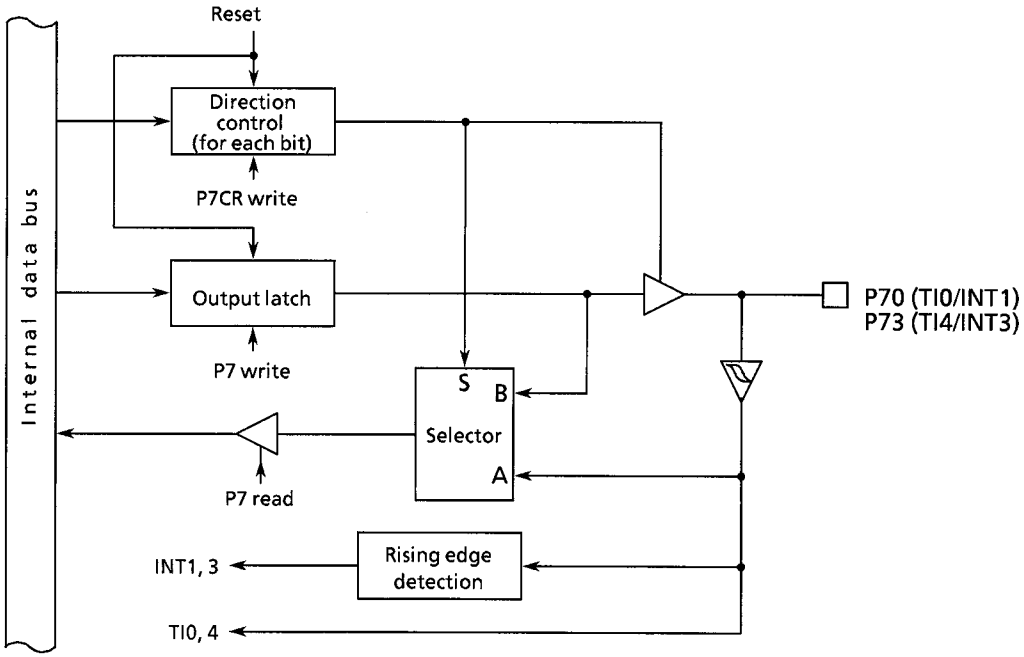


Figure 3.5.23 Port 7 (P70, P73)

(2) Port 71, 74 (TO1, TO5)

In addition to functioning as a general-purpose input/output port, port 71 also functions as TO1 for output of timers 0 and 1. In addition to functioning as a general-purpose input/output port, port 74 also functions as TO5 for output of timers 4 and 5.

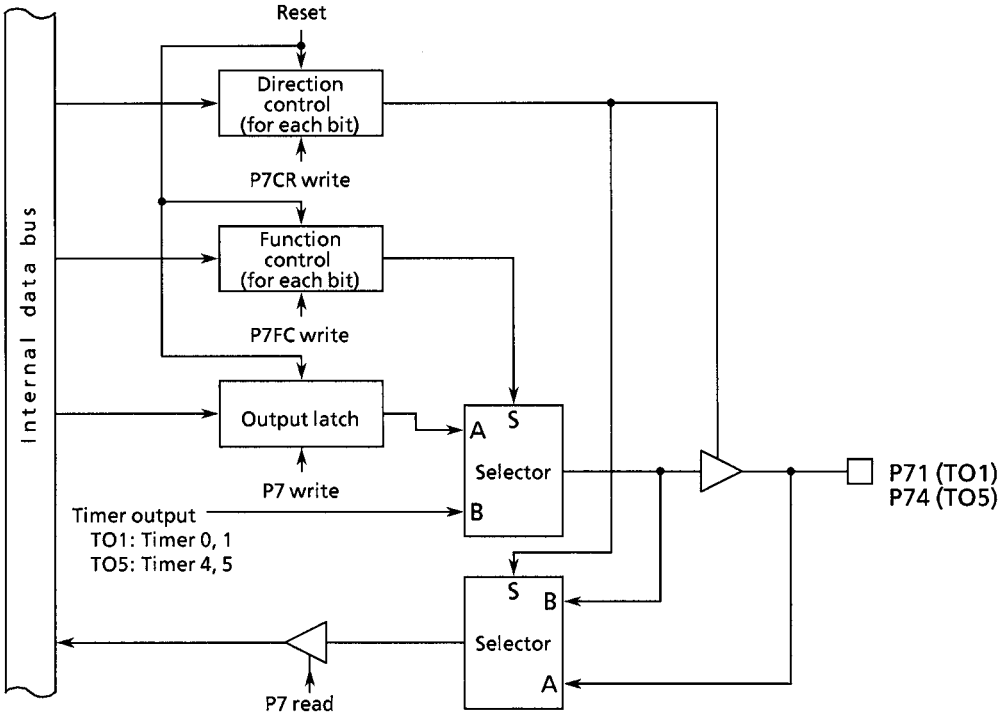


Figure 3.5.24 Port 7 (P71, P74)

(3) Port 72, 75 (TO3/INT2, TO7/INT4)

In addition to functioning as a general-purpose input/output port, port 72 also functions as TO3 for output of timers 2 and 3 and as the external interrupt request input INT2.

In addition to functioning as a general-purpose input/output port, port 75 also functions as TO7 for output of timers 6 and 7 and as the external interrupt request input INT4.

Caution when using INT2 or INT4 interrupts

Input is always enabled for the INT2 and INT4 external interrupt requests.

Caution is required if port 72 or 75 is used as a general-purpose input/output port or timer event count input port while the INT2 and INT4 interrupt functions are in use. This is because rising edges on these input/output signals generate interrupt requests.

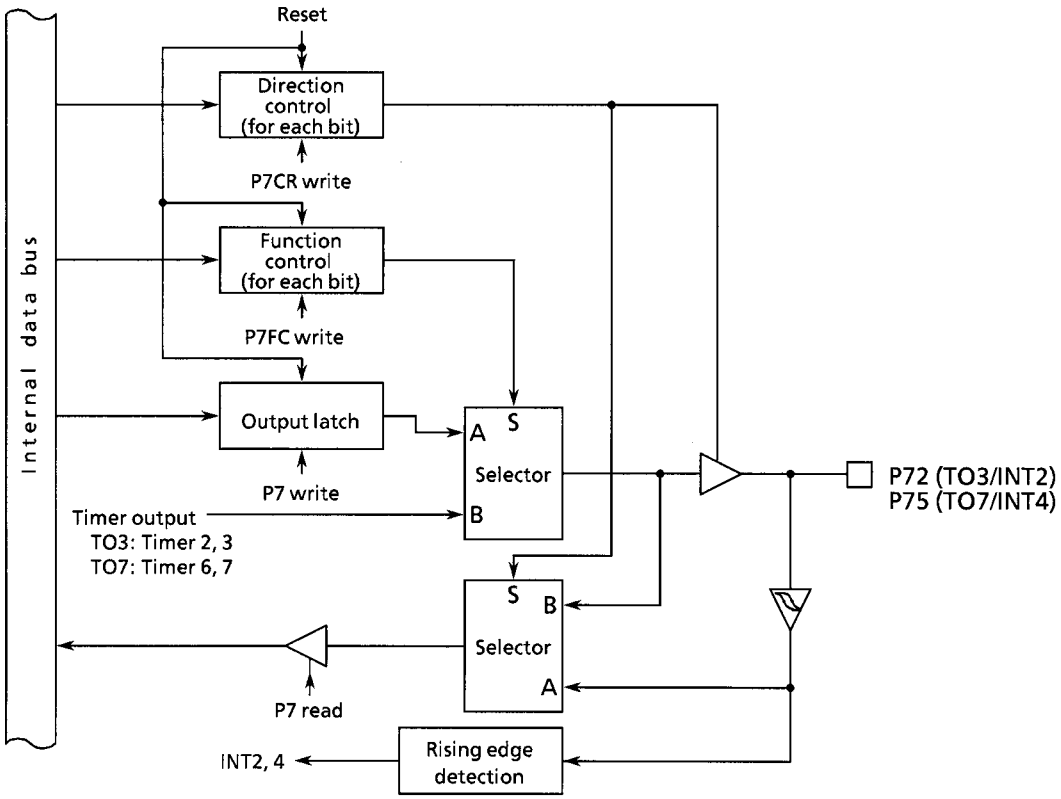


Figure 3.5.25 Port 7 (P72, P75)

Port 7 Register

P7 (0013H)

	7	6	5	4	3	2	1	0
bit Symbol			P75	P74	P73	P72	P71	P70
Read/Write			R/W					
After reset			Input mode (output latch register cleared to 0)					
Function			Also functions as TO7/INT4 (Note)	Also functions as TO5	Also functions as TI4/INT3 (Note)	Also functions as TO3/INT2 (Note)	Also functions as TO1	Also functions as TI0/INT1 (Note)

Port 7 Control Register

P7CR (0016H)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol			P75C	P74C	P73C	P72C	P71C	P70C
Read/Write			W					
After reset			0	0	0	0	0	0
Function			Port 7 input/output settings 0: Input 1: Output					

Port 7 Function Register

P7FC (0017H)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol			P75F	P74F		P72F	P71F	
Read/Write			W			W		
After reset			0	0		0	0	
Function			0: Port 1: TO7	0: Port 1: TO5		0: Port 1: TO3	0: Port 1: TO1	

Port 71 function settings

<P71F> / <P71C>	0	1
0	Input port	
1	Output port	TO1 output

Port 72 function settings

<P72F> / <P72C>	0	1
0	Input port/INT2 input	
1	Output port	TO3 output

Port 74 function settings

<P74F> / <P74C>	0	1
0	Input port	
1	Output port	TO5 output

Port 75 function settings

<P75F> / <P75C>	0	1
0	Input port/INT4 input	
1	Output port	TO7 output

Note: No register is provided to switch between the external interrupt input, event counter input, and port input/output functions. Therefore, even when port 7 is used as a port, the interrupt signal input and event counter input are enabled. When using port 7 exclusively as a port, disable the interrupt and event counter input.

Figure 3.5.26 Register for Port 7

3.5.9 Port 8 (P80 to P87)

Port 8 is an 8-bit general-purpose input/output port with each port bit settable as an input or output.

In addition to being a general-purpose input/output port, port 8 also functions as a serial channel TxD output, RxD input, SCLK input/output, and CAN controller Tx output and Rx input.

Port 8 control register P8CR and port 8 function register P8FC set the functions.

Reset sets all bits of the output latch to 1. It also clears all bits of the P8CR and P8FC registers to 0, setting port 8 to input mode using pull-up resistors.

Port pins 80 and 83 have a programmable open drain function.

(1) Ports 80, 83, 86 (TxD0, TxD1, Tx)

Ports 80, and 83 function as the serial channel TxD0 and TxD1 outputs as well as input/output ports.

These ports have a programmable open drain function. Setting open drain disables pull-up.

Port 86 functions as the CAN controller Tx output as well as input/output ports.

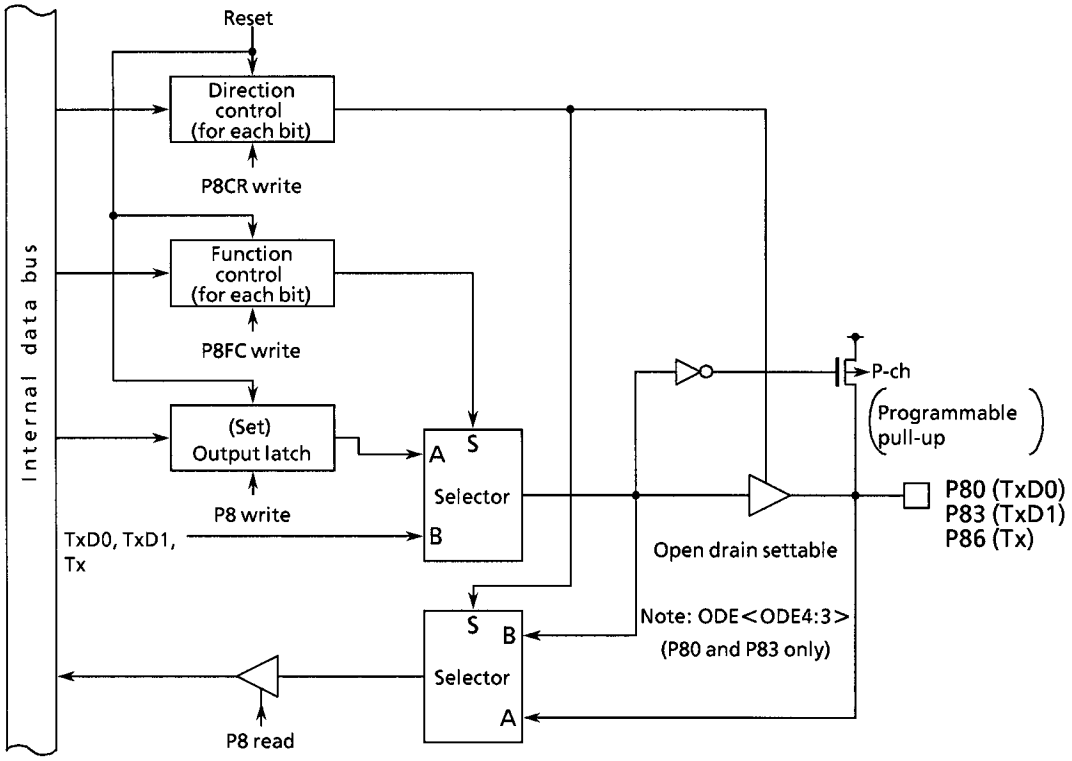


Figure 3.5.27 Port 8 (P80, P83, P86)

(2) Port 81, 84, 87 (RxD0, RxD1, Rx)

Ports 81 and 84 function as serial channel RxD0 and RxD1 inputs as well as input/output ports.

Port 87 functions as the CAN controller Rx input as well as input/output port.

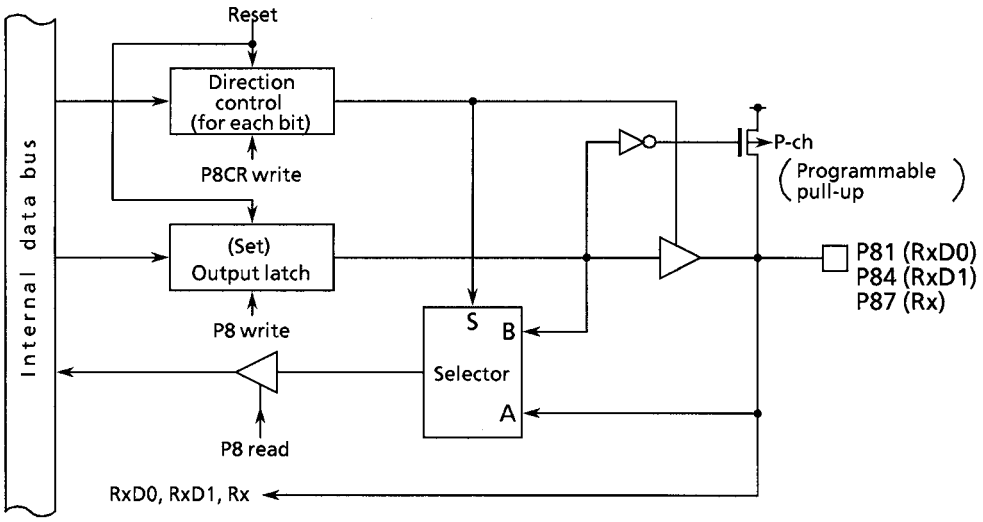


Figure 3.5.28 Port 8 (P81, P84, P87)

(3) Port 82 (SCLK0/$\overline{\text{CTS0}}$)

Port 82 functions as the SCLK0 input/output for serial channel 0 as well as an input/output port. The port also functions as the $\overline{\text{CTS0}}$ input.

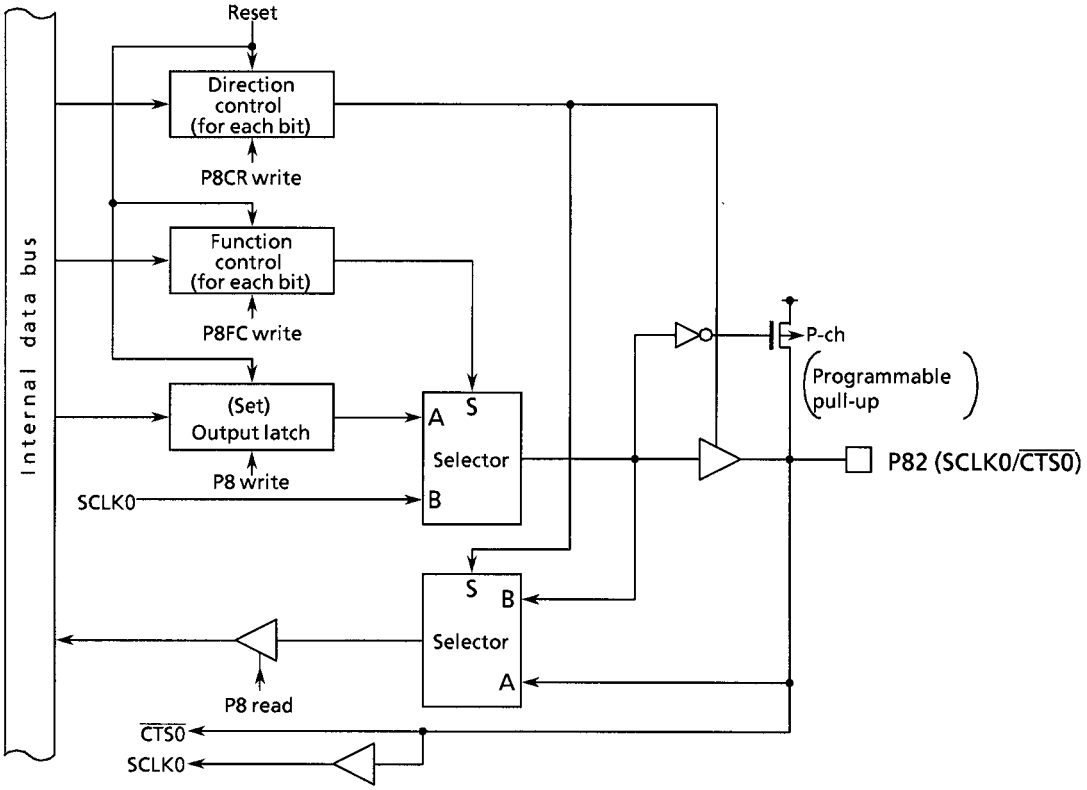


Figure 3.5.29 Port 8 (P82)

(4) Port 85 (SCLK1/$\overline{\text{CTS1}}$)

Port 85 functions as the SCLK1 input/output for serial channel 1 as well as an input/output port. The port also functions as the $\overline{\text{CTS1}}$ input.

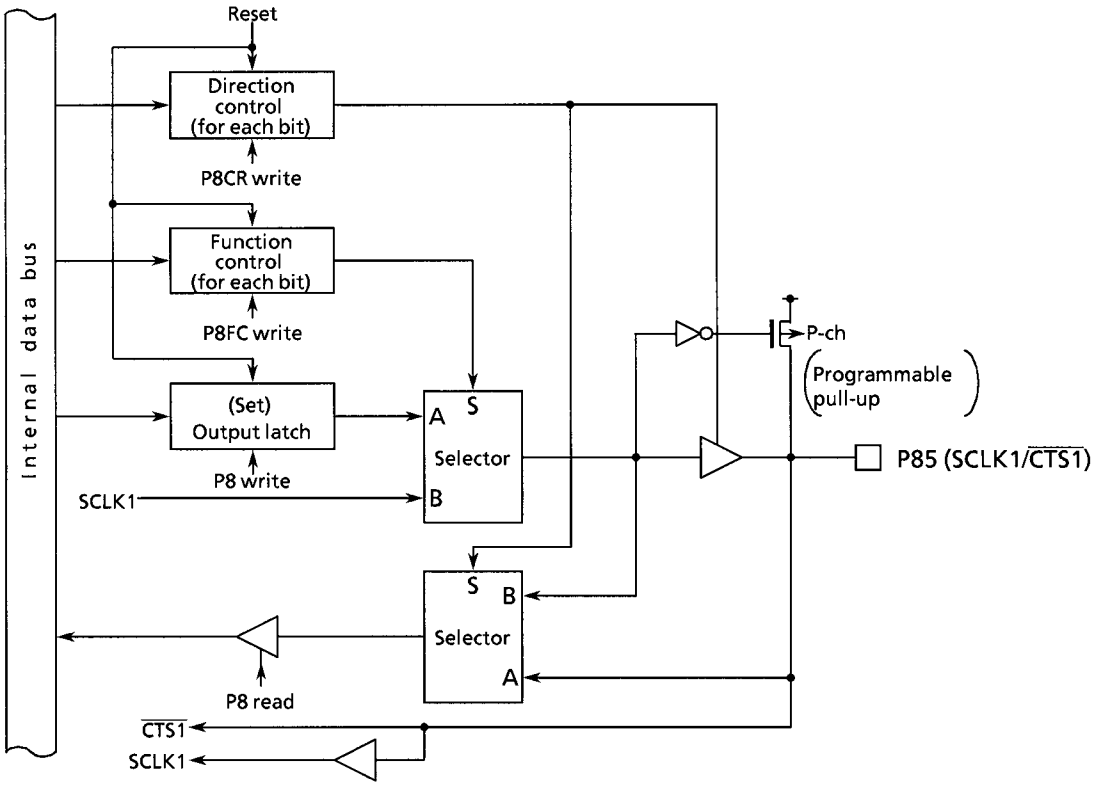


Figure 3.5.30 Port 8 (P85)

Port 8 Register

P8 (0018H)

	7	6	5	4	3	2	1	0
bit Symbol	P87	P86	P85	P84	P83	P82	P81	P80
Read/Write	R/W							
After reset	Input mode (Set to 1/pulled up)							
Function	Also functions as Rx	Also functions as Tx	Also functions as SCLK1/$\overline{CTS1}$	Also functions as RxD1	Also functions as TxD1	Also functions as SCLK0/$\overline{CTS0}$	Also functions as RxD0	Also functions as TxD0

Note: When port 8 is in input mode, the P8 register controls the internal pull-up resistor. When using port 8 in input mode or in both input and output modes (if a bit is set to input), do not execute read-modify-write instructions. The internal pull-up resistor setting may change depending on the state of the input pin.

Open Drain Enable Register

ODE (0058H)

	7	6	5	4	3	2	1	0
bit Symbol				ODE4	ODE3	ODE2	ODE1	ODE0
Read/Write				R/W				
After reset				0	0	0	0	0
Function				Port 83 output settings 0: CMOS 1: Open drain	Port 80 output settings 0: CMOS 1: Open drain	Port 63 output settings 0: CMOS 1: Open drain	Port 62 output settings 0: CMOS 1: Open drain	Port 61 output settings 0: CMOS 1: Open drain

Port 83 output settings

0	CMOS output
1	Open drain output

Port 80 output settings

0	CMOS output
1	Open drain output

Port 61 output settings

0	CMOS output
1	Open drain output

Port 62 output settings

0	CMOS output
1	Open drain output

Port 63 output settings

0	CMOS output
1	Open drain output

Figure 3.5.31 Register for Port 8 (1/2)

Port 8 Control Register

P8CR (001AH)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol	P87C	P86C	P85C	P84C	P83C	P82C	P81C	P80C
Read/Write	W							
After reset	0	0	0	0	0	0	0	0
Function	Port 8 input / output setting 0: Input 1: Output							

Port 8 Function Register

P8FC (001BH)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol		P86F	P85F		P83F	P82F		P80F
Read/Write		W			W			W
After reset		0	0		0	0		0
Function		0: Port 1: Tx	0: Port 1: SCLK1 /$\overline{CTS1}$		0: Port 1: TxD1	0: Port 1: SCLK0 /$\overline{CTS0}$		0: Port 1: TxD0

Port 85 function settings

<P85F> <P85C>	0	1
0	Input port SCLK1/$\overline{CTS1}$ (When <P85> = 0, not pulled up; when <P85> = 1, pulled up)	Do not set
1	Output port	SCLK1 output

Port 86 function settings

<P86F> <P86C>	0	1
0	Input port (When <P86> = 0, not pulled up; when <P86> = 1, pulled up)	Do not set
1	Output port	Tx output

Port 80 function settings

<P80F> <P80C>	0	1
0	Input port (When <P80> = 0, not pulled up; when <P80> = 1, pulled up)	Do not set
1	Output port	TxD0 output

Port 82 function settings

<P82F> <P82C>	0	1
0	Input port SCLK0/$\overline{CTS0}$ (When <P82> = 0, not pulled up; when <P82> = 1, pulled up)	Do not set
1	Output port	SCLK0 output

Port 83 function settings

<P83F> <P83C>	0	1
0	Input port (When <P83> = 0, not pulled up; when <P83> = 1, pulled up)	Do not set
1	Output port	TxD1 output

Note: When setting pins TxD0 and TxD1 to open drain output, set the open drain enable register ODE<ODE4:3> to 1. Pins P81/RxD0, P84/RxD1, and P87/RX do not have port/function switching registers. Therefore, even when the pins are used as input ports, data are still input to SIO or CAN as receive data.

Figure 3.5.31 Register for Port 8 (2/2)

3.5.10 Port 9 (P90 to P96)

Port 9 is a 7-bit general-purpose input/output port with each port bit settable as an input or output.

In addition to its input/output port functions, port 9 also functions as a 16-bit timer input clock pin, a 16-bit timer output pin, and inputs for INT5 to 8. Port 9 control register P9CR and port 9 function register P9FC set the port 9 functions.

A reset clears all bits of the P9 output latch and all bits of the P9CR and P9FC registers to 0, setting port 9 to input mode.

To enable the timer output function, write 1 to the corresponding bit in P9FC.

(1) Ports 90, 91, 94, 95 (TI8/INT5, TI9/INT6, TIA/INT7, TIB/INT8)

In addition to functioning as general-purpose input/output ports, ports 90 and 91 can also function as timer 8 event count inputs TI8 and TI9, and as external interrupt request inputs INT5 and INT6. Ports 94 and 95, in addition to being general-purpose input/output ports, can also function as the timer 9 event count inputs TIA and TIB, and as the external interrupt request inputs INT7 and INT8.

Caution when using INT5 to INT8 interrupts

Input is always enabled for the INT5 to INT8 external interrupt requests.

Caution is required if ports 90, 91, 94, or 95 are used as general-purpose input/output ports or timer event count inputs while the INT5 to INT8 interrupt functions are in use. This is because rising or falling edges on these input/output signals generate interrupt requests.

Caution when using timer event count inputs TI8 to TIB

Input is always enabled for timer event count inputs TI8 to TIB.

Caution is required if ports 90, 91, 94, or 95 are used as general-purpose input/output ports or INT5 to INT8 interrupts during event counting based on TI8 to TIB. This is because these input/output signals trigger an event count on the timer.

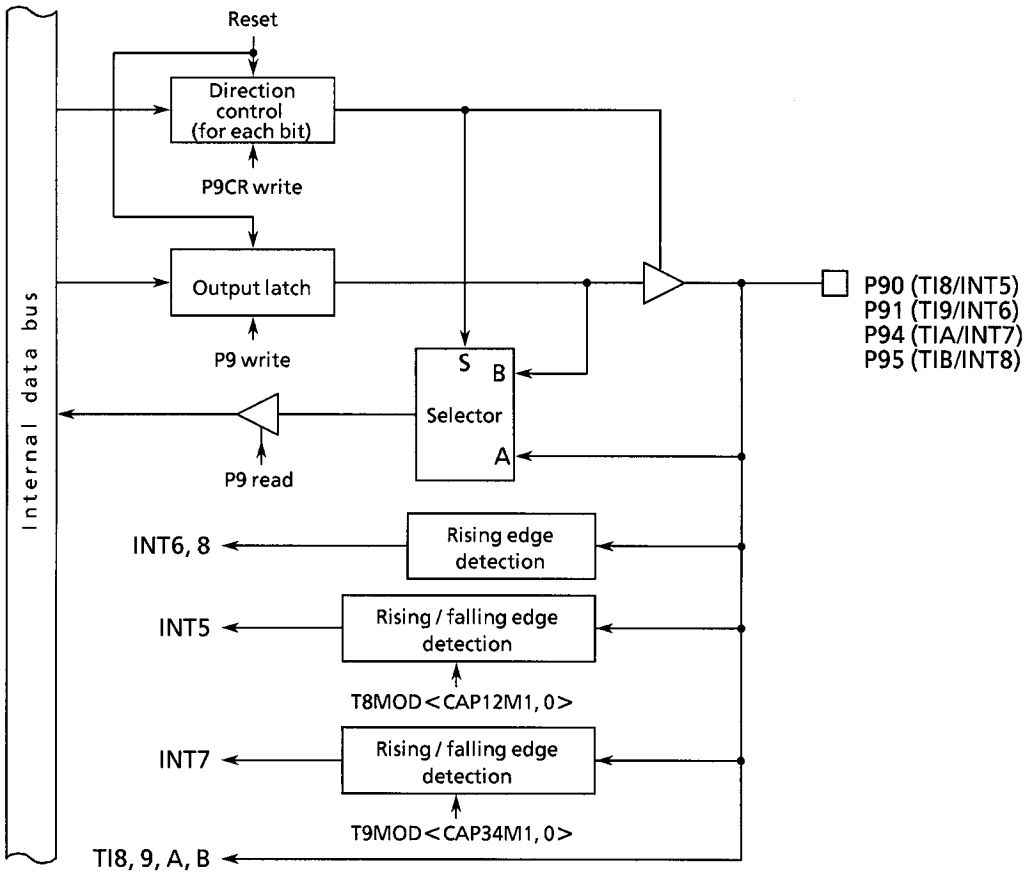


Figure 3.5.32 Port 9 (P90, P91, P94, P95)

(2) Ports 92, 93 (TO8, TO9)

In addition to operating as a general-purpose input/output port, port 92 also functions as the TO8 output for timer 8. Port 93 operates as the TO9 output for timer 8 as well as functioning as a general-purpose input/output port.

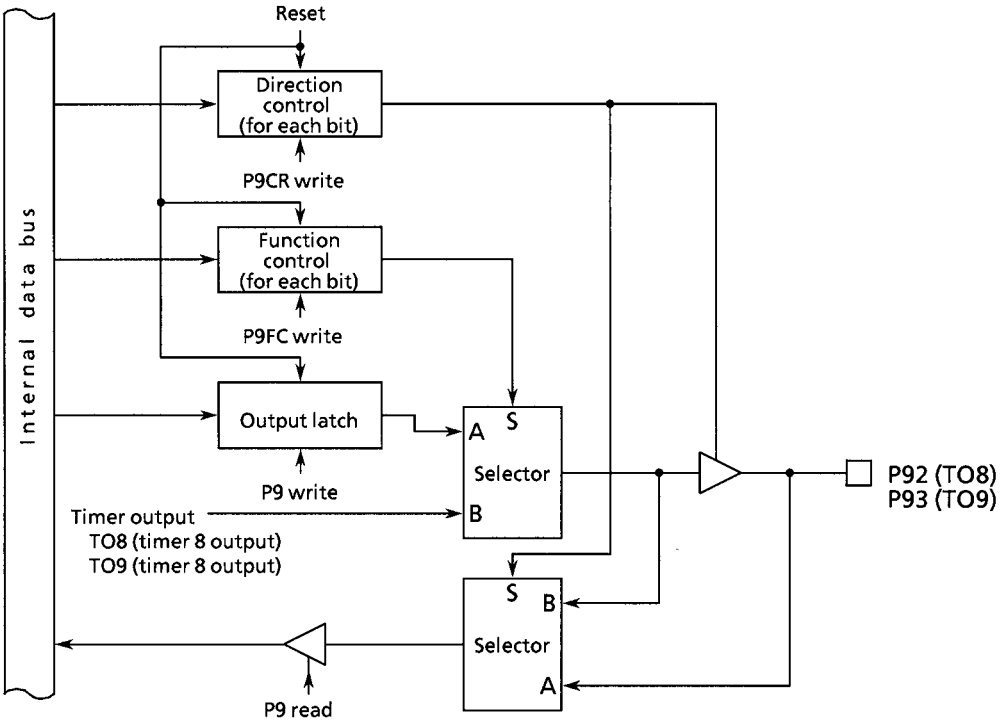


Figure 3.5.33 Port 9 (P92, P93)

(3) Port 96 (TOA/TOB)

In addition to functioning as a general-purpose input/output port, port 96 also functions as the TOA and TOB outputs for timer 9.

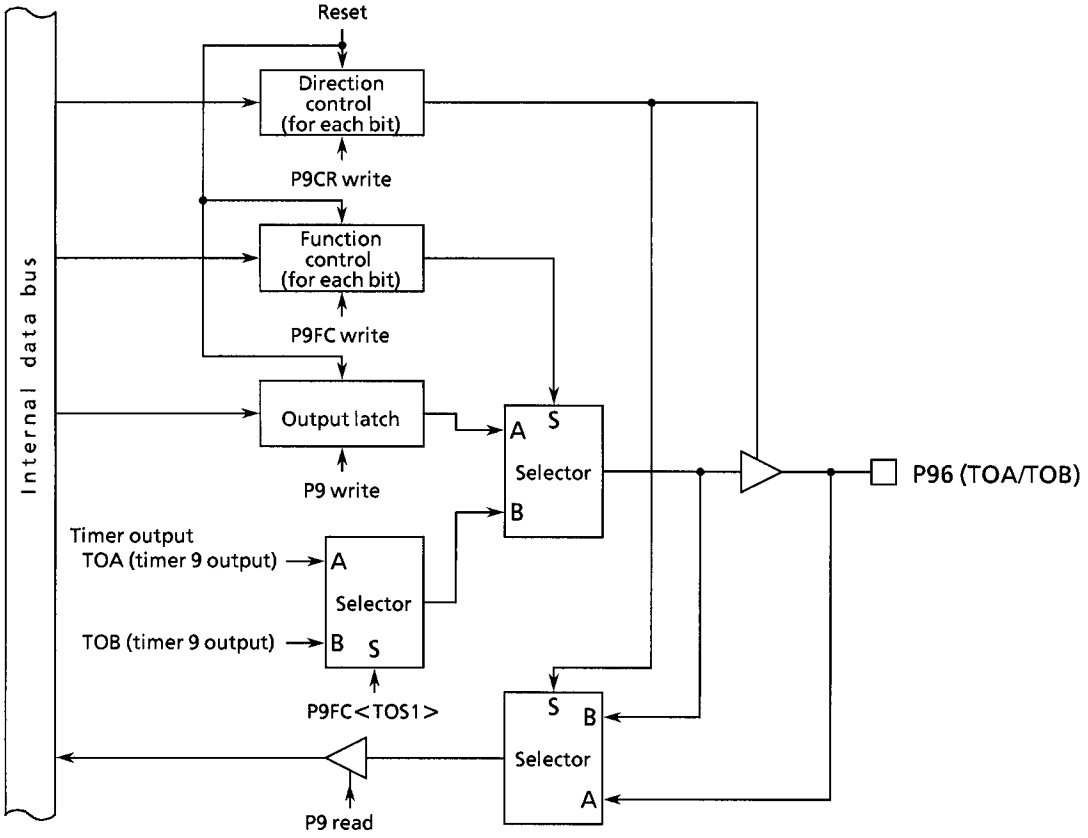


Figure 3.5.34 Port 9 (P96)

Port 9 Register

P9 (0019H)

	7	6	5	4	3	2	1	0
bit Symbol		P96	P95	P94	P93	P92	P91	P90
Read/Write		R/W						
After reset		Input mode (output latch register cleared to 0)						
Function		Also functions as TOA/TOB	Also functions as TIB/INT8 (Note)	Also functions as TIA/INT7 (Note)	Also functions as TO9	Also functions as TO8	Also functions as TI9/INT6 (Note)	Also functions as TI8/INT5 (Note)

Port 9 Control Register

P9CR (001CH)
Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol		P96C	P95C	P94C	P93C	P92C	P91C	P90C
Read/Write		W						
After reset		0	0	0	0	0	0	0
Function		Port 9 input / output settings 0: Input 1: Output						

Port 9 Function Register

P9FC (001DH)
Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol	TOS1	P96F			P93F	P92F		
Read/Write	W				W			
After reset	0	0			0	0		
Function	TOA/TOB output selection 0: TOA 1: TOB	0: Port 1: TOA /TOB			0: Port 1: TO9	0: Port 1: TO8		

Port 96 function settings

<P96F> / <P96C>	0	1
0	Input port	
1	Output port	TOA/TOB output

Timer out A/B output selection

0	Timer out A output
1	Timer out B output

Port 92 function settings

<P92F> / <P92C>	0	1
0	Input port	
1	Output port	TO8 output

Port 93 function settings

<P93F> / <P93C>	0	1
0	Input port	
1	Output port	TO9 output

Note: No register is provided to switch between the external interrupt input, event counter input, and port input/output functions. Therefore, even when port 9 is used as a port, the interrupt signal input and event counter input are enabled.
When using port 9 exclusively as a port, disable the external interrupts (INT5 to 8) and event count inputs (TI8 to B).

Figure 3.5.35 Register for Port 9

3.5.11 Port A (PA0 to PA7)

Port A is an 8-bit input-only port with analog input pins (AN0 to AN7). The PA3 pin also functions as the external trigger input for analog conversion ($\overline{\text{ADTRG}}$).

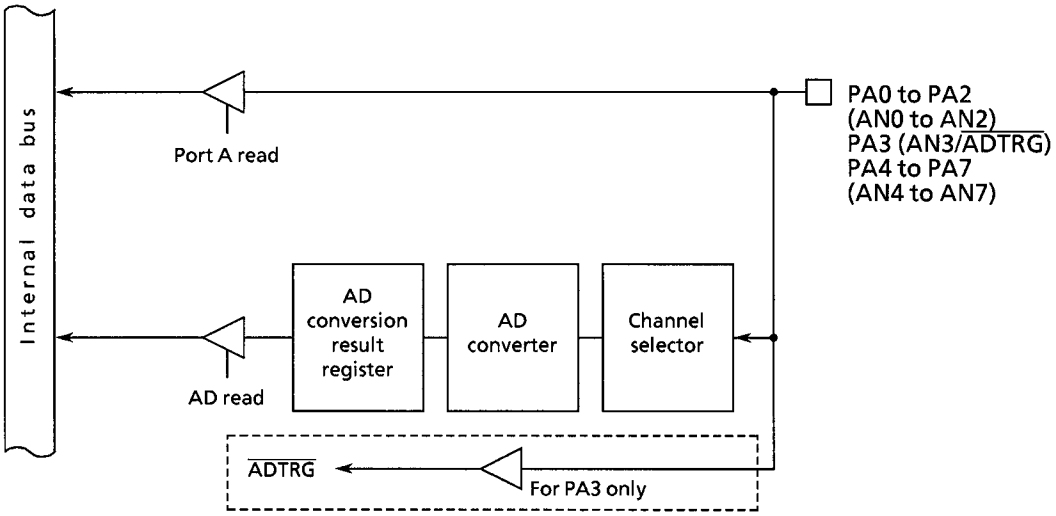


Figure 3.5.36 Port A (PA0 to PA7)

Port A Register

PA (001EH)

	7	6	5	4	3	2	1	0
bit Symbol	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
Read/Write	R							
After reset	Input only							
Function	Also functions as AN7	Also functions as AN6	Also functions as AN5	Also functions as AN4	Also functions as AN3 /ADTRG	Also functions as AN2	Also functions as AN1	Also functions as AN0

Note: AD mode register 1, ADMOD1, selects the AD converter input channel.

Figure 3.5.37 Register for Port A

3.6 Bus Width/Wait Controller

In the TMP95CU54A, four user-specifiable address area blocks can be set. The data bus width and number of waits can be set independently for each address area and for others.

Address areas 0 to 3 are set by a combination of memory start address registers MSAR0 to MSAR3 and memory address mask registers MAMR0 to MAMR3.

Use bus width/wait control registers WAITC0 to WAITC3 and WAITCEX to specify the master enable, data bus width, and number of waits for each address area.

The input pins controlling these states are the bus wait request pin ($\overline{WAIT}$), the external data bus selection pin ($AM8/\overline{16}$), and the external memory access pin ($\overline{EA}$). (See 3.1.2, External Data Bus Width Selection Function.)

3.6.1 Specifying address areas

Address areas 0 to 3 are specified using the start address registers MSAR0 to MSAR3 and memory address mask registers MAMR0 to MAMR3.

At each bus cycle, a compare operation is performed to determine if the address on the bus specifies a location in address areas 0 to 3. If the result of the comparison is a match, this indicates an access to the corresponding address area. In this case, the bus cycle operates in accordance with the settings in bus width/wait control register WAITC0 to WAITC3. If the result of the comparison is not a match, this indicates an access to another address area. In this case, the bus cycle operates in accordance with the settings in bus width/wait control register WAITCEX. (See 3.6.2, Bus Width/Wait Control Register.)

(1) Memory Start Address Registers

Figure 3.6.1 shows the memory start address registers. Memory start address registers MSAR0 to MSAR3 set the start address for address areas 0 to 3. Set the upper eight bits (A23 to A16) of the start address in <S23:16>. The lower 16 bits of the start address (A15 to A0) are permanently set to 0. Accordingly, the start address can only be set in 64 Kbyte increments, starting from 000000H. Figure 3.6.2 shows the relationship between the start address and the start address register value.

Memory Start Address Register (Address areas 0 to 3)

		7	6	5	4	3	2	1	0
MSAR0 (0094H) / MSAR1 (0096H)	bit Symbol	S23	S22	S21	S20	S19	S18	S17	S16
	Read/Write	R/W							
MSAR2 (0098H) / MSAR3 (009AH)	After reset	1	1	1	1	1	1	1	1
	Function	Sets start address A23 to A16							

Sets start address of address areas 0 to 3

Figure 3.6.1 Memory Start Address Register

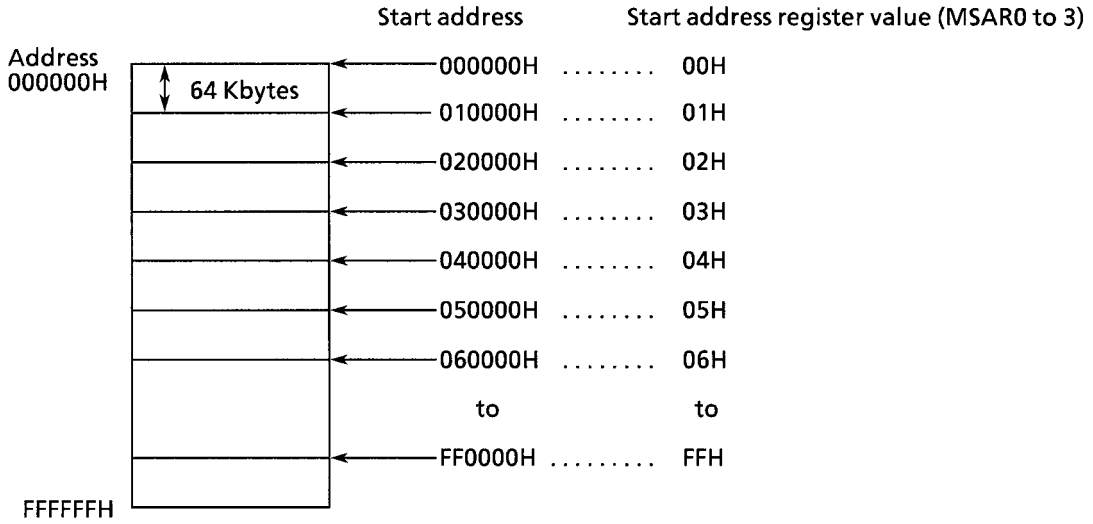


Figure 3.6.2 Relationship Between Start Address and Start Address Register Value

(2) Memory address mask registers

Figure 3.6.3 shows the memory address mask registers. Memory address mask registers MAMR0 to MAMR3 are used to set the size of address areas 0 to 3 by specifying a mask for each bit of the start address set in memory start address registers MSAR0 to MSAR3. The compare operation used to determine if an address is in the address areas 0 to 3 is only performed for bus address bits corresponding to bits set to 0 in these registers.

Also, the address bits that can be masked by MAMR0 to MAMR3 differ between address areas 0 to 3. Accordingly, the size that can be set for each area is different.

Memory address mask register (address area 0)

MAMR0 (0095H)

	7	6	5	4	3	2	1	0
bit Symbol	V20	V19	V18	V17	V16	V15	V14 to 9	V8
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Sets size of address area 0 0: Used for address compare							

Address area 0 can be set within the following range: 256 bytes to 2 Mbytes.

Memory address mask register (address area 1)

MAMR1 (0097H)

	7	6	5	4	3	2	1	0
bit Symbol	V21	V20	V19	V18	V17	V16	V15 to 9	V8
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Sets size of address area 1 0: Used for address compare							

Address area 1 can be set within the following range: 256 bytes to 4 Mbytes.

Memory address mask register (address areas 2 and 3)

MAMR2 (0099H) / MAMR3 (009BH)

	7	6	5	4	3	2	1	0
bit Symbol	V22	V21	V20	V19	V18	V17	V16	V15
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Sets size of address areas 2 and 3 0: Used for address compare							

Address areas 2 and 3 can be set within the following range: 32 Kbyte to 8 Mbytes.

Figure 3.6.3 Memory Address Mask Registers

(3) How to set memory start addresses and address areas

Figure 3.6.4 shows an example of specifying a 64 Kbyte address area starting from 010000H using address area 0.

Set 01H in memory start address register MSAR0<S23:16> (corresponding to the upper 8 bits of the start address). Next, calculate the difference between the start address and the anticipated end address (01FFFFH) based on the size of address area 0. Bits 20 to 8 of the result correspond to the mask value to be set for address area 0. Setting this value in memory address mask register MAMR0<V20:8> sets the area size. This example sets 07H in MAMR0 to specify a 64 Kbyte area.

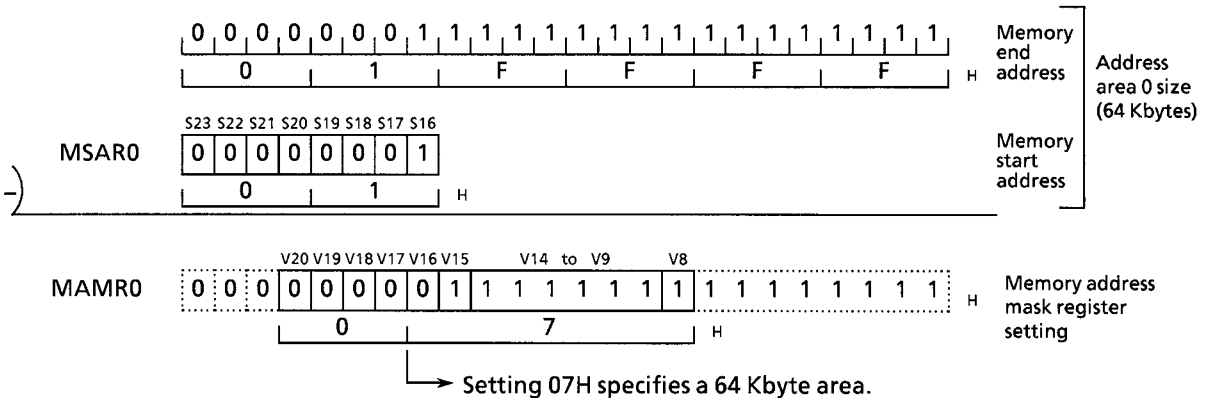


Figure 3.6.4 Address Area 0 Setting Example

After a reset, MSAR0 to MSAR3 and MAMR0 to MAMR3 are set to FFH. WAITC0<B0E>, WAITC1<B1E>, and WAITC3<B3E> are reset to 0. This disables address areas 0, 1 and 3. However, as WAITC2 <B2M> is reset to 0 and WAITC2<B2E> to 1, address area 2 is enabled from 000CA0H to 0021FFH and from 002340H to FE7FFFH. Also, the bus width and number of waits specified in WAITCEX are used for accessing addresses outside the specified address areas 0 to 3. (See 3.6.2 Bus Width/Wait Control Register.)

(4) Address area size specifications

Table 3.6.1 shows the relationship between address area and area size. Δ indicates areas that cannot be set by memory start address register and memory address mask register combinations. When setting an area size using a combination indicated by Δ, set the start address in the desired steps starting from 000000H.

If the address area 2 is set to 16M-byte or if two or more areas overlap, the smaller address area number has the higher priority.

Example: When setting address area 0 as a 128 Kbyte area:

[1] Available start addresses

000000H
020000H
040000H
060000H
⋮

128 Kbytes
128 Kbytes
128 Kbytes

Any of these start addresses can be set.

[2] Unavailable start addresses

000000H
010000H
030000H
050000H
⋮

64 Kbytes
128 Kbytes
128 Kbytes

This exceeds the size of the steps that can be set. In this case, the following start addresses cannot set the desired area size.

Table 3.6.1 Address Area and Area Size

Address area \ Size (bytes)	256	512	32 K	64 K	128 K	256 K	512 K	1 M	2 M	4 M	8 M
0	O	O	O	O	Δ	Δ	Δ	Δ	Δ		
1	O	O		O	Δ	Δ	Δ	Δ	Δ	Δ	
2			O	O	Δ	Δ	Δ	Δ	Δ	Δ	Δ
3			O	O	Δ	Δ	Δ	Δ	Δ	Δ	Δ

3.6.2 Bus Width/Wait Control Registers

Figure 3.6.5 lists the bus width/wait control registers. The master enable/disable, data bus width, and number of wait states for each address area 0 to 3 and others are set in their respective bus width/wait control registers, WAITC0 to WAITC3 and WAITCEX.

Bus Width/Wait Control Register

		7	6	5	4	3	2	1	0
WAITC0 (0090H)	bit Symbol	B0E				B0BUS	B0W2	B0W1	B0W0
	Read/Write	W				W			
	After reset	0				0	0	0	0
Read-modify-write instructions prohibited.	Function	0: Disable 1: Enable				Data bus width 0: 16-bit 1: 8-bit	Number of Waits setting 000: 2 WAIT 100: 0 + N WAIT 001: 1 WAIT 101 } Do not set 010: 1 WAIT + N 110 } Do not set 011: 0 WAIT 111 } Do not set		
WAITC1 (0091H)	bit Symbol	B1E				B1BUS	B1W2	B1W1	B1W0
	Read/Write	W				W			
	After reset	0				0	0	0	0
Read-modify-write instructions prohibited.	Function	0: Disable 1: Enable				Data bus width 0: 16-bit 1: 8-bit	Number of Waits setting 000: 2 WAIT 100: 0 + N WAIT 001: 1 WAIT 101 } Do not set 010: 1 WAIT + N 110 } Do not set 011: 0 WAIT 111 } Do not set		
WAITC2 (0092H)	bit Symbol	B2E	B2M			B2BUS	B2W2	B2W1	B2W0
	Read/Write	W				W			
	After reset	1	0			0	0	0	0
Read-modify-write instructions prohibited.	Function	0: Disable 1: Enable	Address area 2 selection 0: 16M-byte area 1: Address specifica-tion area			Data bus width 0: 16-bit 1: 8-bit	Number of Waits setting 000: 2 WAIT 100: 0 + N WAIT 001: 1 WAIT 101 } Do not set 010: 1 WAIT + N 110 } Do not set 011: 0 WAIT 111 } Do not set		
WAITC3 (0093H)	bit Symbol	B3E				B3BUS	B3W2	B3W1	B3W0
	Read/Write	W				W			
	After reset	0				0	0	0	0
Read-modify-write instructions prohibited.	Function	0: Disable 1: Enable				Data bus width 0: 16-bit 1: 8-bit	Number of Waits setting 000: 2 WAIT 100: 0 + N WAIT 001: 1 WAIT 101 } Do not set 010: 1 WAIT + N 110 } Do not set 011: 0 WAIT 111 } Do not set		
WAITCEX (009CH)	bit Symbol					BEXBUS	BEXW2	BEXW1	BEXW0
	Read/Write					W			
	After reset					0	0	0	0
Read-modify-write instructions prohibited.	Function					Data bus width 0: 16-bit 1: 8-bit	Number of Waits setting 000: 2 WAIT 100: 0 + N WAIT 001: 1 WAIT 101 } Do not set 010: 1 WAIT + N 110 } Do not set 011: 0 WAIT 111 } Do not set		

Master enable bit

0	Address area disable
1	Address area enable

Address area 2 selection

0	16 Mbyte area
1	Address specification area

Number of address area waits setting
(See 3.6.2, (3) Wait Control.)

Data bus width selection

0	16-bit data bus
1	8-bit data bus

Figure 3.6.5 Bus Width/Wait Control Registers

(1) Master enable bits

Bit 7 (<B0E>, <B1E>, <B2E>, and <B3E>) of the bus width/wait control registers is the master bit used to enable or disable settings for the address area. Writing 1 to the bit enables the settings. Reset disables (sets to 0) <B0E>, <B1E>, and <B3E>, and enables (sets to 1) <B2E>.

(2) Selection of data bus width

Bit 3 (<B0BUS>, <B1BUS>, <B2BUS>, <B3BUS>, and <BEXBUS>) of the bus width/wait control registers specifies the width of the data bus. Set 0 to access memory when using a 16-bit data bus. Set 1 when using an 8-bit data bus.

Connect the $\overline{EA}$ and AM8/$\overline{16}$ pins to VCC. This enables external memory to be accessed using the data bus width set in the data bus width select bit.

This method of changing the data bus width depending on the address being accessed is called dynamic bus sizing. For details of this bus operation, see Table 3.6.2.

Table 3.6.2 Dynamic Bus Sizing

Operand Data Bus Width	Operand Start Address	Memory Data Bus Width	CPU Address	CPU Data	
				D15 to D8	D7 to D0
8-bit	2n + 0 (Even number)	8 bits	2n + 0	xxxxx	b7 to b0
		16 bits	2n + 0	xxxxx	b7 to b0
	2n + 1 (Odd number)	8 bits	2n + 1	xxxxx	b7 to b0
		16 bits	2n + 1	b7 to b0	xxxxx
16-bit	2n + 0 (Even number)	8 bits	2n + 0	xxxxx	b7 to b0
			2n + 1	xxxxx	b15 to b8
		16 bits	2n + 0	b15 to b8	b7 to b0
	2n + 1 (Odd number)	8 bits	2n + 1	xxxxx	b7 to b0
			2n + 2	xxxxx	b15 to b8
		16 bits	2n + 1	b7 to b0	xxxxx
			2n + 2	xxxxx	b15 to b8
32-bit	2n + 0 (Even number)	8 bits	2n + 0	xxxxx	b7 to b0
			2n + 1	xxxxx	b15 to b8
			2n + 2	xxxxx	b23 to b16
			2n + 3	xxxxx	b31 to b24
		16 bits	2n + 0	b15 to b8	b7 to b0
			2n + 2	b31 to b24	b23 to b16
	2n + 1 (Odd number)	8 bits	2n + 1	xxxxx	b7 to b0
			2n + 2	xxxxx	b15 to b8
			2n + 3	xxxxx	b23 to b16
			2n + 4	xxxxx	b31 to b24
		16 bits	2n + 1	b7 to b0	xxxxx
			2n + 2	b23 to b16	b15 to b8
			2n + 4	xxxxx	b31 to b24

xxxxx: Indicates that the input data from these bits are ignored during a read. During a write, indicates that the bus for these bits goes to high impedance; also, that the write strobe signal for the bus remains inactive.

(3) Wait control

Bits 2 to 0 (<B0W2:0>, <B1W2:0>, <B2W2:0>, <B3W2:0>, and <BEXW2:0>) of the bus width/wait control registers specify the number of waits to insert.

The following types of wait operation can be specified using combinations of these bits. Do not set combinations other than those listed in the table.

Table 3.6.3 Wait Operation Settings

<BxW2:0>	No. of Waits	Wait Operation
000	2WAIT	Inserts a wait of two states, irrespective of the $\overline{WAIT}$ pin state.
001	1WAIT	Inserts a wait of one state, irrespective of the $\overline{WAIT}$ pin state.
010	1WAIT + N	Samples the state of the $\overline{WAIT}$ pin after inserting a wait of one state. If the $\overline{WAIT}$ pin is low, the waits continue and the bus cycle is extended until the pin goes high.
011	0WAIT	Ends the bus cycle without a wait, regardless of the $\overline{WAIT}$ pin state.
100	0 + NWAIT	Continuously samples the $\overline{WAIT}$ pin state and inserts waits if the pin is low, extending the bus cycle until the pin goes high.

Figures 3.6.6 and 3.6.7 show the timing for N = 0, 1 when the setting is 0 + NWAIT.

For the timings for settings other than 0 + NWAIT, see Figures 7.1 to 7.5 in 7. Basic Timing, Chapter 3, TLCS-900/H CPU.

Reset sets these bits to 000 (2 WAIT).

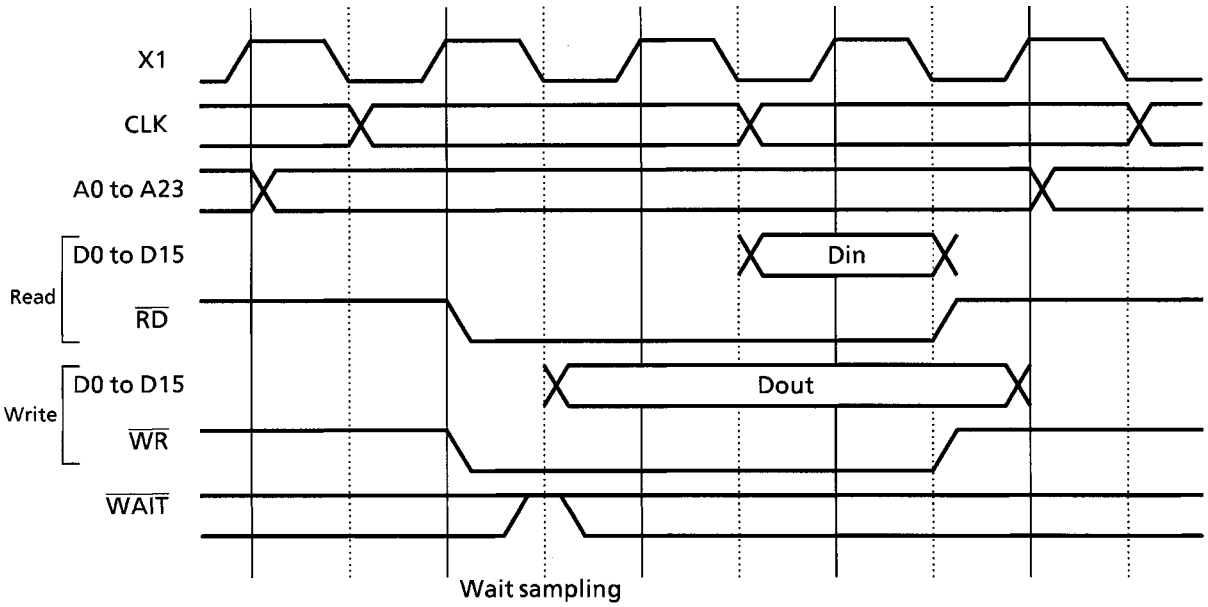


Figure 3.6.6 0 + NWAIT Read/Write Cycle (When N = 0)

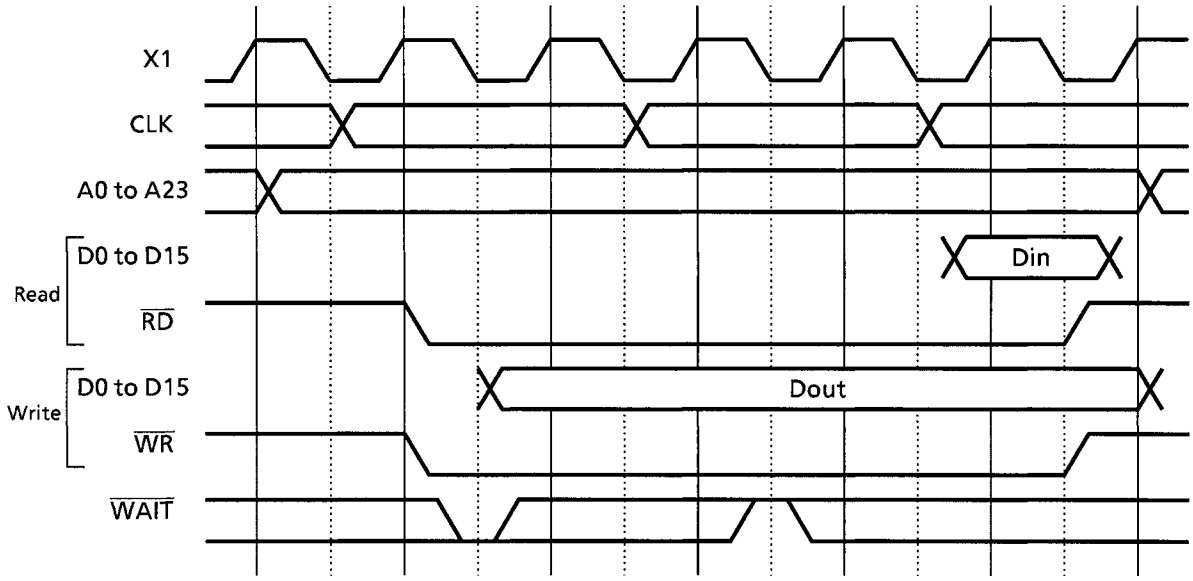


Figure 3.6.7 0 + NWAIT Read/Write Cycle (When N = 1)

(4) Bus width and wait control outside address areas 0 to 3

The bus width/wait control register WAITCEX controls the bus width and number of waits when locations outside the four user-specified address area blocks 0 to 3 are accessed. The WAITCEX register settings are always enabled for areas other than address areas 0 to 3.

(5) 16M-byte area/address setting area selection

Setting the bus width/wait control register WAITC2<B2M> to 0 selects a 16M-byte address area (000CA0H to 0021FFH, and 002340H to FE7FFFH) for address area 2. Setting WAITC2<B2M> to 1 selects the address area specified by start address register MSAR2 and address mask register MAMR2 for address area 2, and likewise for address area 0, 1, and 3. Reset clears this bit to 0 and selects a 16M-byte address area.

(6) Bus width/wait control setting procedure

When using the bus width/wait control function, set the registers as follows:

[1] Set memory start address registers MSAR0 to MSAR3.

Set the start addresses of address areas 0 to 3.

[2] Set memory address mask registers MAMR0 to MAMR3.

Set the size of address areas 0 to 3.

[3] Set control registers WAITC0 to WAITC3.

Set the data bus width, number of waits, and master enable/disable for address areas 0 to 3.

In the case of addresses, if one of the address areas 0 to 3 is set, but an internal I/O, RAM or ROM area is specified, the CPU accesses the internal area.

Setting example:

This example sets the address area 0 as 010000H to 01FFFFH (64 Kbyte area) with a 16-bit bus and zero waits:

MSAR0 = 01H..... Start address: 010000H
MAMR0 = 07H..... Address area: 64 Kbyte
WAITC0 = 83H ..... 16-bit data bus, zero waits, address area 0 settings enabled

3.7 8-Bit Timers

The TMP95CU54A incorporates eight 8-bit timers (timers 0 to 7).

Each timer can operate independently or be cascaded to form four 16-bit timers. The 8-bit timers have the following four operating modes.

- 8-bit interval timer mode (8 channels)
- 16-bit interval timer mode (4 channels)

These modes can be combined (for example, four 8-bit timers and two 16-bit timers).

- 8-bit programmable square wave pulse generation (PPG: variable cycle, variable duty) output mode (4 channels)
- 8-bit PWM (pulse width modulation: variable duty at fixed cycle) output mode (4 channels)

Figure 3.7.1 shows the block diagram of 8-bit timers 0 and 1. The other 8-bit timers (timers 2 and 3, 4 and 5, and 6 and 7) have the same circuit configuration as timers 0 and 1.

Each 8-bit timer consists of an 8-bit up-counter, an 8-bit comparator, and an 8-bit timer register. One timer flip-flop each (TFF1, TFF3, TFF5, and TFF7) is provided for the timer pairs, consisting of timers 0 and 1, timers 2 and 3, timers 4 and 5, and timers 6 and 7.

Of the input clock sources for the 8-bit timers, the φT1, φT4, φT16, and φT256 internal clocks are obtained from the 9-bit internal prescaler.

The 8-bit timers are controlled by nine control registers (T01MOD, T23MOD, T45MOD, T67MOD, T02FFCR, T46FFCR, T8RUN, T16RUN, and TRDC).

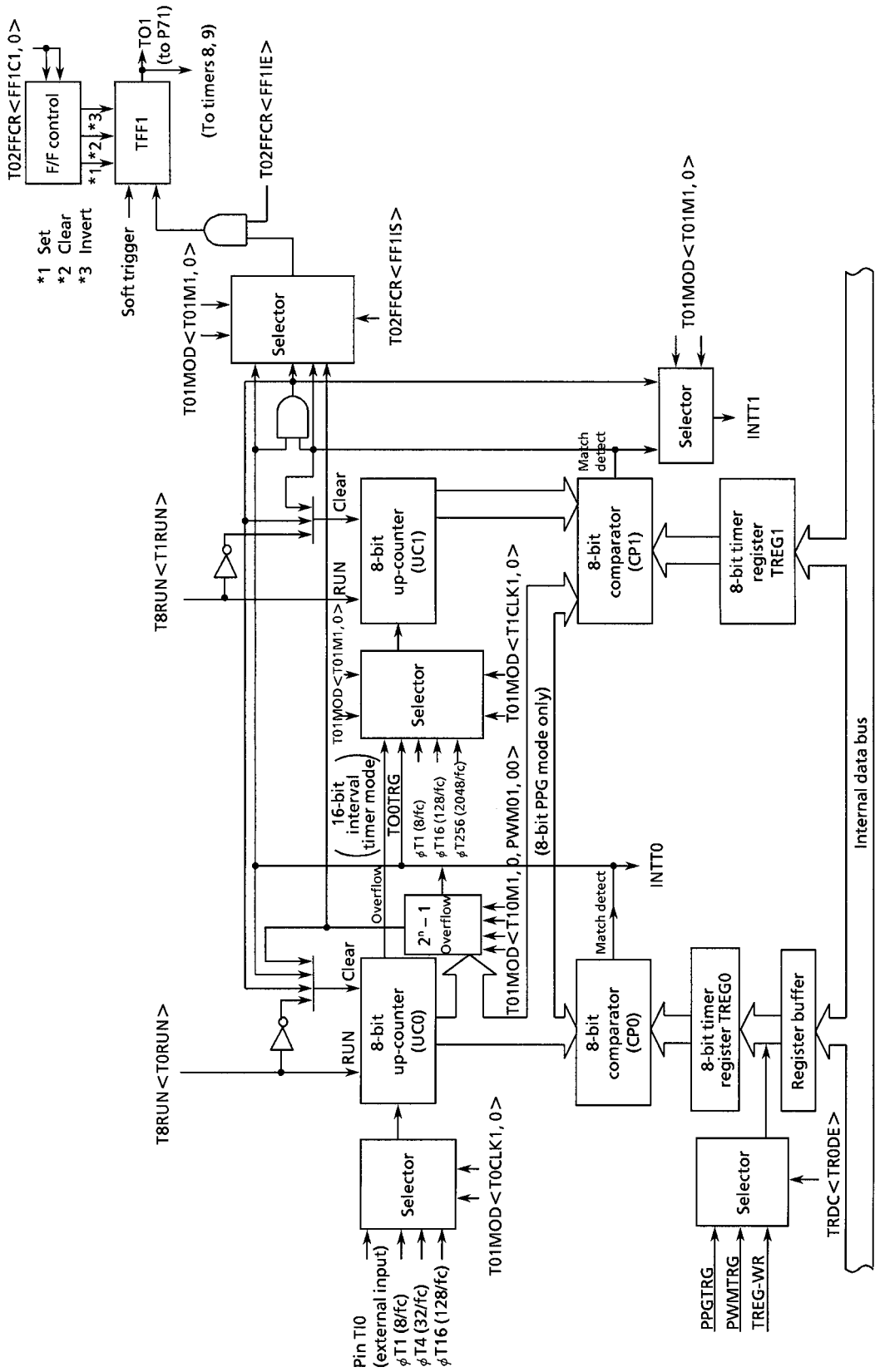


Figure 3.7.1 8-Bit Timer Block Diagram (Timers 0, 1)

3.7.1 8-Bit Timer Registers

Figure 3.7.2 shows the 8-bit timer registers. Setting these registers controls the operation of the 8-bit timers.

8-bit Timer Operation Control Register

T8RUN (0020H)

	7	6	5	4	3	2	1	0
bit Symbol	T7RUN	T6RUN	T5RUN	T4RUN	T3RUN	T2RUN	T1RUN	T0RUN
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Timer 7 operation control 0: Stop and clear 1: Run	Timer 6 operation control 0: Stop and clear 1: Run	Timer 5 operation control 0: Stop and clear 1: Run	Timer 4 operation control 0: Stop and clear 1: Run	Timer 3 operation control 0: Stop and clear 1: Run	Timer 2 operation control 0: Stop and clear 1: Run	Timer 1 operation control 0: Stop and clear 1: Run	Timer 0 operation control 0: Stop and clear 1: Run

8-bit timer up-counter operation control

0	Stop and clear
1	Run

16-Bit Timer Operation Control Register

T16RUN (003BH)

	7	6	5	4	3	2	1	0
bit Symbol	PRRUN		T9RUN	T8RUN				
Read/Write	R/W		R/W					
After reset	0		0	0				
Function	Prescaler operation control 0: Stop and clear 1: Run		Timer 9 operation control 0: Stop and clear 1: Run	Timer 8 operation control 0: Stop and clear 1: Run				

16-bit timer up-counter operation control
(See 3.8, 16-bit Timers/Event Counters.)

Prescaler operation control (Note)

0	Stop and clear
1	Run

Note: Set T16RUN<PRRUN> to 1 when using an 8-bit timer.

Figure 3.7.2 Register for 8-Bit Timer (1/8)

Timer Register Double Buffer Control Register

TRDC (0021H)

	7	6	5	4	3	2	1	0
bit Symbol					TR6DE	TR4DE	TR2DE	TR0DE
Read/Write					R/W			
After reset					0	0	0	0
Function					TREG6 double buffer control 0: Disable 1: Enable	TREG4 double buffer control 0: Disable 1: Enable	TREG2 double buffer control 0: Disable 1: Enable	TREG0 double buffer control 0: Disable 1: Enable

Timer register 0 double buffer control

0	Disable
1	Enable

Timer register 2 double buffer control

0	Disable
1	Enable

Timer register 4 double buffer control

0	Disable
1	Enable

Timer register 6 double buffer control

0	Disable
1	Enable

Figure 3.7.2 Register for 8-Bit Timer (2/8)

Timers 0, 1 Mode Control Register

T01MOD (0024H)

	7	6	5	4	3	2	1	0
bit Symbol	T01M1	T01M0	PWM01	PWM00	T1CLK1	T1CLK0	T0CLK1	T0CLK0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Set operating mode for timers 0, 1 00: 8-bit interval timer 01: 16-bit interval timer 10: 8-bit PPG 11: 8-bit PWM		PWM0 cycle selection 00: Don't care 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		Timer 1 input clock selection 00: TO0TRG 01: ϕT1 10: ϕT16 11: ϕT256		Timer 0 input clock selection 00: TI0 01: ϕT1 10: ϕT4 11: ϕT16	

Timer 0 input clock selection

00	External clock TI0
01	Internal clock ϕT1 (8/fc)
10	Internal clock ϕT4 (32/fc)
11	Internal clock ϕT16 (128/fc)

Timer 1 input clock selection

	<T01M1, 0> ≠ 01	<T01M1, 0> = 01
00	Timer 0 comparator output	Timer 0 overflow output (16-bit interval timer mode)
01	Internal clock ϕT1 (8/fc)	
10	Internal clock ϕT16 (128/fc)	
11	Internal clock ϕT256 (2048/fc)	

PWM0 cycle selection
(Except for PWM mode (<T01M1, 0> = 11), don't care)

00	Don't care
01	$(2^6 - 1)$ × timer 0 input clock cycle
10	$(2^7 - 1)$ × timer 0 input clock cycle
11	$(2^8 - 1)$ × timer 0 input clock cycle

Timers 0, 1 operating mode settings

00	8-bit interval timer x 2 channels (timers 0, 1)
01	16-bit interval timer
10	8-bit PPG output
11	8-bit PWM output (timer 0) + 8-bit interval timer (timer 1)

Figure 3.7.2 Register for 8-Bit Timer (3/8)

Timers 2, 3 Mode Control Register

T23MOD (0028H)

	7	6	5	4	3	2	1	0
bit Symbol	T23M1	T23M0	PWM21	PWM20	T3CLK1	T3CLK0	T2CLK1	T2CLK0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Set operating mode for timers 2, 3 00: 8-bit interval timer 01: 16-bit interval timer 10: 8-bit PPG 11: 8-bit PWM		PWM2 cycle selection 00: Don't care 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		Timer 3 input clock selection 00: TO2TRG 01: ϕT1 10: ϕT16 11: ϕT256		Timer 2 input clock selection 00: Don't care 01: ϕT1 10: ϕT4 11: ϕT16	

Timer 2 input clock selection

00	Don't care
01	Internal clock ϕT1 (8/fc)
10	Internal clock ϕT4 (32/fc)
11	Internal clock ϕT16 (128/fc)

Timer 3 input clock selection

	<T23M1, 0> ≠ 01	<T23M1, 0> = 01
00	Timer 2 comparator output	Timer 2 overflow output (16-bit interval timer mode)
01	Internal clock ϕT1 (8/fc)	
10	Internal clock ϕT16 (128/fc)	
11	Internal clock ϕT256 (2048/fc)	

PWM2 cycle selection
(Except for PWM mode (<T23M1, 0> = 11), don't care)

00	Don't care
01	$(2^6 - 1)$ × timer 2 input clock cycle
10	$(2^7 - 1)$ × timer 2 input clock cycle
11	$(2^8 - 1)$ × timer 2 input clock cycle

Timers 2, 3 operating mode settings

00	8-bit interval timer × 2 channels (timers 2, 3)
01	16-bit interval timer
10	8-bit PPG output
11	8-bit PWM output (timer 2) + 8-bit interval timer (timer 3)

Figure 3.7.2 Register for 8-Bit Timer (4/8)

Timers 4, 5 Mode Control Register

T45MOD (002BH)

	7	6	5	4	3	2	1	0
bit Symbol	T45M1	T45M0	PWM41	PWM40	T5CLK1	T5CLK0	T4CLK1	T4CLK0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Set operating mode for timers 4, 5 00: 8-bit interval timer 01: 16-bit interval timer 10: 8-bit PPG 11: 8-bit PWM		PWM4 cycle selection 00: Don't care 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		Timer 5 input clock selection 00: TO4TRG 01: ϕT1 10: ϕT16 11: ϕT256		Timer 4 input clock selection 00: TI4 01: ϕT1 10: ϕT4 11: ϕT16	

Timer 4 input clock selection

00	External clock TI4
01	Internal clock ϕT1 (8/fc)
10	Internal clock ϕT4 (32/fc)
11	Internal clock ϕT16 (128/fc)

Timer 5 input clock selection

	<T45M1, 0> ≠ 01	<T45M1, 0> = 01
00	Timer 4 comparator output	Timer 4 overflow output (16-bit interval timer mode)
01	Internal clock ϕT1 (8/fc)	
10	Internal clock ϕT16 (128/fc)	
11	Internal clock ϕT256 (2048/fc)	

PWM4 cycle selection
(Except for PWM mode (<T45M1, 0> = 11), don't care)

00	Don't care
01	$(2^6 - 1)$ × timer 4 input clock cycle
10	$(2^7 - 1)$ × timer 4 input clock cycle
11	$(2^8 - 1)$ × timer 4 input clock cycle

Timers 4, 5 operating mode settings

00	8-bit interval timer × 2 channels (timers 4, 5)
01	16-bit interval timer
10	8-bit PPG output
11	8-bit PWM output (timer 4) + 8-bit interval timer (timer 5)

Figure 3.7.2 Register for 8-Bit Timer (5/8)

Timers 6, 7 Mode Control Register

T67MOD (002FH)

	7	6	5	4	3	2	1	0
bit Symbol	T67M1	T67M0	PWM61	PWM60	T7CLK1	T7CLK0	T6CLK1	T6CLK0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0
Function	Set operating mode for timers 6, 7 00: 8-bit interval timer 01: 16-bit interval timer 10: 8-bit PPG 11: 8-bit PWM		PWM6 cycle selection 00: Don't care 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		Timer 7 input clock selection 00: TO6TRG 01: ϕT1 10: ϕT16 11: ϕT256		Timer 6 input clock selection 00: Don't Care 01: ϕT1 10: ϕT4 11: ϕT16	

Timer 6 input clock selection

00	Don't care
01	Internal clock ϕT1 (8/fc)
10	Internal clock ϕT4 (32/fc)
11	Internal clock ϕT16 (128/fc)

Timer 7 input clock selection

	<T67M1, 0> ≠ 01	<T67M1, 0> = 01
00	Timer 6 comparator output	Timer 6 overflow output (16-bit interval timer mode)
01	Internal clock ϕT1 (8/fc)	
10	Internal clock ϕT16 (128/fc)	
11	Internal clock ϕT256 (2048/fc)	

PWM6 cycle selection
(Except for PWM mode (<T67M1, 0> = 11), don't care)

00	Don't care
01	$(2^6 - 1)$ × timer 6 input clock cycle
10	$(2^7 - 1)$ × timer 6 input clock cycle
11	$(2^8 - 1)$ × timer 6 input clock cycle

Timers 6, 7 operating mode settings

00	8-bit interval timer × 2 channels (timers 6, 7)
01	16-bit interval timer
10	8-bit PPG output
11	8-bit PWM output (timer 6) + 8-bit interval timer (timer 7)

Figure 3.7.2 Register for 8-Bit Timer (6/8)

Timers 0, 2 Flip-Flop Control Register

T02FFCR (0025H)

	TFF3				TFF1			
	7	6	5	4	3	2	1	0
bit Symbol	FF3C1	FF3C0	FF3IE	FF3IS	FF1C1	FF1C0	FF1IE	FF1IS
Read/Write	W		R/W		W		R/W	
After reset	1	1	0	0	1	1	0	0
Function	TFF3 control 00: Invert TFF3 01: Set TFF3 10: Clear TFF3 11: Don't care		TFF3 inversion control 0: Disable 1: Enable	TFF3 inversion signal selection 0: Inversion by timer 2 1: Inversion by timer 3	TFF1 control 00: Invert TFF1 01: Set TFF1 10: Clear TFF1 11: Don't care		TFF1 inversion control 0: Disable 1: Enable	TFF1 inversion signal selection 0: Inversion by timer 0 1: Inversion by timer 1

Timer flip-flop TFF1 inversion signal selection
(Except for 8-bit interval timer mode, don't care)

0	Inversion by timer 0
1	Inversion by timer 1

Timer flip-flop TFF1 inversion control

0	Disable
1	Enable

Timer flip-flop TFF1 control

00	Invert TFF1 value (software inversion)
01	Set TFF1 to 1
10	Clear TFF1 to 0
11	Don't care (11 when read)

Timer flip-flop TFF3 inversion signal selection
(Except for 8-bit interval timer mode, don't care)

0	Inversion by timer 2
1	Inversion by timer 3

Timer flip-flop TFF3 inversion control

0	Disable
1	Enable

Timer flip-flop TFF3 control

00	Invert TFF3 value (software inversion)
01	Set TFF3 to 1
10	Clear TFF3 to 0
11	Don't care (11 when read)

Figure 3.7.2 Register for 8-Bit Timer (7/8)

Timers 4, 6 Flip-Flop Control Register

T46FFCR (002CH)

	7	6	5	4	3	2	1	0
bit Symbol	FF7C1	FF7C0	FF7IE	FF7IS	FF5C1	FF5C0	FF5IE	FF5IS
Read/Write	W		R/W		W		R/W	
After reset	1	1	0	0	1	1	0	0
Function	TFF7 control 00: Invert TFF7 01: Set TFF7 10: Clear TFF7 11: Don't care		TFF7 inversion control 0: Disable 1: Enable	TFF7 inversion signal selection 0: Inversion by timer 6 1: Inversion by timer 7	TFF5 control 00: Invert TFF5 01: Set TFF5 10: Clear TFF5 11: Don't care		TFF5 inversion control 0: Disable 1: Enable	TFF5 inversion signal selection 0: Inversion by timer 4 1: Inversion by timer 5

TFF7 TFF5

Timer flip-flop TFF5 inversion signal selection
(Except for 8-bit interval timer mode, don't care)

0	Inversion by timer 4
1	Inversion by timer 5

Timer flip-flop TFF5 inversion control

0	Disable
1	Enable

Timer flip-flop TFF5 control

00	Invert TFF5 value (software inversion)
01	Set TFF5 to 1
10	Clear TFF5 to 0
11	Don't care (11 when read)

Timer flip-flop TFF7 inversion signal selection
(Except for 8-bit interval timer mode, don't care)

0	Inversion by timer 6
1	Inversion by timer 7

Timer flip-flop TFF7 inversion control

0	Disable
1	Enable

Timer flip-flop TFF7 control

00	Invert TFF7 value (software inversion)
01	Set TFF7 to 1
10	Clear TFF7 to 0
11	Don't care (11 when read)

Figure 3.7.2 Register for 8-Bit Timer (8/8)

3.7.2 Block Structure

(1) Prescaler

The prescaler is a 9-bit divider circuit that divides its supplied clock (4/fc) by 2^n (n = 1, ..., 6, 9). The clock supplied to the prescaler is the CPU clock (fc) divided by four (4/fc). The divided clock is used as the input clock for such functions as the 8-bit timers, 16-bit timer/event counters, and baud rate generator.

The prescaler count can be turned on and off using timer operation control register T16RUN<PRRUN>. Setting T16RUN<PRRUN> to 1 starts the count.

Setting 0 clears the divided clock to zero and stops the prescaler. A reset clears <PRRUN> to 0, clearing and stopping the prescaler.

Input clock \ fc = 24 MHz	Cycle
φT1 (8/fc)	333 ns
φT4 (32/fc)	1.33 μs
φT16 (128/fc)	5.33 μs
φT256 (2048/fc)	85.33 μs

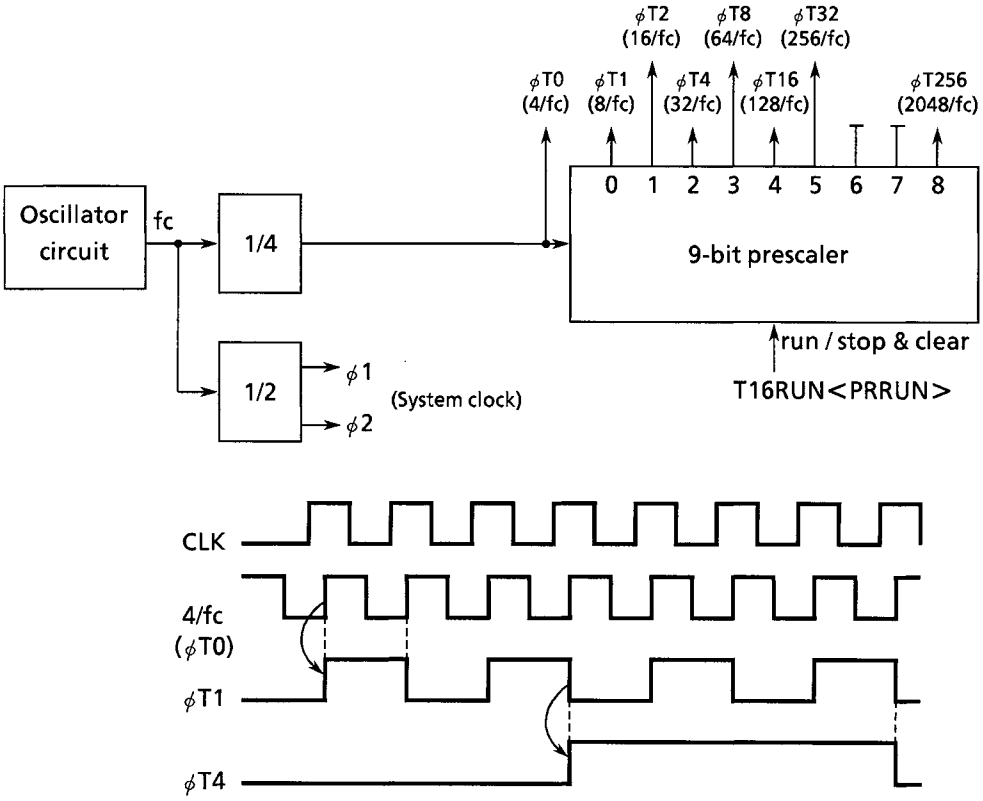


Figure 3.7.3 Prescaler

(2) 8-bit up-counters

The 8-bit up-counters UC0 to 7 are the 8-bit binary counters for timers 0 to 7. The up-counters count up on the internal or external clock selected by 8-bit timer mode control registers T01MOD, T23MOD, T45MOD, and T67MOD. The 8-bit timer operation control register T8RUN settings control the up-counter operation.

The available input clocks for UC0, 2, 4, 6 are the internal clocks φT1, φT4, or φ16. UC0 and 4 can use the external clocks input from the timer input pin (TI0 and TI4) signals.

The input clocks for UC1, 3, 5, 7 vary according to the operating mode.

In 16-bit timer mode, the overflow output signals of timer 0, 2, 4, 6 are used as the input clocks.

In other than 16-bit timer mode, the available input clocks are internal clocks φT1, φT16, φT256 or TOxTRG (timer 0, 2, 4, 6 match detect signals).

A reset clears T8RUN and stops UC0 to 7.

(3) 8-bit timer registers

The 8-bit timer registers are 8-bit registers for setting count values.

The comparator outputs a match detect signal when the value set in 8-bit timer register TREG0 to 7 matches the 8-bit up-counter UC0 to 7 value. If 00H is set, the match detect signal is output when the 8-bit up-counter overflows.

8-bit timer registers TREG0, 2, 4, 6 have a double-buffer configuration (each has a dedicated register buffer).

Timer register double-buffer control registers TRDC<TR0/2/4/6DE> enable or disable the double buffer. Setting <TR0/2/4/6DE> to 0 disables the double-buffer; setting <TR0/2/4/6DE> to 1 enables the double buffer.

When the double buffer is enabled, data are transferred from the register buffer to the timer register at a $2^n - 1$ overflow in pulse width modulation (PWM) mode, or at a cycle compare match in programmable pulse generation (PPG) mode.

Always disable the double buffer in 8-bit and 16-bit interval timer modes.

A reset clears TRDC to 0 and disables the double buffer. When using the double buffer, first write data to TREG0, 2, 4, 6 and set TRDC<TR0/2/4/6DE> to 1, then write the next settings.

As TREG0 to 7 are undefined after a reset, set the registers before using the 8-bit timers.

Figure 3.7.4 shows the configuration of timer registers 0, 2, 4, 6.

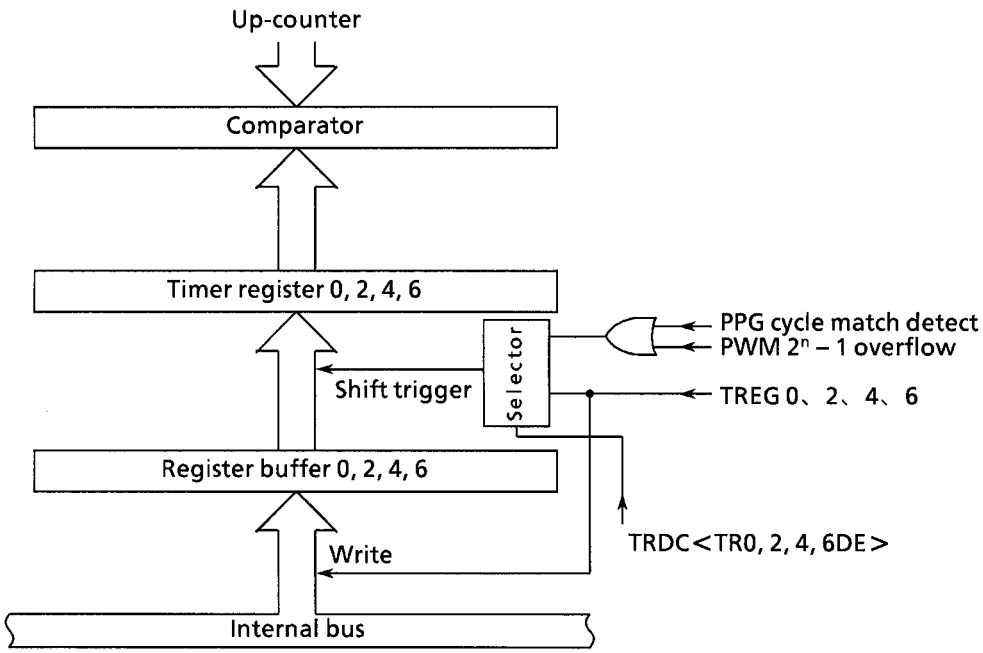


Figure 3.7.4 Configuration of Timer Registers 0, 2, 4, 6

Note: The timer register and register buffer are allocated to the same address in memory. When TRDC<TR0/2/4/6DE> is set to 0, the same value is written to both the register buffer and the timer register. When TRDC<TR0/2/4/6DE> is set to 1, the value is written to the register buffer only. Accordingly, when writing the initial values to the timer registers, first disable the register buffers.

The timer registers are located in memory as follows.

TREG0	TREG1	TREG2	TREG3
8 bits	8 bits	8 bits	8 bits
000022H	000023H	000026H	000027H

TREG4	TREG5	TREG6	TREG7
8 bits	8 bits	8 bits	8 bits
000029H	00002AH	00002DH	00002EH

All registers are write-only and therefore cannot be read.

(4) 8-bit comparator

The 8-bit comparator compares the 8-bit up-counter value with the 8-bit timer register value and detects when the values are equal (match). If the values match, a match detect signal is output, the 8-bit up-counter is cleared to zero, and an interrupt is generated (INTT0 to 7).

(5) Timer flip-flops

The timer flip-flops (TFF1, TFF3, TFF5, TFF7) are inverted by a match detect signal from the 8-bit comparator.

Timer flip-flop control registers T02FFCR<FF3IE>, <FF1IE>, and T46FFCR <FF7IE>, <FF5IE> enable or disable inversion. Setting these bits to 0 disables inversion; setting to 1 enables inversion.

The timer flip-flop values after a reset are undefined. Writing 01 or 10 to T02FFCR<FF3C1, 0>, <FF1C1, 0>, or T46FFCR<FF7C1, 0>, <FF5C1, 0> sets the timer flip-flop to 0 or 1. Writing 00 to the bits inverts the timer flip-flop value (software inversion).

The TFF1, TFF3, TFF5, and TFF7 values can be output to the timer output pins TO1 (shared with P71), TO3 (shared with P72), TO5 (shared with P74), and TO7 (shared with P75) respectively.

As the timer output pins also function as P71, P72, P74, and P75, be sure to set the port 7 function register P7FC before performing timer output.

(See Figure 3.5.26 Register for Port 7)

3.7.3 Operation Description for Each Mode

(1) 8-bit interval timer mode

The eight interval timers 0 to 7 can be used independently. When setting the functions and count data, first stop timers 0 to 7.

The following describes the example of timer 1 only.

[1] Generate interrupts at fixed intervals

Use T01MOD to select the operating mode and input clock. Set the interval time (cycle) in TREG1. Enable interrupt INTT1 such that INTT1 is generated when a match occurs between UC1 and TREG1. After setting the registers, start the timer counting.

Table 3.7.1 shows the input clock selection.

Example: To generate a timer 1 interrupt every 33 μs (at fc = 24 MHz), set the registers in the following order:

		MSB 7	6	5	4	3	2	1	LSB 0	
T8RUN	←	–	–	–	–	–	–	0	–	Stop timer 1 and clear to zero.
T01MOD	←	0	0	X	X	0	1	–	–	Set 8-bit interval timer mode and set input clock to ϕT1 (0.33 μs @fc = 24 MHz).
TREG1	←	0	1	1	0	0	1	0	0	Set 33 μs ÷ ϕT1 = 100 (64H) in timer register.
INTET01	←	1	1	0	1	–	–	–	–	Enable INTT1 and set interrupt level to 5.
T16RUN	←	1	X	–	–	X	X	X	X	
T8RUN	←	–	–	–	–	–	–	1	–	Start timer 1 counting.

Note: X : Don't care – : No change

Table 3.7.1 Selecting Interval and Input Clock for 8-Bit Timer Interrupt

Input Clock	Interrupt Interval (@fc = 24 MHz)	Resolution
ϕT1 (8/fc)	0.33 μs to 85.33 μs	0.33 μs
ϕT4 (32/fc)	1.33 μs to 341.3 μs	1.33 μs
ϕT16 (128/fc)	5.33 μs to 1.365 ms	5.33 μs
ϕT256 (2048/fc)	85.33 μs to 21.85 ms	85.33 μs

[2] Generate square wave with 50%-duty cycle

To output a square wave with a duty cycle of 50%, set a count value equivalent to half the desired cycle and TFF1 to invert on a match detect signal from timer 1 (T02FFCR<FF1IE, FF1IS> = 11).

Also, set P71 as a timer output (P7CR<P71C> = 1, P7FC<P71F> = 1)

Example: To output a square wave from pin TO1 with an interval of 2 µs (at fc = 24 MHz), set the registers in the following order:

		MSB 7	6	5	4	3	2	1	LSB 0	
T8RUN	←	–	X	–	–	–	–	0	–	Stop timer 1 and clear to zero.
T01MOD	←	0	0	X	X	0	1	–	–	Set 8-bit interval timer mode and set input clock to ϕT1.
TREG1	←	0	0	0	0	0	0	1	1	Set 2 μs ÷ ϕT1 (0.33 μs) ÷ 2 = 3 in timer register.
T02FFCR	←	–	–	–	–	1	0	1	1	Clear TFF1 to 0 and set to invert on match detect signal from timer 1.
P7CR	←	X	X	–	–	–	–	1	–	Set P71 as TO1 pin.
P7FC	←	X	X	–	–	X	–	1	X	Set P71 as TO1 pin.
T16RUN	←	1	X	–	–	X	X	X	X	
T8RUN	←	–	–	–	–	–	–	1	–	Start timer 1 counting.

Note: X : Don't care – : No change

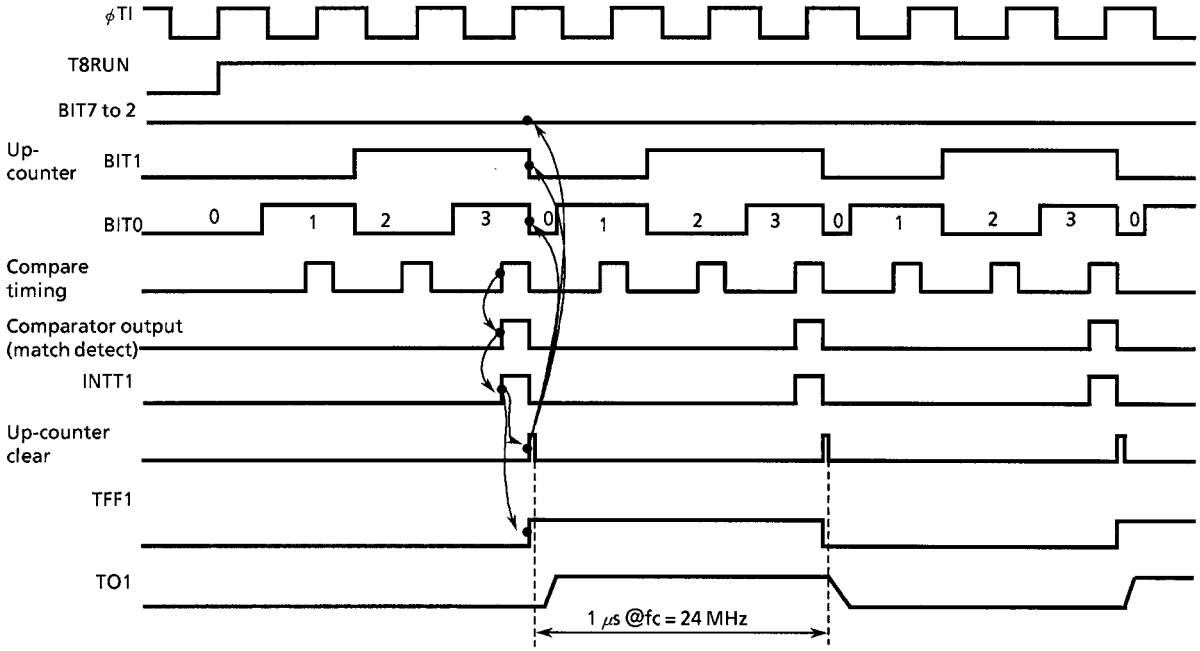


Figure 3.7.5 Square Wave (50% Duty Cycle) Output Timing Chart

[3] To count up at each timer 0 match output, set timer 1

Set 8-bit timer mode and the timer 0 comparator output as the timer 1 input clock (T01MOD<T1CLK1, 0> = 00).

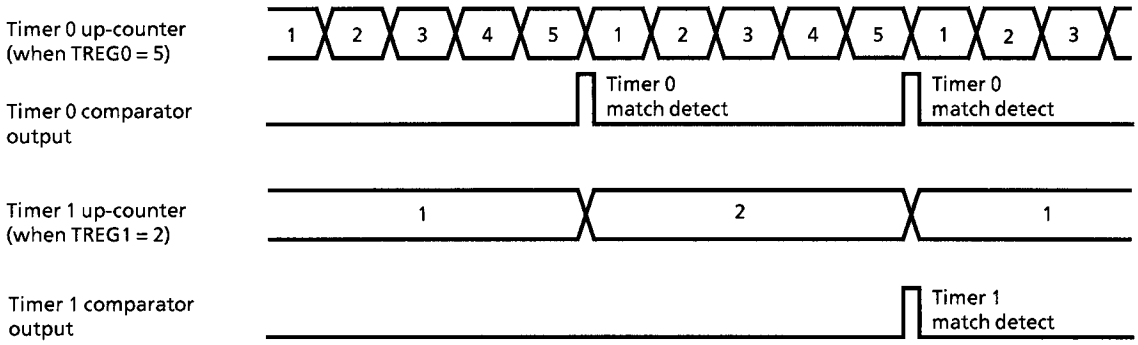


Figure 3.7.6 Using Timer 0 to Drive Timer 1 Count

(2) 16-bit interval timer mode

The 8-bit timers can be cascaded in pairs (timers 0 and 1, 2 and 3, 4 and 5, 6 and 7) to create 16-bit interval timers.

Timers 0 and 1, 2 and 3, 4 and 5, 6 and 7 operate the same. Each pair can be used independently.

The following describes the example of timers 0 and 1.

To cascade timers 0 and 1 to form a 16-bit interval timer, set the timer 0, 1 mode control register T01MOD<T01M1, 0> to 01.

When 16-bit interval timer mode is set, the T01MOD<T1CLK1, 0> setting is ignored and the timer 0 overflow output is forcibly set as the timer 1 input clock.

Table 3.7.2 shows the relationship between the timer (interrupt) interval and the input clock selection.

Table 3.7.2 16-Bit Timer (Interrupt) Interval and Input Clock Selection

Input Clock	Interrupt Interval (fc = 24 MHz)	Resolution
ϕT1 (8/fc)	0.33 μs to 21.845 ms	0.33 μs
ϕT4 (32/fc)	1.33 μs to 87.381 ms	1.33 μs
ϕT16 (128/fc)	5.33 μs to 349.525 ms	5.33 μs

To set the timer interrupt interval, set the lower eight bits in timer register TREG0 and the upper eight bits in TREG1. Be sure to set TREG0 first (as entering data in TREG0 temporarily disables compare, while entering data in TREG1 starts compare).

Example: To generate interrupt INTT1 every 0.33s at fc = 24 MHz, set the following values in timer registers TREG0 and TREG1:

Using φT16 (= 5.33 μs @ 24 MHz) as a timer input clock

0.33 s ÷ 5.33 μs = 62500 = F424H

Therefore, set TREG1 to F4H, and TREG0 to 24H.

Whenever 8-bit up-counter UC0 and TREG0 match, the timer 0 comparator outputs a match detect signal, but up-counter UC0 is not cleared. No INTT0 interrupt is generated.

When up-counter UC1 and TREG1 match, at comparator timing the timer 1 comparator outputs a match detect signal.

When comparator match detect signals for both timer 0 and timer 1 are output at the same time, up-counter 0 and up-counter 1 are cleared to 0 and interrupt INTT1 only is generated. When the timer flip-flop inversion is enabled, the value of timer flip-flop TFF1 is inverted.

Table 3.7.3 Differences Between 16-Bit Timer Mode and 8-Bit Timer Mode
(Timer 1 Input Clock: TO0TRG)

	Timer 0			Timer 1		
	INTT0 interrupt	TO1 output	Counter operation when match detected	INTT1 interrupt	TO1 output	Counter operation when match detected
16-bit timer mode (count-up timer 1 on each timer 0 overflow)	No interrupt generated	Output disabled (of a signal indicating a match with TREG0 is disabled)	TREG0 (count-up even when a match occurs. Clear at match with TREG1)	Interrupt generated	Output enabled (can output a match signal for both timers 0 and 1)	TREG1 × 2^8 + TREG0: Full 16 bits (clear at match)
8-bit timer mode (count up timer 1 on each timer 0 match)	Interrupt generated	Output enabled (either from timer 0 or timer 1)	TREG0 (clear at match)	Interrupt generated	Output enabled (either from timer 0 or timer 1)	TREG1 × TREG0: Multiplication value (clear at match)

Example: When TREG1 = 04H and TREG0 = 80H:

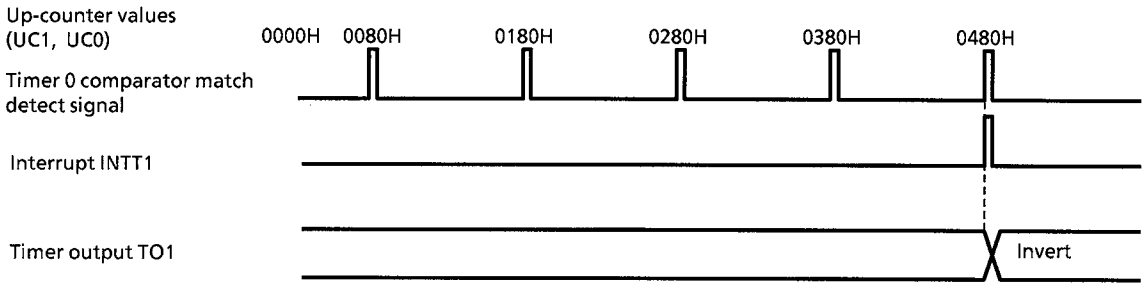


Figure 3.7.7 Timer Output for 16-Bit Timer Mode

(3) 8-bit programmable pulse generation (PPG) output mode

Timers 0, 2, 4, or 6 can output square waves with variable frequencies and variable duty (programmable pulse generation). The output pulse can be set to either active low or active high. Timers 1, 3, 5, and 7 cannot be used in this mode.

Timer 0 outputs from pin TO1 (shared with pin P71), timer 2 outputs from pin TO3 (shared with pin P72), timer 4 outputs from TO5 (shared with pin P74), and timer 6 outputs from TO7 (shared with pin P75).

The following describes the example of timer 0. (Timers 2, 4, 6 operate the same as timer 0.)

A programmable square wave can be output from pin TO1 by setting 8-bit programmable square wave output mode and enabling inversion of the timer flip-flop TFF1.

The TFF1 value is inverted by a match between 8-bit up-counter UC0 and TREG0, and by a match with TREG1. UC0 is cleared by a match with TREG1.

In PPG mode, timer 1 cannot be used, but timer 1 up-counter UC1 must be run (T8RUN<T1RUN> = 1).

Also, the TREG0 and TREG1 settings in PPG mode must satisfy the following condition.

(TREG0 setting value) < (TREG1 setting value)

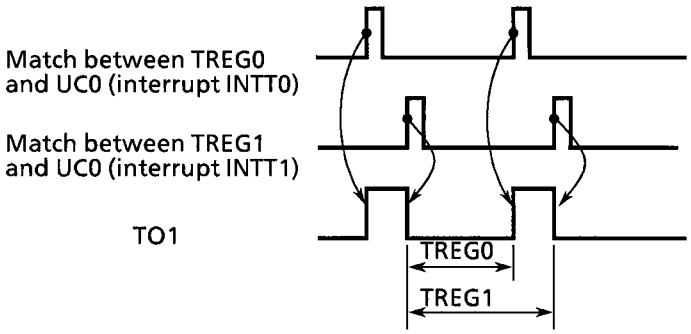


Figure 3.7.8 8-Bit PPG Output Waveform

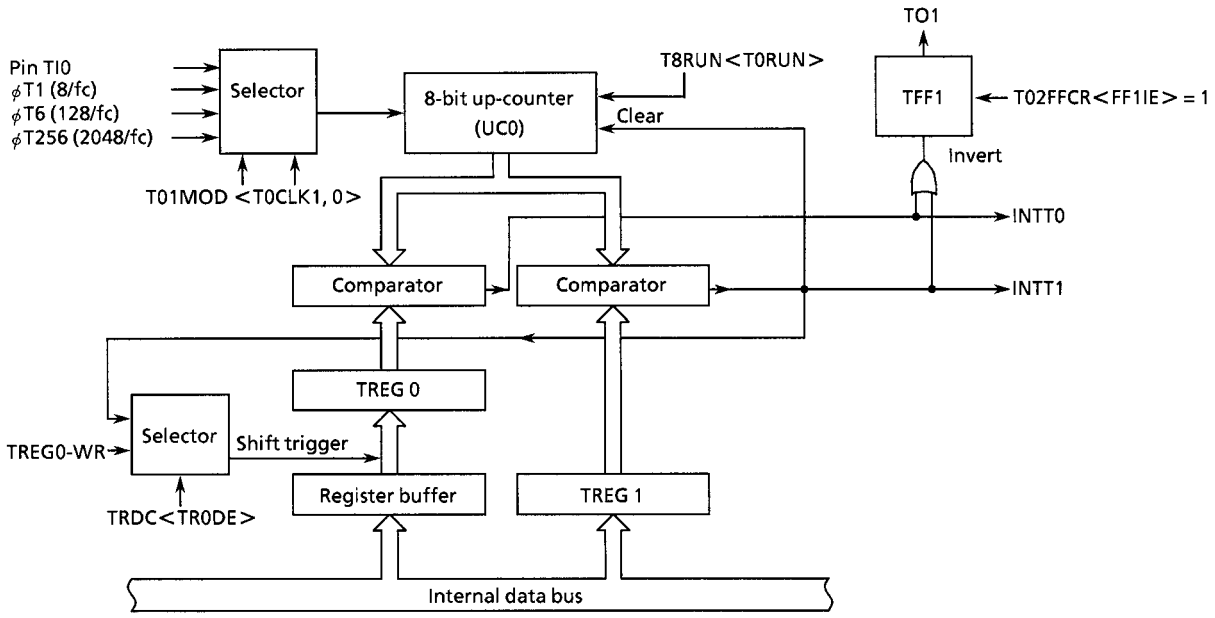


Figure 3.7.9 Block Diagram of 8-Bit PPG Output Mode

Enabling the timer register TREG0 double buffer in this mode shifts the register buffer value to TREG0 when timer register TREG1 matches 8-bit up-counter UC0.

Using the double buffer facilitates handling of small duty waves (when changing the duty).

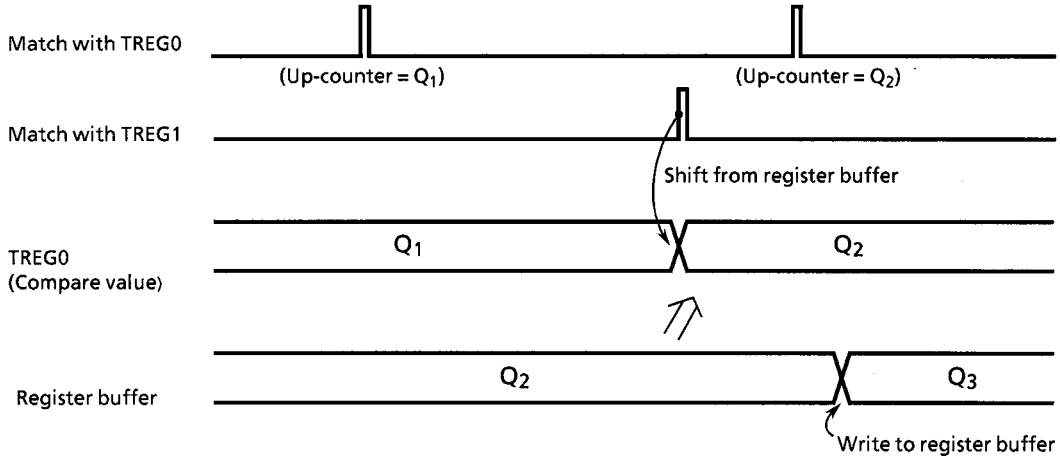


Figure 3.7.10 Register Buffer Operation

Example: Output 1/4-duty 75 kHz-pulse (@ fc = 24 MHz)

Calculate the setting of the timer register.

Setting the frequency to 75 kHz creates a square wave with a cycle of t = 1/75 kHz = 13.3 μs.

Using φT1 = 0.33 μs (@ fc = 24 MHz) results in

13.3 μs ÷ 0.33 μs = 40

Accordingly, set TREG1 = 40 = 28H.

Next, set the duty to 1/4 as follows:

t × 1/4 = 13.3 μs × 1/4 = 3.3 μs

As with TREG1,

3.3 μs ÷ 0.33 μs = 10

Accordingly, set TREG0 = 10 = 0AH.

	MSB 7	6	5	4	3	2	1	LSB 0	
T8RUN ←	-	-	-	-	-	-	0	0	Stop timers 0 and 1, and clear to 0.
T16RUN ←	0	X	-	-	X	X	X	X	
T01MOD ←	1	0	X	X	0	1	0	1	Set 8-bit PPG mode and set input clock to φT1.
T02FFCR ←	-	-	-	-	0	1	1	x	Set TFF1 and enable inversion. Setting to 10 obtains negative logic output wave.
TREG0 ←	0	0	0	0	1	0	1	0	Write 0AH.
TREG1 ←	0	0	1	0	1	0	0	0	Write 28H.
P7CR ←	X	X	-	-	-	-	1	-	Set P71 to TO1 pin.
P7FC ←	X	X	-	-	X	-	1	X	
T16RUN ←	1	X	-	-	X	X	X	X	
T8RUN ←	-	-	-	-	-	-	1	1	Start timers 0 and 1 counting.

Note: X : Don't care − : No change

(4) 8-bit pulse width modulation (PWM) output mode

Only timers 0, 2, 4, 6 can be set to this mode, which allows up to four pulse width modulation outputs with 8-bit resolution. Timers 1, 3, 5, and 7 can be used as 8-bit timers.

In the case of timer 0, PWM is output to pin TO1 (shared with P71). In the case of timers 2, 4, 6, PWM is output to pins TO3 (shared with P72), TO5 (shared with P74), and TO7 (shared with P75) respectively.

Here, the example of timer 0 is used. (Timers 2, 4, 6 operate the same as timer 0.)

Timer output inversion occurs when the 8-bit up-counter UC0 setting and the timer register TREG0 setting match, or when $2^n - 1$ (T01MOD specifies one of n = 6, n = 7, or n = 8) counter overflow occurs. UC0 is cleared by the $2^n - 1$ counter overflow.

In addition, the following conditions must be satisfied when using 8-bit PWM output mode:

(Timer register setting) < ($2^n - 1$ counter overflow setting)

(Timer register setting) ≠ 0

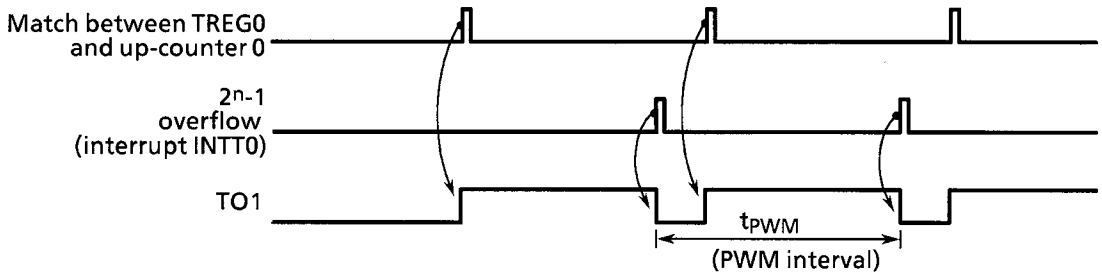


Figure 3.7.11 8-Bit PWM Output Waveform

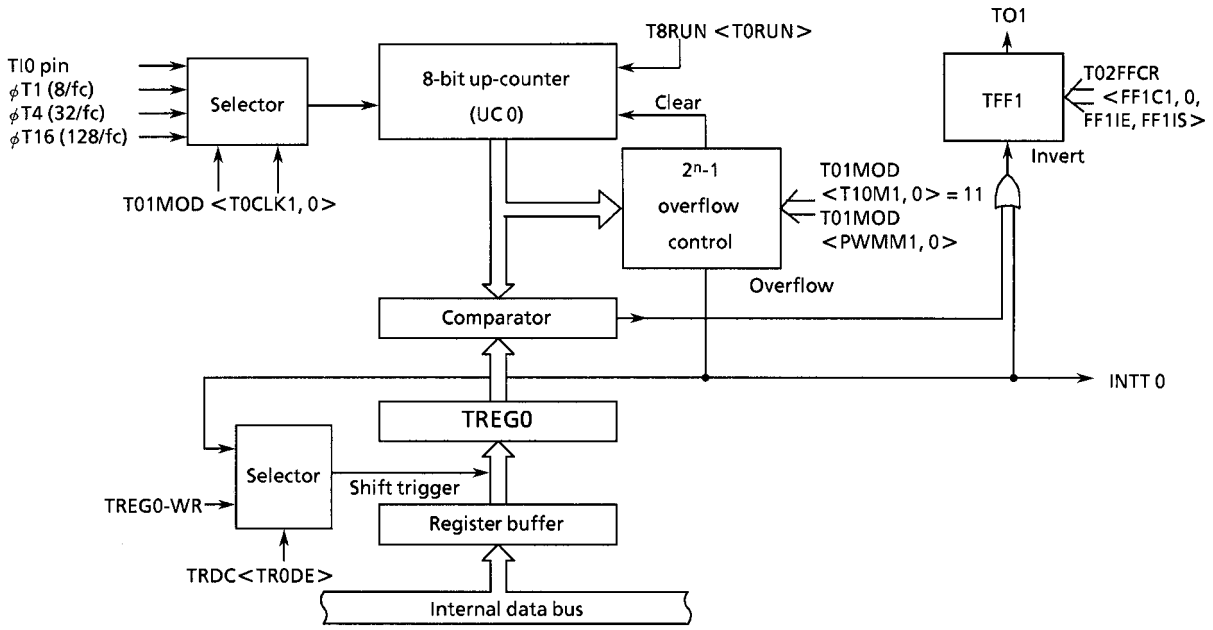


Figure 3.7.12 Block Diagram of 8-Bit PWM Output Mode

Enabling the TREG0 double-buffer in this mode shifts the register buffer value to TREG0 when $2^n - 1$ counter overflow is detected.

Using the double buffer facilitates handling of small duty waves.

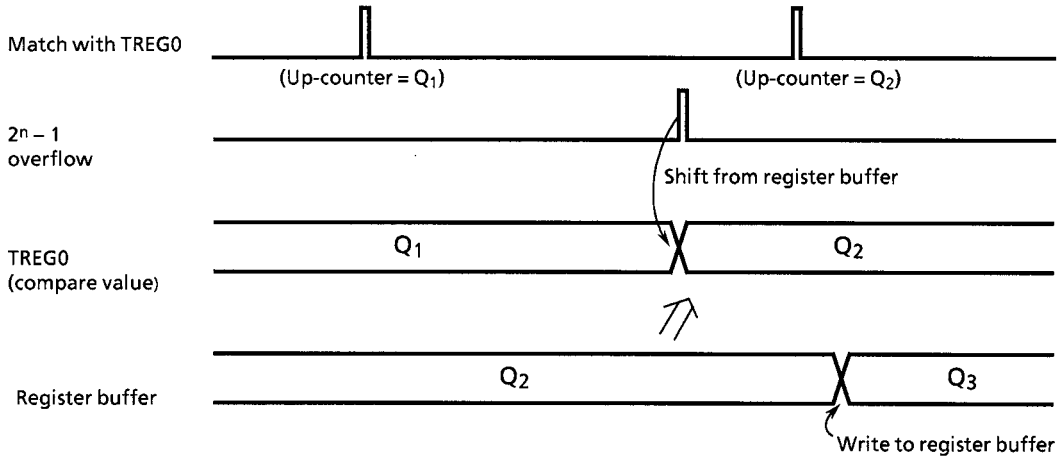
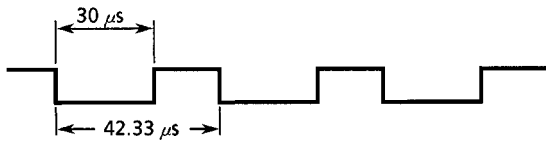


Figure 3.7.13 Register Buffer Operation

Example: Output following PWM waveform to pin TO1 (@ fc = 24 MHz)



To realize a PWM interval of 42.33 μs using φT1 = 0.33 μs (@ fc = 24 MHz):

$42.33\ \mu s \div 0.33\ \mu s = 127 = 2^n - 1$

Accordingly, set n = 7.

As the low level cycle is 30 μs, at φT1 = 0.33 μs,

$30\ \mu s \div 0.33\ \mu s = 90$

Accordingly, set TREG0 = 90 = 5AH.

		MSB 7	6	5	4	3	2	1	LSB 0	
T8RUN	←	-	-	-	-	-	-	-	0	Stop timer 0 and clear to 0.
T01MOD	←	1	1	1	0	-	-	0	1	Set 8-bit PWM mode (interval = 2^7 -1) and set input clock to ϕT1.
T02FFCR	←	-	-	-	-	1	0	1	X	Clear TFF1 and enable inversion.
TREG0	←	0	1	0	1	1	0	1	0	Write 5AH.
P7CR	←	X	X	-	-	-	-	1	-	Set P71 to pin TO1.
P7FC	←	X	X	-	-	X	-	1	X	
T16RUN	←	1	X	-	-	X	X	X	X	
T8RUN	←	-	-	-	-	-	-	-	1	Start timer 0 counting.

Note: X : Don't care − : No change

Table 3.7.4 shows the timer input clock source and the PWM interval determined by the ($2^n - 1$) counter.

Table 3.7.4 Setting of PWM Interval (@ fc = 24 MHz)

Input Clock / ($2^n - 1$) Counter	ϕT1	ϕT4	ϕT16
$2^6 - 1$	21 μs (47.6 kHz)	84 μs (11.9 kHz)	336 μs (3.0 kHz)
$2^7 - 1$	42.3 μs (23.6 kHz)	169.3 μs (5.9 kHz)	677.3 μs (1.5 kHz)
$2^8 - 1$	85 μs (11.8 kHz)	340 μs (2.9 kHz)	1.36 ms (0.7 kHz)

(5) Timer mode list

The 8-bit timers 0 to 7 can be set to 8-bit timer mode, 16-bit timer mode, 8-bit PPG mode, or 8-bit PWM mode. Table 3.7.5 lists settings for the timer modes.

Table 3.7.5 Settings for All Timer Modes

Register Name	TxxMOD				TxxFFCR
bit Symbol / Timer mode (for 8-bit timer channels × 2)	Timer mode <T01M1, 0> <T23M1, 0> <T45M1, 0> <T67M1, 0>	PWM interval <PWM01, 00> <PWM21, 20> <PWM41, 40> <PWM61, 60>	Upper timer input clock <T1CLK1, 0> <T3CLK1, 0> <T5CLK1, 0> <T7CLK1, 0>	Lower timer input clock <T0CLK1, 0> <T2CLK1, 0>(note) <T4CLK1, 0> <T6CLK1, 0>(note)	Inversion select <FF1IS> <FF3IS> <FF5IS> <FF7IS>
16-bit timer (full 16 bits) × 1ch	01	–	–	00: External input 01: ϕT1 10: ϕT4 11: ϕT16	–
8-bit timer (8-bit × 8-bit mode) × 1ch (Inputs lower timer comparator output to upper timer)	00	–	00	00: External input 01: ϕT1 10: ϕT4 11: ϕT16	0: Lower timer 1: Upper timer
8-bit timer × 2ch	00	–	00: Don't care 01: ϕT1 10: ϕT16 11: ϕT256	00: External input 01: ϕT1 10: ϕT4 11: ϕT16	0: Lower timer 1: Upper timer
8-bit PPG × 1ch	10	–	–	00: External input 01: ϕT1 10: ϕT4 11: ϕT16	–
8-bit PWM × 1ch (lower) 8-bit timer × 1ch (upper)	11	00: Don't care 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$	00: Don't care 01: ϕT1 10: ϕT16 11: ϕT256	00: External input 01: ϕT1 10: ϕT4 11: ϕT16	–

Note: External clock is not input to timer 2 or timer 6.

3.8 16-Bit Timers/Event Counters

The TMP95CU54A incorporates two multi-function 16-bit timer/event counters (timers 8 and 9). Timers 8 and 9 have the same functions and can operate independently. The 16-bit timers have the following three operating modes.

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

The capture function can also be used to perform the following operations.

- One-shot pulse output from the external trigger pulse
- Frequency measurement
- Pulse width measurement
- Time differential measurement

Also, the 16-bit timers can be used to output a signal with any phase difference.

Figure 3.8.1 is a block diagram of the 16-bit timer/event counters (timer 8). Timer 9 has the same circuit configuration.

Each 16-bit timer/event counter consists of a 16-bit up-counter, a 16-bit comparator, a 16-bit timer register, and a 16-bit capture register. Timers 8 and 9 each have two timer flip-flops (TFF8/9 and TFFA/B).

Clock sources φT1, φT4, and φT16 input to the 16-bit timers are obtained from the internal 9-bit prescaler (see 3.7.2 (1), Prescaler).

The 16-bit timer/event counters are controlled by six control registers (T8MOD, T9MOD, T8FFCR, T9FFCR, T16RUN, and T89CR).

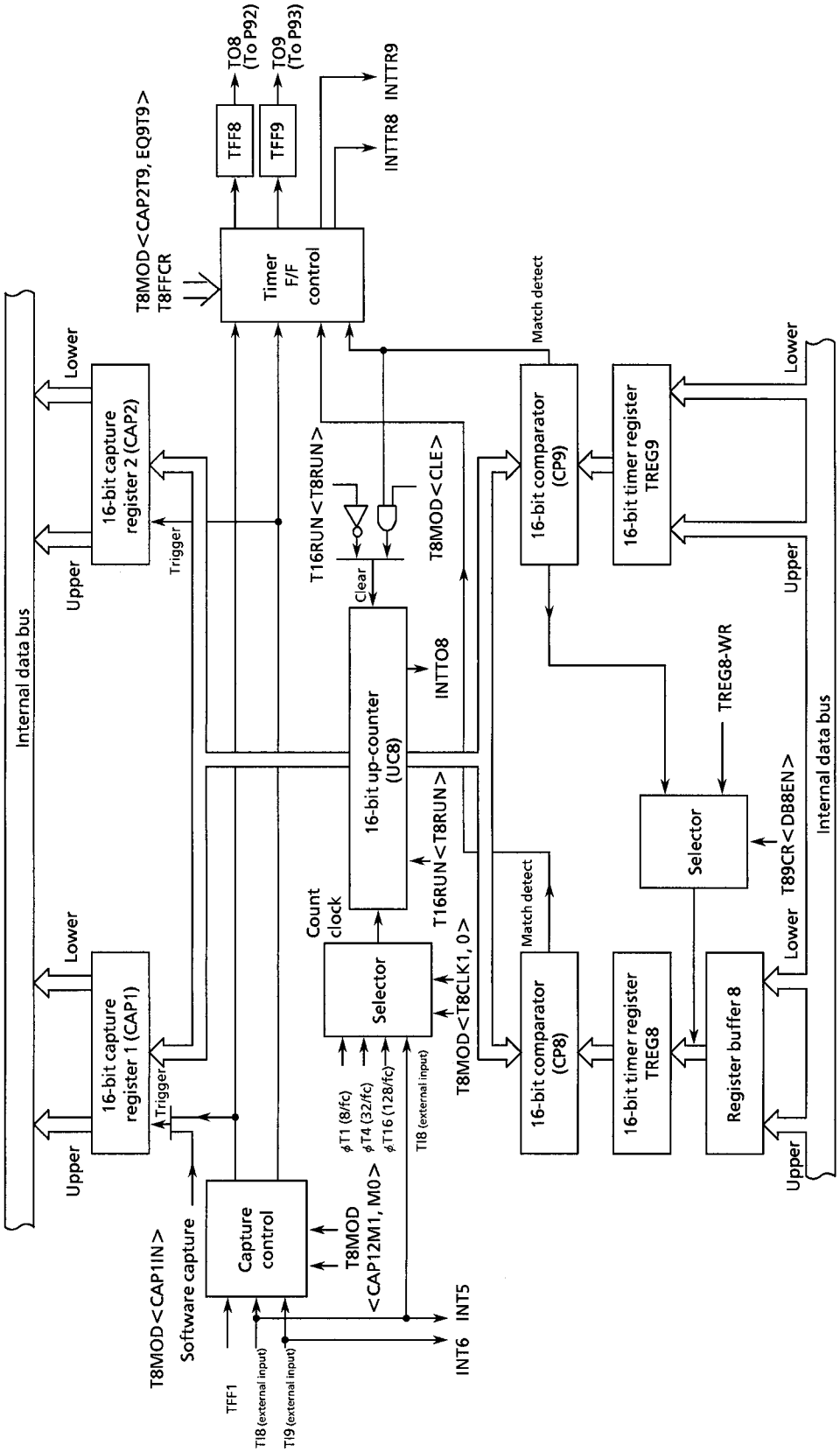


Figure 3.8.1 16-Bit Timer Block Diagram (Timer 8)

3.8.1 16-Bit Timer/Event Counter Registers

Figure 3.8.2 shows the 16-bit timer/event counter related registers.
These register settings control the 16-bit timer/event counter operations.

Timer 8, 9 Control Register

T89CR (003AH)

	7	6	5	4	3	2	1	0
bit Symbol	–					–	DBAEN	DB8EN
Read/Write	R/W					R/W		
After reset	0					0	0	0
Function	Note: Always fixed to 0					Note: Always fixed to 0	TREGA double buffer control 0: Disable 1: Enable	TREG8 double buffer control 0: Disable 1: Enable

Timer register A double buffer control

0	Disable
1	Enable

Timer register 8 double buffer control

0	Disable
1	Enable

16-bit Timer Operation Control Register

T16RUN (003BH)

	7	6	5	4	3	2	1	0
bit Symbol	PRRUN		T9RUN	T8RUN				
Read/Write	R/W		R/W					
After reset	0		0	0				
Function	Prescaler operation control 0: Stop and clear 1: Run		Timer 9 operation control 0: Stop and clear 1: Run	Timer 8 operation control 0: Stop and clear 1: Run				

Timer 8 up-counter operation control

0	Stop and clear
1	Run

Prescaler operation control (Note)

0	Stop and clear
1	Run

Timer 9 up-counter operation control

0	Stop and clear
1	Run

Note: When running a 16-bit timer, set T16RUN<PRRUN> to 1.

Figure 3.8.2 16-Bit Timer/Event Counter Related Registers (1/5)

Timer 8 Mode Control Register

T8MOD (0038H)	7	6	5	4	3	2	1	0
bit Symbol	CAP2T9	EQ9T9	CAP1IN	CAP12M1	CAP12M0	CLE	T8CLK1	T8CLK0
Read/Write	R/W		W	R/W				
After reset	0	0	1	0	0	0	0	0
Function	TFF9 inversion trigger 0: Trigger disable 1: Trigger enable At loading of up-counter value to CAP2	At match between up-counter and TREG9	Software capture control 0: Software capture 1: Don't care	Capture timing 00: Disable INT5 at rising edge 01: TI8↑ TI9↑ INT5 on rising edge 10: TI8↑ TI8↓ INT5 on falling edge 11: TFF1↑ TFF1↓ INT5 on rising edge		Timer 8 up-counter control 0: Clear disable 1: Clear enable	Timer 8 source clock selection 00: TI8 01: ϕT1 10: ϕT4 11: ϕT16	

Timer 8 capture timing

	Capture control	INT5 control
00	Capture disable	INT5 generated on TI8 rising edge
01	CAP1 on TI8 rising edge CAP2 on TI9 rising edge	
10	CAP1 on TI8 rising edge CAP2 on TI8 falling edge	INT5 generated on TI8 falling edge
11	CAP1 on TFF1 rising edge CAP2 on TFF1 falling edge	INT5 generated on TI8 rising edge

Timer 8 input clock

00	External input clock (TI8)
01	ϕT1 (8/fc)
10	ϕT4 (32/fc)
11	ϕT16 (128/fc)

Up-counter (UC8) clear

0	Disable up-counter clear
1	Clear at match with TREG9

Software capture control
(always read as 1)

0	At loading of up-counter 8 value to CAP1
1	Don't care

At match between up-counter and TREG9
Invert trigger of timer flip-flop 9 (TFF9)

0	Trigger disable (inversion disabled)
1	Trigger enable (inversion enabled)

At loading of up-counter value to CAP2
Invert trigger of timer flip-flop 9 (TFF9)

0	Trigger disable (inversion disabled)
1	Trigger enable (inversion enabled)

Figure 3.8.2 16-Bit Timer/Event Counter Related Register (2/5)

Timer 8 Flip-Flop Control Register

T8FFCR (0039H)	7	6	5	4	3	2	1	0
bit Symbol	TFF9C1	TFF9C0	CAP2T8	CAP1T8	EQ9T8	EQ8T8	TFF8C1	TFF8C0
Read/Write	W		R/W				W	
After reset	1	1	0	0	0	0	1	1
Function	TFF9 control 00: Invert TFF9 01: Set TFF9 10: Clear TFF9 11: Don't care		TFF8 inversion trigger 0: Trigger disable 1: Trigger enable				TFF8 control 00: Invert TFF8 01: Set TFF8 10: Clear TFF8 11: Don't care	
			At loading of up-counter value to CAP2	At loading of up-counter value to CAP1	At match between up-counter and TREG9	At match between up-counter and TREG8		

Timer flip-flop 8 (TFF8) control

00	Invert TFF8 value (software inversion)
01	Set TFF8 to 1
10	Clear TFF8 to 0
11	Don't care (always read as 11)

At match between up-counter and TREG8
Invert trigger of timer flip-flop 8 (TFF8)

0	Trigger disable (inversion disabled)
1	Trigger enable (inversion enabled)

At match between up-counter and TREG9
Invert trigger of timer flip-flop 8 (TFF8)

0	Trigger disable (inversion disabled)
1	Trigger enable (inversion enabled)

At loading of up-counter value to CAP1
Invert trigger of timer flip-flop 8 (TFF8)

0	Trigger disable (inversion disabled)
1	Trigger enable (inversion enabled)

At loading of up-counter value to CAP2
Invert trigger of timer flip-flop 8 (TFF8)

0	Trigger disable (inversion disabled)
1	Trigger enable (inversion enabled)

Timer flip-flop 9 (TFF9) control

00	Invert TFF9 value (software inversion)
01	Set TFF9 to 1
10	Clear TFF9 to 0
11	Don't care (always read as 11)

Figure 3.8.2 16-Bit Timer/Event Counter Related Register (3/5)

Timer 9 Mode Control Register

T9MOD (0048H)	7	6	5	4	3	2	1	0
bit Symbol	CAP4TB	EQBTB	CAP3IN	CAP34M1	CAP34M0	CLE	T9CLK1	T9CLK0
Read/Write	R/W		W	R/W				
After reset	0	0	1	0	0	0	0	0
Function	TFFB inversion trigger 0: Trigger disable 1: Trigger enable At loading of up-counter value to CAP4	At match between up-counter and TREGB	Software capture control 0: Software capture 1: Don't care	Capture timing 00: Disable INT7 is rising edge 01: TIA↑ TIB↑ INT7 on rising edge 10: TIA↑ TIA↓ INT7 on falling edge 11: TFF1↑ TFF1↓ INT7 on rising edge		Timer 9 up-counter control 0: Clear disable 1: Clear enable	Timer 9 source clock selection 00: TIA 01: ϕT1 10: ϕT4 11: ϕT16	

Timer 9 capture timing

	Capture control	INT7 control
00	Capture disable	INT7 generated on TIA rising edge ↗
01	CAP3 on TIA rising edge CAP4 on TIB rising edge	
10	CAP3 on TIA rising edge CAP4 on TIA falling edge	INT7 generated on TIA falling edge ↘
11	CAP3 on TFF1 rising edge CAP4 on TFF1 falling edge	INT7 generated on TIA rising edge ↗

Timer 9 input clock

00	External input clock (TIA)
01	ϕT1 (8/fc)
10	ϕT4 (32/fc)
11	ϕT16 (128/fc)

Up-counter (UC9) clear

0	Disable up-counter clear
1	Clear at match with TREGB

Software capture control
(always read as 1)

0	At loading of up-counter 9 value to CAP3
1	Don't care

At match between up-counter and TREGB
Invert trigger of timer flip-flop B (TFFB)

0	Trigger disable (inversion disabled)
1	Trigger enable (inversion enabled)

At loading of up-counter value to CAP2
Timer flip-flop B (TFFB) inversion trigger

0	Trigger disable (inversion disabled)
1	Trigger enable (inversion enabled)

Figure 3.8.2 16-Bit Timer/Event Counter Related Register (4/5)

Timer 9 Flip-Flop Control Register

T9FFCR (0049H)	7	6	5	4	3	2	1	0
bit Symbol	TFFBC1	TFFBC0	CAP4TA	CAP3TA	EQBTA	EQATA	TFFAC1	TFFAC0
Read/Write	W		R/W				W	
After reset	1	1	0	0	0	0	1	1
Function	TFFB control 00: Invert TFFB 01: Set TFFB 10: Clear TFFB 11: Don't care		TFFA inversion trigger 0: Trigger disable 1: Trigger enable				TFFA control 00: Invert TFFA 01: Set TFFA 10: Clear TFFA 11: Don't care	
			At loading of up-counter value to CAP4	At loading of up-counter value to CAP3	At match between up-counter and TREG9	At match between up-counter and TREG8		

Timer flip-flop A (TFFA) control

00	Invert TFFA value (software inversion)
01	Set TFFA to 1
10	Clear TFFA to 0
11	Don't care (always read as 11)

At match between up-counter and TREGA
Timer flip-flop A (TFFA) inversion trigger

0	Trigger disable (inversion disabled)
1	Trigger enable (inversion enabled)

At match between up-counter and TREGB
Timer flip-flop A (TFFA) inversion trigger

0	Trigger disable (inversion disabled)
1	Trigger enable (inversion enabled)

At loading of up-counter value to CAP3
Timer flip-flop A (TFFA) inversion trigger

0	Trigger disable (inversion disabled)
1	Trigger enable (inversion enabled)

At loading of up-counter value to CAP4
Timer flip-flop A (TFFA) inversion trigger

0	Trigger disable (inversion disabled)
1	Trigger enable (inversion enabled)

Timer flip-flop B (TFFB) control

00	Invert TFFB value (software inversion)
01	Set TFFB to 1
10	Clear TFFB to 0
11	Don't care (always read as 11)

Figure 3.8.2 16-Bit Timer/Event Counter Related Register (5/5)

3.8.2 Block Structure

(1) 16-bit up-counters

16-bit up-counters UC8 and 9 are 16-bit binary counters for timers 8 and 9.

These up-counters count up on the external and internal clocks selected by 16-bit timer mode control registers T8MOD and T9MOD. To control the up-counter operations, use 16-bit timer operation control register T16RUN.

The UC8, 9 input clock is selected from either internal clocks φT1, φT4, and φT16, or the external clocks input from the timer input pin (TI8 and TI9).

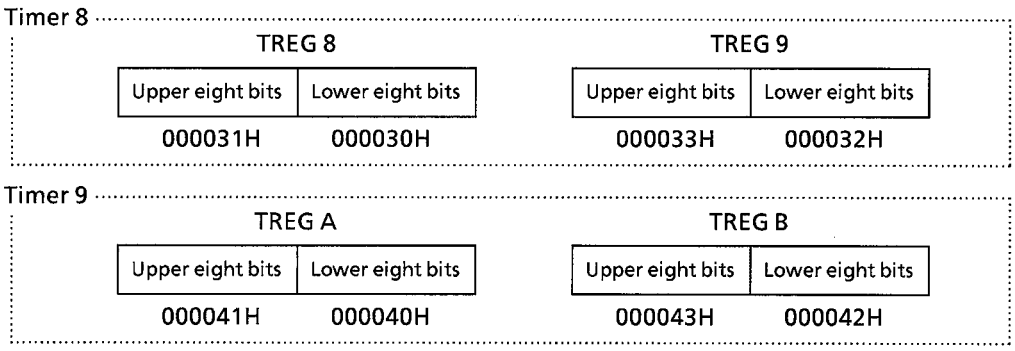
Any overflow from UC8 or 9 triggers interrupt request INTTO8 or INTTO9.

At a reset, T16RUN is cleared, and the prescaler and UC8, 9 are stopped.

(2) 16-bit timer registers

Each timer has two internal 16-bit timer registers for setting counters. A match between these timer register settings and the value of the 16-bit up-counter UC8, 9 outputs a comparator match detect signal.

Data set to 16-bit timer registers TREG8, TREG9 and TREGA, TREGB use a 2-byte data transfer instruction, or two 1-byte data transfer instructions; first for the lower eight bits, then for the upper eight bits.



TREG8 to TREGB are write-only registers and therefore cannot be read.

Of the 16-bit timer registers, TREG8 and TREGA have a double-buffer configuration (each has a register buffer).

Timer 8, 9 control register T89CR<DB8EN, DBAEN> enables/disables the double buffer. Setting <DB8EN, DBAEN> to 0 disables the double buffer; setting <DB8EN, DBAEN> to 1 enables the double buffer.

With the double buffer enabled, data are transmitted from the register buffer to the timer register at a match between up-counter UC8 and TREG9, or between UC9 and timer register TREGB.

As TREG8 to TREGB are undefined after a reset, when using a 16-bit timer write the data first.

A reset clears T89CR to 0 and disables the double buffer. When using the double buffer, write data to TREG8, TREGA, set T89CR<DB8EN, DBAEN> to 1, then write the next data to the register buffer.

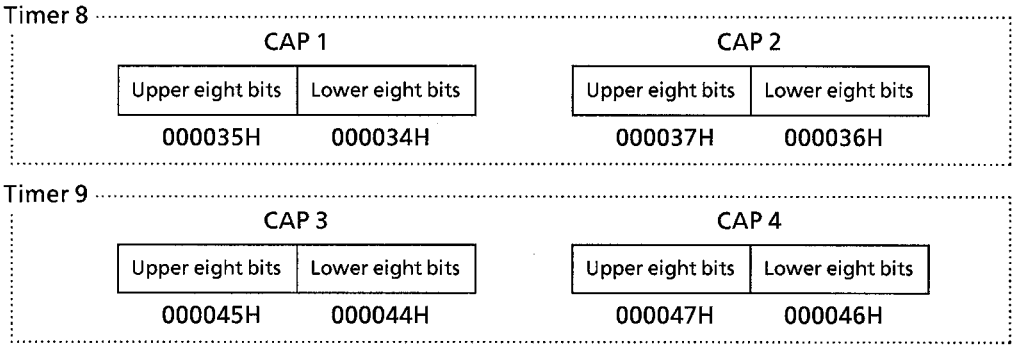
The 16-bit timer registers and register buffers are allocated to the same addresses in memory. When T89CR<DB8EN, DBAEN> is set to 0, the same value is written to the timer register and register buffer.

When <DB8EN, DBAEN> is set to 1, the value is written to the register buffer only. Therefore, the register buffer must be disabled before writing the initial value to the timer register.

(3) Capture register

The capture register is a 16-bit register for latching the 16-bit up-counter UC8, 9 value.

When reading the capture register, use a 2-byte data load instruction, or two 1-byte data load instructions; first to read the lower eight bits, then to read the upper eight bits.



CAP1 to CAP4 are read-only registers and cannot be written by software.

(4) Capture input control

The capture input control circuit controls the timing of the latching of the 16-bit up-counter UC8, 9 value to capture registers CAP1, CAP2, CAP3, and CAP4. Set the capture register latch timing with the timer 8, 9 mode control registers T8MOD<CAP12M1, 0>, T9MOD<CAP34M1, 0>.

The following describes the latch timing setting and operation.

- When T8MOD<CAP12M1, 0>, T9MOD<CAP34M1, 0> are set to 00:
  The capture function is disabled. A reset also disables the capture function.
- When T8MOD<CAP12M1, 0>, T9MOD<CAP34M1, 0> are set to 01:
  On the external input rising edge of TI8 (shared with P90/INT5) and TIA (shared with P94/INT7), capture register CAP1, CAP3 loads the up-counter value. On the external input rising edge of TI9 (shared with P91/INT6) and TIB (shared with P95/INT8), capture register CAP2, CAP4 loads the up-counter value. (Time differential measurement)
- When T8MOD<CAP12M1, 0>, T9MOD<CAP34M1, 0> are set to 10:
  On the TI8, TIA external input rising edge, capture register CAP1, CAP3 loads the up-counter value. On the input falling edge, capture register CAP2, CAP4 loads the up-counter value. Interrupt INT4, INT6 is generated on a falling edge in this mode only. (Pulse width measurement)
- When T8MOD<CAP12M1, 0>, T9MOD<CAP34M1, 0> are set to 11:
  On the timer flip-flop TFF1 rising edge, capture register CAP1, CAP3 loads the up-counter value. On the falling edge, capture register CAP2, CAP4 loads the up-counter value.
  The UC8, 9 up-counter value can also be loaded to a capture register on a software request. When 0 is written to T8MOD<CAP1IN>, T9MOD<CAP3IN>, the UC8, 9 up-counter value at that time is loaded to capture register CAP1, 3.
  The prescaler must first be set to RUN (set T16RUN<PRRUN> = 1).

(5) Comparator

To detect a match, the 16-bit comparator compares the 16-bit up-counter UC8, 9 with the 16-bit timer register TREG8, 9 and TREGA, B settings.

On detection of a match, the comparator outputs a match detect signal and generates interrupts INTTR8, 9 or INTTRA, B from the respective 16-bit timer.

UC8 is cleared by a match between the UC8 value and the TREG9 value. UC9 is cleared by a match between the UC9 value and the TREGB value. UC8, 9 clearing can be disabled by setting the timer 8, 9 mode control registers T8MOD<CLE>, T9MOD<CLE> to 0.

(6) Timer flip-flops

Timers 8 and 9 have two timer flip-flops each. The flip-flops of each timer have different functions.

[1] TFF8, TFFA

Flip-flops TFF8 and TFFA are inverted by a match signal from the comparator and a latch signal to the capture register.

In timer 8 and timer 9, two different capture operations and two types of match detection can be specified as inversion triggers. Use bits 2 to 5 of the T8FFCR and T9FFCR registers to set the inversion triggers.

[2] TFF9, TFFB

Timer flip-flops TFF9 and TFFB are inverted by a match signal from the comparator and a latch signal to the capture register.

In timers 8 and 9, one type of capture operation and one type of match detection can be specified as inversion triggers. Use bits 6 and 7 of the T8MOD and T9MOD registers to set the inversion triggers.

After a reset, the timer flip-flop values are undefined. Writing 01 to T8FFCR <TFF8C1, 0>, <TFF9C1, 0> or T9FFCR <TFFAC1, 0>, <TFFBC1, 0> sets the timer flip-flop to 0; writing 10 to the bits sets the timer flip-flop to 1. Writing 00 to the bits inverts the timer flip-flop value (software inversion).

The TFF8, TFF9, TFFA, and TFFB values can be output to timer output pins TO8 (shared with P92), TO9 (shared with P93), TOA (shared with P96), and TOB (shared with P96) respectively.

As the timer output pins also function as P92, P93, and P96, set port 9 function register P9FC before performing timer output. (See Figure 3.5.35 Register for Port 9)

3.8.3 Operation Description for Each Mode

(1) 16-bit interval timer mode

Interval timers 8 and 9 can be used independently as 16-bit interval timers. The following describes the example of timer 8 only.

Example: Generate interrupts at fixed intervals

To generate timer interrupts at fixed intervals, set the interval time (cycle) in 16-bit timer register TREG9 and use interrupt INTTR9.

Set the registers as follows.

```
            7 6 5 4 3 2 1 0
T16RUN  ← - X - 0 X X X X      Stop timer 8.
INTET89 ← 1 1 0 0 1 0 0 0      Enable INTTR9, set interrupt level to 4, and disable
                               INTTR8.
T8FFCR  ← 1 1 0 0 0 0 1 1      Disable trigger.
T8MOD   ← 0 0 1 0 0 1 * *      Set internal clock to input clock, disable capture
            (** = 01, 10, 11)  function, clear and enable up-counter.
TREG9   ← * * * * * * * *      Set interval time. (16 bits)
          * * * * * * * *
T16RUN  ← 1 X - 1 X X X X      Start timer 8.
```

Note: X : Don't care − : No change

(2) 16-bit event counter mode

Timers 8 and 9 can be set to operate as event counters by setting external inputs TI8 and TIA as the timer clock sources. The following describes timer 8 only.

The 16-bit up-counter UC8 counts up on the rising edge of the TI8 input. The count value can be read by performing a software capture and reading the capture value.

Timer input pin TI8 is shared with P90. However, there is no selection function. Therefore, event counter operation can be performed at any time by setting timer 8 to operating state. Set the registers as follows.

```
            7 6 5 4 3 2 1 0
T16RUN  ← - X - 0 X X X X      Stop timer 8.
P9CR    ← - - - - - - - 0      Set P90 to input mode.
INTET89 ← 1 1 0 0 1 0 0 0      Enable INTTR9 (level 4) and disable INTTR8.
T8FFCR  ← 1 1 0 0 0 0 1 1      Disable trigger.
T8MOD   ← 0 0 1 0 0 1 0 0      Set input clock to TI8.
TREG9   ← * * * * * * * *      Set number of counts (16 bits).
          * * * * * * * *
T16RUN  ← 1 X - 1 X X X X      Start timer 8.
```

Note 1: X : Don't care − : No change

Note 2: The prescaler must also be running when using a 16-bit timer as an event counter (T16RUN<PRRUN> = 1).

(3) 16-Bit programmable pulse generation (PPG) output mode

Timers 8 and 9 can output a square wave with a user-specified frequency and duty (programmable square wave). The output pulse can be either active-low or active-high.

Timer 8 outputs a square wave from pin TO8 (shared with P92) ; timer 9, from TOA (shared with P96).

The following describes timer 8 only.

A programmable pulse (square wave) can be output from pin TO8 by triggering inversion of timer flip-flop TFF8 when a match occurs between the 16-bit up-counter UC8 and TREG8, or between UC8 and TREG9. The TREG8 and TREG9 settings must satisfy the following condition:

(TREG8 setting) < (TREG9 setting)

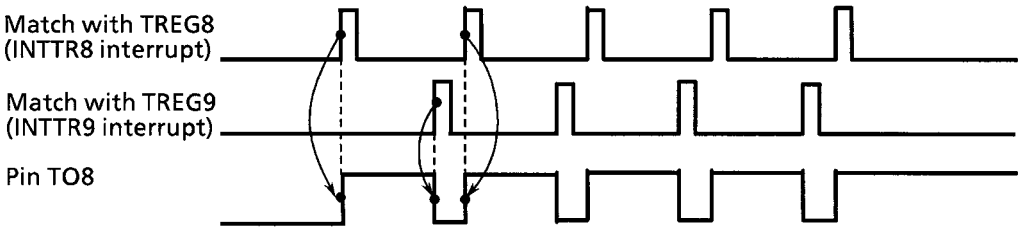


Figure 3.8.3 16-Bit Programmable Pulse Generation (PPG) Output Waveform

Enabling the TREG8 double-buffer in this mode shifts the value of register buffer 8 to TREG8 when TREG9 matches UC8. Using the double-buffer facilitates handling of small duty waves.

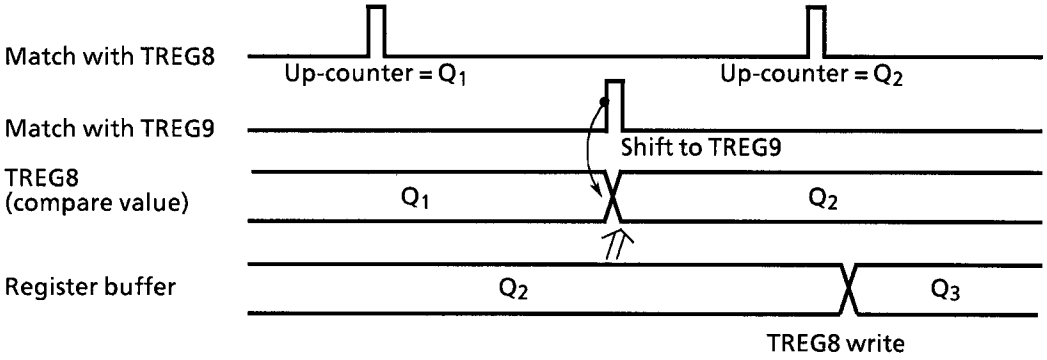


Figure 3.8.4 Register Buffer Operation

Figure 3.8.5 is a block diagram of 16-bit PPG output mode.

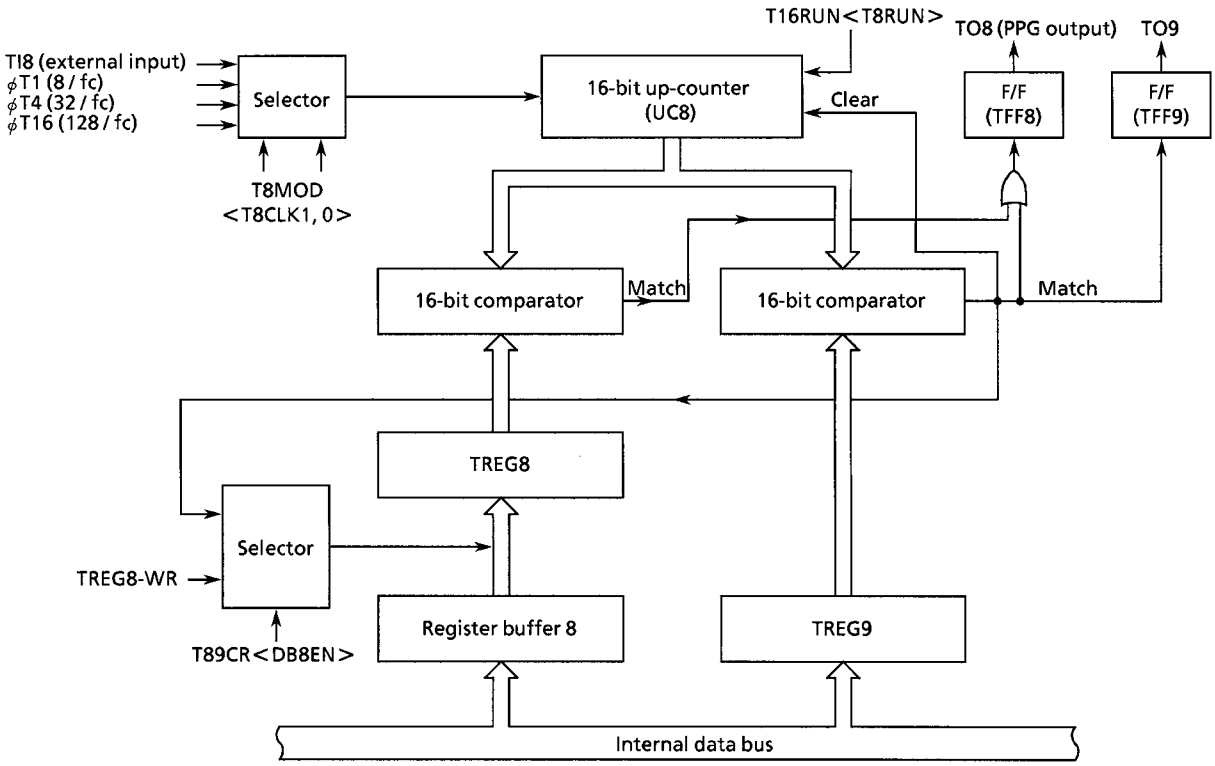


Figure 3.8.5 16-Bit PPG Output Mode Block Diagram

In 16-bit PPG output mode, set the registers as follows.

		7 6 5 4 3 2 1 0	
T16RUN	←	- X - 0 X X X X	Stop timer 8.
TREG8	←	* * * * * * * *	Set the duty. (16 bits)
		* * * * * * * *	
TREG9	←	* * * * * * * *	Set the interval. (16 bits)
		* * * * * * * *	
T89CR	←	0 X X X X 0 - 1	Enable TREG8 double-buffer (Duty/interval modified by INTTR9 interrupt)
T8FFCR	←	1 1 0 0 1 1 1 0	Set TFF8 to invert at detection of match with TREG8 or TREG9. Set TFF8 initial value to 0.
T8MOD	←	0 0 1 0 0 1 **	Set input clock to internal clock, and disable capture function.
		(** = 01, 10, 11)	
P9CR	←	- - - - - 1 - -	Set P92 as TO8.
P9FC	←	X - X X - 1 X X	
T16RUN	←	1 X - 1 X X X X	Start timer 8.

Note: X : Don't care − : No change

(4) Example of capture function application

Use the capture function to realize many applications, including the following examples.

[1] One-shot pulse output from the external trigger pulse

[2] Frequency measurement

[3] Pulse width measurement

[4] Time differential measurement

The following describes these applications based on timer 8.

[1] One-shot pulse output from external trigger pulse

Obtain one-shot pulse output from the external trigger pulse as follows.

Set 16-bit up-counter UC8 to free-running count-up using an internal clock.

Input the external trigger pulse from pin TI8. Load the up-counter value to capture register CAP1 on the rising edge of the external trigger pulse using the capture function.

Interrupt INT5 is generated on the rising edge of the external trigger pulse. Add the value of capture register CAP1 at this interrupt (c) to the delay time (d), and set timer register TREG8 to the sum of these values (c + d). Add the pulse width of the one-shot pulse (p) to TREG8, and set timer register TREG9 to the result (c + d + p).

In addition, set the timer 8 flip-flop control register T8FFCR<EQ9T8, EQ8T8> to 11 and enable the trigger to invert timer flip-flop TFF8 when a match occurs between UC8 and TREG8 or UC8 and TREG9. Then, after output of the one-shot pulse, set the trigger back to disabled state during INTTR9 interrupt processing.

The (c), (d), and (p) notation above corresponds to c, d, and p in Figure 3.8.6, One-Shot Pulse Output from External Trigger Pulse (With Delay).

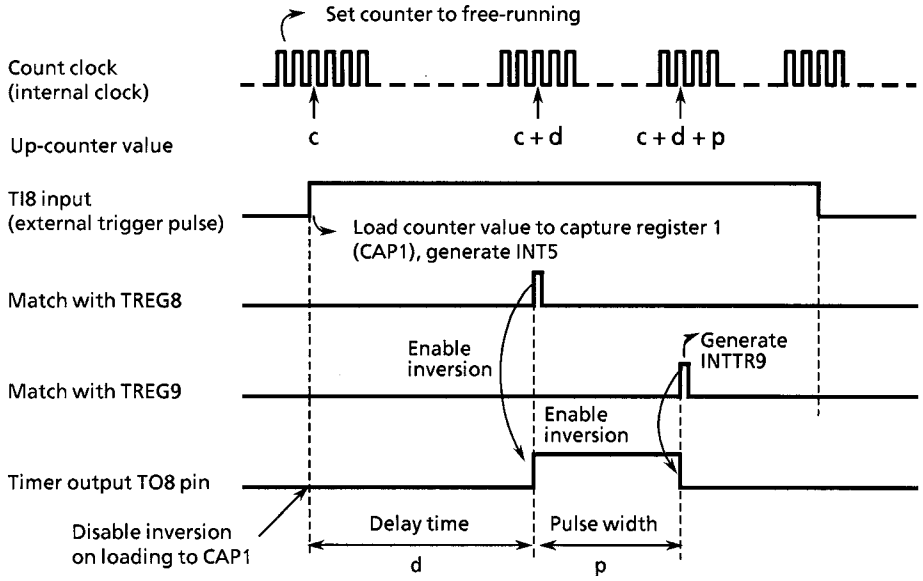


Figure 3.8.6 One-Shot Pulse Output from External Trigger Pulse (With Delay)

Example: On pin TI8, output 2ms one-shot pulse with 3ms-delay after external trigger pulse.

Main settings

```
T8MOD  ← - - 1 0 1 0 0 1   Set to free-running.
                           Set source clock to φT1.
                           Load counter value to CAP1 at TI8 input rising edge.
T8FFCR ← 1 1 0 0 0 0 1 0   Zero clear TFF8 output.
                           Disable TFF8 inversion.
P9CR   ← - - - - - 1 - -   Set P92 as TO8.
P9FC   ← X - X X - 1 X X
INTE45 ← - - - - 1 1 0 0   Enable INT5 and disable INTTR8 and INTTR9.
INTET89← 1 0 0 0 1 0 0 0
T16RUN ← 1 X - 1 X X X X   Start timer 8.
```

Settings at INT5

```
TREG8  ← * * * * * * * *   CAP1+3ms/φT1
TREG9  ← * * * * * * * *   TREG8+2ms/φT1
T8FFCR ← - - - - 1 1 - -   Enable TFF8 inversion on match with TREG8 or TREG9.
INTET89← 1 1 0 0 - - - -   Enable INTTR9.
```

Settings at INTTR9

```
T8FFCR ← - - - - 0 0 - -   Disable TFF8 inversion on match with TREG8 or TREG9.
INTET89← 1 0 0 0 - - - -   Disable INTTR9.
```

Note: X : Don't care − : No change

If delay time is not required, invert timer flip-flop TFF8 by loading capture register 1 (CAP1). Set timer register TREG9 to the sum of the one-shot pulse width (p) and the value of CAP1 at interrupt INT5 (c) (c + p). Set the TFF8 inversion on a match between TREG9 and UC8, and select inversion enable. On interrupt INTTR9, disable the TFF8 inversion.

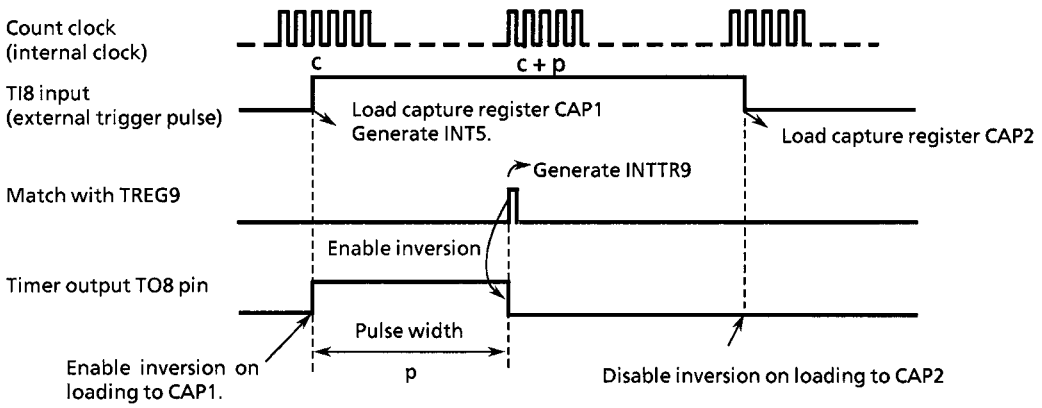


Figure 3.8.7 External Trigger Pulse One-Shot Pulse Output (No Delay)

[2] Frequency measurement

The frequency of an external clock can be measured by the capture function.

The frequency is measured by combining the 8-bit timers (timers 0, 1) in 16-bit event counter mode. (Timers 0 and 1 are used to set the measuring time by inverting TFF1.)

Select the TI8 input as the timer 8 count clock and count timer 8 on the external clock input. Set timer 8 mode control register T8MOD<CAP12M1, 0> to 11. This setting loads the counter value of 16-bit up-counter UC8 into capture register CAP1 on the rising edge of timer flip-flop TFF1. It also loads the counter value into capture register CAP2 on the falling edge of timer flip-flop TFF1. TFF1 is the timer flip-flop of the 8-bit timers (timers 0, 1).

Based on the measuring time, the frequency is calculated from the difference between capture registers CAP1 and CAP2 at the 8-bit timer interrupts (INTT0 or INTT1).

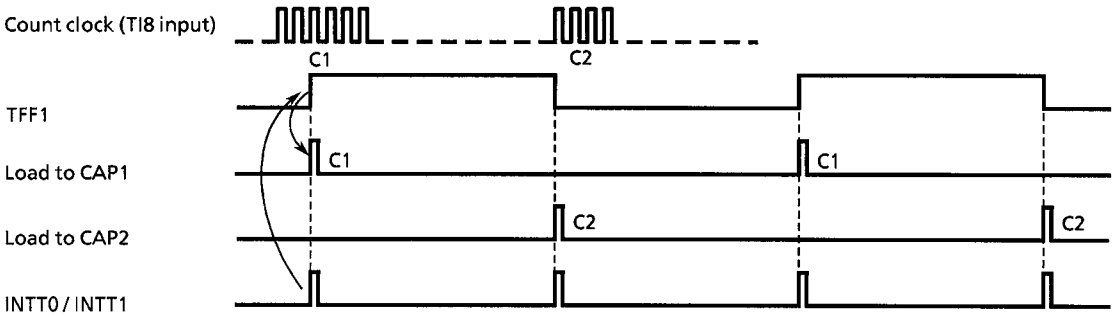


Figure 3.8.8 Frequency Measurement

For example, if TFF1 (8-bit timer flip-flop) is set to 1 for 0.5s, and the difference between CAP1 and CAP2 is 100, the frequency is 100 ÷ 0.5 s = 200 Hz.

[3] Pulse width measurement

The high-level width of an external pulse can be measured using the 16-bit timer capture function.

To measure the pulse width, first set 16-bit up-counter UC8 to operate as a free-running up-counter driven by an internal clock. Using the capture function, load the up-counter value into capture registers CAP1 and CAP2 on the rising and falling edges respectively of the external pulse being measured on the TI8 pin.

Using these settings, the high-level pulse width can be calculated during INT5 interrupt processing by multiplying the difference between CAP1 and CAP2 by the internal clock cycle.

For example, if the difference between CAP1 and CAP2 is 100 and the internal clock cycle is 0.8 µs, the pulse width is 100 × 0.8 µs = 80 µs.

Caution is required when the width of the pulse being measured exceeds the maximum UC8 count time (which is determined by the clock source). Software processing is required in this case.

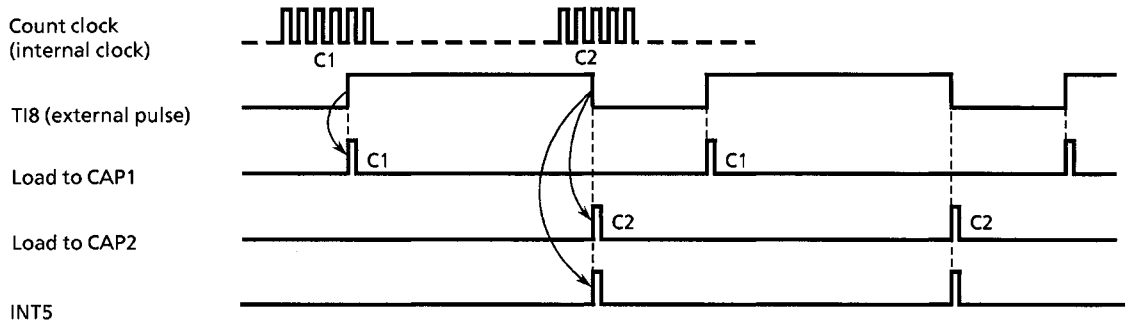


Figure 3.8.9 Pulse Width Measurement

Note: Measure pulse width by setting the timer 8 mode control register T8MOD<CAP12M1, 0> to 10. External interrupt INT5 is generated on the falling edge of the TI8 input pin. At other settings, INT5 is generated on the rising edge of TI8.

It is also possible to measure the width of low level external pulses. In this case, the pulse width is calculated during the interrupt processing for the second INT5 interrupt by multiplying the internal clock cycle by the difference between the value of C2 at the first INT5 interrupt and the value of C1 at the second INT5 interrupt. However, as the first C2 value has been overwritten by the time of the second INT5 interrupt, the C2 value must be saved during processing of the first INT5 interrupt.

[4] Time difference measurement

The time difference between two events can be measured using the 16-bit timer capture function.

To measure time difference, first set the 16-bit up-counter UC8 to operate as a free-running up-counter driven by an internal clock. Load the value of up-counter UC8 into capture register CAP1 on a rising edge detected on the TI8 pin input pulse. Interrupt INT5 is generated at this time.

Similarly, on a rising edge detected on the TI9 pin input pulse, load the up-counter UC8 value into capture register CAP2. Interrupt INT6 is generated at this time.

When both values have been loaded into the capture registers, calculate the time difference by multiplying the difference between CAP2 and CAP1 by the internal clock cycle.

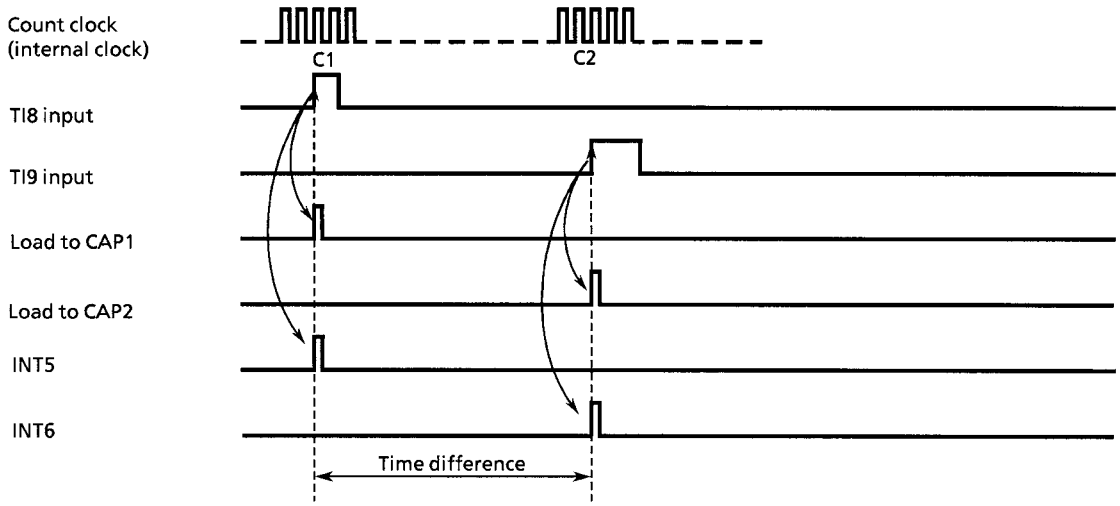


Figure 3.8.10 Time Difference Measurement

(5) Phase output (only available on timer 8)

Signals with a user-specified phase difference can be output using the 16-bit timer.

Select an internal clock as the clock source and set the 16-bit up-counter UC8 to free-running. Set the phase difference in 16-bit timer registers TREG8 and TREG9, set timer flip-flops TFF8 and TFF9 to invert when a match is detected for TREG8 and TREG9, and output the flip-flop values from TO8 and TO9.

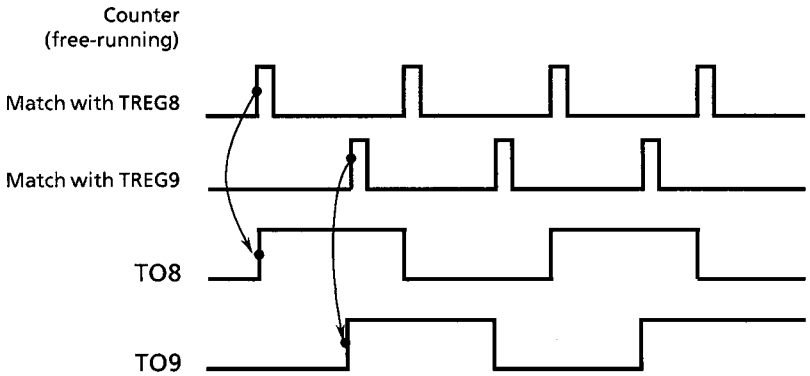


Figure 3.8.11 Phase Output

Table 3.8.1 lists the cycles (counter overflow times) that can be set for each clock source.

Table 3.8.1 16-Bit Up-Counter Overflow Times

	24 MHz
ϕT1	21.85 ms
ϕT4	87.38 ms
ϕT16	349.53 ms

3.9 Serial Channels

The TMP95CU54A has two internal serial input/output channels. The serial channels have the following four operating modes.

- I/O interface mode
  - Mode 0: Can be used to expand the I/O by sending and receiving I/O data and the associated synchronizing signal (SCLK).
- Universal asynchronous receiver transmitter (UART) mode
  - Mode 1: Send/receive data length: 7 bits
  - Mode 2: Send/receive data length: 8 bits
  - Mode 3: Send/receive data length: 9 bits

A parity bit can be added in modes 1 and 2. Mode 3 has a wake-up function that allows a master controller to activate slave controllers via a serial link (multi-controller system).

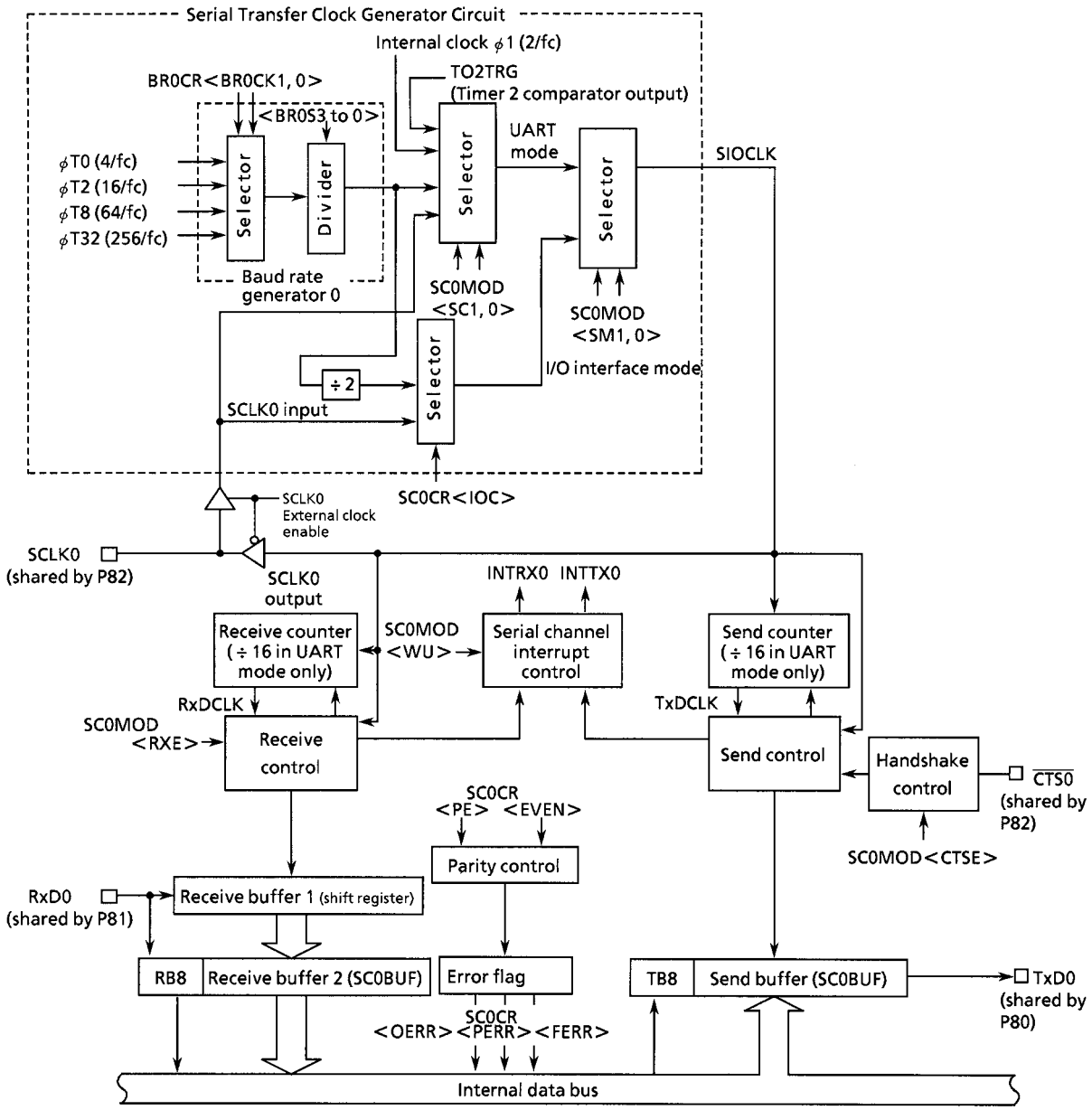


Figure 3.9.1 Block Diagram of Serial Channel 0

3.9.1 Serial Channel Registers

Each serial channel is controlled by three control registers (SC0CR, SC0MOD, and BR0CR in the case of channel 0). Data sent and received are stored in the serial send/receive buffer register in each channel (SC0BUF in the case of channel 0).

(1) Serial channel 0

Serial Channel 0 Mode Control Register

SC0MOD (004EH)

	7	6	5	4	3	2	1	0
bit Symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
Read/Write	R/W							
After reset	Undefined	0	0	0	0	0	0	0
Function	Bit 8 of send data	Handshake function control 0: $\overline{CTS0}$ disable 1: $\overline{CTS0}$ enable	Receive control 0: Receive disable 1: Receive enable	Wake-up function 0: Disable 1: Enable	Serial transfer mode selection 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial transfer clock selection (UART mode) 00: TO2 trigger 01: Baud rate generator 0 10: Internal clock ϕ1 11: SCLK0 pin input (external clock)	

Serial transfer clock selection (UART mode)

00	Timer 2 comparator output
01	Baud rate generator 0
10	Internal clock ϕ1 (2/fc)
11	SCLK0 pin input (external clock)

Serial transfer mode selection

00	I/O interface mode
01	7-bit UART mode
10	8-bit UART mode
11	9-bit UART mode

Wake-up function
(Other than 9-bit UART mode, don't care)

0	Disable
1	Enable

Receive control

0	Receive disable
1	Receive enable

Handshake function ($\overline{CTS0}$ pin) enable

0	Disable (send always enabled)
1	Enable

Store bit 8 of send data

8-bit UART mode (with parity)	Store send parity bit
9-bit UART mode	Store bit 8 of receive data

Figure 3.9.2 Serial Channel Related Register (1/6)

Serial Channel 0 Control Register

SC0CR (004DH)

	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	Undefined	0	0	0	0	0	0	0
Function	Bit 8 of receive data	Odd/even parity selection 0: Odd 1: Even	Parity bit addition control 0: No parity bit 1: Parity bit	1: Error Overrun error flag	Parity error flag	Framing error flag	SCLK0 pin input edge selection 0: Rising edge 1: Falling edge	Serial transfer clock selection (I/O interface mode) 0: Baud rate generator 0 1: SCLK0 pin input

Serial transfer clock selection (I/O interface mode)

0	Baud rate generator 0 (Note 1)
1	SCLK0 pin input (external clock)

Edge selection in SCLK0 pin input operations

0	Data send/receive at SCLK0 rising edge
1	Data send/receive at SCLK0 falling edge

Framing error flag
Parity error flag
Overrun error flag
} Cleared to 0 when read. (Note 2)

Parity bit addition control

0	No parity bit
1	Parity bit

Even/odd parity selection

0	Odd parity
1	Even parity

Receive data store bit 8

8-bit UART mode (with parity)	Store receive parity bit
9-bit UART mode	Store bit 8 of receive data

Note 1: To use the baud rate generator, set T16RUN<PRRUN> to 1 and run the prescaler.
Note 2: As the error flags are all cleared to 0 after reading, don't test only one bit with a bit test instruction.

Figure 3.9.2 Serial Channel Related Register (2/6)

Baud Rate Generator 0 Control Register

BR0CR (004FH)

	7	6	5	4	3	2	1	0
bit Symbol	–		BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
Read/Write	R/W		R/W					
After reset	0		0	0	0	0	0	0
Function	Note: Always fixed to 0.		Baud rate generator 0 input clock selection 00: ϕT0 (4/fc) 01: ϕT2 (16/fc) 10: ϕT8 (64/fc) 11: ϕT32 (256/fc)		Divisor setting for baud rate generator 0			

Baud rate generator 0 divisor setting

0000	Divide by 16
0001 to 1111	Divide by 1 (no division) (Note 1) to Divide by 15

Baud rate generator 0 input clock selection

00	Internal clock ϕT0 (4/fc)
01	Internal clock ϕT2 (16/fc)
10	Internal clock ϕT8 (64/fc)
11	Internal clock ϕT32 (256/fc)

Note 1: The baud rate generator can be divided by 1 in UART mode only. Do not use this setting in I/O interface mode.

Note 2: Don't read from or write to BR0CR register during sending or receiving.

Serial Channel 0 Buffer Register

SC0BUF (004CH)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol	RB07	RB06	RB05	RB04	RB03	RB02	RB01	RB00
	TB07	TB06	TB05	TB04	TB03	TB02	TB01	TB00
Read/Write	R (receive) / W (send)							
After reset	Undefined							

Figure 3.9.2 Serial Channel Related Registers (3/6)

(2) Serial channel 1

Serial Channel 1 Mode Control Register

SC1MOD (0052H)

	7	6	5	4	3	2	1	0
bit Symbol	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
Read/Write	R/W							
After reset	Undefined	0	0	0	0	0	0	0
Function	Bit 8 of send data	Handshake function control 0: $\overline{CTS1}$ disable 1: $\overline{CTS1}$ enable	Receive control 0: Receive disable 1: Receive enable	Wake-up function 0: Disable 1: Enable	Serial transfer mode selection 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		Serial transfer clock selection (UART mode) 00: TO2 trigger 01: Baud rate generator 1 10: Internal clock ϕ1 11: SCLK1 pin input (external clock)	

Serial transfer clock selection (UART mode)

00	Timer 2 comparator output
01	Baud rate generator 1 output
10	Internal clock ϕ1
11	SCLK1 pin input (external clock)

Serial transfer mode selection

00	I/O interface mode
01	7-bit UART mode
10	8-bit UART mode
11	9-bit UART mode

Wake-up function
(Other than 9-bit UART mode, don't care)

0	Disable
1	Enable

Receive control

0	Receive disable
1	Receive enable

Handshake function ($\overline{CTS1}$ pin) enable

0	Disable (send always enabled)
1	Enable

Store bit 8 of send data

8-bit UART mode (with parity)	Store send parity bit
9-bit UART mode	Store bit 8 of receive data

Figure 3.9.2 Serial Channel Related Register (4/6)

Serial Channel 1 Control Register

SC1CR (0051H)

	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	Undefined	0	0	0	0	0	0	0
Function	Bit 8 of receive data	Even/odd parity selection 0: Odd 1: Even	Parity bit addition control 0: No parity bit 1: Parity bit	1: Error Overrun error flag	Parity error flag	Framing error flag	SCLK1 pin input edge selection 0: Rising edge 1: Falling edge	Serial transfer clock selection (I/O interface mode) 0: Baud rate generator 1 1: SCLK1 pin input

Serial transfer clock selection (I/O interface mode)

0	Baud rate generator 1 (Note 1)
1	SCLK1 pin input (external clock)

Edge selection in SCLK1 pin input operations

0	Data send/receive at SCLK1 rising edge
1	Data send/receive at SCLK1 falling edge

Framing error flag
Parity error flag
Overrun error flag
} Cleared to 0 when read. (Note 2)

Parity bit addition control

0	No parity bit
1	Parity bit

Even/odd parity selection

0	Odd parity
1	Even parity

Store bit 8 of receive data

8-bit UART mode (with parity)	Store receive parity bit
9-bit UART mode	Store bit 8 of receive data

Note 1: To use the baud rate generator, set T16RUN<PRRUN> to 1 and run the prescaler.
Note 2: As the error flags are all cleared to 0 after reading, don't test only one bit with a bit test instruction.

Figure 3.9.2 Serial Channel Related Register (5/6)

Baud Rate Generator 1 Control Register

BR1CR (0053H)

	7	6	5	4	3	2	1	0
bit Symbol	–		BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
Read/Write	R/W		R/W					
After reset	0		0	0	0	0	0	0
Function	Note: Always fixed to 0.		Baud rate generator 1 input clock selection 00: ϕT0 (4/fc) 01: ϕT2 (16/fc) 10: ϕT8 (64/fc) 11: ϕT32 (256/fc)		Divisor setting for baud rate generator 1			

Baud rate generator 1 divisor setting

0000	Divide by 16
0001 to 1111	Divide by 1 (no division) (Note 1) to Divide by 15

Baud rate generator 1 input clock selection

00	Internal clock ϕT0 (4/fc)
01	Internal clock ϕT2 (16/fc)
10	Internal clock ϕT8 (64/fc)
11	Internal clock ϕT32 (256/fc)

Note 1: The baud rate generator can be divided by 1 in UART mode only. Do not use this setting in I/O interface mode.

Note 2: Don't read from or write to BR1CR register during sending or receiving.

Serial Channel 1 Buffer Register

SC1BUF (0050H)

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol	RB17	RB16	RB15	RB14	RB13	RB12	RB11	RB10
	TB17	TB16	TB15	TB14	TB13	TB12	TB11	TB10
Read/Write	R (receive) / W (send)							
After reset	Undefined							

Figure 3.9.2 Serial Channel Related Registers (6/6)

3.9.2 Block Structure

As serial channels 0 and 1 operate identically, the following uses channel 0 as an example.

(1) Serial transfer clock generator circuit

The serial transfer clock generator circuit generates SIOCLK (internal signal), which is the send/receive basic clock. To generate SIOCLK, select the clock source required for the generation.

[1] I/O interface mode

As the clock source, select either baud rate generator 0, or SCLK0 from an external source. Set the clock source in bit 0 (<IOC>) of serial channel 0 control register SC0CR.

When baud rate generator 0 is selected (<IOC> = 0), this circuit generates SIOCLK by dividing the output of the baud rate generator by 2.

When external SCLK0 is selected (<IOC> = 1), SIOCLK is set to the same value as the external source.

[2] UART mode

In addition to the clock sources in I/O interface mode, the comparator output of timer 2 and the internal clock φ1 (2/fc) can also be selected as clock sources.

Bits 1 and 0 of serial channel 0 mode control register SC0MOD<SC1, 0> select the clock source. SIOCLK is set to the same value as the selected clock source.

(2) Receive counter

The receive counter is a 4-bit binary counter used in UART mode.

The receive counter uses SIOCLK as the count clock to generate receive sampling clock RxDCLK(internal signal).

(3) Receive control

[1] I/O Interface mode

In I/O interface mode, the receive data input to the RxD0 pin are sampled synchronously with transfer clock SCLK0.

Setting serial channel 0 control register SC0CR<IOC> to 0 samples the received data on the rising edge of SCLK0. Setting SC0CR<IOC> to 1 samples the data on the rising or the falling edge of SCLK0 as determined by the setting of SC0CR<SCLKS>.

[2] UART mode

The receive data are sampled bit by bit using RxDCLK, which is generated by the receive counter. Each bit of data is sampled three times, using majority rule. If two or more instances of the same value are detected among three samples, the circuit recognizes the data as receive data. If the sampled data are 1, 0, 1, for example, the data are evaluated as 1; if 0, 0, 1, the data are evaluated as 0.

(4) Receive buffer

The receive buffer has a double-buffer configuration to prevent overrun error. Receive buffer 1 stores the data received bit by bit.

When receive buffer 1 contains seven or eight bits of data, the data are transferred to receive buffer 2 (SC0BUF), generating interrupt INTRX0.

Reading the data in receive buffer 2 clears the interrupt request flag INTRX0<IRX0C>.

Even before the CPU reads the data in receive buffer 2, the next data can be received and stored in receive buffer 1.

However, receive buffer 2 must be read before all bits of the next data frame are received by buffer 1. If not, an overrun error occurs and the contents of receive buffer 1 are lost, although the contents of receive buffer 2 and the serial channel 0 control register SC0CR<RB8> are preserved.

In 8-bit UART mode (mode 2) with parity added, the parity bit is stored in SC0CR<RB8>. In 9-bit UART mode (mode 3), the MSB is stored in SC0CR<RB8>.

(5) Send counter

The send counter is a 4-bit binary counter used in UART mode.

The send counter uses SIOCLK as its count clock, generating send clock TxDCLK (internal signals).

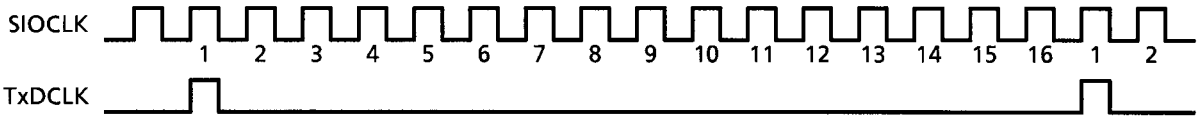


Figure 3.9.3 Send Clock Generation

(6) Send control

[1] I/O interface mode

In I/O interface mode, the TMP95CU54A outputs send data from the TxD0 pin synchronously with transfer clock SCLK0.

Setting serial channel 0 control register SC0CR<IOC> to 0 outputs send data on the rising edge of transfer clock SCLK0.

Setting SC0CR<IOC> to 1 outputs the send data on the rising or falling edge of SCLK0 as determined by the setting of SC0CR<SCLKS>.

[2] UART mode

In UART mode, the send data are output synchronously with the rising edge of the TxDCLK send clock generated by the send counter.

(7) Send buffer

Send buffer (SC0BUF) outputs the send data written by the CPU, beginning with the least significant bit.

When all bits are output, the empty send buffer generates interrupt request INTTX0.

(8) Parity control

Parity bit addition can only be set in 7-bit UART mode (mode 1) and 8-bit UART mode (mode 2).

When serial channel 0 control register SC0CR<PE> is set to 1, data can be sent with a parity bit added. SC0CR<EVEN> selects even parity or odd parity.

A send operation automatically generates the parity bit determined by the send data. In mode 1, SC0BUF<TB7> stores the parity bit; in mode 2, serial channel 0 mode control register SC0MOD<TB8> stores the parity bit.

Set both <PE> and <EVEN> before writing the send data in SC0BUF.

When receiving, parity is calculated from the received data and compared with the received parity bit. If the parities differ, a parity error occurs and parity error flag SC0CR<PERR> is set to 1.

(9) Error flags

To improve the reliability of data reception, serial channel 0 control register SC0CR contains the following three error flags.

[1] Overrun error <OERR>

When all bits of the next data frame have been received in receive buffer 1 while valid data are stored in receive buffer 2 (SC0BUF), an overrun error occurs.

At an overrun error, the data received in buffer 1 are lost.

[2] Parity error <PERR>

The parity bit determined by the data stored in receive buffer 2 (SC0BUF) is compared with the received parity bit. If the parities differ, a parity error occurs.

[3] Framing error <FERR>

The stop bit of data received is sampled three times. If the majority of samples are 0, a framing error occurs.

If an error occurs, these error flags are set to 1. Reading the SC0CR register clears the error flags to 0. If an error occurs, fix by software.

(10) Handshake function control (only supported in UART mode)

The serial channels use the $\overline{CTS0}$ input pin to send data in one-frame units, thus preventing an overrun error. The serial channel 0 mode control register SC0MOD<CTSE> enables or disables the handshake function.

In send operations, sending starts when a low level signal is input to the $\overline{CTS0}$ pin.

When $\overline{CTS0}$ goes high, data sending is halted when sending of the current data completes and the pin is set to wait state. Sending is not restarted until $\overline{CTS0}$ goes low again.

Although an $\overline{RTS0}$ pin is not provided, any port can be assigned to the $\overline{RTS0}$ function. When the receiving side has completed reception, the receiving interrupt processing routine outputs a high-level signal from the port assigned to the $\overline{RTS0}$ function. A handshake function can be easily configured by connecting the sending side $\overline{CTS0}$ pin and the receiving side $\overline{RTS0}$ pin.

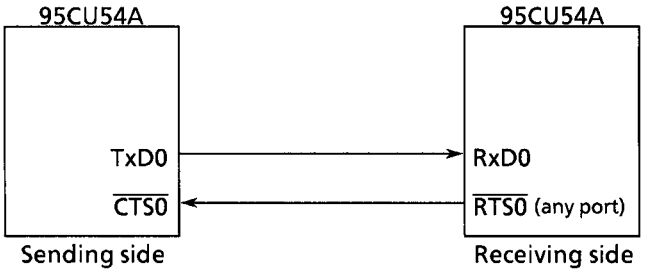
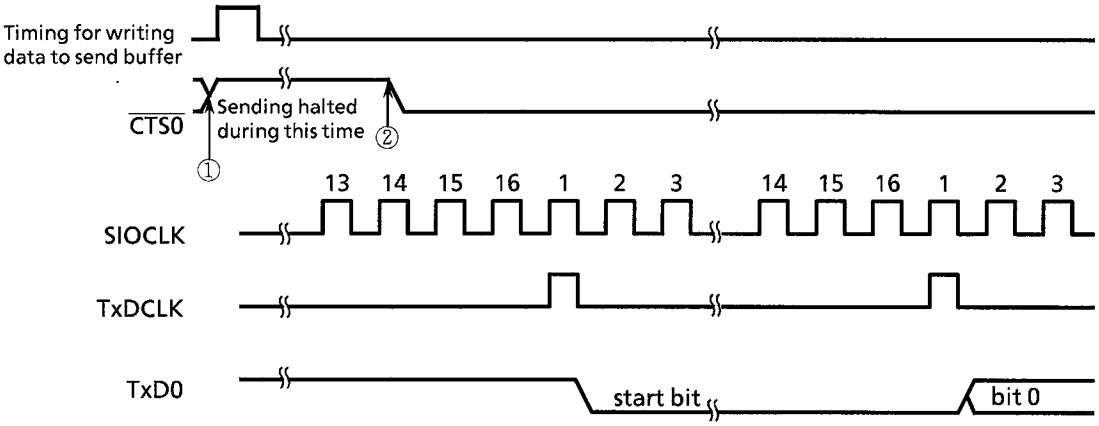


Figure 3.9.4 Handshake Function



[1] When the $\overline{CTS0}$ signal rises during sending, sending of the current data frame is completed, and sending of the next data frame halts.

[2] Sending begins at the first TxDCLK clock falling edge after the $\overline{CTS0}$ signal drops.

Figure 3.9.5 $\overline{CTS0}$ (Clear to Send) Signal Timing

3.9.3 Description of Operation

As serial channels 0 and 1 operate identically, the following uses channel 0 as an example.

(1) Setting send/receive clock transfer rate

[1] Transfer rate setting with baud rate generator selected

The baud rate generator is a circuit used to generate a clock source for the send/receive clock that controls the serial channel transfer rate.

The input clock for generating the clock source can be selected from among φT0 (4/fc), φT2 (16/fc), φT8 (64/fc), or φT32 (256/fc) from the 9-bit prescaler (see 3.7.2 (1), Prescaler). The 8-bit and 16-bit timers share the prescaler. Bits 5, 4 of baud rate generator control register BR0CR<BR0CK1:0> select the input clock.

The selected input clock is divided by the 4-bit divider performing 1 to 16 divisions. Bits 3 to 0 of BR0CR<BR0S3:0> set the divider. The divided clock is the output clock for the baud rate generator.

The following are the transfer rate calculation formulas when the baud rate generator is selected:

- I/O interface mode

$$\text{Transfer rate [bps]} = \frac{\text{Baud rate generator input clock [Hz]}}{\text{Baud rate generator divisor (2 to 16)}} \div 2$$

Note: In I/O interface mode, do not set divisor to 1.

- UART mode

$$\text{Transfer rate [bps]} = \frac{\text{Baud rate generator input clock [Hz]}}{\text{Baud rate generator divisor (1 to 16)}} \div 16$$

The relationship between the input clock and the source clock (fc) is:

φT0 = 4/fc

φT2 = 16/fc

φT8 = 64/fc

φT32 = 256/fc

Accordingly, with the source clock set to 12.288MHz, when φT2 (16/fc) is selected as the input clock and the divisor is 5, the transfer rate in UART mode is:

$$\text{Transfer rate} = \frac{fc/16}{5} \div 16 = 12.288 \times 10^6 \div 16 \div 5 \div 16 = 9600\text{[bps]}$$

Table 3.9.1 shows examples of transfer rate settings in UART mode.

[2] Transfer rate settings with the timer 2 comparator output selected (UART mode only)

The following are the transfer rate calculation formulas when the timer 2 comparator output is selected:

$$\text{Transfer rate [bps]} = \frac{\text{Timer 2 input clock [Hz]}}{\text{TREG2 (1 to 256)}} \div 16$$

The relationship between the timer 2 input clock and the source clock (fc) is:

φT1 = 8/fc

φT4 = 32/fc

φT16 = 128/fc

Accordingly, with the source clock set to 25MHz, when the timer 2 input clock is set to φT1 and TREG2 is set to 1, the transfer rate is:

$$\text{Transfer rate} = \frac{\text{fc}/8}{\text{TREG2}} \div 16 = 24 \times 10^6 \div 8 \div 1 \div 16 = 187500 \text{ [bps]}$$

Table 3.9.2 shows examples of the transfer rate settings.

[3] Transfer rate settings with external SCLK input selected

The following are the transfer rate calculation formulas when the external SCLK input is selected:

- I/O interface mode

  Transfer rate [bps] = external SCLK input [Hz] ÷ 2

- UART mode

  Transfer rate [bps] = external SCLK input [Hz] ÷ 16

Table 3.9.1 UART Mode Transfer Rate Setting Example (1) (Using Baud Rate Generator)

Unit: Kbps

fc [MHz]	Divisor \ Input clock	φT0 (4/fc)	φT2 (16/fc)	φT8 (64/fc)	φT32 (256/fc)
9.830400	1	153.600	38.400	9.600	2.400
	2	76.800	19.200	4.800	1.200
	4	38.400	9.600	2.400	0.600
	8	19.200	4.800	1.200	0.300
	16	9.600	2.400	0.600	0.150
12.288000	5	38.400	9.600	2.400	0.600
	10	19.200	4.800	1.200	0.300
14.745600	1	230.400	57.600	14.400	3.600
	3	76.800	19.200	4.800	1.200
	6	38.400	9.600	2.400	0.600
	12	19.200	4.800	1.200	0.300
17.2032	7	38.400	9.600	2.400	0.600
	14	19.200	4.800	1.200	0.300
19.6608	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
	16	19.200	4.800	1.200	0.300
22.1184	9	38.400	9.600	2.400	0.600

Note: In I/O interface mode, the transfer rates are 8 times the values in this table.
In I/O interface mode, do not set the baud rate generator divisor to 1.

Table 3.9.2 UART Mode Transfer Rate Setting Example (2) (Using Timer 2 Input Clock φT1)

Unit: kbps

TREG2 \ fc	12.288MHz	12MHz	9.8304MHz	8MHz	6.144MHz
1H	96		76.8	62.5	48
2H	48		38.4	31.25	24
3H	32	31.25			16
4H	24		19.2		12
5H	19.2				9.6
8H	12		9.6		6
AH	9.6				4.8
10H	6		4.8		3
14H	4.8				2.4

(2) Data format

Figure 3.9.6 shows the data format for each mode.

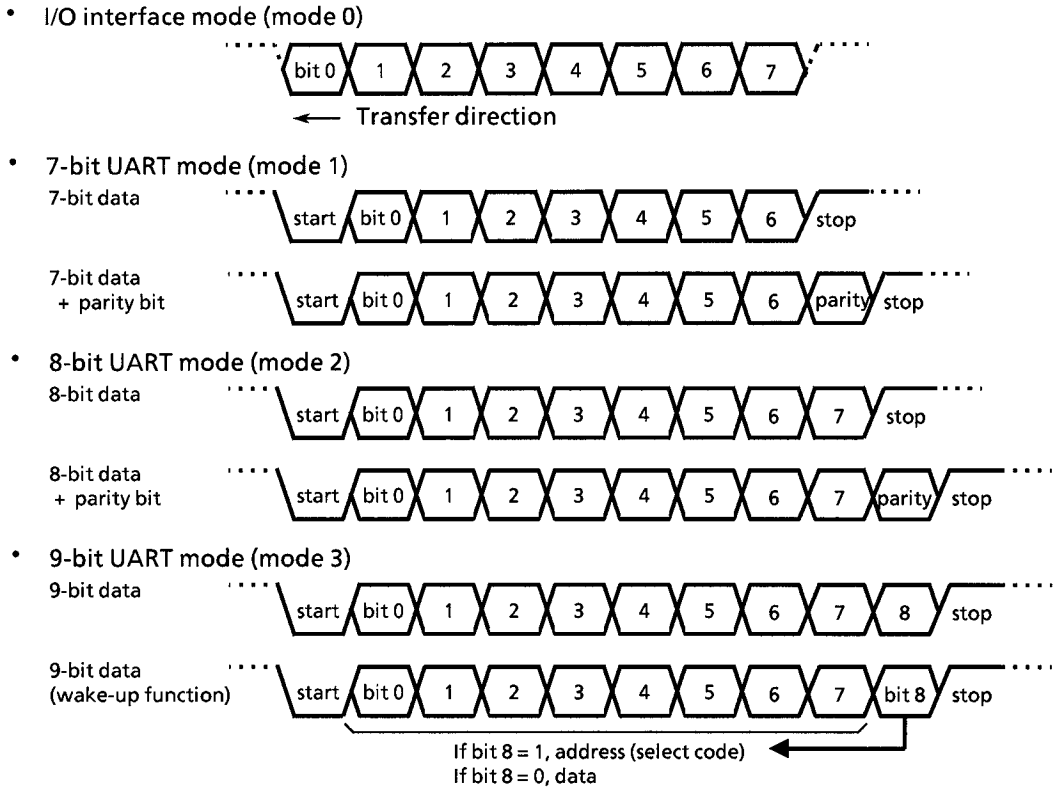


Figure 3.9.6 Data Formats

(3) I/O interface mode (Mode 0)

In this mode, data transfer to an external device is synchronous with the transfer clock.

This mode is used to increase the number of I/O pins for sending or receiving data to an external shift register or other external destinations.

This mode consists of SCLK0 output mode, which outputs a synchronous clock (SCLK0), and SCLK0 input mode, which inputs a synchronous clock (SCLK0) from an external source.

Figures 3.9.7 and 3.9.8 show connection examples of SCLK0 output and input modes.

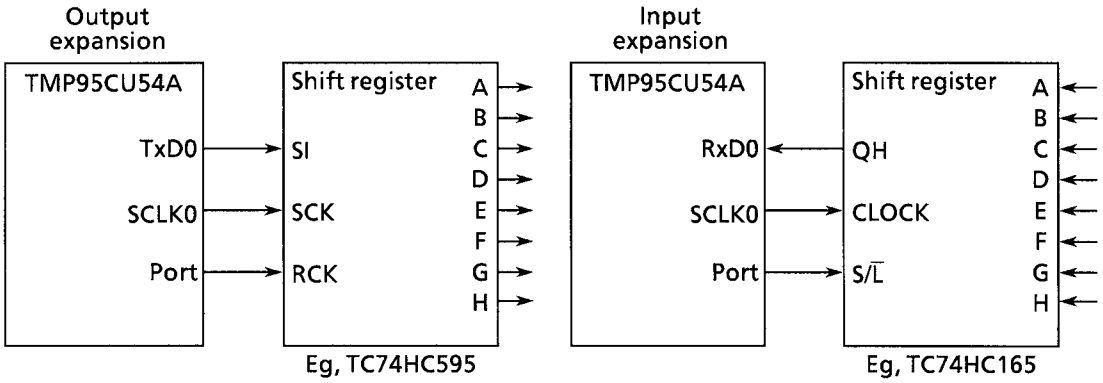


Figure 3.9.7 Example of SCLK0 Output Mode Connection

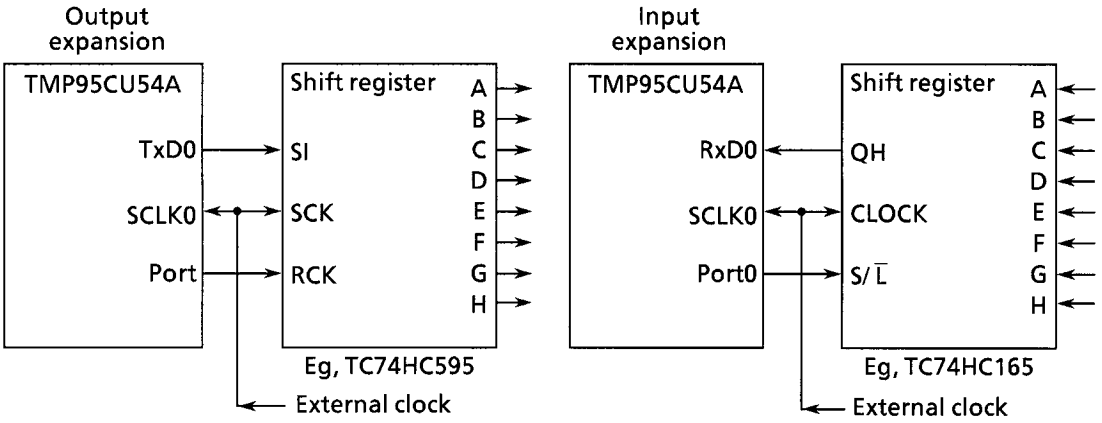


Figure 3.9.8 Example of SCLK0 Input Mode Connection

[1] Sending

In SCLK0 output mode, each time the CPU writes data in the send buffer, eight data bits are output from the TxD0 pin, and a transfer clock signal is output from the SCLK0 pin. When all data have been sent, INTES0<ITX0C> is set, triggering an INTTX0 interrupt request.

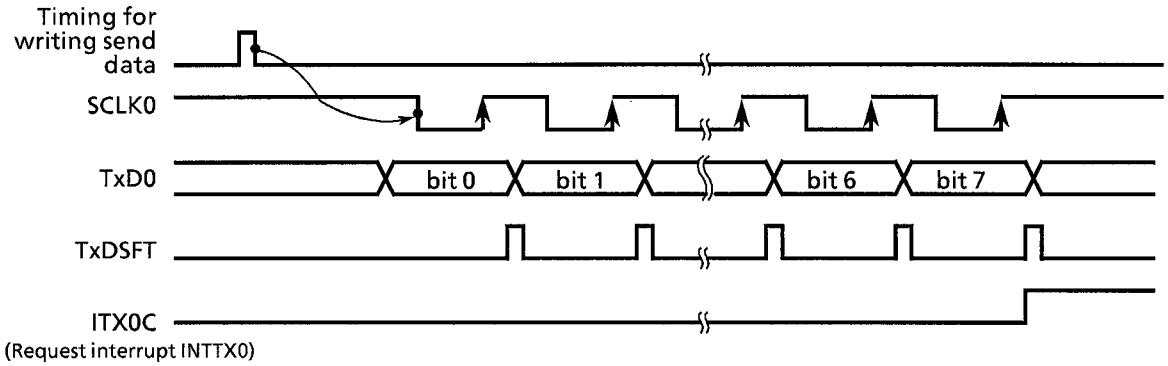


Figure 3.9.9 Sending in I/O Interface Mode (SCLK0 Output Mode)

In SCLK0 input mode, pin TxD0 outputs eight transfer data bits when SCLK0 input is supplied and data are written to the send buffer by the CPU.

When all data have been sent, INTES0<ITX0C> is set, triggering an INTTX0 interrupt request.

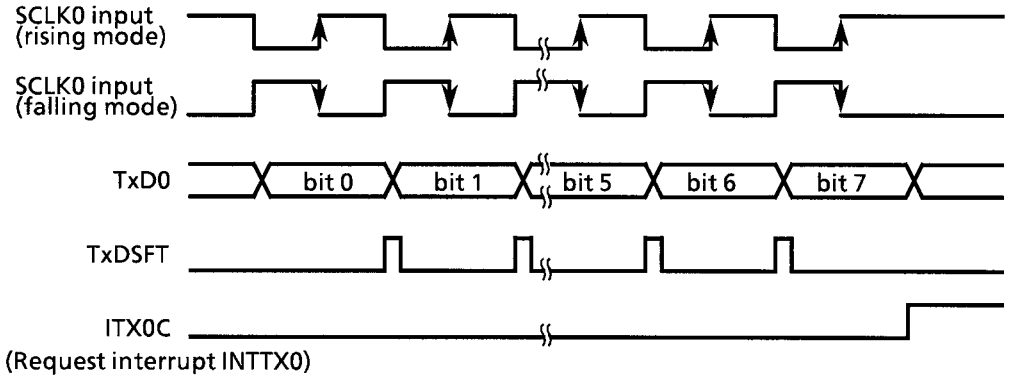


Figure 3.9.10 Sending in I/O Interface Mode (SCLK0 Input Mode)

[2] Receiving

In SCLK0 output mode, whenever the receive interrupt flag INTES0<IRX0C> is cleared by the CPU reading the received data, a synchronous clock is output from the SCLK0 pin and the next data frame is shifted to receive buffer 1. When an 8-bit data frame is received, it is transferred to receive buffer 2 (SC0BUF), and INTES0<IRX0C> is set again, triggering an INTRX0 interrupt request.

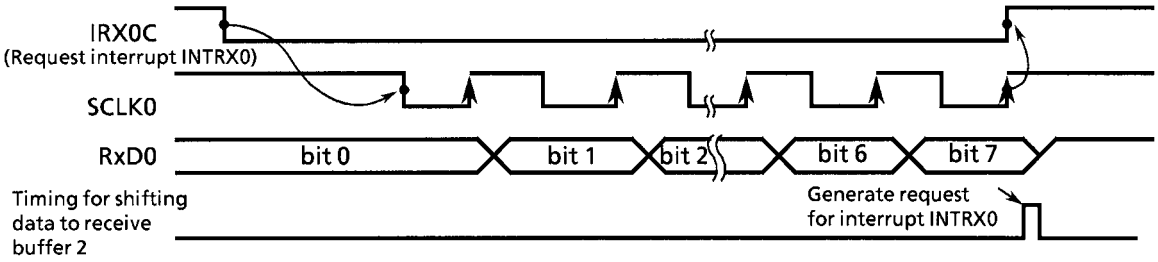


Figure 3.9.11 Receiving in I/O Interface Mode (SCLK0 Output Mode)

In SCLK0 input mode, if SCLK0 input is supplied when received data are read by the CPU, thus clearing receive interrupt flag INTES0<IRX0C>, the next data frame is shifted into receive buffer 1.

When an 8-bit data frame is received, it is shifted to receive buffer 2 (SC0BUF) and INTES0<IRX0C> is set again, triggering an INTRX0 interrupt request.

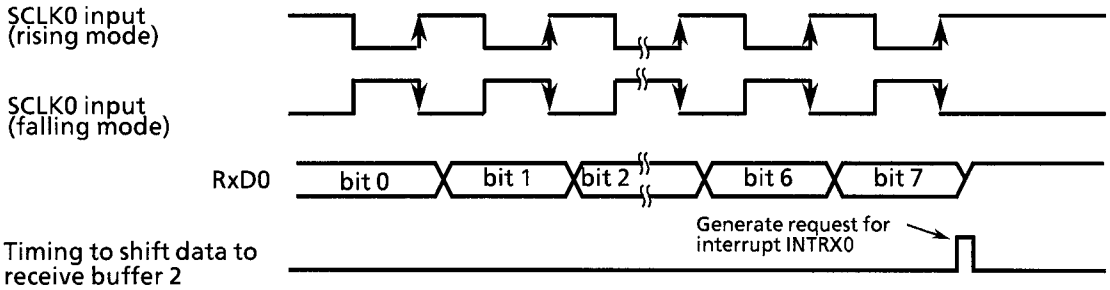


Figure 3.9.12 Receiving in I/O Interface Mode (SCLK0 Input Mode)

Note: To receive data, first enable reception (set SC0MOD<RXE> to 1) for either SCLK0 input mode or output mode.

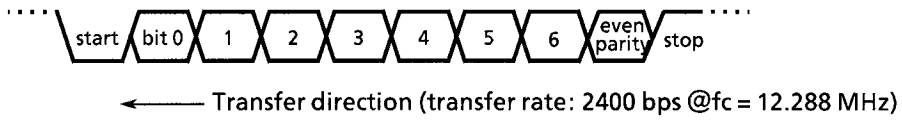
(4) 7-bit UART mode (Mode 1)

Setting serial channel 0 mode control register SC0MOD<SM1:0> to 01 specifies 7-bit UART mode.

A parity bit may be added in this mode. Enable or disable the addition of a parity bit by serial channel 0 control register SC0CR<PE>.

With <PE> set to 1 (parity bit added), SC0CR<EVEN> selects even or odd parity.

Setting example: send 7-bit data with an even parity bit added:



		7	6	5	4	3	2	1	0	
P8CR	←	-	-	-	-	-	-	-	1	Select P80 as TxD0 pin.
P8FC	←	X	-	-	X	-	-	X	1	
SC0MOD	←	X	0	-	X	0	1	0	1	Set 7-bit UART mode.
SC0CR	←	X	1	1	X	X	X	0	0	Add even parity.
BR0CR	←	0	X	1	0	0	1	0	1	Set transfer rate to 2400bps.
T16RUN	←	1	X	-	-	-	-	-	-	Start prescaler for baud rate generator.
INTES0	←	1	1	0	0	-	-	-	-	Enable interrupt INTTX0 and set interrupt level to 4.
SC0BUF	←	*	*	*	*	*	*	*	*	Set send data.

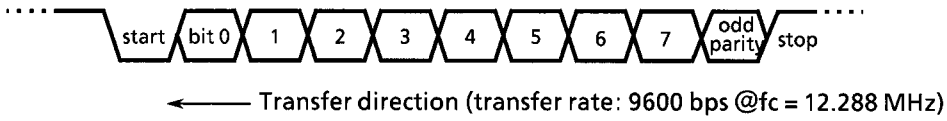
Note: X : Don't care − : No change

(5) 8-bit UART mode (Mode 2)

Setting serial channel 0 mode control register SC0MOD<SM1:0> to 10 selects 8-bit UART mode.

A parity bit may be added in this mode. Enable or disable the addition of a parity bit by serial channel 0 control register SC0CR<PE>. With <PE> set to 1 (parity bit added), SC0CR<EVEN> selects even or odd parity.

Setting example: send 8-bit data with an odd parity bit added:



Main routine settings:

```
          7 6 5 4 3 2 1 0
P8CR    ← - - - - - - 0 -     Select P81 (RxD0) as input pin.
SC0MOD  ← - 0 1 X 1 0 0 1     Set 8-bit UART mode and enable reception.
SC0CR   ← X 0 1 X X X 0 0     Add odd parity.
BR0CR   ← 0 X 0 1 0 1 0 1     Set transfer rate to 9600bps.
T16RUN  ← 1 X - - - - - -     Start the prescaler for baud rate generator.
INTES0  ← - - - - 1 1 0 0     Enable interrupt INTRX0 and set interrupt level 4.
```

Note: X : Don't care − : No change

Interrupt routine processing example:

Check for errors with SC0CR error flags (<OERR>, <PERR>, <FERR>). If there are no errors, read the data received.

(6) 9-bit UART mode (Mode 3)

Setting the serial channel 0 mode control register SC0MOD<SM1:0> to 11 selects 9-bit UART mode.

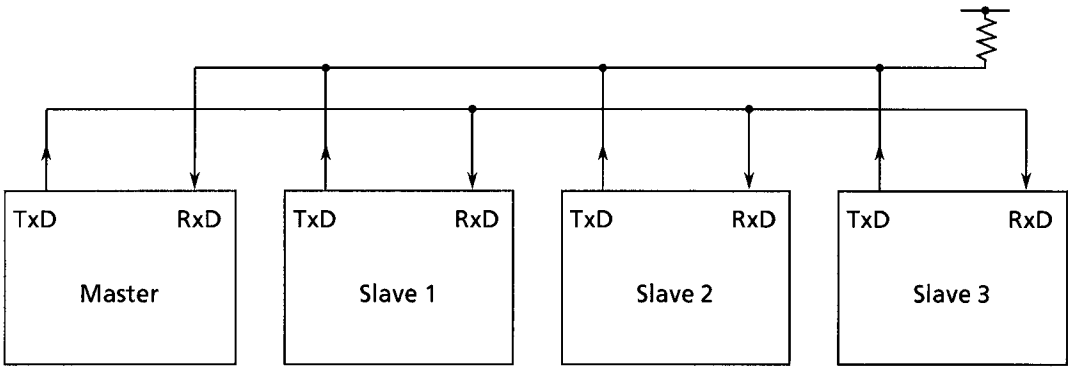
A parity bit cannot be added in this mode.

When sending, the most significant bit (bit 9) is written to SC0MOD<TB8>.

When receiving, the most significant bit is saved in serial channel control register SC0CR<RB8>. When the buffer is written to or read from, the most significant bit is always read or written first.

Wake-Up function

In 9-bit UART mode, select the slave controller wake-up function by setting SC0MOD<WU> to 1. When SC0CR<RB8> = 1, received data are interpreted as select code, and an INTRX0 interrupt request occurs.



Note: The TxD pin of the slave controller must always be set to open-drain output mode using the ODE register.

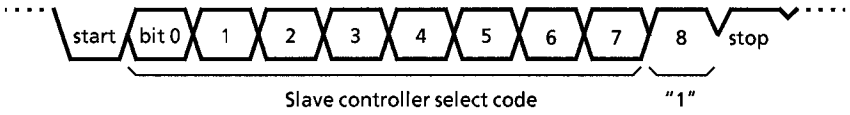
Figure 3.9.13 Serial Link using Wake-Up function

Protocol

[1] Set the master controller and all slave controllers to 9-bit UART mode.

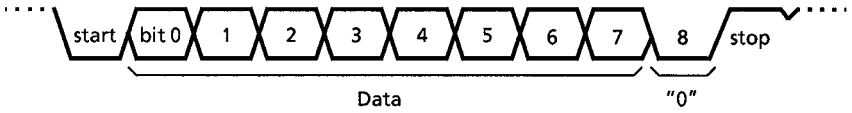
[2] Set the serial channel 0 mode control register SC0MOD<WU> of each slave controller to 1 to enable data reception.

[3] The master controller sends one frame with the most significant bit (bit 8) SC0MOD<TB8> set to 1. This frame contains the 8-bit select code of a slave controller.



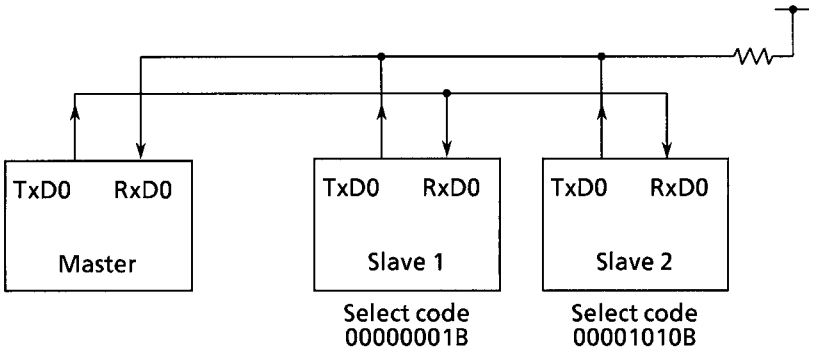
[4] The slave controllers receive the above data frame. The slave controller whose select code matches the select code in the data frame received clears its SC0MOD<WU> bit to 0.

[5] The master controller sends data frames with their most significant bit (bit 8) SC0MOD<TB8> set to 0 to the specified slave controller (the controller whose SC0MOD<WU> bit is cleared to 0).



[6] The slave controllers whose SC0MOD<WU> bit is 1 ignore the received data as interrupt INTRX0 is not generated when the most significant bit (bit 8) SC0CR<RB8> remains cleared to 0 (when data are sent).
The slave controller whose SC0MOD<WU> bit is cleared to 0 can inform the master controller of the termination of a send it received by sending data to the master controller.

Setting example: When linking two slave controllers serially with the master controller using internal clock φ1 as the transfer clock.



As serial channels 0 and 1 operate identically in this mode, the following describes channel 0 only.

- Setting the master controller

  Main routine:

```
P8CR   ← - - - - - - - 0 1   } Select P80 as TxD0 pin, and P81 as RxD0 pin.
P8FC   ← X - - X - - X 1
INTES0 ← 1 1 0 0 1 1 0 1     Enable interrupt INTTX0 and set interrupt level to 4.
                             Enable interrupt INTRX0 and set interrupt level to 5.
SC0MOD ← 1 0 1 0 1 1 1 0     Set φ1 as transfer clock and set 9-bit UART mode.
SC0BUF ← 0 0 0 0 0 0 0 1     Set select code for slave controller 1.
```

INTTX0 interrupt routine:

```
SC0MOD ← 0 - - - - - - -     Set SC0MOD<TB8> to 0.
SC0BUF ← * * * * * * * *     Set send data.
```

Note: X : Don't care − : No change

- Setting slave controller 2

  Main routine:

```
P8CR   ← - - - - - - - 0 1   } Select P80 as TxD0 pin (open-drain output), and P81 as RxD0
P8FC   ← X - - X - - X 1     } pin.
ODE    ← X X X - 1 - - -     }
INTES0 ← 1 1 0 1 1 1 1 0     Enable interrupts INTTX0 and INTRX0.
SC0MOD ← 0 0 1 1 1 1 1 0     Set 9-bit UART mode using transfer clock φ1 (2/fc), and enable
                             wake-up mode (set <WU> to 1).
```

INTRX0 interrupt routine:

Compare SC0BUF and select code (00001010B). If these match, clear SC0MOD<WU> to 0.

Note: X : Don't care − : No change

(7) Signal generation timing

[1] In I/O Interface mode

Timing for send interrupt generation	SCLK0 output mode	Immediately after rise of last SCLK0 signal (See Figure 3.9.9)
	SCLK0 input mode	Immediately after rise (rising mode) or fall (falling mode) of last SCLK0 signal (See Figure 3.9.10)
Timing for receive interrupt generation	SCLK0 output mode	Immediately after final SCLK0 (When received data are transferred to receive buffer 2 (SC0BUF)) (See Figure 3.9.11)
	SCLK0 input mode	Immediately after final SCLK0 (When received data are transferred to receive buffer 2 (SC0BUF)) (See Figure 3.9.12)

[2] In UART mode

Receive

Mode	9-Bit	8-Bit + Parity	8-Bit, 7-Bit + Parity, 7-Bit
Timing for interrupt generation	Around center of bit 8	Around center of parity bit	Around center of stop bit
Timing for framing error generation	Around center of stop bit	Around center of stop bit	Around center of stop bit
Timing for parity error generation	————	Around center of parity bit	←
Timing for overrun error generation	Around center of bit 8	Around center of parity bit	Around center of stop bit

Send

Mode	9-Bit	8-Bit + Parity	8-Bit, 7-Bit + Parity, 7-Bit
Timing for interrupt generation	Immediately before stop bit sent	←	←

3.10 Analog/Digital Converter

The TMP95CU54A incorporates a high-speed, high-precision 10-bit successive approximation type analog/digital converter (AD converter) with 8-channel analog input.

Figure 3.10.1 is a block diagram of the AD converter. The 8-channel analog input pins (AN0 to AN7) are shared by input-only port A and can thus be used as an input port.

Note: When the power is reduced by setting IDLE2, IDLE1, or STOP mode, with some timings, the system may enter standby mode even though the internal comparator is still enabled. Therefore, be sure to check that AD converter operations are halted before executing a HALT instruction.

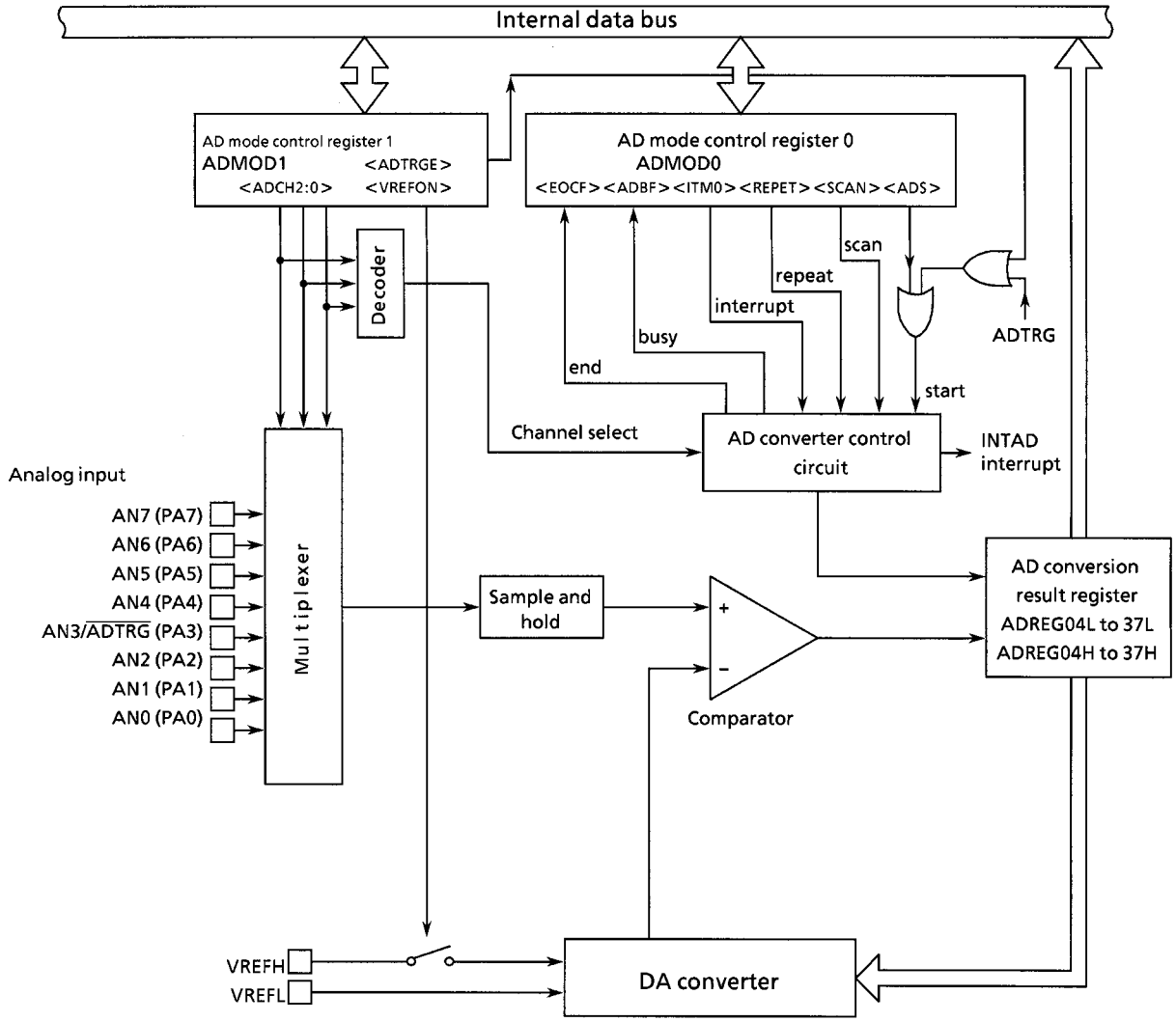


Figure 3.10.1 AD Converter Block Diagram

3.10.1 Analog/Digital Converter Registers

The AD converter is controlled by two AD mode control registers: ADMOD0 and ADMOD1. Eight AD conversion data upper and lower registers (ADREG04H/L, ADREG15H/L, ADREG26H/L, and ADREG37H/L) store the AD conversion results.

Figures 3.10.2 (1-4) show registers related to the AD converter.

AD Mode Control Register 0

ADMOD0 (005EH)	7	6	5	4	3	2	1	0
bit Symbol	EOCF	ADBF	–	–	ITM0	REPET	SCAN	ADS
Read/Write	R		R/W					
After reset	0	0	0	0	0	0	0	0
Function	AD conversion end flag 0:Conversion in progress 1:Conversion complete	AD conversion busy flag 0:Conversion stopped 1:Conversion in progress	Note: Always fixed to 0.	Note: Always fixed to 0.	Interrupt specification in conversion channel fixed repeat mode 0:Every conversion 1:Every fourth conversion	Repeat mode specification 0:Single conversion mode 1:Repeat conversion mode	Scan mode specification 0:Conversion channel fixed mode 1:Conversion channel scan mode	AD conversion start 0:Don't care 1:Start conversion Always read as 0.

AD conversion start

0	Don't care
1	Start AD conversion

Note: Always read as 0.

AD scan mode setting

0	AD conversion channel fixed mode
1	AD conversion channel scan mode

AD repeat mode setting

0	AD single conversion mode
1	AD repeat conversion mode

Specify AD conversion interrupt for channel fixed repeat conversion mode

	Channel fixed repeat conversion mode <SCAN> = 0, <REPET> = 1
0	Generate interrupt every conversion
1	Generate interrupt every fourth conversion

AD conversion busy flag

0	AD Conversion stopped
1	AD conversion in progress

AD conversion end flag

0	Before or during AD conversion
1	AD conversion complete

Figure 3.10.2 Register for AD Converter (1/4)

AD Mode Control Register 1

ADMOD1 (005FH)

	7	6	5	4	3	2	1	0
bit Symbol	VREFON				ADTRGE	ADCH2	ADCH1	ADCH0
Read/Write	R/W				R/W			
After reset	1				0	0	0	0
Function	VREF application control 0: OFF 1: ON				AD external trigger start control 0: Disable 1: Enable	Analog input channel selection		

Analog input channel selection

<ADCH2,1,0> \ <SCAN>	0 (Channel fixed)	1 (Channel scan)
000	AN0	AN0
001	AN1	AN0→AN1
010	AN2	AN0→AN1→AN2
011 (Note)	AN3	AN0→AN1→AN2→AN3
100	AN4	AN4
101	AN5	AN4→AN5
110	AN6	AN4→AN5→AN6
111	AN7	AN4→AN5→AN6→AN7

AD conversion start control by external trigger ($\overline{\text{ADTRG}}$ input)

0	Disable
1	Enable

Control of application of reference voltage to AD converter

0	OFF
1	ON

Before starting conversion (before writing 1 to ADMOD0<ADS>), set the <VREFON> bit to 1.

Note: As pin AN3 also functions as the $\overline{\text{ADTRG}}$ input pin, do not set <ADCH2 to 0> = 011 when using $\overline{\text{ADTRG}}$ with <ADTRGE> set to 1.

Figure 3.10.2 Register for AD Converter (2/4)

AD Conversion Data Lower Register 0/4

ADREG04L (0060H)	7	6	5	4	3	2	1	0
bit Symbol	ADR01	ADR00						ADR0RF
Read/Write	R							R
After reset	Undefined							0
Function	Stores lower 2 bits of AD conversion result							AD conversion data storage flag 1: Conversion result stored

AD Conversion Data Upper Register 0/4

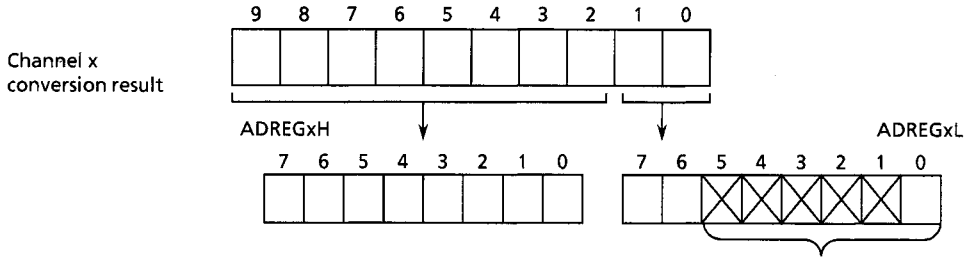
ADREG04H (0061H)	7	6	5	4	3	2	1	0
bit Symbol	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
Read/Write	R							
After reset	Undefined							
Function	Stores upper eight bits of AD conversion result.							

AD Conversion Data Lower Register 1/5

ADREG15L (0062H)	7	6	5	4	3	2	1	0
bit Symbol	ADR11	ADR10						ADR1RF
Read/Write	R							R
After reset	Undefined							0
Function	Stores lower 2 bits of AD conversion result							AD conversion result flag 1: Conversion result stored

AD Conversion Data Upper Register 1/5

ADREG15H (0063H)	7	6	5	4	3	2	1	0
bit Symbol	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
Read/Write	R							
After reset	Undefined							
Function	Stores upper eight bits of AD conversion result.							



- Bits 5 to 1 are always read as 1.
- Bit 0 is the AD conversion data storage flag <ADRxRF>. When the AD conversion result is stored, the flag is set to 1. When either of the registers (ADREGxH, ADREGxL) is read, the flag is cleared to 0.

Figure 3.10.2 Register for AD Converters (3/4)

AD Conversion Result Lower Register 2/6

ADREG26L (0064H)	7	6	5	4	3	2	1	0
bit Symbol	ADR21	ADR20						ADR2RF
Read/Write	R							R
After reset	Undefined							0
Function	Stores lower 2 bits of AD conversion result							AD conversion data storage flag 1: Conversion result stored

AD Conversion Data Upper Register 2/6

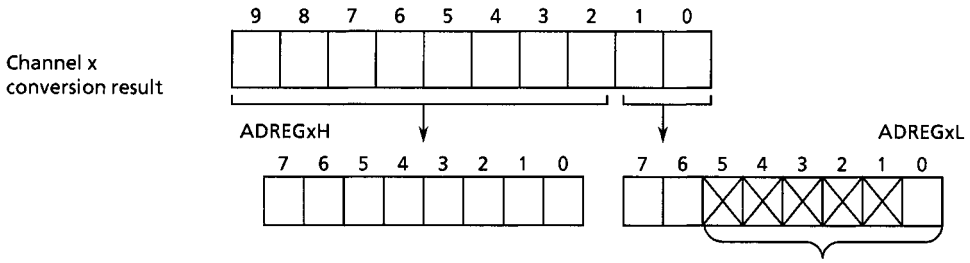
ADREG26H (0065H)	7	6	5	4	3	2	1	0
bit Symbol	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
Read/Write	R							
After reset	Undefined							
Function	Stores upper eight bits of AD conversion result.							

AD Conversion Data Lower Register 3/7

ADREG37L (0066H)	7	6	5	4	3	2	1	0
bit Symbol	ADR31	ADR30						ADR3RF
Read/Write	R							R
After reset	Undefined							0
Function	Stores lower 2 bits of AD conversion result							AD conversion data storage flag 1: Conversion result stored

AD Conversion Result Upper Register 3/7

ADREG37H (0067H)	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 eight bits of AD conversion result.							



- Bits 5 to 1 are always read as 1.
- Bit 0 is the AD conversion data storage flag <ADRxRF>. When the AD conversion result is stored, the flag is set to 1. When either of the registers (ADREGxH, ADREGxL) is read, the flag is cleared to 0.

Figure 3.10.2 Register for AD Converters (4/4)

3.10.2 Description of Operation

(1) Analog reference voltage

A high level analog reference voltage is applied to the VREFH pin; a low level analog reference voltage to the VREFL pin. To perform AD conversion, the reference voltage (the difference between VREFH and VREFL) is divided by 1024 using string resistance. Then, the result of the division is compared with the analog input voltage.

To turn off the switch between VREFH and VREFL, write 0 to AD mode control register 1 ADMOD1<VREFON>. To start AD conversion from the off state, first write 1 to <VREFON>, wait 3 μs until the internal reference voltage stabilizes (not related to the fc), then write 1 to AD mode register ADMOD0<ADS>.

(2) Analog input channel selection

The analog input channel selection varies according to the operating mode of the AD converter.

- In analog input channel fixed mode (ADMOD0<SCAN> = 0)
  Setting ADMOD1<ADCH2 to 0> selects one channel from among analog input pins AN0 to AN7.
- In analog input channel scan mode (ADMOD0<SCAN> = 1)
  Setting ADMOD1<ADCH2 to 0> selects one scan mode from among eight scan modes.

Table 3.10.1 shows the analog input channel selection for each operating mode.

After a reset, ADMOD0<SCAN> is set to 0 and ADMOD1<ADCH2 to 0> is initialized to 000, thus selecting pin AN0 as the channel fixed input. Pins not used as analog input channels can be used as standard input ports.

Table 3.10.1 Analog Input Channel Selection

<ADCH2 to 0>	Channel fixed <SCAN> = "0"	Channel scan <SCAN> = "1"
000	AN0	AN0
001	AN1	AN0→AN1
010	AN2	AN0→AN1→AN2
011	AN3	AN0→AN1→AN2→AN3
100	AN4	AN4
101	AN5	AN4→AN5
110	AN6	AN4→AN5→AN6
111	AN7	AN4→AN5→AN6→AN7

(3) Starting AD conversion

To start AD conversion, write 1 to AD mode control register 0 ADMOD0<ADS> or AD mode control register 1 ADMOD1<ADTRGE> and input a falling edge on the $\overline{\text{ADTRG}}$ pin. When AD conversion starts, the AD conversion busy flag ADMOD0<ADBF> is set to 1, indicating AD conversion is in progress.

Writing 1 to <ADS> during AD conversion restarts conversion. At that time, to determine whether the AD conversion results are preserved, check the conversion data storage flag ADREGxL<ADRxRF>.

During AD conversion, inputting a falling edge to the $\overline{\text{ADTRG}}$ pin is ignored.

(4) AD conversion modes and AD conversion end interrupt

The four AD conversion modes are:

- Channel fixed single conversion mode
- Channel scan single conversion mode
- Channel fixed repeat conversion mode
- Channel scan repeat conversion mode

AD mode control register 0 ADMOD0<REPET>, <SCAN> selects the AD mode.

Completion of AD conversion triggers the AD conversion end INTAD interrupt request. Also, ADMOD0<EOCF> is set to 1 to indicate that AD conversion is complete.

[1] Channel fixed single conversion mode

Setting ADMOD0<REPET>, <SCAN> to 00 sets conversion channel fixed single conversion mode.

In this mode, one specified channel is converted once only. When the conversion is complete, the ADMOD0<EOCF> flag is set to 1, ADMOD0<ADBF> is cleared to 0, and an INTAD interrupt request is generated.

[2] Channel scan single conversion mode

Setting ADMOD0<REPET>, <SCAN> to 01 sets conversion channel scan single conversion mode.

In this mode, the specified scan channels are converted once only. When scan conversion is complete, ADMOD0<EOCF> is set to 1, ADMOD0<ADBF> is cleared to 0, and an INTAD interrupt request is generated.

[3] Channel fixed repeat conversion mode

Setting ADMOD0<REPET>, <SCAN> to 10 sets conversion channel fixed repeat conversion mode.

In this mode, one specified channel is converted repeatedly. When conversion is complete, ADMOD0<EOCF> is set to 1 and ADMOD0<ADBF> is not cleared to 0 but held at 1. The INTAD interrupt request generation timing is selected by ADMOD0<ITM0>.

Setting <ITM0> to 0 generates an interrupt request when every AD conversion is complete.

Setting <ITM0> to 1 generates an interrupt request when every fourth conversion is complete.

[4] Channel scan repeat conversion mode

Setting ADMOD0<REPET>, <SCAN> to 11 sets conversion channel scan repeat conversion mode.

In this mode, the specified scan channels are converted repeatedly. When each scan conversion is complete, ADMOD0<EOCF> is set to 1 and an INTAD interrupt request is generated. ADMOD0<ADBF> is not cleared to 0 but held at 1.

To stop conversion in a repeat conversion mode (mode [3] or [4]), write 0 to ADMOD0<REPET>. After the current conversion is complete, the repeat conversion mode terminates and ADMOD0<ADBF> is cleared to 0.

Switching to a halt state (IDLE2, IDLE1, or STOP) immediately stops the AD converter even with AD conversion still in progress. In repeat conversion modes (modes [3] and [4]), after the halt is released, conversion restarts from the beginning. In single conversion modes (modes [1] and [2]), conversion does not restart (the converter remains stopped).

Table 3.10.2 shows the relationship between AD conversion modes and interrupt requests.

Table 3.10.2 Relationship Between AD Conversion Modes and Interrupt Requests

Mode	Interrupt request generation	ADMOD0		
		<ITM0>	<REPET>	<SCAN>
Channel fixed single conversion mode	After completion of conversion	X	0	0
Channel scan single conversion mode	After completion of scan conversion	X	0	1
Channel fixed repeat conversion mode	Every conversion	0	1	0
	Every fourth conversion	1		
Channel scan repeat conversion mode	After completion of every scan conversion	X	1	1

X: Don't care

(5) AD conversion time

84 states (7 μs @ fc = 24 MHz) are required for AD conversion of one channel.

(6) Storing and reading AD conversion result

The AD conversion data upper and lower registers (ADREG04H/L to ADREG37H/L) store the AD conversion results. (ADREG04H/L to ADRG37H/L are read-only registers.)

In channel fixed repeat conversion mode, the conversion results are stored successively in registers ADREG04H/L to ADRG37H/L. In other modes, the AN0 and AN4, AN1 and AN5, AN2 and AN6, and AN3 and AN7 conversion results are stored in ADREG04H/L, ADREG15H/L, ADREG26H/L, and ADREG37H/L respectively.

Table 3.10.3 shows the correspondence between analog input channels and AD conversion result registers.

Table 3.10.3 Correspondence Between Analog Input Channels and AD Conversion Result Registers

Analog input channel (port A)	AD conversion result register	
	Conversion modes other than at right	Channel fixed repeat conversion mode (every 4th conversion)
AN0	ADREG04H/L	ADREG04H/L → ADREG15H/L → ADREG26H/L → ADREG37H/L → ADREG04H/L
AN1	ADREG15H/L	
AN2	ADREG26H/L	
AN3	ADREG37H/L	
AN4	ADREG04H/L	
AN5	ADREG15H/L	
AN6	ADREG26H/L	
AN7	ADREG37H/L	

The AD conversion data storage flag <ADRxRF> uses bit 0 of the AD conversion data lower register. The storage flag indicates whether the AD conversion result register was read or not. When a conversion result is stored in the AD conversion result register, the flag is set to 1. When either of the AD conversion result registers (ADREGxH or ADREGxL) is read, the flag is cleared to 0.

Reading the AD conversion result also clears the AD conversion end flag ADMOD0<EOCF> to 0.

Setting example:

[1] Convert the analog input voltage at the AN3 pin and write the result to memory address 0800H using the AD interrupt (INTAD) processing routine.

Main routine setting:

```
            7 6 5 4 3 2 1 0
INTEOAD ← 1 1 0 0 - - - -      Enable INTAD and set level to 4.
ADMOD1  ← 1 X X X 0 0 1 1      Set analog input channel to pin AN3.
ADMOD0  ← X X 0 0 0 0 0 1      Start conversion in channel fixed single conversion mode.
```

Interrupt routine processing example:

```
WA      ← ADREG37              Read value of ADREG37L and ADREG37H to general-
                               purpose register WA (16 bits).
WA      >> 6                   Shift contents read in WA six times to right and zero-fill
                               upper bits.
(0800H) ← WA                   Write contents of WA to memory address 0800H.
```

[2] This example repeatedly converts the analog input voltages at the three pins AN0 to AN2, using channel scan repeat conversion mode.

```
INTEOAD ← 1 0 0 0 - - - -      Disable INTAD.
ADMOD1  ← 1 X X X 0 0 1 0      Set pins AN0 to AN2 as analog input channels.
ADMOD0  ← X X 0 0 0 1 1 1      Start conversion in channel scan repeat conversion mode.
```

Note: X : Don't care – : No change

3.11 CAN Controller

(1) Overview

- Supports CAN version 2.0B
- Supports standard format and Extended format
- Supports data frames and remote frames in both formats
- 16 Mailboxes (15 Receive & Transmit + 1 Receive only)
- Baud rate up to 1Mbps on the CAN bus (at operation frequency 20 to 24 MHz)
- Programmable baud rate with bit time parameter
- Built in baud rate prescaler
- 2 selectable mechanisms for internal arbitration of transmit messages
  [1] mailbox number
  [2] identifier priority
- Time stamp for receive and transmit messages
- Operation mode
  [1] Normal operation mode
  [2] Configuration mode
  [3] Sleep mode (Wake up on CAN bus activity or CPU access)
  [4] Halt mode
  [5] Test loopback mode (stand alone operation enabled by self acknowledge)
  [6] Test error mode (Write enabled to error counter)
- Message acceptance filter
  [1] Programmable global mask for mailboxes 0 to 14
  [2] Programmable local mask for mailbox 15
- Acceptance mask bit for identifier extended bit
- Flexible interrupt structure (3 interrupts)
  [1] Receive interrupt: INTCR
  [2] Transmit interrupt: INTCT
  [3] Global interrupt: INTCG (includes warning level, error passive, bus off, etc.)

(2) Nomenclature

- R/W Read and write access by the CPU
- R Read access by the CPU
- W Write access by the CPU
- R/S Read access and set (write with 1) by the CPU
- R/C Read access and clear (write with 1) by the CPU
- The mailbox RAM symbol "–" following a Reset indicates that the initial value is indeterminate.
- The mailbox RAM bit Symbol "\" denotes blank bits. The values of these bits are indeterminate when read.
- The control register bit Symbol "\" denotes reserved bits. They indicate that value is indeterminate when read.
  Always write "0" when write.

(3) Architecture

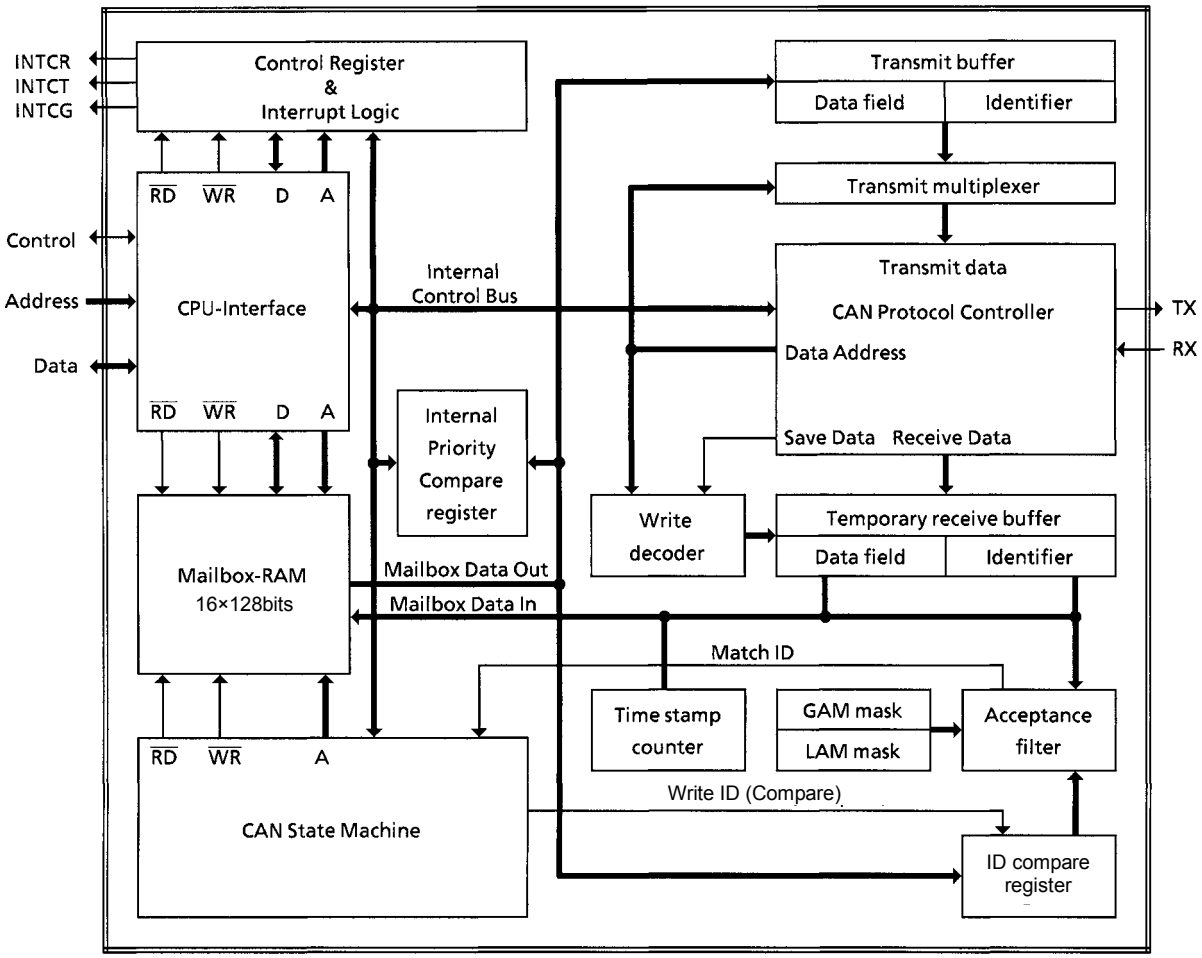


Figure 3.11.1 Block Diagram of CAN Controller

(4) CAN bus interface

The interface to the CAN bus is a simple two-wire line, consisting of an input pin RX and an output pin TX. This CAN bus interface is suitable for operation with CAN bus transceivers based on ISO/DIS 11898.

3.11.1 Memory Map

The mailboxes and control registers used by the CAN are mapped to the memory locations shown below.

Table 3.11.1 CAN Mailboxes and Control Registers

Address	Register	Description
002200H *	MB0	Mailbox RAM
:	:	
0022FFH *	MB15	
002300H	MC	Mailbox Configuration Register
002302H	MD	Mailbox Direction Register
002304H *	TRS	Transmit Request Set Register
002306H *	—	(Reserved)
002308H *	TA	Transmission Acknowledge Register
00230AH *	—	(Reserved)
00230CH *	RMP	Receive Message Pending Register
00230EH *	RML	Receive Message Lost Register
002310H	LAM0 (high)	Local Acceptance Mask Register 0 (bit 28 to 16)
002312H	LAM1 (low)	Local Acceptance Mask Register 1 (bit 15 to 0)
002314H	GAM0 (high)	Global Acceptance Mask Register 0 (bit 28 to 16)
002316H	GAM1 (low)	Global Acceptance Mask Register 1 (bit 15 to 0)
002318H	MCR	Master Control Register
00231AH	GSR	Global Status Register
00231CH	BCR1	Bit Configuration Register 1
00231EH	BCR2	Bit Configuration Register 2
002320H *	GIF	Global Interrupt Flag Register
002322H	GIM	Global Interrupt Mask Register
002324H *	MBTIF	Mailbox Transmit Interrupt Flag Register
002326H *	MBRIF	Mailbox Receive Interrupt Flag Register
002328H	MBIM	Mailbox Interrupt Mask Register
00232AH	CDR	Change Data Request Register
00232CH *	RFP	Remote Frame Pending Register
00232EH *	CEC	CAN Error Counter Register
002330H	TSP	Time Stamp Counter Prescaler Register
002332H *	TSC	Time Stamp Counter Register

Note1: * Read- modify-write prohibited.
Note2: Do not access the reserved address.

3.11.2 Mailboxes

The mailbox is configured with RAM to store identifiers and transmit/receive data, which can be accessed by the CAN controller and the CPU. The CPU controls the CAN controller by modifying the contents of the mailboxes and control registers. The contents of the mailboxes and control registers are used to perform the functions of acceptance filtering, transmit message and interrupt handling.

In order to initiate a transfer, the transmission request bit has to be written to the corresponding register. The entire transmission procedure is done then without any CPU involvement. If a mailbox has been configured as receive messages the CPU easily reads its data registers using CPU read instructions. The mailbox may be configured to interrupt the CPU after every successful message transmission or reception.

The mailbox module provides 16 mailboxes, each of which has 8 bytes long data, 29-bit identifier and several control bits. Each mailbox is 16 bytes in size. Each mailbox, except the last one, can be set for either transmit or receive operation.

Mailbox 15 is a receive-only mailbox with a special acceptance mask designed to allow groups of different message identifiers to be received.

The receive-only mailbox 15 masks all bits when receiving a message whose ID does not correspond to any of the mailboxes 0 to 14.

In addition, when using mailbox 15 as a usual receiving mailbox, the automatic remote frame response function cannot be used. Set <RFH> bit to 0 to disable automatic remote frame response. In this case, each time a message is received, the ID of all mailboxes is checked by software. If the ID is rewritten to a different ID from that which was originally received, the received data is invalid. Once mailbox prohibit (MC=0) is set, after waiting maximum frame length, set the correct ID again.

Address	Mailboxes
2200H to 220FH	MB0 (Used for transmit/receive)
2210H to 221FH	MB1 (Used for transmit/receive)
:	:
:	:
22E0H to 22EFH	MB14 (Used for transmit/receive)
22F0H to 22FFH	MB15 (Used for receive-only)

Each mailbox is configured as shown below.

(Mailbox "n")	b15	b0	
MBn + 00H	MI0		(Message identifier field 0)
02H	MI1		(Message identifier field 1)
04H	MCF		(Message control field)
06H	D1	D0	(Data field 0, 1)
08H	D3	D2	(Data field 2, 3)
0AH	D5	D4	(Data field 4, 5)
0CH	D7	D6	(Data field 6, 7)
0EH	TSV		(Time stamp value)

Note: MBn = 2200H + n × 10H

The components of each mailbox are explained in the following pages.

Message Identifier Field 0 (MI0)

Message Identifier Field 0 Low

MI0L (MBn + 00H) RMW instructions prohibited

	7	6	5	4	3	2	1	0
bit Symbol	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
Read/Write	R/W							
After reset	–	–	–	–	–	–	–	–

Identifiers <ID23> to <ID16> are stored.

Message Identifier Field 0 High

MI0H (MBn + 01H) RMW instructions prohibited

	15	14	13	12	11	10	9	8
bit Symbol	IDE	GAME	RFH	ID28	ID27	ID26	ID25	ID24
Read/Write	R/W							
After reset	–	–	–	–	–	–	–	–

Identifiers <ID28> to <ID24> are stored.

Remote frame processing bit Note1) Note2)

0	For transmit mailbox, remote frame are not responded to.
1	For transmit mailbox, remote frame are responded to. (The <TRS> bit is set.)
0/1	For receive mailbox, they are processed as data frames. (The <RMP> and <RFP> bits are set.)

Global (local) acceptance mask enable

0	Acceptance mask is not used for acceptance filtering.
1	Acceptance mask is used for acceptance filtering.

For mailbox 15, it functions as local acceptance mask enable bit <LAME>.

Identifier extension bit

0	Standard format (11-bit identifier) Identifiers <ID28> to <ID18> are used.
1	Extended format (29-bit identifier) Identifiers <ID28> to <ID0> are used.

The priority of a message ID becomes so high that 0 continues from the MSB (<ID28> bit) of ID.

Note1: When the receive-only mailbox MB15 is used as a usual receiving mailbox, the automatic remote frame response function cannot be used. The remote frame processing bit <RFH> is set to “0”.

Note2: When the ID of the received remote frame corresponds to the ID of the transmission mailbox <RFH> = “1” and <GAME> = “1”, the ID of the remote frame is overwritten to this mailbox. Thereafter, it responds by automatically applying the overwritten ID.

Message Identifier Field 1 (MI1)

Message Identifier Field 1 Low

MI1L (MBn + 02H) RMW instructions prohibited	7	6	5	4	3	2	1	0
bit Symbol	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
Read/Write	R/W							
After reset	–	–	–	–	–	–	–	–

Identifiers <ID7> to <ID0> are stored.

Message Identifier Field 1 High

MI1H (MBn + 03H) RMW instructions prohibited	15	14	13	12	11	10	9	8
bit Symbol	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
Read/Write	R/W							
After reset	–	–	–	–	–	–	–	–

Identifiers <ID15> to <ID8> are stored.

Note1: For standard format, identifiers <ID17> to <ID0> are indeterminate.

Note2: Set the mailbox ID at initial configuration. Once a mailbox has been enabled, when writing to the MI0 or MI1 field of the mailbox, reset the <MC> bit, and after a mailbox has been prohibited by the CAN controller, wait the maximum frame length time before executing the operation.

Message Control Field (MCF)

Message Control Field Low

MCFL (MBn + 04H) RMW instructions prohibited	7	6	5	4	3	2	1	0
bit Symbol				RTR	DLC3	DLC2	DLC1	DLC0
Read/Write				R/W				
After reset				–	–	–	–	–

Data length code

<DLC3:0>	Data Bytes	Corresponding Mailbox Data
0000	0 byte	None
0001	1 byte	D0
0010	2 bytes	D1, D0
0011	3 bytes	D2, D0, D0
0100	4 bytes	D3, D2, D1, D0
0101	5 bytes	D4, D3, D2, D1, D0
0110	6 bytes	D5, D4, D3, D2, D1, D0
0111	7 bytes	D6, D5, D4, D3, D2, D1, D0
1000	8 bytes	D7, D6, D5, D4, D3, D2, D1, D0

Note: Do not use data length codes other than those listed above.

Remote transmit request bit

0	Data frame
1	Remote fame

Message Control Field High

MCFH (MBn + 05H) RMW instructions prohibited	7	6	5	4	3	2	1	0
bit Symbol								
Read/Write								
After reset								

Note: There is no necessity for initial configuration of receive mailboxes. RTR and DLC of the received message are stored in the MCF register. Please set transmit mailboxes at the initial configuration.

Data field (D0 to D7)

This is a read/write register that stores up to 8 bytes of transmit/receive data. However, in the case of receive mailboxes, the write access to the data field is disabled.

For transmit, data in a length of bytes set by the mailbox's data length code is transmitted.

For receive, the data length code in the receive message is copied to the mailbox's data length code, so that the byte in a length equal to this data length code is received as valid data.

Data field 0

D0 (MBn + 06H) RMW instructions prohibited

	7	6	5	4	3	2	1	0
bit Symbol	D07	D06	D05	D04	D03	D02	D01	D00
Read/Write	R/W							
After reset	–	–	–	–	–	–	–	–

Data field 1

D1 (MBn + 07H) RMW instructions prohibited

	15	14	13	12	11	10	9	8
bit Symbol	D17	D16	D15	D14	D13	D12	D11	D10
Read/Write	R/W							
After reset	–	–	–	–	–	–	–	–

Data field 2

D2 (MBn + 08H) RMW instructions prohibited

	7	6	5	4	3	2	1	0
bit Symbol	D27	D26	D25	D24	D23	D22	D21	D20
Read/Write	R/W							
After reset	–	–	–	–	–	–	–	–

Data field 3

D3 (MBn + 09H) RMW instructions prohibited

	15	14	13	12	11	10	9	8
bit Symbol	D37	D36	D35	D34	D33	D32	D31	D30
Read/Write	R/W							
After reset	–	–	–	–	–	–	–	–

Data field 4

D4 (MBn + 0AH) RMW instructions prohibited

	7	6	5	4	3	2	1	0
bit Symbol	D47	D46	D45	D44	D43	D42	D41	D40
Read/Write	R/W							
After reset	–	–	–	–	–	–	–	–

Data field 5

D5 (MBn + 0BH) RMW instructions prohibited

	15	14	13	12	11	10	9	8
bit Symbol	D57	D56	D55	D54	D53	D52	D51	D50
Read/Write	R/W							
After reset	–	–	–	–	–	–	–	–

Data field 6

D6 (MBn + 0CH) RMW instructions prohibited

	7	6	5	4	3	2	1	0
bit Symbol	D67	D66	D65	D64	D63	D62	D61	D60
Read/Write	R/W							
After reset	–	–	–	–	–	–	–	–

Data field 7

D7 (MBn + 0DH) RMW instructions prohibited

	15	14	13	12	11	10	9	8
bit Symbol	D77	D76	D75	D74	D73	D72	D71	D70
Read/Write	R/W							
After reset	–	–	–	–	–	–	–	–

Time Stamp Value (TSV)

Time Stamp Value Low

TSVL (MBn + 0EH)

	7	6	5	4	3	2	1	0
bit Symbol	TSV7	TSV6	TSV5	TSV4	TSV3	TSV2	TSV1	TSV0
Read/Write	R							
After reset	–	–	–	–	–	–	–	–

Time Stamp Value High

TSVH (MBn + 0FH)

	15	14	13	12	11	10	9	8
bit Symbol	TSV15	TSV14	TSV13	TSV12	TSV11	TSV10	TSV9	TSV8
Read/Write	R							
After reset	–	–	–	–	–	–	–	–

This is a 16-bit read-only register into which the value of the time stamp counter is loaded when data is successfully transmitted or received.

The counter value is not loaded into this register when transmit or receive operations fail.

Maximum frame length

Rewrite message ID field after MCn is prohibited and one frame time passes.
General one frame time is as follows:
N means the number of data byte (0-8 byte).

- Standard frame format (at data frame) = (44 + 8N) × 1 bit time
- Extended frame format (at data frame) = (64 + 8N) × 1 bit time

Furthermore, the maximum frame length to which bit stuffing applies is as follows:
Since the maximum number of bits is eight byte data of the extended frame format,
64-bit(fixed length) + 64-bit (number of data bytes) = 128-bit.
Moreover, the bit stuffing rule is not applied to
EOF + ACK field + CRC delimiter = 10-bit,
Therefore the maximum number of bits to which the bit stuffing rule applies is
128-bit – 10-bit = 118-bit length.
(It is calculated on the assumption of the longest case of insertion of the stuffing bit in the CRC field.)
At the bit stuffing rule, as a reversing bit is inserted when the same level is 5-bit successive, the maximum inserted number of bits is
118-bit ÷ 5 → 24-bit.
Hence, the maximum frame length is
128-bit + 24-bit = 152-bit.
Therefore, when the message ID field is rewritten at a baud-rate of 500kbps, is the necessary waiting time is:
152-bit × 2 us = 304 μs

3.11.3 Control Registers

(1) Mailbox control registers

Mailbox configuration register (MC)

Mailbox Configuration Register Low

MCL (2300H)

	7	6	5	4	3	2	1	0
bit Symbol	MC7	MC6	MC5	MC4	MC3	MC2	MC1	MC0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Mailbox Configuration Register High

MCH (2301H)

	15	14	13	12	11	10	9	8
bit Symbol	MC15	MC14	MC13	MC12	MC11	MC10	MC9	MC8
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Each bit corresponds to mailboxes 0 through 15.

Each mailbox can be enabled or disabled.

When <MCn> = 1, access to mailbox “n” is enabled.

When <MCn> = 0, access to mailbox “n” is disabled.

If, during CAN controller transmission, <MCn>=0, access may be permitted depending on the transmission stage. In this case, there is the possibility of conflict between the mailbox transmit/receive complete flag and the transmit/receive interrupt flag.

Set the mailbox ID at initial configuration. After disabling a mailbox by resetting the <MC> bit, wait the maximum frame length time before rewriting to the MI0 or MI1 field of the mailbox which is permitted.

The transmit mailbox data and control fields can be accessed for write at any time. However, in the case of transmit mailboxes where the <RFH> bit is set, the write access to the message control field is enabled when the <MC> bit is reset.

Mailbox direction register (MD)

Mailbox Direction Register Low

MDL (2302H)

	7	6	5	4	3	2	1	0
bit Symbol	MD7	MD6	MD5	MD4	MD3	MD2	MD1	MD0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Mailbox Direction Register High

MDH (2303H)

	15	14	13	12	11	10	9	8
bit Symbol	MD15	MD14	MD13	MD12	MD11	MD10	MD9	MD8
Read/Write	R	R/W						
After reset	1	0	0	0	0	0	0	0

Each bit corresponds to mailbox 0 through 15.

Each mailbox except mailbox 15 can be directed for transmit or receive.

When <MDn> = 0, the mailbox MBn is directed for transmit.

When <MDn> = 1, the mailbox MBn is directed for receive.

Mailbox 15 is a receive-only mailbox, so that <MD15> bit is fixed to 1. This bit can only be read; you cannot write to it.

MD registers are set at initial configuration. When the setting is changed while transmitting or receiving, the following operations occur.

(1) When changing to <MDn> = 0 (transmission) while receiving.

Reception of the message currently being received continues, and after the reception is completed, the <RMPn> bit is set to 1. However, the <MBRIFn>flag is not set even if <MBIMn> is set to 1 (interrupt enabled), and the receive interrupt is not generated.

(2) When changing to <MDn>=1 (reception) while transmitting.

Transmission of the message currently being transmitted continues, and after the transmission is completed, <TAn> bit is set to 1. However, the <MBTIFn> flag is not set even if <MBIMn> is set to 1 (interrupt enabled), and the transmit interrupt is not generated.

(2) Transmit control registers

Transmission request set register (TRS)

Transmission Request Set Register Low

TRSL (2304H) RMW instructions prohibited	7	6	5	4	3	2	1	0
bit Symbol	TRS7	TRS6	TRS5	TRS4	TRS3	TRS2	TRS1	TRS0
Read/Write	R/S							
After reset	0	0	0	0	0	0	0	0

Transmission Request Set Register High

TRSH (2305H) RMW instructions prohibited	15	14	13	12	11	10	9	8
bit Symbol		TRS14	TRS13	TRS12	TRS11	TRS10	TRS9	TRS8
Read/Write		R/S						
After reset		0	0	0	0	0	0	0

Each bit corresponds to mailboxes 0 through 15. Since mailbox 15 is a receive-only mailbox, bit 15 is nonexistent.

If after writing data and identifier to mailbox "n" that has been set for transmit (<MDn> = 0), the <TRSn> bit is set when the said mailbox is enabled (<MCn> = 1), a message is transmitted from mailbox "n". If there are multiple transmit requests, messages are transmitted sequentially. The order in which messages are transmitted depends on the master control register MCR bit 3 <MTOS>.

If <MTOS> bit is set to "0", the mailbox with the lower number has the higher priority. For example: if the mailboxes MB0, MB2 and MB5 are configured for transmission and the corresponding TRS bits are set, then the messages will be transmitted in the following order: MB0, MB2 and MB5. If a new transmission request is set for MB0 during the processing of MB2 then in the next internal arbitration-run MB0 will be selected for the next transmission. This will also happen, when the CAN controller loses arbitration while transmitting MB2. In this case, MB0 will be sent at the next opportunity instead of MB2.

If <MTOS> bit is set to "1", the priority of the identifier stored in the mailbox will determine the sending order. The mailbox with the higher-priority identifier will be sent first. In case of a lost arbitration on the CAN bus line, a new internal arbitration run will be started and the message with the highest priority will be sent at the next possible time.

The <TRSn> bit is reset when transmit has succeeded.

If transmit has failed, transmit is retried repeatedly until it succeeds.

When the <TRSn> bit is 1, the write access to the corresponding mailbox is denied.

The <TRSn> bit cannot be set from the CPU if mailbox "n" is set for receive.

When mailbox "n" is set for transmit, the <TRSn> bit is set by writing a 1 from the CPU and is reset by the internal logic. Writing a 0 from the CPU has no effect.

Transmission acknowledge register (TA)

Transmission Acknowledge Register Low

TAL (2308H) RMW instructions prohibited

	7	6	5	4	3	2	1	0
bit Symbol	TA7	TA6	TA5	TA4	TA3	TA2	TA1	TA0
Read/Write	R/C							
After reset	0	0	0	0	0	0	0	0

Transmission Acknowledge Register High

TAH (2309H) RMW instructions prohibited

	15	14	13	12	11	10	9	8
bit Symbol		TA14	TA13	TA12	TA11	TA10	TA9	TA8
Read/Write	R/C							
After reset		0	0	0	0	0	0	0

Each bit corresponds to mailboxes 0 through 15. Since mailbox 15 is a receive-only mailbox, bit 15 is nonexistent.

The <TAn> bit is set when the message of mailbox "n" has been transmitted successfully. In this case, a transmission successful interrupt is generated if it has been enabled.

The <TAn> bit is reset by writing a "1" to the <TAn> bit or the <TRSn> bit from the CPU. Writing a "0" from the CPU has no effect.

Change data request register (CDR)

Change Data Request Register Low

CDRL (232AH)

	7	6	5	4	3	2	1	0
bit Symbol	CDR7	CDR6	CDR5	CDR4	CDR3	CDR2	CDR1	CDR0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Change Data Request Register High

CDRH (232BH)

	15	14	13	12	11	10	9	8
bit Symbol		CDR14	CDR13	CDR12	CDR11	CDR10	CDR9	CDR8
Read/Write	R/W							
After reset		0	0	0	0	0	0	0

Each bit corresponds to mailboxes 0 through 15. Since mailbox 15 is a receive-only mailbox, bit 15 is nonexistent.

If the <CDRn> bit is set, a transmission request for mailbox "n" will be ignored. That means, that a mailbox "n" with the <TRSn> and <CDRn> bit set will not be considered in the internal arbitration-run: the mailbox "n" is locked for transmission. The processing of mailbox "n" in the arbitration-run will be considered again after clearing the <CDRn> bit.

The <CDR> bit is useful for dealing with remote frames. It is intended for updating the data field of a transmit mailbox, which is configured for automatic reply to remote frames (the <RFH> bit is set). By using the <CDR> bit, the user can update the data field without needing to take additional care with regard to data consistency.

(3) Receive control registers

The identifier of each incoming message is compared with the identifiers held in the mailboxes that have been set for receive operation. The comparison of the identifiers depends on the value of the global/local acceptance mask enable bits <GAME>/<LAME> in the mailbox and the data held in the global/local acceptance mask registers GAM/LAM.

When a matching identifier is detected, the received identifier, control bits, and data bytes are written to the mailbox that has matched. At this time, the corresponding receive message pending bit <RMPn> is set and a receive successful interrupt is generated if it has been enabled. Once a matching identifier is found, no other identifiers are compared.

If no match is detected, the message is rejected.

The CPU must reset the <RMPn> bit after reading the data. If a second message is received for this mailbox when the <RMPn> bit has already been set, the corresponding receive message lost bit <RMLn> is set. In this case, the data stored in mailbox "n" is overwritten with the new data. In this case, a global interrupt (receive message lost) is generated if it has been enabled.

Receive-only mailbox

Only if the identifier of a received message does not match any identifiers of the mailboxes 0 through 14 is the identifier compared with the identifier of the receive-only mailbox 15. When a matching identifier is detected, the contents of the received message are written to the mailbox 15.

Receive message pending register (RMP)

Receive Message Pending Register Low

RMPL (230CH) RMW instructions prohibited

	7	6	5	4	3	2	1	0
bit Symbol	RMP7	RMP6	RMP5	RMP4	RMP3	RMP2	RMP1	RMP0
Read/Write	R/C							
After reset	0	0	0	0	0	0	0	0

Receive Message Pending Register High

RMPH (230DH) RMW instructions prohibited

	15	14	13	12	11	10	9	8
bit Symbol	RMP15	RMP14	RMP13	RMP12	RMP11	RMP10	RMP9	RMP8
Read/Write	R/C							
After reset	0	0	0	0	0	0	0	0

Each bit corresponds to mailbox 0 through 15.

When a message is received and its content is stored in mailbox "n", the <RMPn> bit is set.

If a second message is received by mailbox "n" for which the <RMPn> bit has been set, mailbox "n" is overwritten with the new data. In this case, the corresponding <RMLn> bit is set.

The <RMPn> bit is set by the internal logic and is reset by writing a 1 to the <RMPn> bit from the CPU. Writing a 0 from the CPU has no effect.

Receive message lost register (RML)

Receive Message Lost Register Low

RMLL (230EH) RMW instructions prohibited

	7	6	5	4	3	2	1	0
bit Symbol	RML7	RML6	RML5	RML4	RML3	RML2	RML1	RML0
Read/Write	R/C							
After reset	0	0	0	0	0	0	0	0

Receive Message Lost Register High

RMLH (230FH) RMW instructions prohibited

	15	14	13	12	11	10	9	8
bit Symbol	RML15	RML14	RML13	RML12	RML11	RML10	RML9	RML8
Read/Write	R/C							
After reset	0	0	0	0	0	0	0	0

Each bit corresponds to mailbox 0 through 15.

If a second message is received by mailbox “n” for which the <RMPn> bit has been set, mailbox “n” is overwritten with the new data and the <RMLn> bit is set.

The <RMLn> bit is set by the internal logic and is reset by writing a 1 to the <RMPn> bit from the CPU. Writing a 0 to <RMPn> bit and writing a 1 or 0 to <RMLn> bit from the CPU have no effect.

(4) Handling of remote frames

If a remote frame is received, it is compared with the identifiers of all mailboxes. The comparison of identifiers depends on the value of the global/local acceptance mask enable bits <GAME>/<LAME> in the mailbox and the data held in the global/local acceptance mask registers GAM/LAM.

If a received remote frame matches the identifier of a mailbox that is set for transmit and the <RFH> bit for that mailbox is set, the <TRSn> bit is set in order to send a message in response to the remote frame. Even when there is a matching identifier, if the <RFH> bit for that mailbox is reset (even though it may be a transmit mailbox), the remote frame is not responded to.

If there is a matching identifier and this mailbox is set for receive, the remote frame is processed as data frame, in which case the <RMP> and <RFP> bits are set.

Once a matching identifier is found, no other identifiers are compared.

Table 3.11.2 Operation when Remote Frame is Received

ID	Mailbox	<RFH> bit	Handling of Remote Frame
Matched	Transmit	0	Not responded to.
		1	Responded to. (<TRS> bit is set) (Note)
	Receive	1/0	Not responded to. Processed as data frame. (<RMP> and <RFP> bits are set.)
Unmatched	Transmit/Receive	1/0	Not responded to.

Note: When the ID matches a <GAME>=1 mailbox, the mailbox ID is overwritten with the remote frame ID and automatically responds to this new ID.

Remote frame pending register (RFP)

Remote Frame Pending Register Low

RFPL (232CH) RMW instructions prohibited		7	6	5	4	3	2	1	0
	bit Symbol	RFP7	RFP6	RFP5	RFP4	RFP3	RFP2	RFP1	RFP0
	Read/Write	R/C							
	After reset	0	0	0	0	0	0	0	0

Remote Frame Pending Register High

RFPH (232DH) RMW instructions prohibited		15	14	13	12	11	10	9	8
	bit Symbol	RFP15	RFP14	RFP13	RFP12	RFP11	RFP10	RFP9	RFP8
	Read/Write	R/C							
	After reset	0	0	0	0	0	0	0	0

When a remote frame is received by mailbox "n" set for receive, the corresponding <RFPn> and <RMPn> bits are set. The <RFPn> bit is reset by writing a "1" to the <RMPn> bit. Writing a "0" has no effect. Also, the <RFPn> bit is reset automatically when the remote frame received in mailbox "n" is overwritten by a newly received data frame.

(5) Acceptance filter

The global acceptance mask registers GAM0 and GAM1 are used for filtering messages when the <GAME> bit for mailboxes 0 through 14 is set to 1. An incoming message is stored in the first mailbox with a matching identifier. Only if there is no matching identifier in the mailboxes 0 to 14 is the incoming message compared with the mailbox 15, a receive-only mailbox. The local acceptance mask registers LAM0, LAM1 are used for filtering messages when the <LAME> bit for mailbox 15 is set.

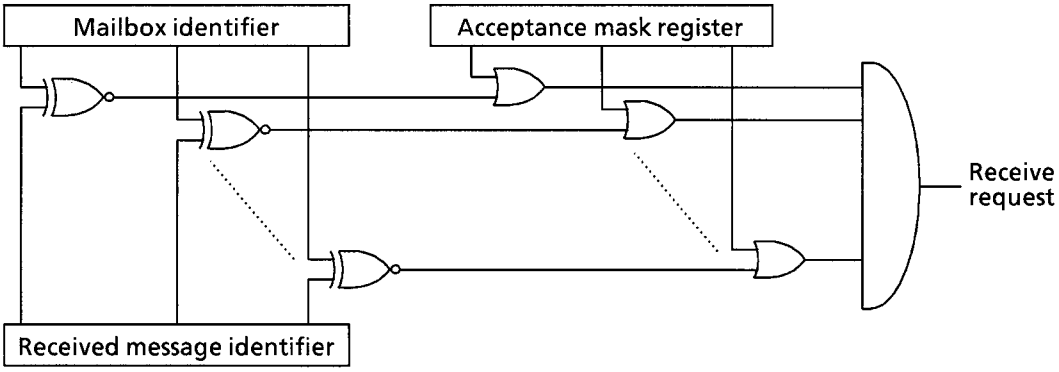


Figure 3.11.2 Acceptance Filter

Local acceptance mask registers (LAM0, LAM1)

Local Acceptance Mask Register 0 Low

LAM0L (2310H)

	7	6	5	4	3	2	1	0
bit Symbol	LAM23	LAM22	LAM21	LAM20	LAM19	LAM18	LAM17	LAM16
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Local Acceptance Mask Register 0 High

LAM0H (2311H)

	15	14	13	12	11	10	9	8
bit Symbol	LAMI			LAM28	LAM27	LAM26	LAM25	LAM24
Read/Write	R/W			R/W				
After reset	0			0	0	0	0	0

Local Acceptance Mask Register 1 Low

LAM1L (2312H)

	7	6	5	4	3	2	1	0
bit Symbol	LAM7	LAM6	LAM5	LAM4	LAM3	LAM2	LAM1	LAM0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Local Acceptance Mask Register 1 High

LAM1H (2313H)

	15	14	13	12	11	10	9	8
bit Symbol	LAM15	LAM14	LAM13	LAM12	LAM11	LAM10	LAM9	LAM8
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

The LAM0 and LAM1 registers are used only for filtering messages for mailbox 15. This feature allows the user to choose whether or not to locally mask any identifier bit of the incoming message for mailbox 15. Incoming messages are first checked to see if they match mailboxes 0 to 14 before being forwarded to mailbox 15.

If the <LAMn> bit is 0, messages are received only when the corresponding bit of the incoming message identifier matches that of the mailbox identifier. If the <LAMn> bit is 1, messages are received regardless of whether the corresponding bit of the incoming message identifier is 0 or 1. The GAM0 and GAM1 registers do not affect mailbox 15.

For messages in extended format, the identifier extension <IDE> bit and the whole 29 bits of the identifier are compared. For messages in standard format, only the <IDE> bit and the first 11bits of the identifier (<ID28> to <ID18>) are compared.

The <LAMI> bit (local acceptance mask identifier extension bit) is used to mask the <IDE> bit of mailbox 15.

If the <LAMI> bit is 0, messages in extended or standard format are received according to the <IDE> bit of mailbox 15.

If the <LAMI> bit is 1, messages in both extended and standard formats are received regardless of whether the <IDE> bit of mailbox 15 is 0 or 1. For messages in extended format, the whole 29bits of the mailbox identifier and the whole 29 mask bits of the LAM register are used for filtering. For messages in standard format, only the first 11bits of the mailbox identifier (<ID28> to <ID18>) and the first 11 bits of the LAM register (<LAM28> to <LAM18>) are used for filtering.

LAM0 and LAM1 are set at initial configuration. Do not change the setting of these registers while operating. If the setting is changed while receiving, the received message IDs are compared with the changed register values.

Global acceptance mask registers (GAM0, GAM1)

Global Acceptance Mask Register 0 Low

GAM0L (2314H)

	7	6	5	4	3	2	1	0
bit Symbol	GAM23	GAM22	GAM21	GAM20	GAM19	GAM18	GAM17	GAM16
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Global Acceptance Mask Register 0 High

GAM0H (2315H)

	15	14	13	12	11	10	9	8
bit Symbol	GAMI			GAM28	GAM27	GAM26	GAM25	GAM24
Read/Write	R/W			R/W				
After reset	0			0	0	0	0	0

Global Acceptance Mask Register 1 Low

GAM1L (2316H)

	7	6	5	4	3	2	1	0
bit Symbol	GAM7	GAM6	GAM5	GAM4	GAM3	GAM2	GAM1	GAM0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Global Acceptance Mask Register 1 High

GAM1H (2317H)

	15	14	13	12	11	10	9	8
bit Symbol	GAM15	GAM14	GAM13	GAM12	GAM11	GAM10	GAM9	GAM8
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

The GAM0 and GAM1 registers are used for filtering messages for mailbox 0 to 14.

If the <GAME> bit for mailboxes 0 to 14 is set, the GAM0 and GAM1 registers are used for incoming messages. A received message is stored in only the first mailbox with a matching identifier.

If the <GAMn> bit is 0, messages are received only when the corresponding bit of the incoming message identifier matches that of the mailbox identifier. If the <GAMn> bit is 1, messages are received regardless of whether the corresponding bit of the incoming message identifier is 0 or 1. For messages in extended format, the identifier extension <IDE> bit and the whole 29 bits of the identifier are compared. For messages in standard format, only the <IDE> bit and the first 11 bits of the identifier (<ID28> to <ID18>) are compared.

The <GAMI> bit (global acceptance mask identifier extension bit) is used to mask the <IDE> bits of mailbox 0 to 14.

If the <GAMI> bit is 0, messages in extended or standard format are received according to the <IDE> bits of mailbox 0 to 14.

If the <GAMI> bit is 1, messages in both extended and standard formats are received regardless of whether the <IDE> bits of mailbox 0 to 14 are 0 or 1. For messages in extended format, the whole 29 bits of the mailbox identifier and the whole 29 mask bits of the GAM register are used for filtering. For messages in standard format, only the first 11bits of the mailbox identifier (<ID28> to <ID18>) and the first 11 bits of the GAM register (<GAM28> to <GAM18>) are used for filtering.

GAM0 and GAM1 are set at initial configuration. Do not change the setting of these registers while operating. If the setting is changed while receiving, the received message IDs are compared with the changed register values.

(6) Control registers

Master control register (MCR)

Master Control Register Low

MCRL (2318H)	7	6	5	4	3	2	1	0
bit Symbol	CCR	SMR	HMR	WUBA	MTOS		TSCC	SRES
Read/Write	R/W						W	
After reset	1	0	0	0	0		0	0

Master Control Register High

MCRH (2319H)	15	14	13	12	11	10	9	8
bit Symbol							TSTLB	TSTERR
Read/Write							R/W	
After reset							0	0

TSTLB: Test Loopback

0: Cancels the test loopback mode. (Normal operation)
1: Requests the test loopback mode.
This mode supports stand-alone operation.

TSTERR: Test Error

0: Cancels the test error mode. (Normal operation)
1: Requests the test error mode.
In this mode it is possible to write the error counter register CEC.

CCR: Change Configuration Request

0: Cancels the configuration mode. (Normal operation)
1: Request the configuration mode.
This mode allows for writing to the bit configuration registers BCR1, BCR2.

SMR: Sleep Mode Request

0: The sleep mode is not requested. (Normal operation)
1: Requests the sleep mode.
When this mode is entered, the CAN controller clock stops oscillating and the error counter and transmit requests are cleared.

HMR: Halt Mode Request

0: Cancels the halt mode. (Normal operation)
1: Requests the halt mode.
When this mode is entered, the CAN controller no longer transmits and receives messages. It only sends error and acknowledge flags.

WUBA: Wake Up on Bus Activity

0: Wakes up the module only by detecting a write access to the MCR register.
1: Wakes up the module when active bus state is detected or detecting a write access to the MCR register.

MTOS: Mailbox Transmission Order Select

0: Mailbox transmission order by mailbox number. The mailbox with the lower number will be sent first.
1: Mailbox transmission order by identifier priority. The mailbox with the higher priority identifier will be sent first.

TSCC: Time Stamp Counter Clear

0: No effect
1: The time stamp counter will be cleared.

Note 1: This is a write-only bit; it is always 0 when read.
Note 2: The time stamp counter is also cleared by a write to the TSP register, or writing a 0 to the TSC register.

SRES: Software Reset

0: No effect
1: Resets the CAN controller in software. All internal registers are initialized.

Note: This is a write-only bit; it is always 0 when read.

Bit configuration register 1 (BCR1)

Bit Configuration Register 1 Low

BCR1L (231CH)

	7	6	5	4	3	2	1	0
bit Symbol	BRP7	BRP6	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

<BRP7:0> is the baud rate prescaler value. It can be set in the range of 0 to 255.

Bit Configuration Register 1 High

BCR1H (231DH)

	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write								
After reset								

Bit configuration register 2 (BCR2)

Bit Configuration Register 2 Low

BCR2L (231EH)

	7	6	5	4	3	2	1	0
bit Symbol	SAM	TSEG22	TSEG21	TSEG20	TSEG13	TSEG12	TSEG11	TSEG10
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Setting of SAM

<SAM>	Sampling Time
0	1
1	3

Setting of TSEG2

<TSEG22:20>	Unit Time of TSCL
000	Not available
001	2×TSCL
010	3×TSCL
011	4×TSCL
100	5×TSCL
101	6×TSCL
110	7×TSCL
111	8×TSCL

Setting of TSEG1

<TSEG13:10>	Unit Time of TSCL
0000	Not available
0001	2×TSCL
0010	3×TSCL
0011	4×TSCL
0100	5×TSCL
0101	6×TSCL
0110	7×TSCL
0111	8×TSCL
1000	9×TSCL
1001	10×TSCL
1010	11×TSCL
1011	12×TSCL
1100	13×TSCL
1101	14×TSCL
1110	15×TSCL
1111	16×TSCL

Bit Configuration Register 2 High

BCR2H (231FH)

	15	14	13	12	11	10	9	8
bit Symbol							SJW1	SJW0
Read/Write							R/W	
After reset							0	0

Setting of SJW

<SJW1:0>	Adjust Time
00	1×TSCL
01	2×TSCL
10	3×TSCL
11	4×TSCL

The bit length is determined by parameters TSEG1, TSEG2, and BRP. All CAN controllers on the CAN bus must operate at the same baud rate. If individual CAN controllers operate with different frequencies, the baud rate has to be adjusted by the mentioned parameters. In the bit timing logic, the conversion of the parameters to the required bit timing is materialized. The configuration registers BCR1 and BCR2 contain the data regarding bit timing.

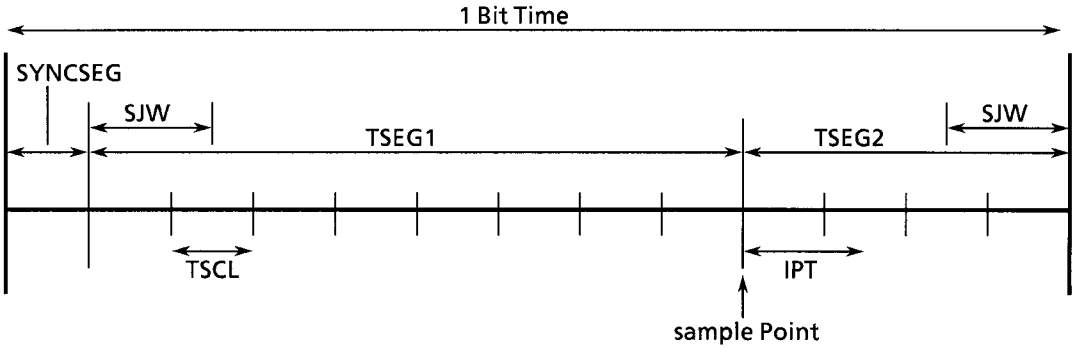


Figure 3.11.3 Bit Timing

The length of TSCL is defined by:

TSCL = (<BRP7:0> + 1) /f_{SYS} (f_{SYS} = external clock divided by 2)

f_{SYS} is used to the CAN controller system clock frequency (input clock of the CAN controller).

The length of one bit is determined by the equation below:

1 Bit Time = SYNCSEG + TSEG1 + TSEG2

1 bit time is equal to or greater than 10 × f_{SYS}

The synchronization segment SYNCSEG has always the length of × TSCL.

The length of TSEG1 should be equal to or greater than the length of TSEG2.

TSEG1 ≥ TSEG2.

The baud rate is defined by:

Baud rate = f_{SYS} ÷ [(<BRP7:0> +1 × ((<TSEG13:10> + 1) + (<TSEG22:20> + 1) + 1))

IPT (information processing time) is the time segment starting with the sample point reserved for processing of the sampled bit level. IPT is equal to 4 f_{SYS} clock cycles.

The parameter SJW (2bit) indicates by how many units of TSCL it is possible to lengthen or to shorten when re-synchronizing. Values between 1 (SJW = 00b) and 4 (SJW = 11b) are adjustable. The bus line is re-synchronized at each falling edge. The maximum length of SJW is equal to the length of TSEG2.

SJW ≤ TSEG2

With the corresponding bit timing, it is possible to reach a multiple sampling of the bus line at the sample point by setting <SAM> bit. The level determined by the CAN bus then corresponds to the result from the majority decision of the last three values. The three-time sampling is not allowed for <BRP7:0> < 4. For <BRP7:0> < 4 a one-time sampling will always be performed regardless of the value of <SAM> bit.

Restrictions are as follows:

<BRP7:0>	TSCL length (CAN clock cycles: fsys)	IPT length (CAN clock cycles: fsys)	TSEG2 minimum length (TSCL)
0	1	4	4
1	2	4	2
>1	<BPR7:0>+1	4	2

Example for setting baud rate

External clock = 24MHz
Internal system clock fsys=12MHz

CAN input clock = fsys
1TSCL=(<BRP7:0>+1)÷fsys

(1)1Mbps

<BRP7:0>	TSCL	<TSEG13:10>	<TSEG22:20>	Sample Point(%)
00h	12	0110b (7TSCL)	011b (4TSCL)	66.7
		0101b (6TSCL)	100b (5TSCL)	58.3
01h	6	0010b (3TSCL)	001b (2TSCL)	66.7

(2) 500kbps

<BRP7:0>	TSCL	<TSEG13:10>	<TSEG22:20>	Sample Point(%)
00h	24	1111b (16TSCL)	110b (7TSCL)	70.8
		1110b (15TSCL)	111b (8TSCL)	66.7
01h	12	1000b (9TSCL)	001b (2TSCL)	83.3
		0111b (8TSCL)	010b (3TSCL)	75.0
		0110b (7TSCL)	011b (4TSCL)	66.7
		0101b (6TSCL)	100b (5TSCL)	58.3
02h	8	0100b (5TSCL)	001b (2TSCL)	75.0
		0011b (4TSCL)	010b (3TSCL)	62.5
03h	6	0010b (3TSCL)	001b (2TSCL)	66.7

(3)250kbps

<BRP7:0>	TSCL	<TSEG13:10>	<TSEG22:20>	Sample Point(%)
01h	24	1111b (16TSCL)	110b (7TSCL)	70.8
		1110b (15TSCL)	111b (8TSCL)	66.7
02h	16	1100b (13TSCL)	001b (2TSCL)	87.5
		1011b (12TSCL)	010b (3TSCL)	81.3
		1010b (11TSCL)	011b (4TSCL)	75.0
		1001b (10TSCL)	100b (5TSCL)	68.8
		1000b (9TSCL)	101b (6TSCL)	62.5
		0111b (8TSCL)	110b (7TSCL)	56.3
03h	12	1000b (9TSCL)	001b (2TSCL)	83.3
		0111b (8TSCL)	010b (3TSCL)	75.0
		0110b (7TSCL)	011b (4TSCL)	66.7
		0101b (6TSCL)	100b (5TSCL)	58.3
05h	8	0100b (5TSCL)	001b (2TSCL)	75.0
		0011b (4TSCL)	010b (3TSCL)	62.5
07h	6	0010b (3TSCL)	001b (2TSCL)	66.7

(4)125kbps

<BRP7:0>	TSCL	<TSEG13:10>	<TSEG22:20>	Sample Point(%)
03h	24	1111b (16TSCL)	110b (7TSCL)	70.8
		1110b (15TSCL)	111b (8TSCL)	66.7
05h	16	1100b (13TSCL)	001b (2TSCL)	87.5
		1011b (12TSCL)	010b (3TSCL)	81.3
		1010b (11TSCL)	011b (4TSCL)	75.0
		1001b (10TSCL)	100b (5TSCL)	68.8
		1000b (9TSCL)	101b (6TSCL)	62.5
		0111b (8TSCL)	110b (7TSCL)	56.3
07h	12	1000b (9TSCL)	101b (2TSCL)	83.3
		0111b (8TSCL)	010b (3TSCL)	75.0
		0110b (7TSCL)	011b (4TSCL)	66.7
		0101b (6TSCL)	100b (5TSCL)	58.3
0Bh	8	0100b (5TSCL)	001b (2TSCL)	75.0
		0011b (4TSCL)	010b (3TSCL)	62.5
0Fh	6	0010b (3TSCL)	001b (2TSCL)	66.7

Example:

A transmission rate of 1 Mbps will be adjusted, i.e. a bit has a length of 1 µs. The CAN input clock frequency f_{SYS} is 12 MHz. The baud rate prescaler is set to 0. That means a bit for this data transmission rate has to be programmed with a length of 12 × TSCL.

E.g. <BRP7:0> = 00H
<TSEG13:10> = 0101B (6 × TSCL)
<TSEG22:20> = 100B (5 × TSCL)

With this setting a threefold sampling of the bus is not possible (<BRP7:0> < 4), thus SAM = 0 should be set. SJW is not allowed to be greater than TSEG2, so the maximum value could be set to <SJW1:0> = 11B (4 × TSCL)

Time stamp feature

There is a free-running 16-bit time stamp counter TSC implemented in the CAN controller to get an indication of the time of reception or transmission of messages. The content of the TSC is written into the time stamp value TSV of the corresponding mailbox when a received message has been stored or a message has been transmitted.

The TSC is driven from the bit clock of the CAN bus line. When the CAN controller is in configuration mode or sleep mode, the TSC will be stopped. After a reset, the TSC can be cleared by writing a value to the time stamp counter prescaler TSP. The TSC can be written and read by CPU in configuration mode and in normal operation mode.

Time stamp counter register (TSC)

Time Stamp Counter Register Low

TSCL (2332H) RMW instructions prohibited		7	6	5	4	3	2	1	0
	bit Symbol	TSC7	TSC6	TSC5	TSC4	TSC3	TSC2	TSC1	TSC0
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0

Time Stamp Counter Register High

TSCH (2333H) RMW instructions prohibited		15	14	13	12	11	10	9	8
	bit Symbol	TSC15	TSC14	TSC13	TSC12	TSC11	TSC10	TSC9	TSC8
	Read/Write	R/W							
	After reset	0	0	0	0	0	0	0	0

Overflow of the TSC can be detected by the time stamp counter overflow flag <TSO> of the global status register GSR and the time stamp counter overflow interrupt flag <TSOIF> of the global interrupt flag register GIF. Both flags are cleared by writing a "1" to the corresponding bit location in GIF.

There is a 4-bit prescaler for the TSC. It is the time stamp counter prescaler register TSP that stores the value to be reloaded into this prescaler. After reset, the TSP register is set to 0, so a value 0 is loaded into the prescaler. The TSC counter's count-up period, TTSC, is shown below:

TTSC = TBIT × (<TSP3:0> + 1) (TBIT: bit cycle)

Time stamp counter prescaler register (TSP)

Time Stamp Counter Prescaler Register Low

TSPL (2330H)	7	6	5	4	3	2	1	0
bit Symbol					TSP3	TSP2	TSP1	TSP0
Read/Write					R/W			
After reset					0	0	0	0

Time Stamp Counter Prescaler Register High

TSPH (2331H)	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write								
After reset								

To ensure that the value of the TSC will not change during the write cycle to the mailbox, a hold register is implemented. The value of the TSC will be copied to this register if a message has been received or transmitted successfully. The reception is successful for the receiver if there is no error until the last but one bit of End-of-frame. The transmission is successful for the transmitter if there is no error until the last bit of End-of-frame. (Refer to the CAN version 2.0B)

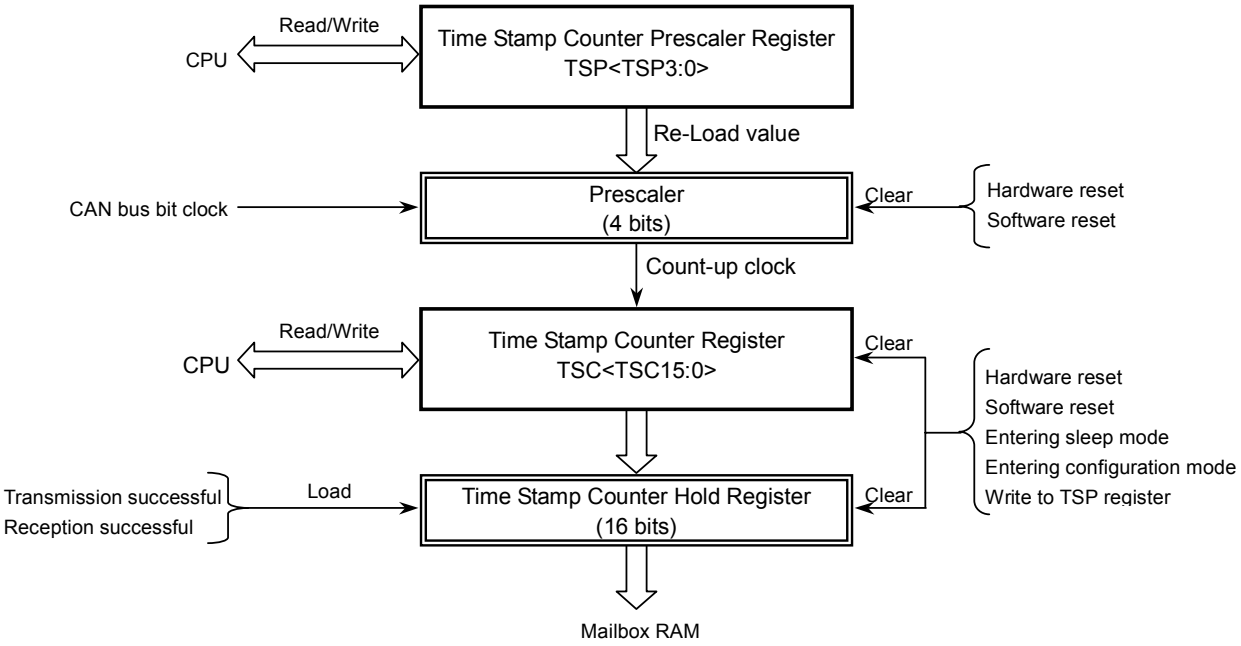


Figure 3.11.4 Time Stamp Counter

The free running time stamp counter and the time stamp hold register will be cleared in the following cases:

- After reset (hardware reset or software reset)
- When the module enters configuration mode
- When the module enters sleep mode
- When a write access to the time stamp prescaler register is performed

(7) Status registers

Global status register (GSR)

Global Status Register Low

GSRL (231AH)

	7	6	5	4	3	2	1	0
bit Symbol	CCE	SMA	HMA		TSO	BO	EP	EW
Read/Write	R				R			
After reset	1	0	0		0	0	0	0

Global Status Register High

GSRH (231BH)

	15	14	13	12	11	10	9	8
bit Symbol	MsgInSlot <3:0>				RM	TM		
Read/Write	R							
After reset	1	1	1	1	0	0		

MsgInSlot: Message In Slot

Indicates a message in the transmission slot.

0000: Message of mailbox 0
0001: Message of mailbox 1
:
1110: Message of mailbox 14
1111: No transmission message

RM: Receive Mode

0: The CAN controller is not receiving a message.
1: The CAN controller is receiving a message. That means the CAN controller is not the transmitter of the message and the bus is not idle.

TM: Transmit Mode

0: The CAN controller is not transmitting a message.
1: The CAN controller is transmitting a message. That means the CAN controller stays transmitter until the bus is idle or it loses arbitration.

CCE: Change Configuration Enable

0: The CAN controller is not in the configuration mode. (Normal operation)
1: The CAN controller has entered the configuration mode.

SMA: Sleep Mode Acknowledge

0: The CAN controller is not in the sleep mode. (Normal operation)
1: The CAN controller has entered the sleep mode.

HMA: Halt Mode Acknowledge

0: The CAN controller is not in the halt mode. (Normal operation)
1: The CAN controller has entered the halt mode.

TSO: Time Stamp Overflow Flag

0: There was no overflow of the time stamp counter.
1: There was at least one overflow of the time stamp counter since this bit has been cleared.
To clear this bit, clear the <TSOIF> bit in the GIF register.

BO: Bus-Off Status

0: The CAN controller is in the bus-on status. (Normal operation)

1: The CAN controller is in the bus-off status.
There is an abnormal rate of occurrences of errors on the CAN bus. This condition occurs when the transmit error counter TEC has reached the limit of 256. During bus-off no messages can be received or transmitted. The CAN controller will go to bus-on automatically after the bus-off recovery sequence. After entering bus-off, the error counters are undefined.

EP: Error Passive Status

0: The CAN controller is in the error active mode.
The values of both transmit error counter TEC and receive error counter REC are less than 128.

1: The CAN controller is in the error passive mode.
Either one of or both the transmit error counter TEC and receive error counter REC have reached the error passive status of 128.

EW: Warning Status

0: Both values of the error counters TEC and REC are less than or equal to 96.

1: At least one of the error counters is greater than 96 and reached the warning level.

CAN error counter register (CEC)

CAN Error Counter Register Low

CECL (232EH) RMW instructions prohibited

	7	6	5	4	3	2	1	0
bit Symbol	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

CAN Error Counter Register High

CECH (232FH) RMW instructions prohibited

	15	14	13	12	11	10	9	8
bit Symbol	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

The CAN controller contains two error counters: receive error counter REC and transmit error counter TEC. The values of both counters can be read by the CPU. A write access to the error counters is only possible in the test error mode, at the same time as and with the same value of the lower 8bit (<TSTERR> bit in MCR register is set). These error counters are incremented or decremented according to the CAN version 2.0B.

The controller enters the following three states depending on the values of REC and TEC.

(1) Error active state (TEC < 128 and REC < 128)
The state where an error rarely occurs.
The CAN controller is in an error active state after reset release.
When an error is detected, an active error flag is transmitted.

(2) Error passive state (TEC ≥ 128 or REC ≥ 128)
The state where many errors have occurred.
When an error is detected, a passive error flag is transmitted.

(3) Bus-off state (TEC ≥ 256)
The CAN controller cannot perform message transmission to and reception from the CAN bus.

Receive error counter REC is not incremented after exceeding the error passive limit (128). After the correct reception of a message when REC = 128, the counter is set to a value between 119 and 127. After reaching the bus-off status, the counts are undefined.

A CAN controller which is in bus-off state will automatically enter error active state if 11 continuous recessive bits are detected 128 times on the CAN bus .

All internal flags are reset, and the error counters are cleared. The configuration registers keep the programmed values. The values of the error counters are undefined during bus-off status.

When the CAN controller enters configuration mode (see 3.11.4 (1) Configuration mode), the error counters will be cleared.

(8) Interrupt control registers

The CAN controller has the following interrupt sources:

- Transmit interrupt
  When a message has been transmitted successfully
- Receive interrupt
  When a message has been received successfully
- Remote frame pending interrupt
  When a remote frame is received
- Wake-up interrupt
  When the CAN controller is awakened from sleep mode
- Receive message lost interrupt
  When a receive message is lost
- Time stamp counter overflow interrupt
  When the time stamp counter has overflowed
- Bus off interrupt
  When the CAN controller enters the bus-off mode
- Error passive interrupt
  When the CAN controller enters the error passive mode
- Warning level interrupt
  When at least one of the two error counters is greater than 96 and has reached the warning level.

These interrupt sources are divided into three groups:

- Receive interrupt (INTCR)
- Transmit interrupt (INTCT)
- Global interrupt (INTCG)

There is one interrupt output line for each group. INTCR is dedicated to receive interrupts, INTCT is dedicated to transmit interrupts and INTCG to the global interrupts.

Global interrupt flag register (GIF)

Global Interrupt Flag Register Low

GIFL (2320H) RMW instructions prohibited	7	6	5	4	3	2	1	0
bit Symbol	RFPF	WUIF	RMLIF	—	TSOIF	BOIF	EPIF	WLIF
Read/Write	R/C							
After reset	0	0	0	—	0	0	0	0

Global Interrupt Flag Register High

GIFH (2321H) RMW instructions prohibited	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write								
After reset								

The interrupt flag bits will be set if the corresponding interrupt condition has occurred. If the corresponding interrupt mask bit is set in the GIM register, an interrupt pulse on the global interrupt line INTCG will be generated. As long as an interrupt flag in the GIF register is set, if the corresponding interrupt source generates a new interrupt event, a new interrupt pulse on INTCG will not be generated. If an interrupt flag in the GIF register is set and another interrupt source generates an interrupt event, then a new interrupt pulse on INTCG will be generated.

If one or more interrupt flags have been cleared and one or more interrupt flags are still set, a new global interrupt pulse INTCG will be generated.

The interrupt flags will be cleared by writing a “1” to the corresponding bit location.

RFPF: Remote Frame Pending Flag

0: No remote frame has been received.
1: A remote frame has been received (in a receive-mailbox).
This bit will not be set if the identifier of the remote frame matches to a transmit-mailbox with <RFH> set.

WUIF: Wake-Up Interrupt Flag

0: The CAN controller is in the sleep mode or the normal operation mode.
1: The CAN controller has left the sleep mode.

RMLIF: Receive Message Lost Interrupt Flag

0: No receive message has been lost.
1: For at least one of the receive-mailboxes, a receive message has been lost.
At least one of the bits in the RML register is set.

TSOIF: Time Stamp Counter Overflow Interrupt Flag

0: There have been no overflows of the time stamp counter since this bit has been cleared.
1: There was at least one overflow of the time stamp counter since this bit has been cleared.

BOIF: Bus-Off Interrupt Flag

0: The CAN controller is still in the bus-on mode.
1: The CAN controller has entered the bus-off mode.

EPIF: Error Passive Interrupt Flag

0: The CAN controller is still in error active mode.
1: The CAN controller has entered the error passive mode.

WLIF: Warning Level Interrupt Flag

0: None of the error counters has reached the warning level.
1: At least one of the error counters has reached the warning level.

Global interrupt mask register (GIM)

Global Interrupt Mask Register Low

GIML (2322H)

	7	6	5	4	3	2	1	0
bit Symbol	RFPM	WUIM	RMLIM	—	TSOIM	BOIM	EPIM	WLIM
Read/Write	R/W							
After reset	0	0	0	—Note)	0	0	0	0

Global Interrupt Mask Register High

GIMH (2323H)

	15	14	13	12	11	10	9	8
bit Symbol								
Read/Write								
After reset								

Note: Write to 0

Each interrupt flag bit in the GIF register is masked by the corresponding mask bit in the GIM register.

If a bit in the GIM register is 0, the interrupt generation for the corresponding global interrupt event is disabled, and if it is 1, the interrupt generation is enabled. After reset, all bits in the GIM register are cleared, thereby disabling global interrupt.

Mailbox interrupts

Separate interrupt outputs are provided for mailbox interrupts independently of global interrupts. These include mailbox transmit interrupt INTCT, and mailbox receive interrupt INTCR, that depend on mailbox settings. A mailbox transmit interrupt flag register MBTIF is provided for mailbox transmit interrupts, and a mailbox receive interrupt flag register MBRIF is provided for mailbox receive interrupts.

In addition, there is a mailbox interrupt mask register MBIM that enables or disables each mailbox interrupt.

Mailbox interrupt mask register (MBIM)

Mailbox Interrupt Mask Register Low

MBIML (2328H)

	7	6	5	4	3	2	1	0
bit Symbol	MBIM7	MBIM6	MBIM5	MBIM4	MBIM3	MBIM2	MBIM1	MBIM0
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Mailbox Interrupt Mask Register High

MBIMH (2329H)

	15	14	13	12	11	10	9	8
bit Symbol	MBIM15	MBIM14	MBIM13	MBIM12	MBIM11	MBIM10	MBIM9	MBIM8
Read/Write	R/W							
After reset	0	0	0	0	0	0	0	0

Each bit corresponds to mailboxes 0 through 15.

The MBIM register settings determine whether to enable or disable each mailbox interrupt.

If a bit in the MBIM register is 0, the interrupt generation for the corresponding mailbox is disabled.

If a bit in the MBIM register is 1, the interrupt generation for the corresponding mailbox is enabled.

Mailbox transmit interrupt flag register (MBTIF)

Mailbox Transmit Interrupt Flag Register Low

MBTIFL (2324H) RMW instructions prohibited		7	6	5	4	3	2	1	0
	bit Symbol	MBTIF7	MBTIF6	MBTIF5	MBTIF4	MBTIF3	MBTIF2	MBTIF1	MBTIF0
	Read/Write	R/C							
	After reset	0	0	0	0	0	0	0	0

Mailbox Transmit Interrupt Flag Register High

MBTIFH (2325H) RMW instructions prohibited		15	14	13	12	11	10	9	8
	bit Symbol		MBTIF14	MBTIF13	MBTIF12	MBTIF11	MBTIF10	MBTIF9	MBTIF8
	Read/Write		R/C						
	After reset		0	0	0	0	0	0	0

This register is provided for mailbox transmit interrupts. Each bit in this register corresponds to mailboxes 0 through 15. The interrupt flag for mailbox 15, the <MBTIF15> flag, is nonexistent because mailbox 15 is the receive-only mailbox. If mailbox "n" is set for receive, the corresponding interrupt flag in this register, the <MBTIFn> flag, will always be read as 0.

If a message in mailbox "n" has been transmitted successfully and the mask bit <MBIMn> is set to 1, the corresponding transmit interrupt flag <MBTIFn> will be set. If no other bit was set before in the MBTIF register, transmit interrupt pulse INTCT will be generated.

If, for any mailbox, the mask bit in the MBIM register is 0, the transmit interrupt flag in the MBTIF register will not be set and no transmit interrupt pulse INTCT will be generated. Information about a successful transmission can be read from the TA register.

If one or more transmit interrupt flags have been set in the MBTIF register and another interrupt condition has occurred, no interrupt will be generated, but the corresponding flag in the MBTIF register will be set.

If there one or more transmit interrupt flags are set after clearing one or more transmit interrupt flags, another mailbox transmit interrupt pulse INTCT will be generated.

The interrupt flags in the MBTIF register will be cleared by writing a 1 from the CPU to the MBTIF register. Writing a 0 has no effect. The corresponding status flags in the TA register must be cleared separately.

Note that interrupt flags in the MBTIF register must be confirmed as 1 (active), before clearing.

Mailbox receive interrupt flag register (MBRIF)

Mailbox Receive Interrupt Flag Register Low

MBRIFL (2326H) RMW instructions prohibited	7	6	5	4	3	2	1	0
bit Symbol	MBRIF7	MBRIF6	MBRIF5	MBRIF4	MBRIF3	MBRIF2	MBRIF1	MBRIF0
Read/Write	R/C							
After reset	0	0	0	0	0	0	0	0

Mailbox Receive Interrupt Flag Register High

MBRIFH (2327H) RMW instructions prohibited	15	14	13	12	11	10	9	8
bit Symbol	MBRIF15	MBRIF14	MBRIF13	MBRIF12	MBRIF11	MBRIF10	MBRIF9	MBRIF8
Read/Write	R/C							
After reset	0	0	0	0	0	0	0	0

This register is provided for mailbox receive interrupts. Each bit in this register corresponds to mailboxes 0 through 15. If mailbox "n" is set for transmit, the corresponding interrupt flag in this register, the <MBRIFn> flag, will always be read as 0.

If a message in mailbox "n" has been received successfully and the mask bit <MBIMn> is set to 1, the corresponding receive interrupt flag <MBRIFn> will be set. If no other bit was set before in MBRIF register, receive interrupt pulse INTCR will be generated.

If for a mailbox the mask bit in MBIM register is 0, the receive interrupt flag in MBRIF register will not be set and no receive interrupt pulse INTCR will be generated. The information about a successful reception could be read from the RMP register respectively.

If one or more receive interrupt flags have been set in MBRIF register and another interrupt condition has been occurred, no interrupt will be generated, but the corresponding flag in MBRIF register will be set.

If there is one or more receive interrupt flags set after clearing one or more receive interrupt flags, another mailbox receive interrupt pulse INTCR will be generated.

The interrupt flags in MBRIF register will be cleared by writing a 1 from the CPU to MBRIF register. Writing a 0 has no effect. The corresponding status flags in RMP register have to be cleared separately.

Note that interrupt flags in the MBRIF register must be confirmed as 1 (active), before clearing.

3.11.4 Description of Mode

(1) Configuration mode

The CAN controller must be initialized (set the bit configuration registers BCR1 and BDR2) before activation. The BCR1 and BCR2 registers can only be modified when the module is in the configuration mode. After reset, the configuration mode is active and the <CCR> bit of the MCR register and the <CCE> bit of the GSR register are set to 1. The CAN controller can be set to the normal operation mode by writing a 0 to the <CCR> bit. After leaving the configuration mode, the <CCE> bit will be set to 0 and the power-up sequence will start. The power-up sequence consists of detecting eleven consecutive recessive bits on the CAN bus line. After the power-up sequence, the CAN controller is bus-on and ready for operation.

When the <CCR> bit is set to 1, the CAN controller will enter the configuration mode from the normal operation mode. After the CAN controller has entered the configuration mode, the <CCE> bit will be set to 1. See also the flowchart in Figure 3.11.5 (Flowchart of CAN Initialization). On entering the configuration mode, the error counter CEC, the time stamp counter TSC and the time stamp hold register will be cleared.

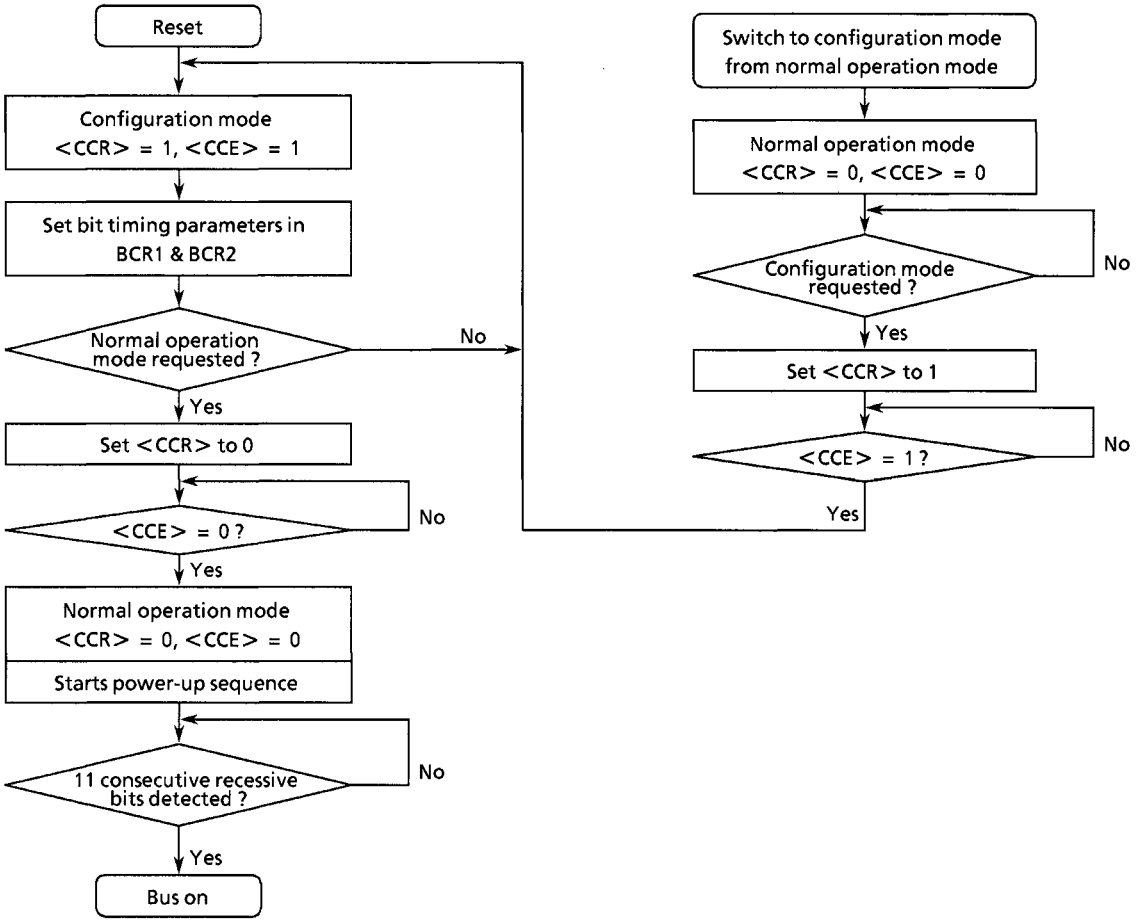


Figure3.11.5 Flowchart of CAN Initialization

(2) Sleep mode

The sleep mode will be requested by writing 1 to the <SMR> bit of the MCR register. When the CAN controller enters the sleep mode, the status bit <SMA> of the GSR register will be set to 1.

During the sleep mode, the clock of the CAN controller is switched off. Only the wake up logic will be active. The read value of the GSR register will be F040H, this means there is no message in transmit buffer and the sleep mode is active (the <SMA> bit is set to 1). Read accesses to all other registers will deliver the value 0000H. Write accesses to all registers other than the MCR register will be denied.

The CAN controller leaves the sleep mode if a write access to the MCR register has been detected or there is any bus activity detected on the CAN bus line (with <WUBA> = 1). The CAN controller then begins its power-up sequence. The CAN controller waits until detecting 11 consecutive recessive bits on the RX input line and goes to bus active after them. The first message that initiates the bus activity cannot be received.

In sleep mode, the CAN error counters and all transmission request set bits <TRSn> will be cleared. After leaving the sleep mode, the <SMR> bit in the MCR register and the <SMA> bit in the GSR register will be cleared.

If the CAN controller is transmitting a message when the <SMR> bit is set, the CAN controller will not switch to the sleep mode immediately. It will continue until a successful transmission or after losing arbitration, or until a successful reception or an error condition occurs on the CAN bus line. By this means, the CAN controller will initiate no error condition on the CAN bus line.

(3) Halt mode

The halt mode will be requested by writing 1 to the <HMR> bit of the MCR register. When the CAN controller enters the halt mode, the <HMA> bit of the GSR register will be set. During the halt mode the CAN controller does not send or receive any messages. The CAN controller is still active on the CAN bus line. Error Flags and Acknowledge Flags will be sent. The CAN controller leaves the halt mode if the <HMR> bit is reset to 0.

If the CAN controller is transmitting a message when the <HMR> bit is set, the transmission will continue until successful , or until a lost arbitration is detected. By this means the CAN controller will initiate no error condition on the CAN bus line.

(4) Test loopback mode

In this mode, the CAN controller can receive its own transmitted message and will generate its own acknowledge bit. No other CAN controller is necessary for this operation. The only supposition is that the RX and TX lines must be connected to a CAN bus transceiver or directly together.

In the test loopback mode, the CAN controller can transmit a message from one mailbox and receive it in another mailbox. The set-up for the mailboxes is the same as in the normal operation mode.

The test loopback mode can only be enabled or disabled in the configuration mode. Figure 3.11.6 shows the flowchart of the test loopback mode and the test error mode set-up.

(5) Test error mode

The error counters can only be written when the CAN controller is in the test error mode.

When the CAN controller is in the test error mode, both error counters will be written at the same time with the same value (lower 8 bits). The maximum value that can be written into the error counters is 255. Thus, the error counter value of 256 which forces the CAN controller into the bus-off mode can not be written into the error counters.

The test error mode can only be enabled or disabled in the configuration mode. Figure 3.11.6 shows the flowchart of the test loopback mode and the test error mode set-up.

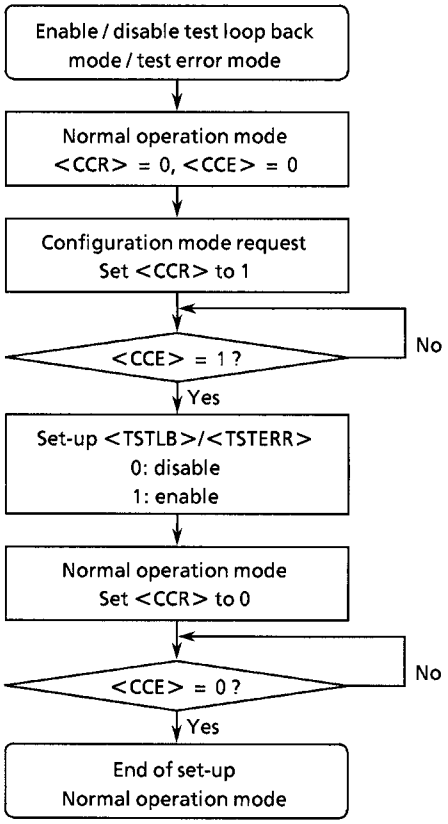


Figure3.11.6 Flowchart of the test loopback mode/ test error mode set-up

3.11.5 Functional Description

(1) Transmit mode

Figure 3.11.7 shows the flowchart of message transmit using the transmit interrupt INTCT.

It is also possible to use polling instead of interrupt. In this case, "Transmit interrupt generated? " is replaced by "<TAn> = 1? ". "Set <MBIMn> to 1" and "Clear <MBTIFn>" must be removed from the flow.

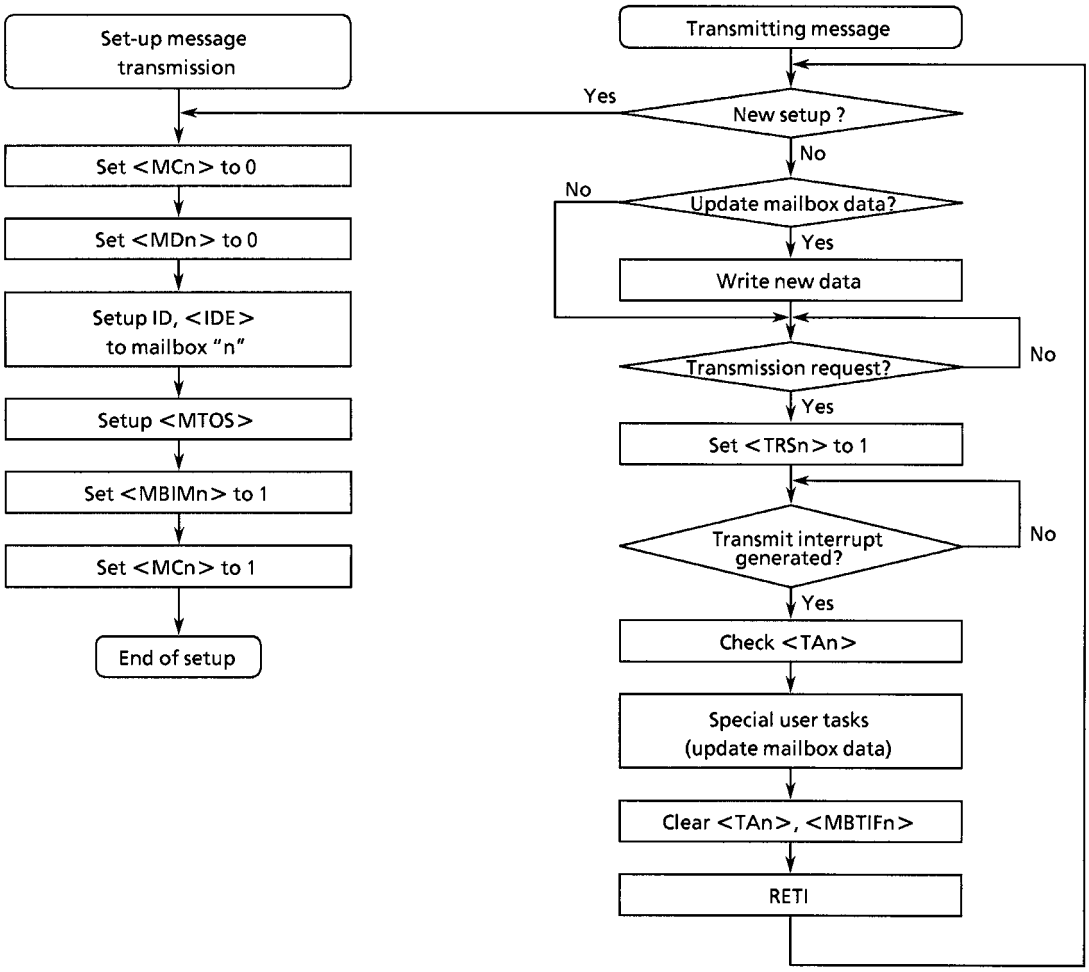


Figure 3.11.7 Flowchart of message transmission

(2) Receive mode

If the CAN controller has received a message from the CAN bus line, this message will be located in the receive buffer. The identifier of the message stored in the receive buffer will be compared to the identifier of the mailbox. If <GAME>/<LAME> bit is set, the global/local acceptance mask register GAM/LAM will be used. If there is one of the following conditions found, no further compare will be performed.

- Data frame and a matching identifier in a mailbox configured as receive
- Remote frame and a matching identifier in a mailbox configured as receive
- Remote frame and a matching identifier in a mailbox configured as transmit and <RFH> bit is set

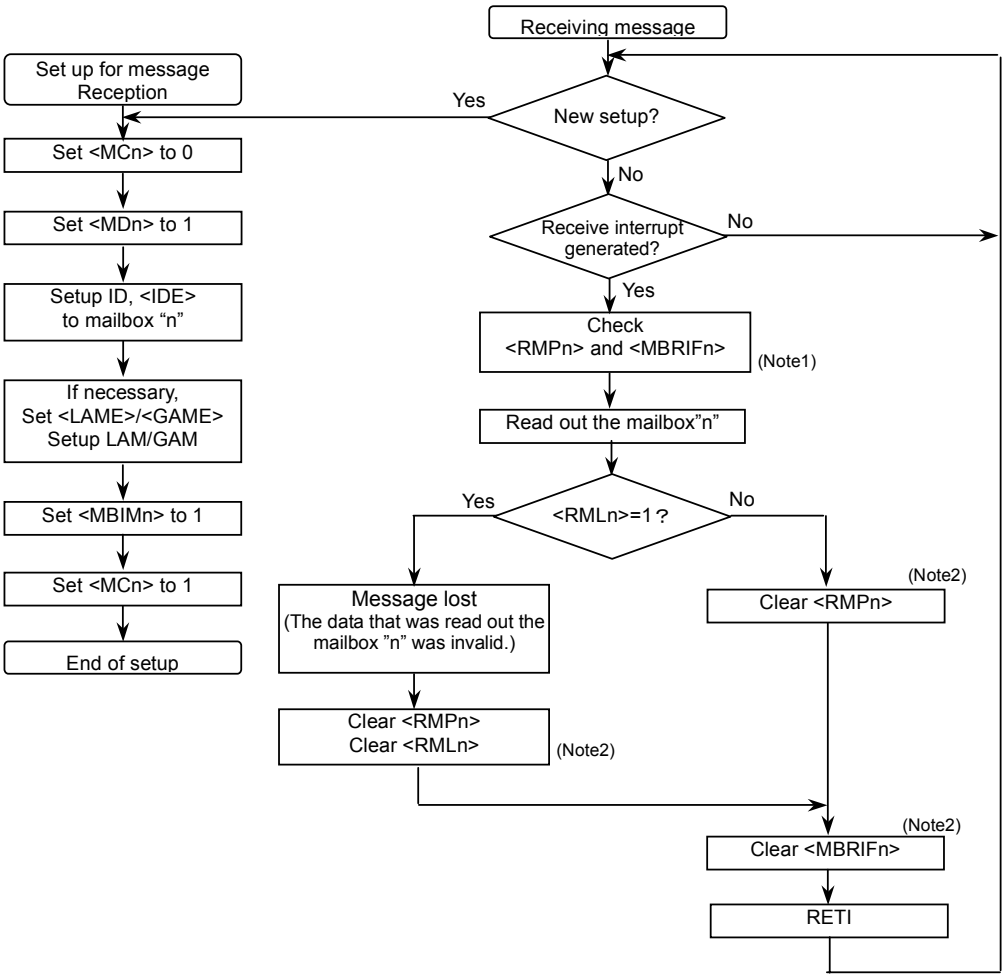
The minimal time to save a next received message after the <RMP> bit set depends on the configured bit timing. In the case of the data length code = 0, the minimal time is as follows.

- Standard format: 47 bit times - 16 f_{SYS}
- Extended format: 67 bit times - 16 f_{SYS}

[1] Data frames

Figure 3.11.8 shows one example of the flowchart of message reception using the receive interrupt INTCR.

It is also possible to use polling instead of the interrupt. In this case, "Receive interrupt generated? " is replaced by "<RMPn> = 1? ". "Set <MBIMn> to 1" and "Clear <MBRIFn>" must be removed from the flow.



Note1: Be sure to check <RMPn> and <MBRIFn>

Note2: If "Clear <RMPn>" is executed, and mailbox "n" receives a message before "Clear <MBRIFn>" is also executed, then it is possible that <RMPn> will be set at 1 (<MBRIFn>=0).

Figure 3.11.8 Flowchart of message reception (example)

[2] Remote frame

Figure 3.11.9 shows the flowchart of one example of the handling of remote frames by using the automatic reply feature. This feature is available when the <RFH> bit of a mailbox configured for transmission is set. To avoid data inconsistency problems when updating the mailbox data, the CDR register is used.

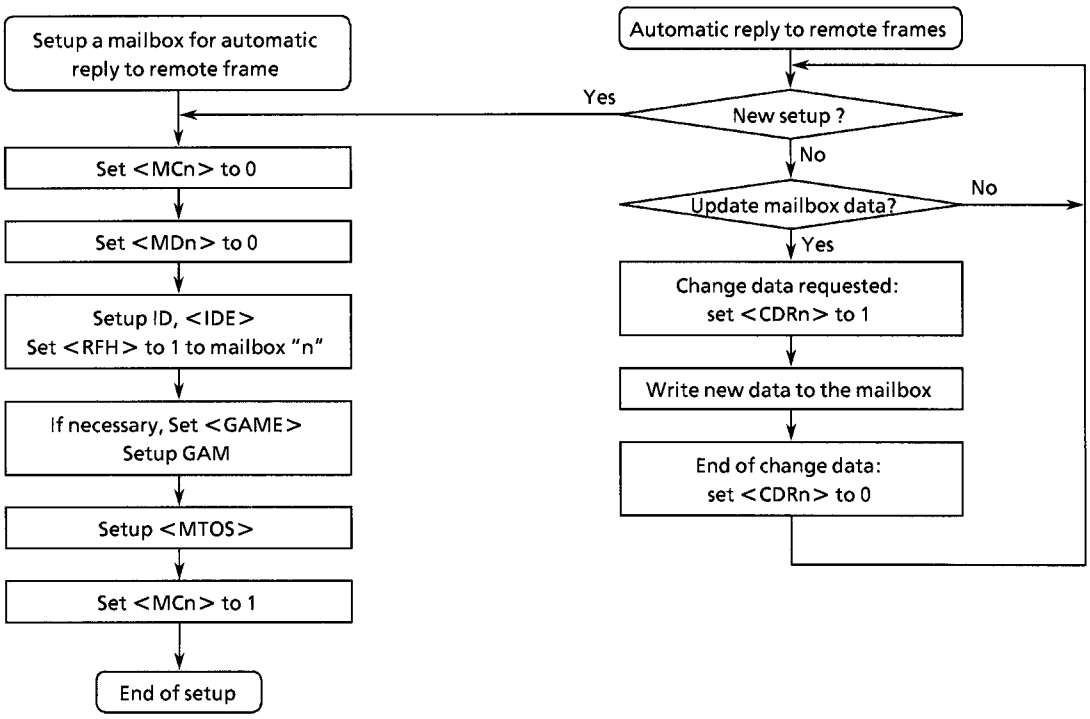


Figure 3.11.9 Flowchart of remote frame handling with the automatic reply feature (example)

3.12 Serial Expansion Interface (SEI)

3.12.1 Overview

The SEI is one of the serial interfaces built into the TMP95CU54A, which can be connected to peripheral devices, by full duplex synchronous communication protocol. The TMP95CU54A incorporates one channel of this serial expansion interface.

The SEI can also support the Micro DMA mode corresponding to the micro DMA transfer.

(1) Features

- The master outputs the shift clock only during data transfer
- The clock polarity and phase are programmable
- The data are 8 bits long
- Either MSB first or LSB first can be selected
- Micro DMA mode support for micro DMA transfers
- Transfer rate: 4 Mbps, 2 Mbps, 1 Mbps or 250 kbps (when operating at 24 MHz)
- Error detection function

  [1] Write collision detection: when writing to the shift register during data transfer

  [2] Overflow detection: when receiving new data while the transfer end flag is set (slave only)

  Note: There is no Mode fault detection function. Set P6FC<P60F> which is the enable / disable bit for Mode fault detection to "1", and disable the Mode fault detection function.

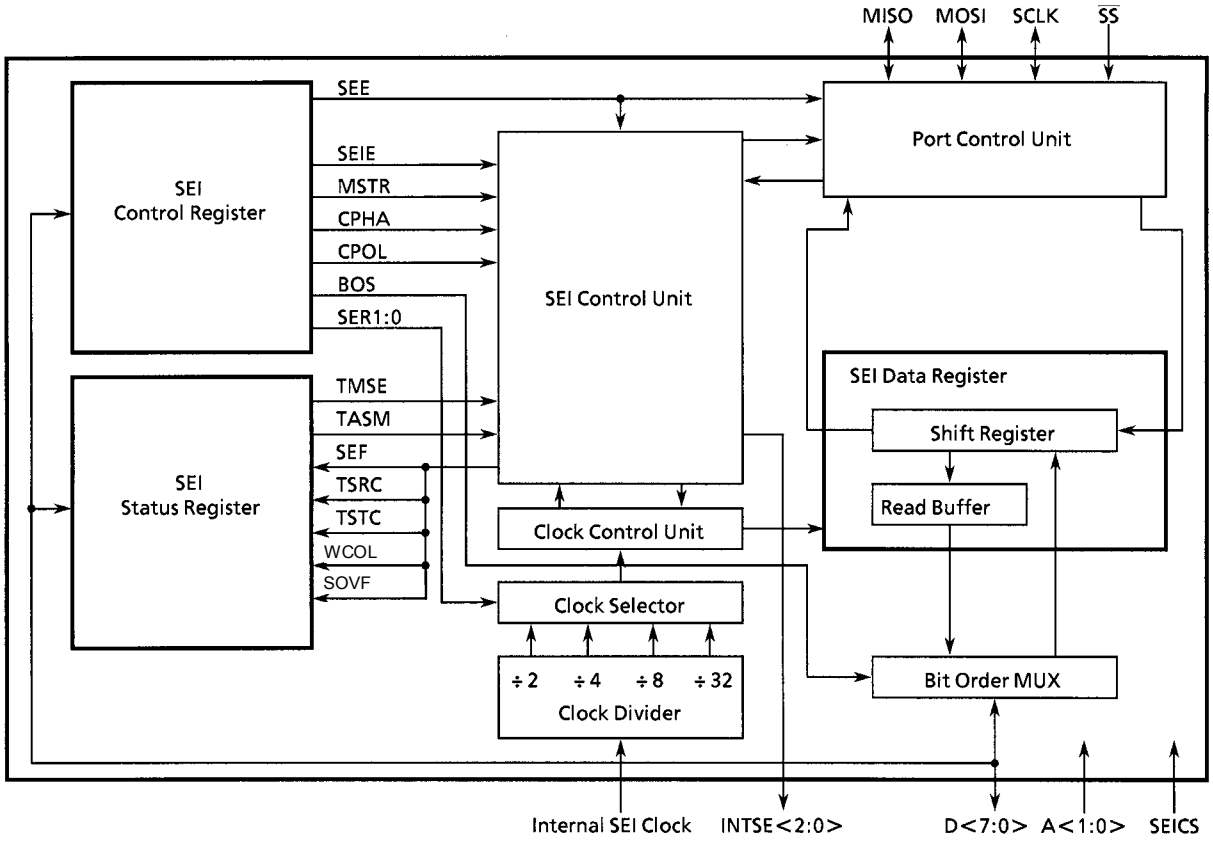


Figure 3.12.1 SEI Block Diagram

3.12.2 SEI Operation

During a SEI transfer, data is simultaneously transmitted (shifted out serially) and received serially (shifted in serially). The SEI clock (SECLK) takes synchronously the two serial data lines (MOSI/MISO) in order to shift and sample information on the lines. A slave selection line ($\overline{SS}$)selects slave devices individually. A slave device not selected cannot use the SEI bus.

(1) SEI clock phase and polarity controls

Software can select any four combinations of serial clock phase and polarity using two bits in the SEI control register (SECR). The clock polarity is set by the <CPOL> bit, and the clock is either active "H" or active "L". The clock phase <CPHA> control bit selects one of two fundamentally different transfer formats. The clock phase and polarity should be identical for the master SEI device and the communicating slave device.

(2) SEI data and clock timing

The SEI programmable clock timing and data can be connected to almost any synchronous serial device. Please see "3.12.4 SEI transfer format" for a detailed description of the transfer format.

3.12.3 SEI Signal Lines

There are four input/output pin signals associated with the SEI transfer. Every signal depends on the mode (master/slave) of the SEI device.

(1) SCLK

The SCLK pin functions as an output pin when the SEI is set for master, and functions as an input pin when the SEI is set for slave.

When the SEI is set for master, the SCLK signal is supplied by the internal SEI clock generation circuit. When the master starts transferring data, eight clock cycles are automatically output at the SCLK pin. When the SEI is set for slave, the SCLK pin functions as an input pin, in which case the SCLK signal from the master synchronizes data transfers between the master and slave. The slave device ignores the SCLK signal if the slave select $\overline{SS}$ pin is high.

In both master and slave SEI devices, data is shifted in or out at each rising or falling edge of the SCLK signal and is sampled at the opposite edge where the data is stable. Edge polarity is determined by the SEI transfer protocol.

(2) MISO/MOSI

The MISO and MOSI pins are used for transmitting and receiving serial data.

When the SEI is configured as a master, MISO is the data input line and MOSI is the data output line.

When the SEI is configured as a slave, these pins reverse roles.

All SCLK pins are connected together, as are all MOSI pins and all MISO pins. Refer to “Figure 3.12.5 Configuration of SEI system”. A single SEI device is configured as a master, while all other SEI devices on the SEI bus are configured as slaves. The single master drives the transfer clock and data out of its SCLK and MOSI pins to the SCLK and MOSI pins of the slaves. One selected slave device optionally drives data out of its MISO pin to the MISO master pin.

The SCLK, MISO and MOSI pins can be set up to function as programmable open-drain pins.

(3) $\overline{SS}$

The $\overline{SS}$ pin is used to enable the SEI slave for transfer and receive. If the $\overline{SS}$ pin of a slave is inactive (high), the device ignores SCLK clocks and keeps the MISO output pin in the high-impedance state.

3.12.4 SEI Transfer Format

The transfer format is determined by the setting of the <CPHA> bit and the <CPOL> bit in the SECR register. The <CPHA> bit switches between two different transfer protocols.

(1) Transfer Format of <CPHA> = 0

Figure 3.12.2 shows the transfer format for a <CPHA>=0 transfer.

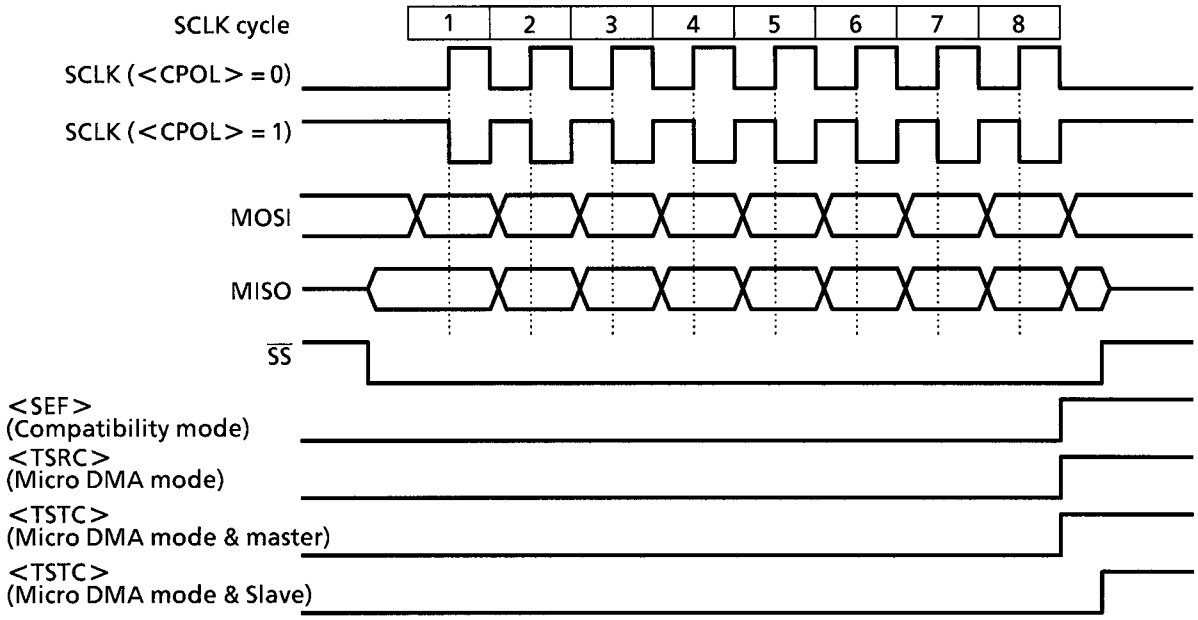


Figure 3.12.2 Transfer format of <CPHA>=0

In this transfer format, the first bit is sampled in on the first clock edge. This will be on a rising edge when <CPOL> = 0 and on a falling edge when <CPOL> = 1. With <CPOL> = 0 the shift clock will idle low, with <CPOL> = 1 it will idle high.

In master mode, when a transfer is initiated by writing new data to the SEDR register, the new data is placed on the MOSI pin for half a clock cycle before the shift clock starts to operate. The <BOS> bit in the SECR register determines whether the data will be shifted out in a MSB or LSB order. After the last shift cycle, the <SEF> flag (in Compatibility mode) or the <TSRC> flag and <TSTC> flags (in Micro DMA mode) will be asserted.

In slave mode, the SEDR register is not allowed to be written if the $\overline{SS}$ signal is low. A write attempt in this state will result in a write collision and the <WCOL> bit will be asserted in the SESR register. Therefore, even if the transfer has been completed and the <SEF> flag or <TSRC> flag bit has been asserted, software has to wait until the $\overline{SS}$ signal goes high again before writing new data to the SEDR register. To allow the use of a micro DMA to transfer data to the SEDR register in slave mode, the <TSTC> flag is delayed until $\overline{SS}$ goes high.

(2) Transfer format of <CPHA> = 1

Figure 3.12.3 shows the transfer format for a <CPHA> = 1 transfer.

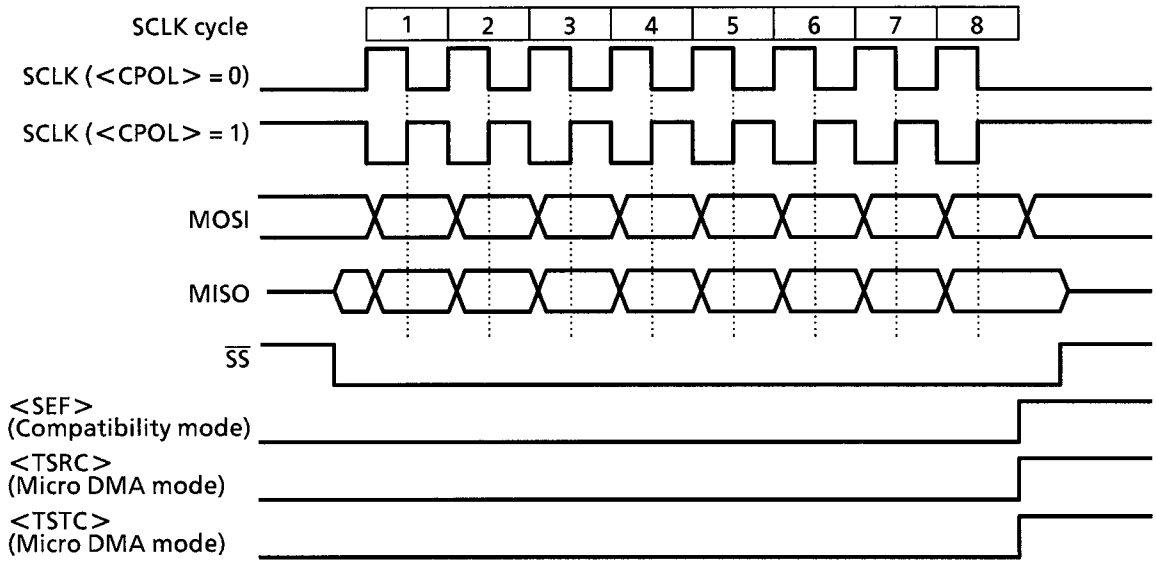


Figure 3.12.3 Transfer format of <CPHA> = 1

In this transfer format, the first bit is sampled in on the second clock edge. This will be on a falling edge when <CPOL> = 0 and on a rising edge when <CPOL> = 1. If <CPOL> = 0, the shift clock is a rising edge; with <CPOL> = 1 it is a falling edge.

In master mode, when a transfer is initiated by writing new data to the SEDR register, the data is placed on the MOSI pin with the first edge of the shift clock. Again, the first bit to be transferred will be determined by the <BOS> bit in the SECR register.

Unlike in the <CPHA> = 0 format, the SEDR register is allowed to be written in slave mode even if the $\overline{SS}$ signal is low.

In both master and slave mode, the <SEF> flag (in Compatibility mode) or the <TSRC> flag and <TSTC> flag (in Micro DMA mode) will be asserted simultaneously after the completion of the last shift cycle. An attempt to write the SEDR register while the data shifting is still in progress will result in a write collision.

3.12.5 Functional Description

Figure 3.12.4 shows master-to-slave connection via the SEI.

The different nodes on a SEI bus function like a distributed shift register. When data is sent from the MOSI pin of the master device to the corresponding pin of the slave device, data from the slave is sent back from the MISO pin of the slave device to the corresponding pin of the master device.

This means that data is communicated in full-duplex mode and data output and data input are synchronized by the same clock signal. After a transfer, the data transmitted from the 8-bit shift register is replaced with receive data.

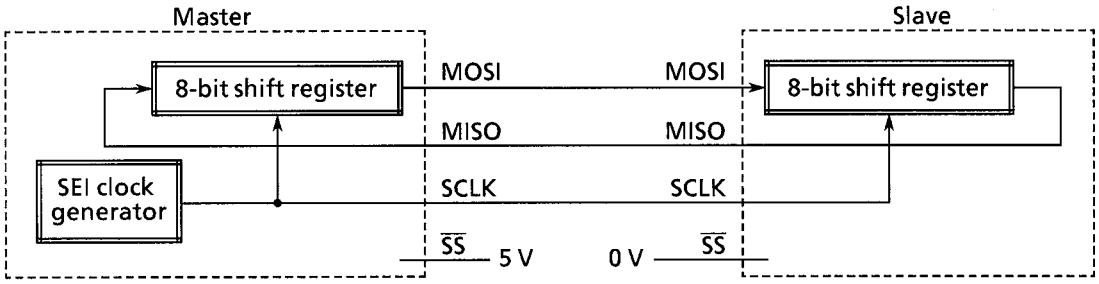


Figure 3.12.4 Connection between Master and Slave in SEI

Figure 3.12.5 shows a configuration of the SEI system.

Port 6, the SEI output, can be set for open-drain output programmable. Therefore, this port can be connected to multiple devices.

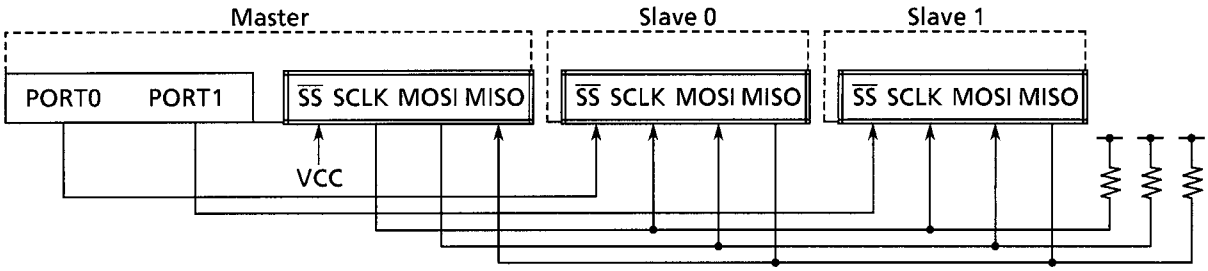


Figure 3.12.5 Configuration of SEI System (comprising one master and two slaves)

3.12.6 Operation Modes

SEI allows the programmer the choice of 2 fundamentally different operation modes - the Compatibility mode and the Micro DMA mode. These operation modes differ in terms of flag clearing, interrupt generation and whether or not - micro DMA is used. The table below shows the differences between the two operation modes.

Table 3.12.1 Differences between the Two Operation Modes

	Compatibility Mode	Micro DMA Mode
Error Flag Clearing	Reading a register with the Status flag set, followed by SECR or SEDR register access	Writing a “1” to the status register
Transfer Status Flag Clearing	Reading a register with the Status flag set, followed by an access to the data register	Writing a “1” to the status register or by reading or writing the data register
Interrupt generation	INTSE0: <SEF>	INTSE0: <WCOL> or <SOVF> INTSE1: <TSRC> INTSE2: <TSTC>
Micro DMA use	No	Yes

SEI can be switched between these operation modes, if SEI is disabled (<SEE> = 0), by setting the <TMSE> bit in the SESR register.

3.12.7 SEI Registers

Use SEI control register SECR, SEI status register SESR and SEI data register SEDR to set SEI.

Note: When accessing the SEI registers, at least 4 states must be inserted between SEI register write and SEI register read. Please remember this when programming.

Example:

```
LD (SEDR), data1   :  write SEDR
NOP                :  } or other instructions which do not access the SEI registers
NOP                :
LD A, (SESR)       :  read SESR
LD (SESR), data2   :  write SESR
NOP                :  } or other instructions which do not access the SEI registers
NOP                :
LD A, (SESR)       :  read SESR
```

(1) SEI control register (SECR)

SEI Control Register

SECR (009DH)

	7	6	5	4	3	2	1	0
bit Symbol	SEIE	SEE	BOS	MSTR	CPOL	CPHA	SER1	SER0
Read/Write	R/W							
After reset	0	0	0	0	0	1	0	0
Function	SEI interrupt 0: Disabled 1: Enabled	SEI operation 0: Stopped 1: Operating	Bit order Select 0: MSB first 1: LSB first	Mode select 0: Slave 1: Master	Clock Polarity select See Fig 3.12.2, 3.12.3	Clock Phase Select See Fig 3.12.2, 3.12.3	SEI transfer rate select See table 3.12.2	

Read-modify-write instructions prohibited.

<SEIE>: SEI interrupt enable

Compatibility mode:

0: SEI interrupts are disabled.

1: SEI interrupts are enabled. A SEI interrupt is requested if the <SEF> flag is being asserted.

Micro DMA mode:

The <SEIE> bit is obsolete in Micro DMA mode. Only the interrupt controller registers are used to enable or disable interrupts.

<SEE>: SEI function enable

0: SEI function is off. It is necessary to disable the SEI function to switch between the Micro DMA mode and the compatibility mode. Wait until the transfer in progress is completed before you clear the <SEE> bit to stop the SEI operation.

1: SEI function is on. Before using the SEI function, make sure that the port function is set for the SEI function.

<BOS>: Bit order select

The bit order selection bit <BOS> selects whether the data to be transferred is MSB first or LSB first.

0: The MSB bit of the SEDR register (bit 7) will be transmitted first.

1: The LSB bit of the SEDR register (bit 0) will be transmitted first.

<MSTR>: Master/Slave mode select

0: SEI is configured as slave.

1: SEI is configured as master..

<CPOL>: Clock polarity select

0: Active "H" level clock is selected. The SECLK clock is at idle "L" level when not transmitting..

1: Active "L" level clock is selected. The SECLK clock is at idle "H" levelwhen not transmitting.

Refer to figures 3.12.2 and 3.12.3.

<CPHA>: Clock phase select

<CPHA> bit selects one of two, fundamentally different transfer formats.

Refer to figures 3.12.2 and 3.12.3.

<SER1:0>:SEI bit rate select

The following table shows the relationship between the <SER1> and <SER0> control bits and the bit rate for transfers when the SEI is operating as a master. When the SEI is operating as a slave, the serial clock is input from the master, therefore the <SER1> and <SER0> control bits are redundant.

Table 3.12.2 SEI transfer bit rate

<SER1>	<SER0>	Divide-by-rate of internal SEI clock	Transfer rate when fc = 24 MHz
0	0	2	4 Mbps
0	1	4	2 Mbps
1	0	8	1 Mbps
1	1	32	250 Kbps

Internal SEI clock: External clock divided by 3.

(2) SEI status register (SESR)

SEI Status Register

SESR (009EH)

Compati-bility mode

	7	6	5	4	3	2	1	0
bit Symbol	SEF	WCOL	SOVF	–	–	–	–	TMSE
Read/Write	R							R/W
After reset	0	0	0					0
Function	SEI transfer complete flag 1: Transfer completed	Write collision flag 1: Write collided	Overflow flag (slave) 1: Overflow occurred					SEI mode select 0: Compati-bility mode 1: Micro DMA mode

SEI Status Register

SESR (009EH)

Micro DMA mode

Read-modify-write instructions prohibited.

	7	6	5	4	3	2	1	0
bit Symbol	–	WCOL	SOVF	–	TSRC	TSTC	TASM	TMSE
Read/Write		R			R		R/W	
After reset		0	0		0	0	0	0
Function		Write collision flag 1: Write collided	Overflow flag (slave) 1: Overflow occurred		SEI receive complete flag 1: Receive completed	SEI transmit complete flag 1: Transmit completed	SEI automated shift mode (master) Interrupt mask (slave)	SEI mode select 0: Compati-bility mode 1: Micro DMA mode

<SEF>: Transfer complete flag

Compatibility mode:

The <SEF> flag is automatically set to 1 at the end of a SEI transfer. The <SEF> flag is automatically cleared by reading the SESR register with <SEF> flag set, followed by an access of the SEDR register.

Micro DMA mode:

Always reads as undefined; writes to this flag have no effect.

<WCOL>: Write collision error flag

Compatibility mode:

The <WCOL> flag is automatically asserted if the SEDR register is written while a transfer is in progress. The write itself has no effect on the running transmission. The <WCOL> flag is automatically cleared by reading the SESR register with the <WCOL> bit set followed by an access to the SEDR or SECR register. No interrupt will be generated on the assertion of this flag.

Micro DMA mode:

The <WCOL> flag is automatically asserted if the SEDR register is written while a transfer is in progress. The write itself has no effect on the running transmission. The flag can only be reset by writing a 1 to it. Writing a 0 has no effect. An interrupt will be generated on INTSE0 on a transition from 0 to 1 if the module is configured as a slave and <TASM> bit is equal to 0.

<SOVF>: Slave mode overflow error flag

Master mode:

Always reads as undefined; writes to this flag have no effect.

Slave mode:

Compatibility mode:

The <SOVF> flag is automatically asserted if a new byte has been completely received and the <SEF> flag is still asserted. The <SOVF> flag is automatically cleared by reading the SESR register with the <SOVF> flag set followed by an access to the SEDR register. The <SOVF> flag will also be cleared by switching to master mode. In Compatibility mode, no interrupt will be generated on the assertion of the <SOVF> flag.

Micro DMA mode:

The <SOVF> flag is automatically asserted if a new byte has been completely received and the <TSRC> flag is still asserted. The <SOVF> flag can only be cleared by writing a 1 to it. Writing 0 to it has no effect. INTSE0 is generated with <TASM> = 1 if the <SOVF> flag changes from 0 to 1.

<TSRC>: Receive completion flag

Compatibility mode:

Always reads as undefined; writes to this flag have no effect.

Micro DMA mode:

The <TSRC> flag is set when a receive has been completed, that is when eight cycles have shifted on the SCLK signal. It is cleared by performing a read operation on the SEI data register, by switching to compatibility mode, or by writing a 1 to this flag. Writing a 0 to this flag has no effect. An interrupt INTSE1 will be generated on the assertion of this flag.

<TSTC>: Transmit completion flag

Compatibility mode:

Always reads as undefined; writes to this flag have no effect.

Micro DMA mode:

The <TSTC> flag is set when the transmission of one byte of data is completed, but the timing of the flag depends on the transfer format and master/slave status. Refer to figures 3.12.2 and 3.12.3. It is cleared by performing a write operation on the SEI data register, by switching to compatibility mode or by writing a 1 to this flag. Writing a 0 to this flag has no effect. An interrupt INTSE2 will be generated on the assertion on this flag.

<TASM>: Automated shift mode (master) /INTSE0 interrupt mask (slave)

Compatibility mode:

Always reads as undefined; writes to this flag have no effect.

Micro DMA mode:

The functioning of this bit is determined by the <MSTR> bit setting.

Master mode:

0: Disables the automated shift mode.

1: Enables the automated shift mode.

In this mode, a read access to the SEI data register SEDR will perform the following functions.

- The SEI data register will be cleared to 00 hex after it has been read.
- A new transfer will be initiated, thus in master mode 8 low bits will be sent, 8 new bits will be received.

The automated shift mode also works when it is combined with a Micro DMA. It has no effect, when SEI is in slave mode.

Slave mode:

This bit functions as a mask for the interrupt INTSE0 generation of the <SOVF> and <WCOL> flags.

0: An interrupt will be generated on the <WCOL> flag, but not on the <SOVF> flag.

1: An interrupt will be generated on the <SOVF> flag, but not on the <WCOL> flag.

<TMSE>: SEI mode select

0: Compatibility mode.

1: Micro DMA mode.

Selects the Micro DMA mode, which also allows Micro DMA transfers. It is necessary to disable the SEI system before switching to the Micro DMA mode.

(3) SEI data register (SEDR)

SEI Data Register

SEDR (009FH)

(for transmission)

	7	6	5	4	3	2	1	0
Bit symbol	SED7	SED6	SED5	SED4	SED3	SED2	SED1	SED0
Read/Write	W							
After Reset	0	0	0	0	0	0	0	0

(for receiving)

	7	6	5	4	3	2	1	0
Bit symbol	SED7	SED6	SED5	SED4	SED3	SED2	SED1	SED0
Read/Write	R							
After Reset	0	0	0	0	0	0	0	0

Note) Read-modify-write prohibited.

This register is used to transmit and receive data. When the SEI system is configured as a master, transfers are started by a software write to the SEDR register.

After once starting transmission, please write after checking that the transmission end flag has been set by interrupt or polling when the master device writes to the SEDR register.

Only when the <SEE> bit of the SECR register is “1”, is a read/write to the SEDR register possible.

When the <SEE> bit is “0”, the write access is ignored and “00H” will be read.

3.12.8 SEI System Errors

Two system errors can be detected by the SEI device. The first type of error, a write collision, indicates that an attempt has been made to write data to the SEDR while a transfer was in progress. The second error occurs when the SEI system is configured as a slave and a new byte of data has been completely shifted in by the remote bus master before the old byte could be read.

(1) Write collision error

A write collision occurs if the SEDR register is written while a transfer is in progress. Because the SEDR register is not a double buffer in the direction of the transmission, writing before transfer to the SEDR register is completed is written directly to the SEI shift register. Because this write corrupts any transfer in progress, a write-collision error is generated. The transfer continues undisturbed, and the write data that caused the error is not written to the shifter.

A write collision is normally a slave error because a slave has no control over when a master will initiate a transfer. A master knows when a transfer is in progress, thus, there is no excuse for a master to generate a write-collision error, although the SEI device can detect write collisions in a master as well as in a slave.

In slave mode, a write collision is likely to occur when the master shifts faster than it can be handled by the slave. This occurs when the slave is transferring a new value to the data register after the master has begun the next shift cycle. In this case a write collision occurs.

In Micro DMA mode, an interrupt on INTSE0 will occur, if the module is configured as slave, when the <TASM> bit is set as 0 and the <WCOL> flag shows a positive edge.

(2) Slave mode overflow error

On an SEI bus, the transmission bit rate is determined by the master. At higher bit rates the problem arises whereby a slave might not be able to follow the transmission of a master. This means that the data is shifted in faster than it can be processed by the slave. Therefore the SEI device offers the <SOVF> flag in its status register which allows the detection of a possible loss of data.

The <SOVF> flag will be asserted, when,

- The SEI module is configured as a slave.
- An old byte of data is still waiting to be read when a new byte of data has been completely received.

When <SOVF> is set, the SEDR has been overwritten by new byte data.

Since this error is only arises in the slave mode, the <TASM> bit can be used as an interrupt mask for this flag. An interrupt is generated on INTSE0 only in Micro DMA mode and the <TASM> bit is 1, if the <SOVF> flag in the status register is asserted.

3.12.9 Interrupt Generation

Interrupt processing differs for the two SEI operating modes, which can be selected using the <TMSE> bit in the SESR register. The SEI module is connected to three interrupt channels named INTSE0, INTSE1 and INTSE2.

(1) Compatibility mode

In compatibility mode only the INTSE0 is used. This channel generates an interrupt, if the <SEF> flag in the SESR register shows a transition from 0 to 1. The <SEIE> bit is used as a global interrupt enable/disable.

SEI interrupt channel 0 (INTSE0)	Interrupt on <SEF>
SEI interrupt channel 1 (INTSE1)	Inactive
SEI interrupt channel 2 (INTSE2)	Inactive

(2) Micro DMA mode

In Micro DMA mode all three interrupt channels are used to allow Micro DMA transfers to and from the SEI data register. Interrupt channel 0 generates an interrupt on two different sources. The first type of interrupt is generated on a transition of the <WCOL> flag from 0 to 1 if the module is in slave mode with the <TASM> bit equal to 0. The second type of interrupt is generated on transition of the <SOVF> flag from 0 to 1 if the module is in slave mode with the <TASM> bit equal to 1.

After a completed transfer, both the <TSRC> flag and the <TSTC> flag in the SESR register are asserted simultaneously. However, there is an exception for <CPHA> = 0 in slave mode. Please see "4.1 transfer format of <CPHA> = 0". Both flags trigger their own interrupt.

The <TSRC> flag generates an interrupt on INTSE1 on a transition from 0 to 1. The <TSRC> flag can be cleared by either reading the SEDR register or by writing a 1 value to this flag.

The <TSTC> flag generates an interrupt on INTSE2 on a transition from 0 to 1. The <TSTC> flag is cleared by either writing the SEDR register or by writing a 1 value to this flag.

For the use of Micro DMAs, INTSE1 and INTSE2 interrupts can be used to trigger a Micro DMA.

INTSE1 interrupt: triggers a Micro DMA read of data from the SEDR register.

INTSE2 interrupt: triggers a Micro DMA write to the SEDR register.

Thus a new transfer is initiated.

SEI interrupt channel 0 (INTSE0)	Interrupt on <WCOL> Note1) or <SOVF> Note2)
SEI interrupt channel 1 (INTSE1)	Interrupt on <TSRC>
SEI interrupt channel 2 (INTSE2)	Interrupt on <TSTC>

Note 1: In slave mode, when <TASM> = 0

Note 2: In slave mode, when <TASM> = 1

In Micro DMA mode the <SEIE> bit is redundant. The Interrupts are individually disabled at the interrupt controller.

3.12.10 Use of Micro DMA for SEI (Micro DMA mode)

The use of Micro DMA for larger SEI transfers enables the speeding up of communication on the SEI by

- reducing CPU load for interrupt processing,
- reducing the time gap between two successive transfers.

The Micro DMA transfers can be used in both master and slave modes.

(1) Read/Write Micro DMA transfer

Set the <TMSE>bit of the SESR register to 1 to set to micro DMA mode. Two Micro DMA channels are configured in a way that both the values to be shifted out can be determined and the values shifted in can be stored. However, the transfer will be handled completely by the Micro DMA controller.

[1] Initiation

Two Micro DMA channels must be set up for the transfer. One Micro DMA is triggered on the SEI interrupt channel 1 (INTSE1) to transfer the value that was received from the SEI data register to memory. The other cannel (INTSE2) is used to write new data from the memory to the SEI data register. This setting is used to re-initiate transfers in the master mode.

The Micro DMA with the lower channel has to be assigned to the INTSE1 interrupt since it takes precedence over the Micro DMA with the higher channel number.

The Micro DMA transfer is initiated the first time by writing the first transfer value to the SEI data register. The following transfers will be handled automatically by the Micro DMA controller.

Table 3.12.3 SEI setting for micro DMA transfer (Read/Write)

<SEIE>	<SEE>	<MSTR>	<TASM>	<TMSE>
X	1	0: Slave	INTSE0 interrupt mask	1
		1: Master	0	

[2] Micro DMA transfer

Once initiated the Micro DMAs wait to be triggered by a completed transfer. On a completed transfer, both <TSRC> and <TSTC> flags are set to 1, and both the SEI receive completed interrupt pulse INTSE1 and the SEI transmit completed interrupt pulse INTSE2 are generated. Since the Micro DMA channel with the lower channel number takes precedence, the Read Micro DMA is performed before the Write Micro DMA. The Read Micro DMA reads the value from the SEI data register and stores the value at the location specified within the Micro DMA control registers. The read access also clears the <TSRC> flag in effect. After this the Write Micro DMA transfers a value from a specified memory address to the SEI data register. The write access to the data register automatically clears the <TSTC> bit in the SEI status register and starts a new transfer when the module is in master mode. After each Micro DMA transfer, the count registers for both Micro DMA are decreased. This procedure continues until the counters reach the value of 0. A Micro DMA interrupt will be generated to indicate the end of the Micro DMA transfer. An interrupt service routine triggered on the end of the Micro DMA transfer can be used to re-initiate Micro DMA transfers.

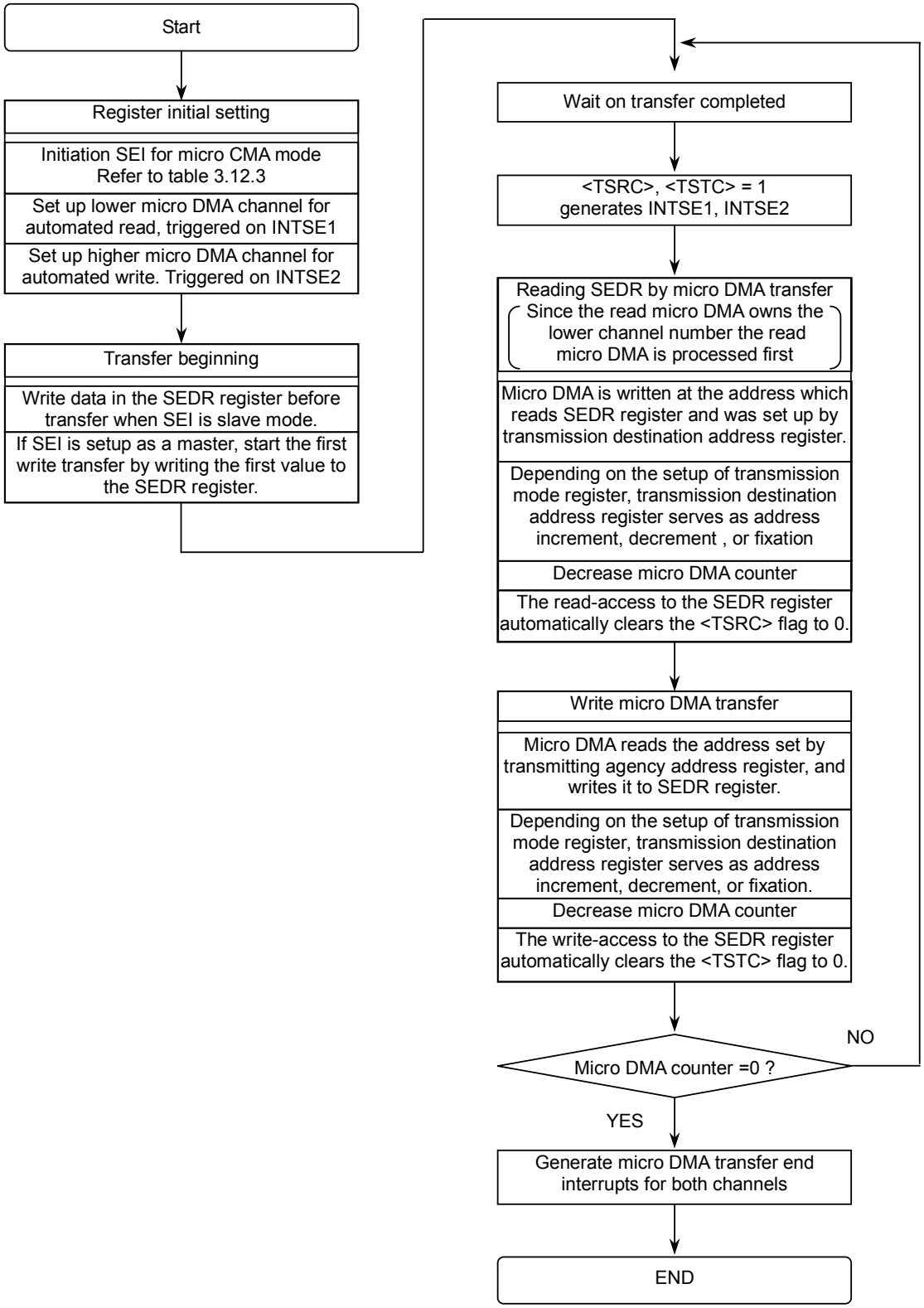


Figure 3.12.7 Flow for Micro DMA read/write transfer

(2) Read-only Micro DMA transfer

This mode is used to shift in lager blocks of data, while "don't care data" is shifted out (e.g.: reads from serial EEPROM). Only a single Micro DMA is used to store the data read from the SEI data register to a specified RAM area.

[1] Initiation

For this mode the module has to be configured for Micro DMA mode by asserting the <TMSE> bit in the SESR register. When SEI is acting as master, the <TASM> bit has to be set additionally to allow the automated shifting. Just one Micro DMA has to be set up to transfer the received data to a memory location specified within the Micro DMA destination address register. The SEI receive completion interrupt INTSE1 is used to trigger this Micro DMA. The SEI transmit completion interrupt INTSE2 is disabled at the interrupt controller. If SEI is set up as a master, the first transfer has to be initiated by writing the SEI data register.

Table 3.12.4 SEI setting for micro DMA transfer (read-only)

<SEIE>	<SEE>	<MSTR>	<TASM>	<TMSE>
X	1	0: Slave	INTSE0 interrupt mask	1
		1: Master	1	

[2] Micro DMA transfer

After initiating the first transfer, the Micro DMA waits for the transfer to be completed. With the completion of the transfer both the <TSRC> and <TSTC> flags in the SEI status register are asserted. On the assertion of the <TSRC> flag an interrupt is generated to trigger the Micro DMA. The <TSTC> flag will be asserted simultaneously and will remain set till the end of the block transfer.

The Micro DMA moves the received value from the SEI data register to the memory location specified in its destination address register. When the SEDR register is read, the SEDR register (shift register) is cleared to "00H" automatically because <TASM> bit is 1. Simultaneously, a new transfer is started automatically. This procedure will repeat until the Micro DMA counter reaches a value of 0.

Moreover, after the first transfer is completed, the <TSTC> flag remains as set at 1, unless cleared.

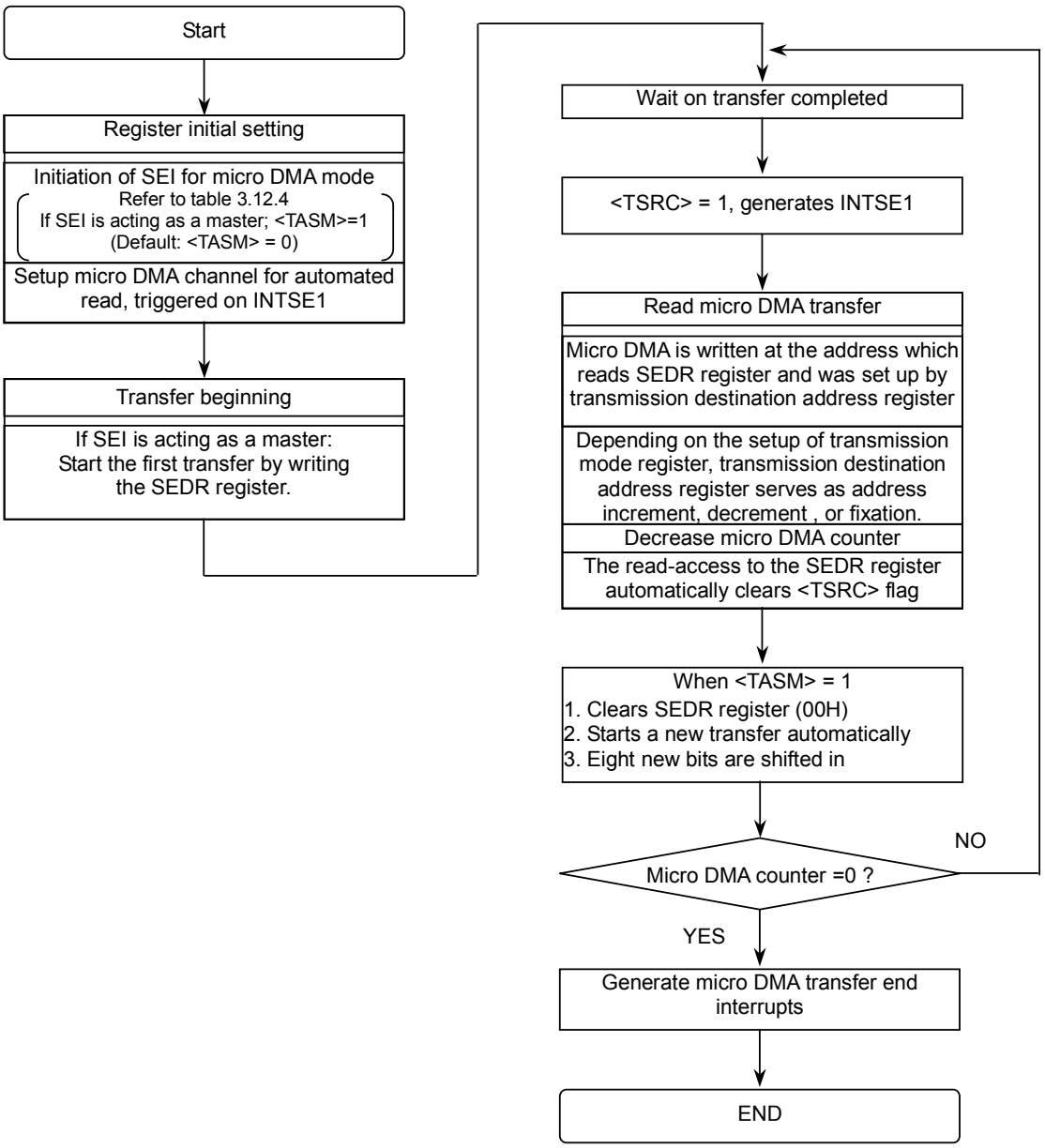


Figure 3.12.8 Flowchart for Micro DMA read-only transfer

(3) Write-only Micro DMA transfer

The write-only transfer mode is used to transmit larger blocks of data while the incoming data is ignored. Only a single Micro DMA is used to transfer new transmit data from a memory location specified by the Micro DMA source address register to the SEI data register.

[1] Initiation

For this mode the module has to be configured for Micro DMA mode by asserting the <TMSE> bit in the SESR register. One of the Micro DMA channels has to be set up for the automated write to the SEI data register. This Micro DMA is triggered by the SEI transmit completion interrupt INTSE2. The SEI receive completion interrupt INTSE1 is disabled at the interrupt controller. If SEI is set up as a master, the first transfer is initiated by writing the first value to the SEI data register.

Table 3.12.5 SEI setting for micro DMA transfer (write-only)

<SEIE>	<SEE>	<MSTR>	<TASM>	<TMSE>
X	1	0: Slave	INTSE0 interrupt mask	1
		1: Master	0	

[2] Micro DMA transfer

After starting the first transfer, the Micro DMA waits for the transfer to be completed. On completion, both the <TSRC> and <TSTC> flags in the SEI status register are asserted. Disregard the <TSRC> flag and the <SOVF> flag as they are no longer used. After the first transfer is completed, the <TSRC> flag remains as set at 1, unless cleared. If once set at 1, the <SOVF> flag also remains at 1 unless cleared. The <TSTC> flag generates an interrupt, which will trigger the Micro DMA transfer.

The Micro DMA reads a value from the memory address specified in its source register and transfers it to the SEI data register. The write access to the SEI data register clears the <TSTC> flag and starts a new transfer on the SEI bus when the module is in master mode. This procedure continues until the Micro DMA counter reaches a value of 0.

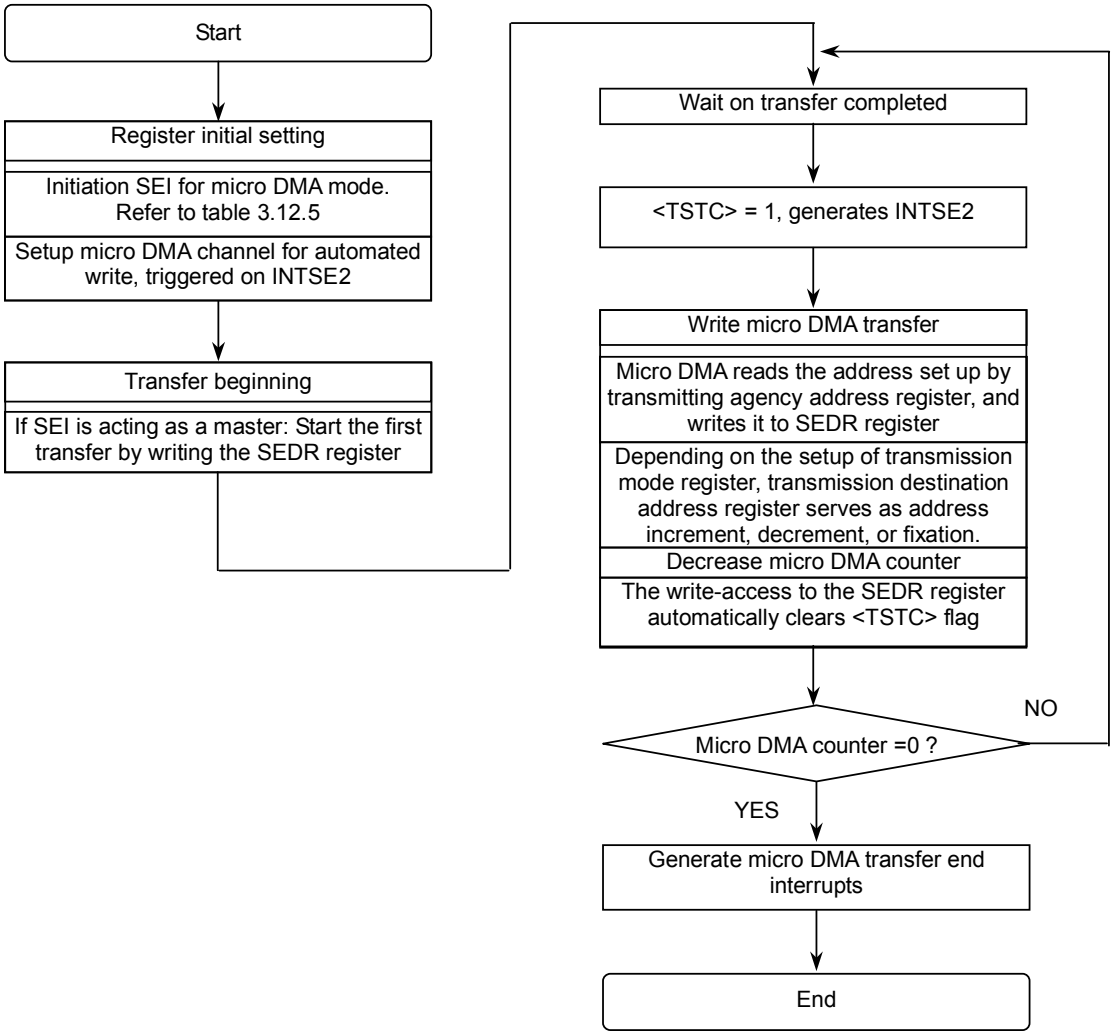


Figure 3.12.9 Flowchart for Micro DMA write-only transfer

3.13 Watchdog Timer (Runaway Detection Timer)

The TMP95CU54A incorporates a watchdog timer for detecting a runaway (out-of-control) condition.

The watchdog timer (WDT) returns the CPU to its normal state when it detects the start of a CPU runaway due to, for example, noise. When the watchdog timer detects a runaway, it generates an INTWD (non-maskable) interrupt to notify the CPU of the condition.

In addition, the runaway detection result can be used for a forcible reset of the microcontroller itself. The watchdog timer consists of a 22-step binary counter with 2/fc as the input clock, and a control block. Figure 3.13.1 is a block diagram of the watchdog timer (WDT).

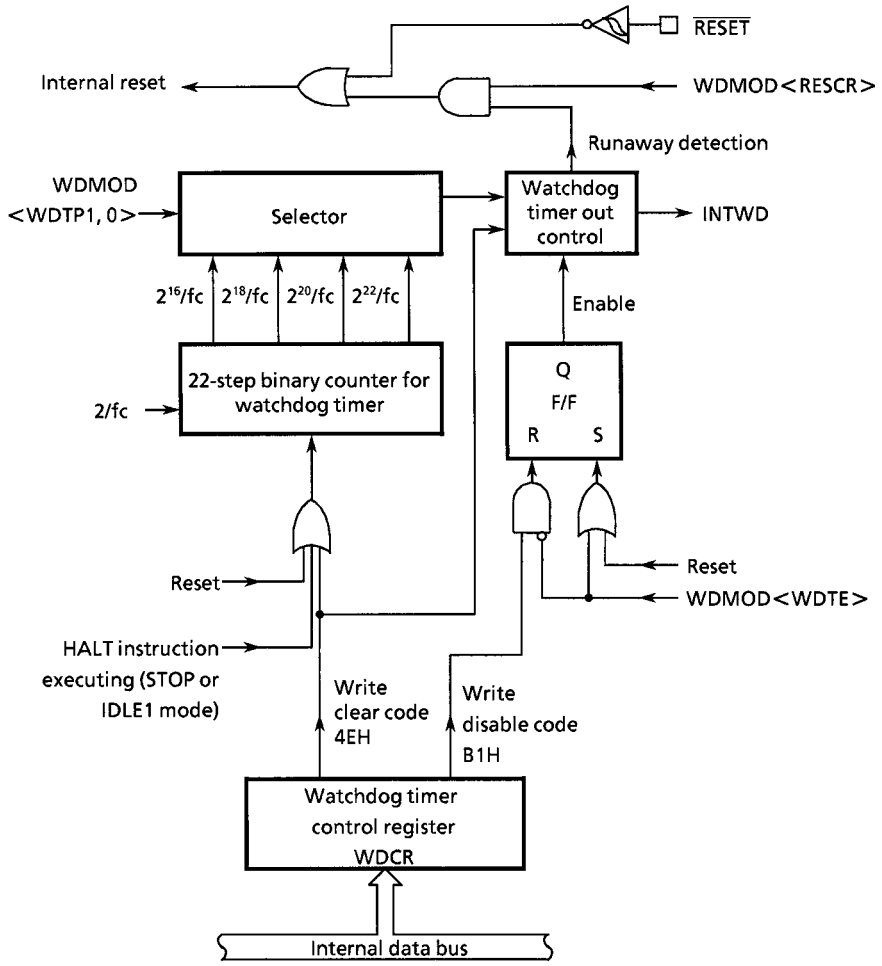


Figure 3.13.1 Block diagram of watchdog timer

3.13.1 Watchdog timer registers

The watchdog timer (WDT) is controlled by two control registers. Figure 3.13.2 shows watchdog timer mode control register WDMOD and watchdog timer control register WDCR.

Watchdog Timer Mode Control Register

WDMOD (006EH)

	7	6	5	4	3	2	1	0
bit Symbol	WDTE	WDTP1	WDTP0	WARM	HALTM1	HALTM0	RESCR	DRVE
Read/Write	R/W							
After reset	1	0	0	0	0	0	0	0
Function	Watchdog timer control 0: Disable 1: Enable	Watchdog timer detection time selection 00: 2^{16}/fc 01: 2^{18}/fc 10: 2^{20}/fc 11: 2^{22}/fc		Warm-up time 0: 2^{14}/fc 1: 2^{16}/fc	HALT mode selection 00: RUN mode 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode		Internal reset control at runaway detection 1: Perform internal reset by runaway detection	Pin control in STOP mode 1: Drive pins in STOP mode

(See 3.4, Standby Function)

Internal reset control at runaway detection

0	Don't care
1	Perform internal reset by runaway detection

Watchdog timer detection time selection

00	2^{16}/fc (approximately 2.7 ms @ 24 MHz)
01	2^{18}/fc (approximately 10.9 ms @ 24 MHz)
10	2^{20}/fc (approximately 43.7 ms @ 24 MHz)
11	2^{22}/fc (approximately 175 ms @ 24 MHz)

Watchdog timer enable/disable control

0	Disable
1	Enable

Watchdog Timer Control Register

WDCR (006FH)

	7	6	5	4	3	2	1	0
bit Symbol	–							
Read/Write	W							
After reset	–							
Function	B1H: WDT disable code				4EH: WDT clear code			

Watchdog timer disable and clear

B1H	Disable code
4EH	Clear code
Others	Don't care

Figure 3.13.2 Watchdog Timer Related Registers

(1) Watchdog timer mode control register (WDMOD)

[1] Setting watchdog timer detection time <WDTP1:0>

This 2-bit register is used to set the watchdog timer interrupt time for detecting a runaway. After a reset, WDMOD <WDTP1:0> is set to 00, which sets a detection time of 2^{16}/fc [s]. (The number of states is approximately 32,768.)

[2] Watchdog timer enable/disable control <WDTE>

After a reset, WDMOD<WDTE> is initialized to 1, enabling the watchdog timer.

Disabling the watchdog timer requires both clearing this bit to 0 and writing the disable code B1H in watchdog timer control register WDCR. This two-step process is an insurance against an out-of-control system disabling the watchdog timer.

To return from disable state to enable state, simply set <WDTE> to 1.

[3] Runaway detection time internal reset control <RESCR>

This register determines whether or not the watchdog timer resets itself on detection of a runaway. Setting WDMOD <RESCR> to 1 forcibly resets the microcontroller after detection of a runaway. On reset, <RESCR> is initialized to 0. Therefore, detection of a runaway will not trigger an internal reset. In such a case, the watchdog timer holds the runaway detection state until the clear code is written to WDCR.

(2) Watchdog timer control register WDCR

This register is used to disable the watchdog timer functions and to clear the binary counter.

- Disable control

After clearing WDMOD<WDTE> to 0, write the disable code B1H to WDCR to disable the watchdog timer.

```
          7 6 5 4 3 2 1 0
WDMOD ← 0 - - - - - X X      Clear <WDTE> to 0.
WDCR  ← 1 0 1 1 0 0 0 1      Write disable code B1H.
```

Note: X : Don't care – : No change

- Watchdog timer clear control

Writing clear code 4EH to WDCR clears the binary counter and resumes the count.

```
WDCR ← 0 1 0 0 1 1 1 0      Write clear code 4EH.
```

3.13.2 Description of operation

After the detection time set by the watchdog timer mode register WDMOD <WDTP1:0> is reached, the watchdog timer generates interrupt INTWD. The watchdog timer detection time can be selected from 2^{16}f/c, 2^{18}f/c, 2^{20}f/c, and 2^{22}f/c. The binary counter for the watchdog timer must be cleared to 0 by software (by instruction) before the INTWD interrupt is generated. If the CPU malfunctions (is out of control) due to factors such as noise, and does not execute an instruction to clear the binary counter, the binary counter overflows and generates interrupt INTWD. The CPU interprets the INTWD interrupt as a malfunction (runaway condition) detection signal, which can be used to start program-based anti-malfunction measures to return the system to normal (normal mode).

Runaway detection can also be used for an internal reset (reset mode). To perform an internal reset by runaway detection, first set WDMOD <RESCR> to 1.

The INTWD interrupt generation cycle is twice the watchdog timer detection time selected by <WDTP1:0>.

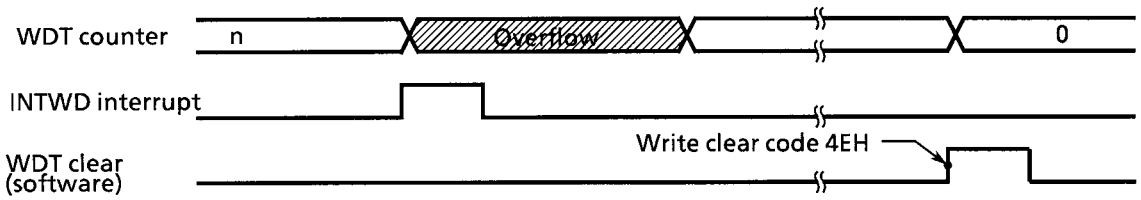


Figure 3.13.3 Normal Mode

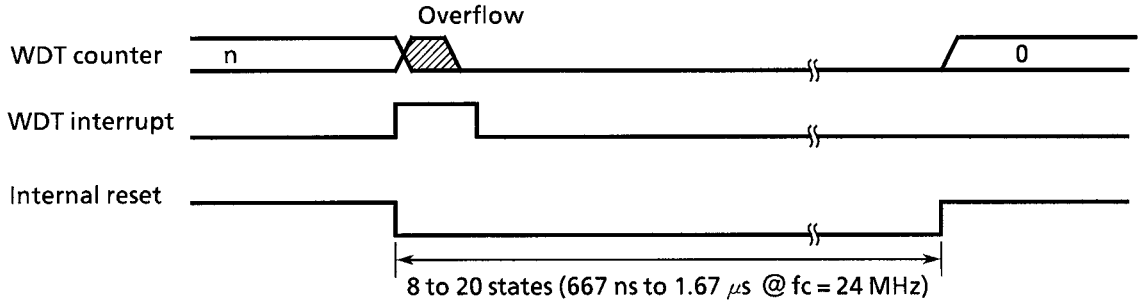


Figure 3.13.4 Reset Mode

The watchdog timer operates during RUN and IDLE2 modes. While an INTWD interrupt does not occur during IDLE2 mode, to prevent an INTWD interrupt being triggered immediately after the halt release, disable the watchdog timer. The watchdog timer is halted in IDLE1 and STOP modes.

As the binary counter continues counting during bus release (when $\overline{\text{BUSAK}}$ goes low), set the runaway detection time in accordance with the bus release time. If the watchdog timer detects a runaway condition during bus release, the watchdog timer generates an INTWD interrupt immediately after the bus release.

The watchdog timer starts operating immediately after reset release.

Example: [1] Clear the binary counter.

```
WDCR  ← 0 1 0 0 1 1 1 0     Write clear code 4EH.
```

[2] Set the watchdog timer detection time to 2^{18}/fc.

```
WDMOD ← 1 0 1 - - - X X
```

[3] Disable the watchdog timer.

```
WDMOD ← 0 - - - - - X X     Clear <WDTE> to 0.
WDCR  ← 1 0 1 1 0 0 0 1     Write disable code B1H.
```

[4] Select IDLE1 mode.

```
WDMOD ← 0 - - - 1 0 X X     Disable WDT and set IDLE1 mode.
WDCR  ← 1 0 1 1 0 0 0 1
Execute HALT instruction.   Set HALT mode.
```

[5] Select IDLE2 mode.

```
WDMOD ← 0 - - - 1 1 X X     Disable WDT and set IDLE2 mode.
WDCR  ← 1 0 1 1 0 0 0 1
Execute HALT instruction.   Set HALT mode.
```

[6] Select STOP mode. (Warm-up time 2^{16}/fc)

```
WDMOD ← - - - 1 0 1 X X     Set STOP mode.
Execute HALT instruction.   Set HALT mode.
```

Note: X : Don't care − : No change

3.14 Bus Release Function

The TMP95CU54A has a bus request pin ($\overline{BUSRQ}$, shared with P53) for releasing the bus, and a bus acknowledge pin ($\overline{BUSAK}$, shared with P54). These pins are set by the P5CR and P5FC registers.

3.14.1 Description of operation

When a low level signal is input to the $\overline{BUSRQ}$ pin, the TMP95CU54A recognizes a bus release request. When the current bus cycle terminates, the address bus (A23 to A0) and the bus control signals ($\overline{RD}$, $\overline{WR}$, $\overline{HWR}$) first go high. Then these signals and the data bus (D15 to D0) output buffer are set to off, and the $\overline{BUSAK}$ pin outputs a low signal. This sequence indicates that the bus is released.

During bus release, TMP95CU54A disables all access to the internal I/O registers, although internal I/O functions are not affected. Accordingly, the watchdog timer continues to count up during bus release. When using the bus release function, set the runaway detection time in accordance with the bus release time.

3.14.2 Pin states when bus is released

Table 3.14.1 shows the pin states when the bus is released.

Table 3.14.1 Pin States at Bus Release

Pin Name	Pin State at Bus Release	
	Port Mode	Function Mode
P07 to P00 (D7 to D0) P17 to P10 (D15 to D8)	No change	Goes to high impedance.
P27 to P20 (A23 to A16) P37 to P30 (A15 to A8) P47 to P40 (A7 to A0) P50 ($\overline{RD}$) P51 ($\overline{WR}$)	No change	Goes to high impedance. (Goes high immediately before bus release.)
P52 ($\overline{HWR}$)	No change	Turns output buffer off. Internal pull-up resistors are added regardless of the output latch value. (Goes high immediately before bus release.)

4. Electrical Characteristics

4.1 Absolute Maximum Ratings

Parameter	Symbol	Rating	Unit
Power Supply Voltage	V_{cc}	− 0.5 to + 6.5	V
Input Voltage	V_{IN}	− 0.5 to Vcc + 0.5	V
Output current (total)	ΣI_{OL}	+ 120	mA
Output current (total)	ΣI_{OH}	− 120	mA
Power Dissipation (Ta = + 85°C)	P_D	600	mW
Soldering Temperature (10 s)	T_{SOLDER}	+ 260	°C
Storage Temperature	T_{STG}	− 65 to + 150	°C
Operating Temperature	T_{OPR}	− 40 to + 85	°C

Note: The absolute maximum ratings are rated values which must not be exceeded during operation, even for an instant. Any one of the ratings must not be exceeded. If any absolute maximum rating is exceeded, a device may break down or its performance may be degraded, causing it to catch fire or explode resulting in injury to the user. Thus, when designing products which include this device, ensure that no absolute maximum rating value will ever be exceeded.

4.2 DC Electrical Characteristics

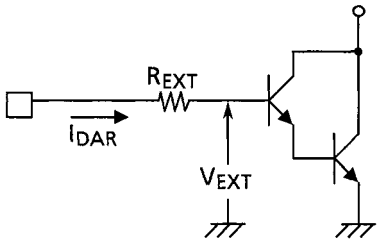
Vcc = +5 V ±10%, Ta = −40 to +85°C (fc = 8 to 24 MHz)

Parameter	Symbol	Test Condition	Min	Max	Unit
Input Low Voltage (D0 to 15) Port 2 to A (except P56, P70, P72, P73, P75) $\overline{RESET}$, $\overline{NMI}$, INT0 to 4 $\overline{EA}$, AM8/$\overline{16}$ X1	V_{IL} V_{IL1} V_{IL2} V_{IL3} V_{IL4}		−0.3 −0.3 −0.3 −0.3 −0.3	0.8 0.3 Vcc 0.25 Vcc 0.3 0.2 Vcc	V V V V V
Input High Voltage (D0 to 15) Port 2 to A (except P56, P70, P72, P73, P75) $\overline{RESET}$, $\overline{NMI}$, INT0 to 4 $\overline{EA}$, AM8/$\overline{16}$ X1	V_{IH} V_{IH1} V_{IH2} V_{IH3} V_{IH4}		2.2 0.7 Vcc 0.75 Vcc Vcc − 0.3 0.8 Vcc	Vcc + 0.3 Vcc + 0.3 Vcc + 0.3 Vcc + 0.3 Vcc + 0.3	V V V V V
Output Low Voltage	V_{OL}	I_{OL} = 1.6 mA		0.45	V
Output High Voltage	V_{OH} V_{OH1} V_{OH2}	I_{OH} = −400 μA I_{OH} = −100 μA I_{OH} = −20 μA	2.4 0.75 Vcc 0.9 Vcc		V V V
Darlington Drive Current (8 Output Pins max.)	I_{DAR}	V_{EXT} = 1.5 V R_{EXT} = 1.1 kΩ	−1.0	−3.5	mA
Input Leakage Current Output Leakage Current	I_{LI} I_{LO}	0.0 ≦ Vin ≦ Vcc 0.2 ≦ Vin ≦ Vcc − 0.2	0.02 (Typ) 0.05 (Typ)	± 5 ± 10	μA μA
Operating Current (NORMAL) RUN IDLE2 IDLE1	I cc	fc = 24 MHz	70 (Typ) 35 (Typ) 30 (Typ) 5 (Typ)	85 50 40 10	mA mA mA mA
STOP (Ta = −40 to + 85°C) (Ta = −20 to + 70°C)		0.2 ≦ Vin ≦ Vcc − 0.2	0.5 (Typ)	100 50	μA μA
Power Down Voltage (@STOP, RAM Back up)	V_{STOP}	V_{IL2} = 0.2 Vcc, V_{IH2} = 0.8 Vcc	2.0	6.0	V
Pull Up Registance	R_{RP}		45	160	kΩ
Pin Capacitance	C_{IO}	fc = 1 MHz		10	pF
Schmitt Width $\overline{RESET}$, $\overline{NMI}$, INT0 to 4	V_{TH}		0.4	1.0 (Typ)	V

Note 1: Typical values are for Ta = +25°C, V_{CC} = +5 V

Note 2: I_{DAR} guarantees up to eight pins from any output port.

Refer: I_{DAR} definition diagram



4.3 AC Electrical Characteristics

Vcc = +5 V ±10%, Ta = −40 to +85°C

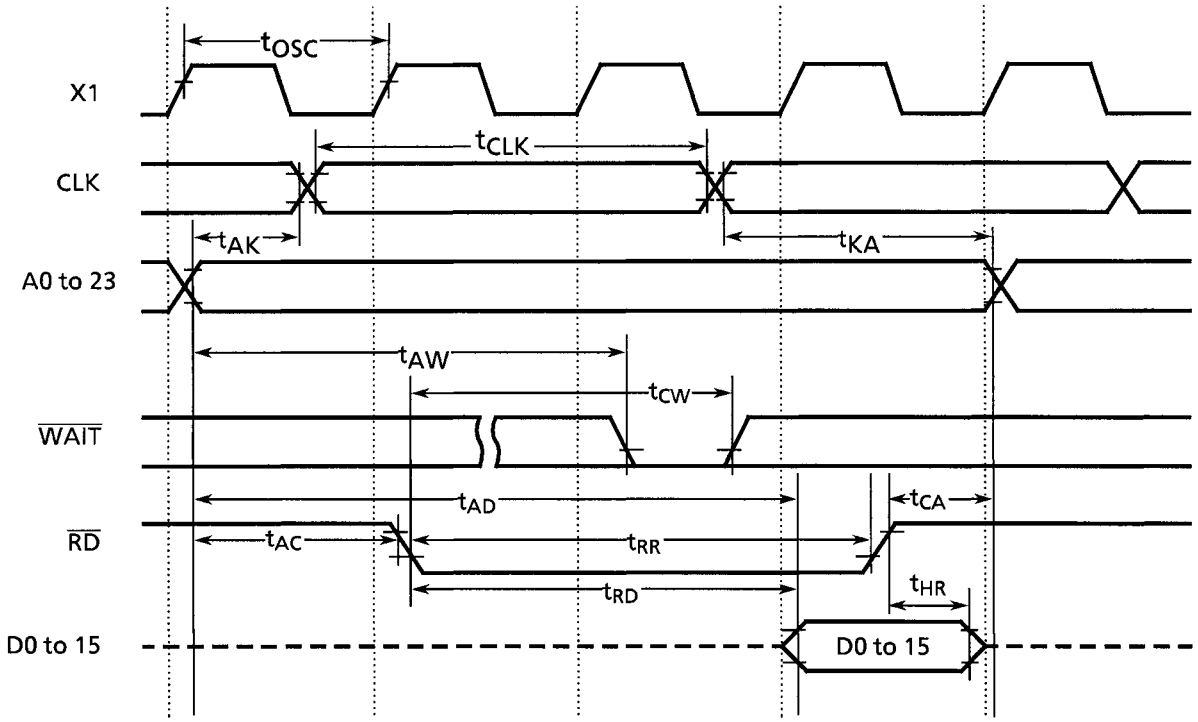
(fc = 8 MHz to 24 MHz)

No.	Parameter	Symbol	Variable		24 MHz		Unit
			Min	Max	Min	Max	
1	Oscillation cycle (= x)	t_{OSC}	42	125	42		ns
2	Clock pulse width	t_{CLK}	2.0x – 40		44		ns
3	A0 to 23 valid → Clock hold	t_{AK}	0.5x – 20		1		ns
4	Clock valid → A0 to 23 hold	t_{KA}	1.5x – 60		3		ns
5	A0 to 23 valid → $\overline{RD}/\overline{WR}$ fall	t_{AC}	1.0x – 20		22		ns
6	$\overline{RD}/\overline{WR}$ rise → A0 to 23 hold	t_{CA}	0.5x – 20		1		ns
7	A0 to 23 valid → D0 to 15 input	t_{AD}		3.5x – 40		107	ns
8	$\overline{RD}$ fall → D0 to 15 input	t_{RD}		2.5x – 45		60	ns
9	$\overline{RD}$ low pulse width	t_{RR}	2.5x – 40		65		ns
10	$\overline{RD}$ rise → D0 to 15 hold	t_{HR}	0		0		ns
11	$\overline{WR}$ low pulse width	t_{WW}	2.5x – 40		65		ns
12	D0 to 15 valid → $\overline{WR}$ rise	t_{DW}	2.0x – 40		44		ns
13	$\overline{WR}$ rise → D0 to 15 hold	t_{WD}	0.5x – 10		11		ns
14	A0 to 23 valid → $\overline{WAIT}$ input ($^{1\ WAIT}_{+\ n\ mode}$)	t_{AW}		3.5x – 90		57	ns
	A0 to 23 valid → $\overline{WAIT}$ input ($^{0+n\ WAIT}_{mode}$)	t_{AW}		1.5x – 40		23	ns
15	$\overline{RD}/\overline{WR}$ fall → $\overline{WAIT}$ hold ($^{1\ WAIT}_{+\ n\ mode}$)	t_{CW}	2.5x + 0		105		ns
	$\overline{RD}/\overline{WR}$ fall → $\overline{WAIT}$ hold ($^{0+n\ WAIT}_{mode}$)	t_{CW}	0.5x + 0		21		ns
16	$\overline{WR}$ rise → PORT valid	t_{CP}		200		200	ns

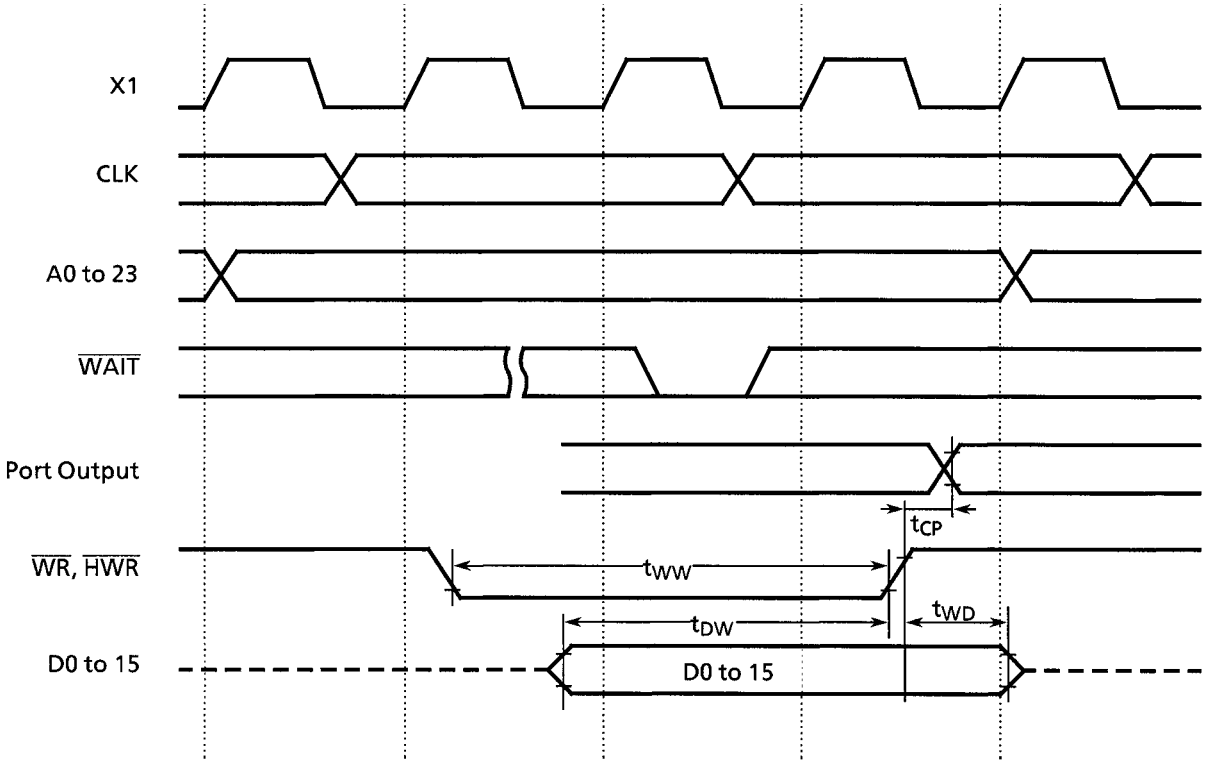
AC measuring conditions

- Output level: High 2.2 V / Low 0.8 V, CL = 50 pF
- Input level: High 2.4 V / Low 0.45 V (D0 to D15)
  High 0.8×Vcc / Low 0.2×Vcc (except for D0 to D15)

(1) Read cycle



(2) Write cycle



4.4 Serial Channel Timing

(1) I/O interface mode

[1] SCLK input mode

Vcc = +5 V ± 10%, Ta = −40 to +85°C (fc = 8 to 24 MHz)

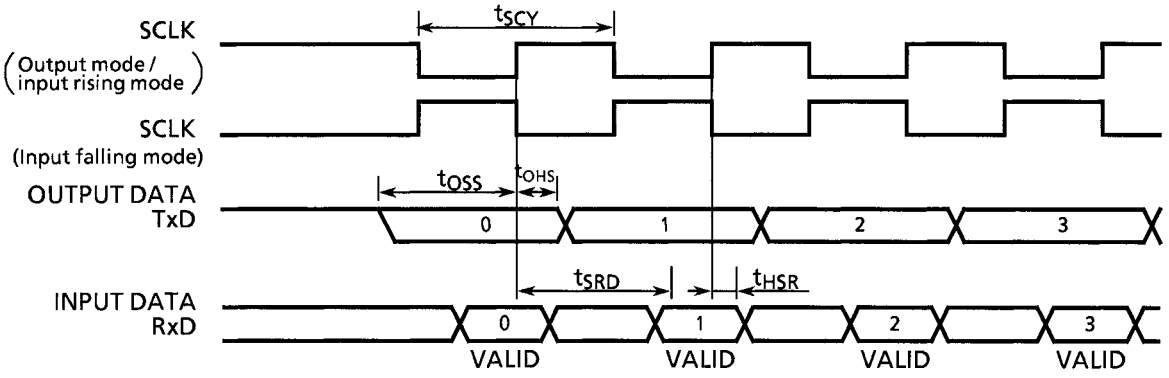
Parameter	Symbol	Variable		24 MHz		Unit
		Min	Max	Min	Max	
SCLK cycle	t_{SCY}	16x		0.667		μs
Output Data →SCLK rise/fall*	t_{OSS}	$t_{SCY}/2 - 5x - 50$		75		ns
SCLK rise/fall* →Output Data hold	t_{OHS}	5x − 100		108		ns
SCLK rise/fall* →input data hold	t_{HSR}	0		0		ns
SCLK rise/fall* →valid data input	t_{SRD}		$t_{SCY} - 5x - 100$		358	ns

*) SCLK rise/fall: In SCLK rising edge mode, SCLK rising edge timing; in SCLK falling edge mode, SCLK falling edge timing

[2] SCLK output mode

Vcc = +5 V ± 10%, Ta = −40 to +85°C (fc = 8 to 24 MHz)

Parameter	Symbol	Variable		24 MHz		Unit
		Min	Max	Min	Max	
SCLK cycle (programmable)	t_{SCY}	16x	8192x	0.667	341.3	μs
Output Data →SCLK rising edge	t_{OSS}	$t_{SCY} - 2x - 150$		433		ns
SCLK rising edge →Output Data hold	t_{OHS}	2x − 80		3		ns
SCLK rising edge →Input Data hold	t_{HSR}	0		0		ns
SCLK rising edge →valid data input	t_{SRD}		$t_{SCY} - 2x - 150$		433	ns



(2) UART mode (SCLK0 to 1 external input)

Vcc = +5 V ± 10%, Ta = −40 to +85°C (fc = 8 to 24 MHz)

Parameter	Symbol	Variable		24 MHz		Unit
		Min	Max	Min	Max	
SCLK cycle	t_{SCY}	4x + 20		187		ns
Low-level SCLK pulse width	t_{SCYL}	2x + 5		88		ns
High-level SCLK pulse width	t_{SCYH}	2x + 5		88		ns

4.5 AD Conversion Characteristics

Vcc = +5 V ± 10%, Ta = −40 to +85°C (fc = 8 to 24 MHz)

Parameter		Symbol	Test Conditions	Min	Typ	Max	Unit
AD analog reference supply voltage (+)		V_{REFH}		Vcc − 0.2		Vcc	V
AD analog reference supply voltage (−)		V_{REFL}		Vss		Vss + 0.2	
Analog reference voltage		AV_{CC}		Vcc − 0.2		Vcc	
Analog reference voltage		AV_{SS}		Vss		Vss + 0.2	
Analog input voltage		V_{AIN}		V_{REFL}		V_{REFH}	
Analog reference voltage supply current	<VREFON> = 1	I_{REF}	Vcc = +5 V ± 10%			3.7	mA
	<VREFON> = 0		Vcc = +5 V ± 10%		0.02	5.0	μA
Total tolerance (excludes quantization error)		E_T	Vcc = +5 V ± 10%		±1	±3	LSB

Note 1: 1LSB = (VREFH − VREFL) / 2^{10} [V]

Note 2: Power supply current ICC from the VCC pin includes the power supply current from the AVCC pin.

4.6 Event Counter (External Input Clocks: TI0, TI4, TI8, TI9, TIA, TIB)

Vcc = +5 V ± 10%, Ta = −40 to +85°C (fc = 8 to 24 MHz)

Parameter	Symbol	Variable		24 MHz		Unit
		Min	Max	Min	Max	
External input clock cycle	t_{VCK}	8x + 100		433		ns
External low-level input clock pulse width	t_{VCKL}	4x + 40		207		ns
External high-level input clock pulse width	t_{VCKH}	4x + 40		207		ns

4.7 Interrupt Operation

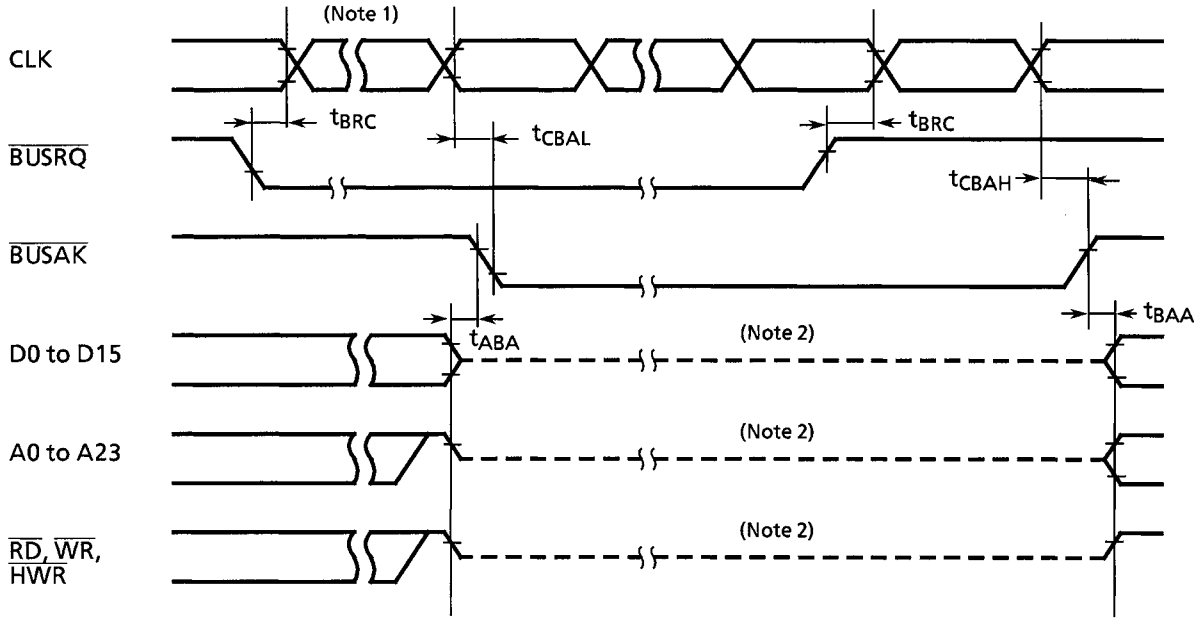
Vcc = +5 V ± 10%, Ta = −40 to +85°C (fc = 8 to 24 MHz)

Parameter	Symbol	Variable		24 MHz		Unit
		Min	Max	Min	Max	
$\overline{NMI}$, INT0 to 4 low-level pulse width	t_{INTAL}	4x		167		ns
$\overline{NMI}$, INT0 to 4 high-level pulse width	t_{INTAH}	4x		167		ns
INT5 to INT8 low-level pulse width	t_{INTBL}	8x + 100		433		ns
INT5 to INT8 high-level pulse width	t_{INTBH}	8x + 100		433		ns

4.8 Bus Request/Bus Acknowledge Timing

Vcc = +5 V ± 10%, Ta = −40 to +85°C (fc = 8 to 24 MHz)

Parameter	Symbol	Variable		24 MHz		Unit
		Min	Max	Min	Max	
$\overline{BUSRQ}$ setup time for CLK	t_{BRC}	120		120		ns
CLK→$\overline{BUSAK}$ fall	t_{CBAL}		2.0x + 120		203	ns
CLK→$\overline{BUSAK}$ rise	t_{CBAH}		0.5x + 40		61	ns
Time from output buffer off until $\overline{BUSAK}$ falling edge	t_{ABA}	0	80	0	80	ns
Time from $\overline{BUSAK}$ rising edge until output buffer on	t_{BAA}	0	80	0	80	ns



Note 1: When $\overline{BUSRQ}$ goes to low level to request bus release, if the current bus cycle is not yet complete due to a wait, the bus is not released until the wait is completed.

Note 2: The dotted line indicates only that the output buffer is off, not that the signal is at middle level. Immediately after bus release, the signal level prior to the bus release is held dynamically by the external load capacitance. Therefore, designs should allow for the fact that when using an external resistor or similar to fix the signal level while the bus is released, after bus release, a delay occurs before the signal goes to its fixed level (due to the CR time constant). The internal programmable pull-up resistor continues to function in accordance with the internal signal level.

5. List of Special Function Registers (SFR) and the Mailbox RAM

The special function registers (SFR), which control the input/output ports and peripheral components, are allocated 160 bytes within the 000000H to 00009FH address range and 64 bytes within the 002300H to 00233FH address range.

The mailbox RAM is allocated 256 bytes within the 002200H to 0022FFH address range.

The registers built into the TMP95CU54A cannot be accessed from outside the TMP95CU54A.

(1) Input/output port

(2) Input/output port control

(3) Timer control

(4) Serial channel control

(5) Interrupt control

(6) Watchdog timer control

(7) Bus Width/wait controller

(8) AD converter control

(9) Serial Expansion interface control

(10) CAN controller

Table structure

Symbol	Name	Address	7	6			1	0	
									→ bit Symbol
									→ Read / Write
									→ Initial value at reset
									→ Remarks

(Supplement for symbols used in Table)

[1] Read/Write

- R/W : Both readable and writable
- R : Readable
- W : Writable
- *R/W : Read-modify-write (RMW) instructions are prohibited for controlling ON/OFF of the pull-up resistors.
- R/S : Enable Read / Set (When “1” is written)
- R/C : Enable Read / Clear (When “1” is written)

[2] RMW prohibited

- Cannot be read, modified, or written. (Cannot use the following instructions: EX, ADD, ADC, SUB, SBC, INC, DEC, AND, OR, XOR, STCF, RES, SET, CHG, TSET, RLC, RRC, RL, RR, SLA, SRA, SLL, SRL, RLD, RRD)

Table 5.1 List of TMP95CU54A Special Function Register Addresses (1/2)

Address	Register Name	Address	Register Name	Address	Register Name	Address	Register Name
000000H	P0	30H	TREG8L	60H	ADREG04L	90H	WAITC0
1H	P1	1H	TREG8H	1H	ADREG04H	1H	WAITC1
2H	P0CR	2H	TREG9L	2H	ADREG15L	2H	WAITC2
3H	(Reserved)	3H	TREG9H	3H	ADREG15H	3H	WAITC3
4H	P1CR	4H	CAP1L	4H	ADREG26L	4H	MSAR0
5H	P1FC	5H	CAP1H	5H	ADREG26H	5H	MAMR0
6H	P2	6H	CAP2L	6H	ADREG37L	6H	MSAR1
7H	P3	7H	CAP2H	7H	ADREG37H	7H	MAMR1
8H	P2CR	8H	T8MOD	8H	(Reserved)	8H	MSAR2
9H	P2FC	9H	T8FFCR	9H	(Reserved)	9H	MAMR2
AH	P3CR	AH	T89CR	AH	SDMACR0	AH	MSAR3
BH	P3FC	BH	T16RUN	BH	SDMACR1	BH	MAMR3
CH	P4	CH	(Reserved)	CH	SDMACR2	CH	WAITCEX
DH	P5	DH		DH	SDMACR3	DH	SECR
EH	P4CR	EH		EH	WDMOD	EH	SESR
FH	P4FC	FH		FH	WDCR	FH	SEDR
10H	P5CR	40H	TREGAL	70H	INTE0AD		
1H	P5FC	1H	TREGAH	1H	INTE12		
2H	P6	2H	TREGBL	2H	INTE34		
3H	P7	3H	TREGBH	3H	INTE56		
4H	P6CR	4H	CAP3L	4H	INTE78		
5H	P6FC	5H	CAP3H	5H	INTET01		
6H	P7CR	6H	CAP4L	6H	INTET23		
7H	P7FC	7H	CAP4H	7H	INTET45		
8H	P8	8H	T9MOD	8H	INTET67		
9H	P9	9H	T9FFCR	9H	INTET89		
AH	P8CR	AH	(Reserved)	AH	INTETAB		
BH	P8FC	BH	(Reserved)	BH	NTETOV		
CH	P9CR	CH	SC0BUF	CH	INTES0		
DH	P9FC	DH	SC0CR	DH	INTES1		
EH	PA	EH	SC0MOD	EH	INTEC01		
FH	(Reserved)	FH	BR0CR	FH	INTEC2S0		
20H	T8RUN	50H	SC1BUF	80H	INTETC01		
1H	TRDC	1H	SC1CR	1H	INTETC23		
2H	TREG0	2H	SC1MOD	2H	INTESE12		
3H	TREG1	3H	BR1CR	3H	(Reserved)		
4H	T01MOD	4H	(Reserved)	4H			
5H	T02FFCR	5H		5H			
6H	TREG2	6H		6H			
7H	TREG3	7H		7H			
8H	T23MOD	8H	ODE	8H			
9H	TREG4	9H	IIMC	9H			
AH	TREG5	AH	DMA0V	AH			
BH	T45MOD	BH	DMA1V	BH			
CH	T46FFCR	CH	DMA2V	CH			
DH	TREG6	DH	DMA3V	DH			
EH	TREG7	EH	ADMOD0	EH			
FH	T67MOD	FH	ADMOD1	FH			

Table 5.1 List of TMP95CU54A Special Function Register Addresses (2/2)

Address	Register Name	Address	Register Name	Address	Register Name	Address	Register Name
002300H	MCL	002310H	LAM0L	002320H	GIFL	002330H	TSPL
2301H	MCH	2311H	LAM0H	2321H	GIFH	2331H	TSPH
2302H	MDL	2312H	LAM1L	2322H	GIML	2332H	TSCL
2303H	MDH	2313H	LAM1H	2323H	GIMH	2333H	TSCH
2304H	TRSL	2314H	GAM0L	2324H	MBTIFL	2334H	(Reserved)
2305H	TRSH	2315H	GAM0H	2325H	MBTIFH	2335H	
2306H	(Reserved)	2316H	GAM1L	2326H	MBRIFL	2336H	
2307H	(Reserved)	2317H	GAM1H	2327H	MBRIFH	2337H	
2308H	TAL	2318H	MCRL	2328H	MBIML	2338H	
2309H	TAH	2319H	MCRH	2329H	MBIMH	2339H	
230AH	(Reserved)	231AH	GSRL	232AH	CDRL	233AH	
230BH	(Reserved)	231BH	GSRH	232BH	CDRH	233BH	
230CH	RMPL	231CH	BCR1L	232CH	RFPL	233CH	
230DH	RMPH	231DH	BCR1H	232DH	RFPH	233DH	
230EH	RMLL	231EH	BCR2L	232EH	CECL	233EH	
230FH	RMLH	231FH	BCR2H	232FH	CECH	233FH	

Note: Don't access reserved registers.

Table 5.2 List of TMP95CU54A Mailbox RAM Addresses (1/2)

Address	Register Name	Address	Register Name	Address	Register Name	Address	Register Name
002200H	MB0MI0L	002220H	MB2MI0L	002240H	MB4MI0L	002260H	MB6MI0L
2201H	MB0MI0H	2221H	MB2MI0H	2241H	MB4MI0H	2261H	MB6MI0H
2202H	MB0MI1L	2222H	MB2MI1L	2242H	MB4MI1L	2262H	MB6MI1L
2203H	MB0MI1H	2223H	MB2MI1H	2243H	MB4MI1H	2263H	MB6MI1H
2204H	MB0MCFL	2224H	MB2MCFL	2244H	MB4MCFL	2264H	MB6MCFL
2205H	MB0MCFH	2225H	MB2MCFH	2245H	MB4MCFH	2265H	MB6MCFH
2206H	MB0D0	2226H	MB2D0	2246H	MB4D0	2266H	MB6D0
2207H	MB0D1	2227H	MB2D1	2247H	MB4D1	2267H	MB6D1
2208H	MB0D2	2228H	MB2D2	2248H	MB4D2	2268H	MB6D2
2209H	MB0D3	2229H	MB2D3	2249H	MB4D3	2269H	MB6D3
220AH	MB0D4	222AH	MB2D4	224AH	MB4D4	226AH	MB6D4
220BH	MB0D5	222BH	MB2D5	224BH	MB4D5	226BH	MB6D5
220CH	MB0D6	222CH	MB2D6	224CH	MB4D6	226CH	MB6D6
220DH	MB0D7	222DH	MB2D7	224DH	MB4D7	226DH	MB6D7
220EH	MB0TSVL	222EH	MB2TSVL	224EH	MB4TSVL	226EH	MB6TSVL
220FH	MB0TSVH	222FH	MB2TSVH	224FH	MB4TSVH	226FH	MB6TSVH
2210H	MB1MI0L	2230H	MB3MI0L	2250H	MB5MI0L	2270H	MB7MI0L
2211H	MB1MI0H	2231H	MB3MI0H	2251H	MB5MI0H	2271H	MB7MI0H
2212H	MB1MI1L	2232H	MB3MI1L	2252H	MB5MI1L	2272H	MB7MI1L
2213H	MB1MI1H	2233H	MB3MI1H	2253H	MB5MI1H	2273H	MB7MI1H
2214H	MB1MCFL	2234H	MB3MCFL	2254H	MB5MCFL	2274H	MB7MCFL
2215H	MB1MCFH	2235H	MB3MCFH	2255H	MB5MCFH	2275H	MB7MCFH
2216H	MB1D0	2236H	MB3D0	2256H	MB5D0	2276H	MB7D0
2217H	MB1D1	2237H	MB3D1	2257H	MB5D1	2277H	MB7D1
2218H	MB1D2	2238H	MB3D2	2258H	MB5D2	2278H	MB7D2
2219H	MB1D3	2239H	MB3D3	2259H	MB5D3	2279H	MB7D3
221AH	MB1D4	223AH	MB3D4	225AH	MB5D4	227AH	MB7D4
221BH	MB1D5	223BH	MB3D5	225BH	MB5D5	227BH	MB7D5
221CH	MB1D6	223CH	MB3D6	225CH	MB5D6	227CH	MB7D6
221DH	MB1D7	223DH	MB3D7	225DH	MB5D7	227DH	MB7D7
221EH	MB1TSVL	223EH	MB3TSVL	225EH	MB5TSVL	227EH	MB7TSVL
221FH	MB1TSVH	223FH	MB3TSVH	225FH	MB5TSVH	227FH	MB7TSVH

Table 5.2 List of TMP95CU54A Mailbox RAM Addresses (2/2)

Address	Register Name	Address	Register Name	Address	Register Name	Address	Register Name
002280H	MB8MI0L	0022A0H	MB10MI0L	0022C0H	MB12MI0L	0022E0H	MB14MI0L
2281H	MB8MI0H	22A1H	MB10MI0H	22C1H	MB12MI0H	22E1H	MB14MI0H
2282H	MB8MI1L	22A2H	MB10MI1L	22C2H	MB12MI1L	22E2H	MB14MI1L
2283H	MB8MI1H	22A3H	MB10MI1H	22C3H	MB12MI1H	22E3H	MB14MI1H
2284H	MB8MCFL	22A4H	MB10MCFL	22C4H	MB12MCFL	22E4H	MB14MCFL
2285H	MB8MCFH	22A5H	MB10MCFH	22C5H	MB12MCFH	22E5H	MB14MCFH
2286H	MB8D0	22A6H	MB10D0	22C6H	MB12D0	22E6H	MB14D0
2287H	MB8D1	22A7H	MB10D1	22C7H	MB12D1	22E7H	MB14D1
2288H	MB8D2	22A8H	MB10D2	22C8H	MB12D2	22E8H	MB14D2
2289H	MB8D3	22A9H	MB10D3	22C9H	MB12D3	22E9H	MB14D3
228AH	MB8D4	22AAH	MB10D4	22CAH	MB12D4	22EAH	MB14D4
228BH	MB8D5	22ABH	MB10D5	22CBH	MB12D5	22EBH	MB14D5
228CH	MB8D6	22ACH	MB10D6	22CCH	MB12D6	22ECH	MB14D6
228DH	MB8D7	22ADH	MB10D7	22CDH	MB12D7	22EDH	MB14D7
228EH	MB8TSVL	22AEH	MB10TSVL	22CEH	MB12TSVL	22EEH	MB14TSVL
228FH	MB8TSVH	22AFH	MB10TSVH	22CFH	MB12TSVH	22EFH	MB14TSVH
2290H	MB9MI0L	22B0H	MB11MI0L	22D0H	MB13MI0L	22F0H	MB15MI0L
2291H	MB9MI0H	22B1H	MB11MI0H	22D1H	MB13MI0H	22F1H	MB15MI0H
2292H	MB9MI1L	22B2H	MB11MI1L	22D2H	MB13MI1L	22F2H	MB15MI1L
2293H	MB9MI1H	22B3H	MB11MI1H	22D3H	MB13MI1H	22F3H	MB15MI1H
2294H	MB9MCFL	22B4H	MB11MCFL	22D4H	MB13MCFL	22F4H	MB15MCFL
2295H	MB9MCFH	22B5H	MB11MCFH	22D5H	MB13MCFH	22F5H	MB15MCFH
2296H	MB9D0	22B6H	MB11D0	22D6H	MB13D0	22F6H	MB15D0
2297H	MB9D1	22B7H	MB11D1	22D7H	MB13D1	22F7H	MB15D1
2298H	MB9D2	22B8H	MB11D2	22D8H	MB13D2	22F8H	MB15D2
2299H	MB9D3	22B9H	MB11D3	22D9H	MB13D3	22F9H	MB15D3
229AH	MB9D4	22BAH	MB11D4	22DAH	MB13D4	22FAH	MB15D4
229BH	MB9D5	22BBH	MB11D5	22DBH	MB13D5	22FBH	MB15D5
229CH	MB9D6	22BCH	MB11D6	22DCH	MB13D6	22FCH	MB15D6
229DH	MB9D7	22BDH	MB11D7	22DDH	MB13D7	22FDH	MB15D7
229EH	MB9TSVL	22BEH	MB11TSVL	22DEH	MB13TSVL	22FEH	MB15TSVL
229FH	MB9TSVH	22BFH	MB11TSVH	22DFH	MB13TSVH	22FFH	MB15TSVH

(1) Input/output ports

Symbol	Name	Address	7	6	5	4	3	2	1	0
P0	Port 0 Register	00H	P07	P06	P05	P04	P03	P02	P01	P00
			R/W							
			Input mode (output latch register undefined)							
			shared with D7 to D0							
P1	Port 1 Register	01H	P17	P16	P15	P14	P13	P12	P11	P10
			R/W							
			Input mode (output latch register cleared to 0)							
			shared with D15 to D8							
P2	Port 2 Register	06H	P27	P26	P25	P24	P23	P22	P21	P20
			R/W							
			Input mode (output latch register cleared to 0)							
			shared with D23 to D16							
P3	Port 3 Register	07H	P37	P36	P35	P34	P33	P32	P31	P30
			R/W							
			Input mode (output latch register cleared to 0)							
			shared with A15 to A8							
P4	Port 4 Register	0CH	P47	P46	P45	P44	P43	P42	P41	P40
			R/W							
			Input mode (output latch register cleared to 0)							
			shared with A7 to A0							
P5	Port 5 Register	0DH	P57	P56	P55	P54	P53	P52	P51	P50
			*R/W							
			Output only (set to 1)	Input mode (set to 1 / Pull-up)					Output only (set to 1) (Note)	
			Shared with CLKOUT	Shared with INT0	Shared with WAIT	Shared with BUSAK	Shared with BUSRQ	Shared with HWR	Shared with WR	Shared with RD
P6	Port 6 Register	12H					P63	P62	P61	P60
							R/W			
							Input mode (set to 1)			
							Shared with SCLK	Shared with MISO	Shared with MOSI	Shared with SS
P7	Port 7 Register	13H			P75	P74	P73	P72	P71	P70
					R/W					
					Input mode (output latch register cleared to 0)					
					Shared with TO7/INT4	Shared with TO5	Shared with TI4/INT3	Shared with TO3/INT2	Shared with TO1	Shared with TI0/INT1
P8	Port 8 Register	18H	P87	P86	P85	P84	P83	P82	P81	P80
			*R/W							
			Input mode (set to 1/pulled up)							
			Shared with Rx	Shared with Tx	Shared with SCLK1/CTS1	Shared with RxD1	Shared with TxD1	Shared with SCLK0/CTS0	Shared with RxD0	Shared with TxD0
P9	Port 9 Register	19H		P96	P95	P94	P93	P92	P91	P90
				R/W						
				Input mode (output latch register cleared to 0)						
				Shared with TOA/TOB	Shared with TIB/INT8	Shared with TIA/INT7	Shared with TO9	Shared with TO8	Shared with TI9/INT6	Shared with TI8/INT5
PA	Port A Register	1EH	PA7	PA6	PA5	PA4	PA3	PA2	PA1	PA0
			R							
			Input-only							
			Shared with AN7	Shared with AN6	Shared with AN5	Shared with AN4	Shared with AN3/ADTRG	Shared with AN2	Shared with AN1	Shared with AN0

Note: When P5<P50> is cleared to 0 with P50 set as an $\overline{RD}$ pin (P5FC<P50F> = 1, the P50 $\overline{RD}$ signal is still output even when the internal address area is accessed (for PSRAM).

(2) Input/output port control (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P0CR	Port 0 Control Register	02H (RMW prohibited)	P07C	P06C	P05C	P04C	P03C	P02C	P01C	P00C
			W							
			0	0	0	0	0	0	0	0
			0: IN 1: OUT							
P1CR	Port 1 Control Register	04H (RMW prohibited)	P17C	P16C	P15C	P14C	P13C	P12C	P11C	P10C
			W							
			0	0	0	0	0	0	0	0
			0: IN 1: OUT							
P1FC	Port 1 Function Register	05H (RMW prohibited)	P17F	P16F	P15F	P14F	P13F	P12F	P11F	P10F
			W							
			0	0	0	0	0	0	0	0
			0: PORT 1: D15 to D8 (P1CR = 00H)							
P2CR	Port 2 Control Register	08H (RMW prohibited)	P27C	P26C	P25C	P24C	P23C	P22C	P21C	P20C
			W							
			0	0	0	0	0	0	0	0
			0: IN 1: OUT							
P2FC	Port 2 Function Register	09H (RMW prohibited)	P27F	P26F	P25F	P24F	P23F	P22F	P21F	P20F
			W							
			0	0	0	0	0	0	0	0
			0: PORT 1: A23 to A16 (P2CR = FFH)							
P3CR	Port 3 Control Register	0AH (RMW prohibited)	P37C	P36C	P35C	P34C	P33C	P32C	P31C	P30C
			W							
			0	0	0	0	0	0	0	0
			0: IN 1: OUT							
P3FC	Port 3 Function Register	0BH (RMW prohibited)	P37F	P36F	P35F	P34F	P33F	P32F	P31F	P30F
			W							
			0	0	0	0		0	0	0
			0: PORT 1: A15 to A8 (P3CR = FFH)							
P4CR	Port 4 Control Register	0EH (RMW prohibited)	P47C	P46C	P45C	P44C	P43C	P42C	P41C	P40C
			W							
			0	0	0	0	0	0	0	0
			0: IN 1: OUT							
P4FC	Port 4 Function Register	0FH (RMW prohibited)	P47F	P46F	P45F	P44F	P43F	P42F	P41F	P40F
			W							
			0	0	0	0	0	0	0	0
			0: PORT 1: A7 to A0 (P4CR = FFH)							
P5CR	Port 5 Control Register	10H (RMW prohibited)		P56C	P55C	P54C	P53C	P52C		
				W						
				0	0	0	0	0		
				0: IN 1: OUT						
P5FC	Port 5 Function Register	11H (RMW prohibited)	P57F			P54F	P53F	P52F	P51F	P50F
			W				W			
			1			0	0	0	0	0
			0: PORT 1: CLKOUT			0: PORT 1: $\overline{BUSAK}$	0: PORT 1: $\overline{BUSRQ}$	0: PORT 1: $\overline{HWR}$	0: PORT 1: $\overline{WR}$	0: PORT 1: $\overline{RD}$

Input/output port control (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
P6CR	Port 6 Control Register	14H (RMW prohibited)					P63C	P62C	P61C	P60C
			W							
			0	0	0	0	0	0	0	0
			0: IN						1: OUT	
P6FC	Port 6 Function Register	15H (RMW prohibited)					P63F	P62F	P61F	P60F
			W							
			0	0	0	0	0	0	0	0
			0: PORT 1: SCLK	0: PORT 1: MISO	0: PORT 1: MOSI	0: PORT 1: $\overline{SS}$				
P7CR	Port 7 Control Register	16H (RMW prohibited)			P75C	P74C	P73C	P72C	P71C	P70C
			W							
			0	0	0	0	0	0	0	0
			0: IN				1: OUT			
P7FC	Port 7 Function Register	17H (RMW prohibited)			P75F	P74F		P72F	P71F	
			W					W		
			0	0	0	0		0	0	
			0: PORT 1: TO7	0: PORT 1: TO5				0: PORT 1: TO3	0: PORT 1: TO1	
P8CR	Port 8 Control Register	1AH (RMW prohibited)	P87C	P86C	P85C	P84C	P83C	P82C	P81C	P80C
			W							
			0	0	0	0	0	0	0	0
			0: IN				1: OUT			
P8FC	Port 8 Function Register	1BH (RMW prohibited)		P86F	P85F		P83F	P82F		P80F
				W			W			W
				0	0		0	0		0
				0: PORT 1: Tx	0: PORT 1: SCLK1 /$\overline{CTS1}$		0: PORT 1: TxD1	0: PORT 1: SCLK0 /$\overline{CTS0}$		0: PORT 1: TxD0
P9CR	Port 9 Control Register	1CH (RMW prohibited)		P96C	P95C	P94C	P93C	P92C	P91C	P90C
			W							
				0	0	0	0	0	0	0
			0: IN				1: OUT			
P9FC	Port 9 Function Register	1DH (RMW prohibited)	TOS1	P96F			P93F	P92F		
			W				W			
			0	0			0	0		
			0: TOA 1: TOB	0: PORT 1: TOA / TOB			0: PORT 1: TO9	0: PORT 1: TO8		

(3) Timer control (1/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
T8RUN	8-bit Timer Run Control Register	20H	T7RUN	T6RUN	T5RUN	T4RUN	T3RUN	T2RUN	T1RUN	T0RUN
			R/W							
			0	0	0	0	0	0	0	0
			8-bit timer 7 0: Stop and clear 1: Count	8-bit timer 6 0: Stop and clear 1: Count	8-bit timer 5 0: Stop and clear 1: Count	8-bit timer 4 0: Stop and clear 1: Count	8-bit timer 3 0: Stop and clear 1: Count	8-bit timer 2 0: Stop and clear 1: Count	8-bit timer 1 0: Stop and clear 1: Count	8-bit timer 0 0: Stop and clear 1: Count
TRDC	Timer Register Double Buffer Control Register	21H					TR6DE	TR4DE	TR2DE	TR0DE
							R/W			
							0	0	0	0
							TREG6 double buffer 0: Disable 1: Enable	TREG4 double buffer 0: Disable 1: Enable	TREG2 double buffer 0: Disable 1: Enable	TREG0 double buffer 0: Disable 1: Enable
TREG0	8-bit Timer Register 0	22H (RMW prohibited)	–							
			W							
			Undefined							
TREG1	8-bit Timer Register 1	23H (RMW prohibited)	–							
			W							
			Undefined							
T01 MOD	8-bit Timer 0, 1 Mode Control Register	24H	T01M1	T01M0	PWM01	PWM00	T1CLK1	T1CLK0	T0CLK1	T0CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Timer 0, 1 operating mode setting 00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		PWM0 cycle selection 00: Don't care 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		Timer 1 input clock selection 00: TO0TRG 01: ϕT1 10: ϕT16 11: ϕT256		Timer 0 input clock selection 00: TI0 input 01: ϕT1 10: ϕT4 11: ϕT16	
T02 FFCR	8-bit Timer 0, 2 Flip-Flop Control Register	25H	FF3C1	FF3C0	FF3IE	FF3IS	FF1C1	FF1C0	FF1IE	FF1IS
			W		R/W		W		R/W	
			1	1	0	0	1	1	0	0
			00: Invert TFF3 01: Set TFF3 10: Clear TFF3 11: Don't care		TFF3 inversion control 0: Disable 1: Enable	0: Inversion by timer 2 1: Inversion by timer 3	00: Invert TFF1 01: Set TFF1 10: Clear TFF1 11: Don't care		TFF1 inversion control 0: Disable 1: Enable	0: Inversion by timer 0 1: Inversion by timer 1
TREG2	8-bit Timer Register 2	26H (RMW prohibited)	–							
			W							
			Undefined							
TREG3	8- bit Timer Register 3	27H (RMW prohibited)	–							
			W							
			Undefined							
T23 MOD	8-bit Timer 2, 3 Mode Control Register	28H	T23M1	T23M0	PWM21	PWM20	T3CLK1	T3CLK0	T2CLK1	T2CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Timer 2, 3 operating mode setting 00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		PWM2 cycle selection 00: Don't care 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		Timer 3 input clock selection 00: TO2TRG 01: ϕT1 10: ϕT16 11: ϕT256		Timer 2 input clock selection 00: Don't care 01: ϕT1 10: ϕT4 11: ϕT16	

Timer control (2/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
TREG4	8-bit Timer Register4	29H (RMW prohibited)	–							
			W							
			Undefined							
TREG5	8-bit Timer Register5	2AH (RMW prohibited)	–							
			W							
			Undefined							
T45 MOD	8-bit Timer 4, 5 Mode Control Register	2BH	T45M1	T45M0	PWM41	PWM40	T5CLK1	T5CLK0	T4CLK1	T4CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Timer 4, 5 operating mode setting 00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		PWM4 cycle selection 00: Don't care 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		Timer 5 input clock selection 00: TO4TRG 01: ϕT1 10: ϕT16 11: ϕT256		Timer 4 input clock selection 00: TI4 input 01: ϕT1 10: ϕT4 11: ϕT16	
T46 FFCR	8-bit Timer 4, 6 Flip-Flop Control Register	2CH	FF7C1	FF7C0	FF7IE	FF7IS	FF5C1	FF5C0	FF5IE	FF5IS
			W		R/W		W		R/W	
			1	1	0	0	1	1	0	0
			00: Invert TFF7 01: Set TFF7 10: Clear TFF7 11: Don't care		TFF7 inversion control 0: Disable 1: Enable	0: Inversion by timer 6 1: Inversion by timer 7	00: Invert TFF5 01: Set TFF5 10: Clear TFF5 11: Don't care		TFF5 inversion control 0: Disable 1: Enable	0: Inversion by timer 4 1: Inversion by timer 5
TREG6	8-bit Timer Register6	2DH (RMW prohibited)	–							
			W							
			Undefined							
TREG7	8-bit Timer Register7	2EH (RMW prohibited)	–							
			W							
			Undefined							
T67 MOD	8-bit Timer 6, 7 Mode Control Register	2FH	T67M1	T67M0	PWM61	PWM60	T7CLK1	T7CLK0	T6CLK1	T6CLK0
			R/W							
			0	0	0	0	0	0	0	0
			Timer 6, 7 operating mode setting 00: 8-bit timer 01: 16-bit timer 10: 8-bit PPG 11: 8-bit PWM		PWM6 cycle selection 00: Don't care 01: $2^6 - 1$ 10: $2^7 - 1$ 11: $2^8 - 1$		Timer 7 input clock selection 00: TO6TRG 01: ϕT1 10: ϕT16 11: ϕT256		Timer 6 input clock selection 00: Don't care 01: ϕT1 10: ϕT4 11: ϕT16	
TREG8L	16-bit Timer Register8L	30H (RMW prohibited)	–							
			W							
			Undefined							
TREG8H	16-bit Timer Register8H	31H (RMW prohibited)	–							
			W							
			Undefined							
TREG9L	16-bit Timer Register9L	32H (RMW prohibited)	–							
			W							
			Undefined							
TREG9H	16-bit Timer Register9H	33H (RMW prohibited)	–							
			W							
			Undefined							

Timer control (3/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
CAP1L	Capture Register1L	34H	–							
			R							
			Undefined							
CAP1H	Capture Register1H	35H	–							
			R							
			Undefined							
CAP2L	Capture Register2L	36H	–							
			R							
			Undefined							
CAP2H	Capture Register2H	37H	–							
			R							
			Undefined							
T8MOD	16-bit Timer 8 Mode Control Register	38H	CAP2T9	EQ9T9	CAP1IN	CAP12M1	CAP12M0	CLE	T8CLK1	T8CLK0
			R/W		W	R/W				
			0	0	1	0	0	0	0	0
			TFF9 inversion trigger 0: Trigger Disable 1: Trigger Enable		0: Software capture 1: Don't care	Capture timing 00: Disable 01: TI8 ↑ TI9 ↑ 10: TI8 ↑ TI8 ↓ 11: TFF1 ↑ TFF1 ↓		Timer 8 up-counter control 0: Clear disabled 1: Clear at match with TREG9	Timer 8 input clock selection 00: TI8 input 01: ϕT1 10: ϕT4 11: ϕT16	
			At loading of up-counter value to CAP2	At match between up-counter and TREG9						
T8FFCR	16-bit Timer 8 Flip-Flop Control Register	39H	TFF9C1	TFF9C0	CAP2T8	CAP1T8	EQ9T8	EQ8T8	TFF8C1	TFF8C0
			W		R/W				W	
			1	1	0	0	0	0	1	1
			00: Invert TFF9 01: Set TFF9 10: Clear TFF9 11: Don't care		TFF8 inversion trigger 0: Trigger Disable 1: Trigger Enable				00: Invert TFF8 01: Set TFF8 10: Clear TFF8 11: Don't care	
					At loading of up-counter value to CAP2	At loading of up-counter value to CAP3	At match between up-counter and TREG9	At match between up-counter and TREG8		
T89CR	Timer 8/9 Control Register	3AH	–					–	DBAEN	DB8EN
			R/W					R/W		
			0					0	0	0
			Note: Always fixed to 0.					Note: Always fixed to 0.	TREGA double buffer 0: Disable 1: Enable	TREG8 double buffer 0: Disable 1: Enable
T16RUN	16-bit Timer Run Control Register	3BH	PRRUN		T9RUN	T8RUN				
			R/W		R/W					
			0		0	0				
			Prescaler 0: Stop and clear 1: Count		16-bit timer 9 0: Stop and clear 1: Count	16-bit timer 8 0: Stop and clear 1: Count				

Timer control (4/4)

Symbol	Name	Address	7	6	5	4	3	2	1	0
TREGAL	16-bit Timer RegisterAL	40H (RMW prohibited)	–							
			W							
			Undefined							
TREGAH	16-bit Timer RegisterAH	41H (RMW prohibited)	–							
			W							
			Undefined							
TREGBL	16-bit Timer RegisterBL	42H (RMW prohibited)	–							
			W							
			Undefined							
TREGBH	16b-it Timer RegisterBH	43H (RMW prohibited)	–							
			W							
			Undefined							
CAP3L	Capture Register3L	44H	–							
			R							
			Undefined							
CAP3H	Capture Register3H	45H	–							
			R							
			Undefined							
CAP4L	Capture Register4L	46H	–							
			R							
			Undefined							
CAP4H	Capture Register4H	47H	–							
			R							
			Undefined							
T9MOD	16-bit Timer 9 Mode Control Register	48H	CAP4TB	EQBTB	CAP3IN	CAP34M1	CAP34M0	CLE	T9CLK1	T9CLK0
			R/W		W	R/W				
			0	0	1	0	0	0	0	0
			TFFB inversion trigger 0: Trigger Disable 1: Trigger Enable		0: Software capture 1: Don't care	Capture timing 00: Disable 01: TIA ↑ TIB ↑ 10: TIA ↑ TIA ↓ 11: TFF1 ↑ TFF1 ↓		Timer 9 up-counter control 0: Clear disabled 1: Clear at match with TREGB	Timer 8 input clock selection 00: TIA input 01: ϕT1 10: ϕT4 11: ϕT16	
			At loading of up-counter value to CAP4	At match between up-counter and TREGB						
T9FFCR	16-bit Timer 9 Flip-Flop Control Register	49H	TFFBC1	TFFBC0	CAP4TA	CAP3TA	EQBTA	EQATA	TFFAC1	TFFAC0
			W		R/W				W	
			1	1	0	0	0	0	1	1
			00: Invert TFFB 01: Set TFFB 10: Clear TFFB 11: Don't care		TFFA inversion trigger 0: Trigger Disable 1: Trigger Enable				00: Invert TFFA 01: Set TFFA 10: Clear TFFA 11: Don't care	
					At loading of up-counter value to CAP4	At loading of up-counter value to CAP3	At match between up-counter and TREGB	At match between up-counter and TREGA		

(4) Serial channel control (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
SC0BUF	Serial Channel 0 Buffer Register	4CH	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
			TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0
			R (receive) /W (send)							
			Undefined							
SC0CR	Serial Channel 0 Control Register	4DH	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
			R	R/W		R (cleared to 0 when read)				
			Undefined	0	0	0	0	0	0	0
			Bit 8 of receive data	Parity 0: Odd 1: Even	Parity addition 0: Disable 1: Enable	1: Error Overrun	Parity	Framing	0: SCLK0 1: SCLK0	I/O interface mode clock selection 0: Baud rate generator 0 1: SCLK0 pin input
SC0-MOD	Serial Channel 0 Mode Control Register	4EH	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			Undefined	0	0	0	0	0	0	0
			Bit 8 of send data	Handshake function 0: CTS Disable 1: CTS Enable	Receive control 0: Disable 1: Enable	Wake-up function 0: Disable 1: Enable	Serial transfer mode selection 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		UART mode clock selection 00: TO2 trigger 01: Baud rate generator 0 10: Internal clock ϕ1 11: SCLK0 pin input (external clock)	
BR0CR	Baud Rate Generator 0 Control Register	4FH	–		BR0CK1	BR0CK0	BR0S3	BR0S2	BR0S1	BR0S0
			R/W		R/W					
			0		0	0	0	0	0	0
			Note: Always fixed to 0.		Baud rate generator 0 input clock selection 00: ϕT0 (4/fc) 01: ϕT2 (16/fc) 10: ϕT8 (64/fc) 11: ϕT32 (256/fc)		Baud rate generator 0 divisor setting 0000: Divide by 16 0001: Divide by 1 (no division) to 1111: Divide by 15			
SC1BUF	Serial Channel 1 Buffer Register	50H	RB7	RB6	RB5	RB4	RB3	RB2	RB1	RB0
			TB7	TB6	TB5	TB4	TB3	TB2	TB1	TB0
			R (receive) /W (send)							
			Undefined							
SC1CR	Serial Channel 1 Control Register	51H	RB8	EVEN	PE	OERR	PERR	FERR	SCLKS	IOC
			R	R/W		R (cleared to 0 when read)			R/W	
			Undefined	0	0	0	0	0	0	0
			Bit 8 of receive data	Parity 0: Odd 1: Even	Parity addition 0: Disable 1: Enable	1: Error Overrun	Parity	Framing	0: SCLK1 1: SCLK1	I/O interface mode clock selection 0: Baud rate generator 1 1: SCLK1 pin input
SC1-MOD	Serial Channel 1 Mode Control Register	52H	TB8	CTSE	RXE	WU	SM1	SM0	SC1	SC0
			R/W							
			Undefined	0	0	0	0	0	0	0
			Bit 8 of send data	Handshake function 0: CTS Disable 1: CTS Enable	Receive control 0: Disable 1: Enable	Wake-up function 0: Disable 1: Enable	Serial transfer mode selection 00: I/O interface mode 01: 7-bit UART mode 10: 8-bit UART mode 11: 9-bit UART mode		UART mode clock selection 00: TO2 trigger 01: Baud rate generator 1 10: Internal clock ϕ1 11: SCLK1 pin input (external clock)	

Serial channel control (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
BR1CR	Baud Rate Generator 1 Control Register	53H	–		BR1CK1	BR1CK0	BR1S3	BR1S2	BR1S1	BR1S0
			R/W		R/W					
			0		0	0	0	0	0	0
			Always fixed to 0.		Baud rate generator 1 input clock selection 00: ϕT0 (4/fc) 01: ϕT2 (16/fc) 10: ϕT8 (64/fc) 11: ϕT32 (256/fc)		Baud rate generator 1 divisor setting 0000: Divide by 16 0001: Divide by 1 (no division) to 1111: Divide by 15			
ODE	Serial Open Drain Enable Register	58H				ODE4	ODE3	ODE2	ODE1	ODE0
						R/W				
						0	0	0	0	0
						P83 output settings 0: CMOS 1: Open drain	P80 output settings 0: CMOS 1: Open drain	P63 output settings 0: CMOS 1: Open drain	P62 output settings 0: CMOS 1: Open drain	P61 output settings 0: CMOS 1: Open drain

(5) Interrupt control (1/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTE-0AD	INT0/AD Enable Register	70H (RMW prohibited)	INTAD				INT0			
			IADC	IADM2	IADM1	IADM0	I0C	I0M2	I0M1	I0M0
			R/W	W			R/W (Note1)	W		
			0	0	0	0	0	0	0	0
INTE12	INT1/2 Enable Register	71H (RMW prohibited)	INT2				INT1			
			I2C	I2M2	I2M1	I2M0	I1C	I1M2	I1M1	I1M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTE34	INT3/4 Enable Register	72H (RMW prohibited)	INT4				INT3			
			I4C	I4M2	I4M1	I4M0	I3C	I3M2	I3M1	I3M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTE56	INT5/6 Enable Register	73H (RMW prohibited)	INT6				INT5			
			I6C	I6M2	I6M1	I6M0	I5C	I5M2	I5M1	I5M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTE78	INT7/8 Enable Register	74H (RMW prohibited)	INT8				INT7			
			I8C	I8M2	I8M1	I8M0	I7C	I7M2	I7M1	I7M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTET01	INTT0/1 Enable Register	75H (RMW prohibited)	INTT1 (timer 1)				INTT0 (timer 0)			
			IT1C	IT1M2	IT1M1	IT1M0	IT0C	IT0M2	IT0M1	IT0M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTET23	INTT2/3 Enable Register	76H (RMW prohibited)	INTT3 (timer 3)				INTT2 (timer 2)			
			IT3C	IT3M2	IT3M1	IT3M0	IT2C	IT2M2	IT2M1	IT2M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	
INTET45	INTT4/5 Enable Register	77H (RMW prohibited)	INTT5 (timer 5)				INTT4 (timer 4)			
			IT5C	IT5M2	IT5M1	IT5M0	IT4C	IT4M2	IT4M1	IT4M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTET67	INTT6/7 Enable Register	78H (RMW prohibited)	INTT7 (timer 7)				INTT6 (timer 6)			
			IT7C	IT7M2	IT7M1	IT7M0	IT6C	IT6M2	IT6M1	IT6M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTET89	INTTR8/9 Enable Register	79H (RMW prohibited)	INTTR9 (timer 8)				INTTR8 (timer 8)			
			IT9C	IT9M2	IT9M1	IT9M0	IT8C	IT8M2	IT8M1	IT8M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTETAB	INTTRA/B Enable Register	7AH (RMW prohibited)	INTTRB (timer 9)				INTTRA (timer 9)			
			ITBC	ITBM2	ITBM1	ITBM0	ITAC	ITAM2	ITAM1	ITAM0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0

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

Note 1: In INT0 level mode, the interrupt request flag cannot be cleared by writing 0 to <I0C>.

IxxC	Function (Read)	Function (Write)
0	Indicates no interrupt request generated	Clears interrupt request flag
1	Indicates interrupt request generated	----- Don't care -----

Interrupt control (2/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTEOV	INTTO8/9 Enable Register	7BH (RMW prohibited)	INTTO9				INTTO8			
			ITO9C	ITO9M2	ITO9M1	ITO9M0	ITO8C	ITO8M2	ITO8M1	ITO8M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTES0	INTRX0/ TX0 Enable Register	7CH (RMW prohibited)	INTTX0				INTRX0			
			ITX0C	ITX0M2	ITX0M1	ITX0M0	IRX0C	IRX0M2	IRX0M1	IRX0M0
			R/W	W			R (Note2)	W		
			0	0	0	0	0	0	0	0
INTES1	INTRX1/ TX1 Enable Register	7DH (RMW prohibited)	INTTX1				INTRX1			
			ITX1C	ITX1M2	ITX1M1	ITX1M0	IRX1C	IRX1M2	IRX1M1	IRX1M0
			R/W	W			R (Note2)	W		
			0	0	0	0	0	0	0	0
INTEC 01	INTCR/ CT Enable Register	7EH (RMW prohibited)	INTCT				INTCR			
			IC1C	IC1M2	IC1M1	IC1M0	IC0C	IC0M2	IC0M1	IC0M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTEC 2S0	INTCG/ SE0 Enable Register	7FH (RMW prohibited)	INTSE0				INTCG			
			ISE0C (Note3)	ISE0M2	ISE0M1	ISE0M0	IC2C	IC2M2	IC2M1	IC2M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTETC 01	INTTC0/1 Enable Register	80FH (RMW prohibited)	INTTC1				INTTC0			
			ITC1C	ITC1M2	ITC1M1	ITC1M0	ITC0C	ITC0M2	ITC0M1	ITC0M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTETC 23	INTTC2/3 Enable Register	81H (RMW prohibited)	INTTC3				INTTC2			
			ITC3C	ITC3M2	ITC3M1	ITC3M0	ITC2C	ITC2M2	ITC2M1	ITC2M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTESE 12	INTSE1/2 Enable Register	82H (RMW prohibited)	INTSE2				INTSE1			
			ISE2C	ISE2M2	ISE2M1	ISE2M0	ISE1C	ISE1M2	ISE1M1	ISE1M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0

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

IxxC	Function (Read)	Function (Write)
0	Indicates no interrupt request generated	Clears interrupt request flag
1	Indicates interrupt request generated	----- Don't care -----

Note 2: As <IRX0C>, <IRX1C> are read-only, an interrupt request cannot be cleared by writing 0 to these flags.
Note 3: Please clear <ISE0C> by writing 0 after clearing the status flag <SEF> or <MODF> of SEI.

Interrupt control (3/3)

Symbol	Name	Address	7	6	5	4	3	2	1	0
IIMC	Interrupt Input Mode Contorol Register	59H (RMW prohibited)			–			I0IE	I0LE	NMIREE
					W			W		
					0			0	0	0
					Note: Always set to 0			INT0 input 0: Disable 1: Enable	INT0 0: ↑ edge 1: level	$\overline{\text{NMI}}$ 0: ↓ edge 1: ↑ ↓ edge
DMA0V	Micro DMA 0 Start Vector Register	5AH (RMW prohibited)	DMA0V7	DMA0V6	DMA0V5	DMA0V4	DMA0V3	DMA0V2		
			W							
			0	0	0	0	0	0		
			Micro DMA0 start vector							
DMA1V	Micro DMA 1 Start Vector Register	5BH (RMW prohibited)	DMA1V7	DMA1V6	DMA1V5	DMA1V4	DMA1V3	DMA1V2		
			W							
			0	0	0	0	0	0		
			Micro DMA1 start vector							
DMA2V	Micro DMA 2 Start Vector Register	5CH (RMW prohibited)	DMA2V7	DMA2V6	DMA2V5	DMA2V4	DMA2V3	DMA2V2		
			W							
			0	0	0	0	0	0		
			Micro DMA2 start vector							
DMA3V	Micro DMA 3 Start Vector Register	5DH (RMW prohibited)	DMA3V7	DMA3V6	DMA3V5	DMA3V4	DMA3V3	DMA3V2		
			W							
			0	0	0	0	0	0		
			Micro DMA3 start vector							

Note: The micro DMA software start is activated in the write cycle of SDMACR0/1/2/3 (6AH/6BH/6CH/6DH). (Data values are not affected by a software start.)

(6) Watchdog timer control

Symbol	Name	Address	7	6	5	4	3	2	1	0
WD-MOD	Watch Dog Timer Mode Control Register	6EH	WDTE	WDTP1	WDTP0	WARM	HALTM1	HALTM0	RESCR	DRVE
			R/W							
			1	0	0	0	0	0	0	0
			WDT control 0: Disable 1: Enable	WDT detection time selection 00: 2^{16}/fc 01: 2^{18}/fc 10: 2^{20}/fc 11: 2^{22}/fc		Warm-up time 0: 2^{14}/fc 1: 2^{16}/fc	HALT mode selection 00: RUN mode 01: STOP mode 10: IDLE1 mode 11: IDLE2 mode		1: Perform internal reset on runaway detection	1: Drive pins in STOP mode
WDCR	Watch Dog Timer Control Register	6FH (RMW prohibited)	–							
			W							
			–							
			B1H: WDT disable code 4EH: WDT clear code							

(7) Bus width/wait controller (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
WAITC0	Block 0 Bus Width/ WAIT Control Register	90H (RMW prohibited)	B0E				B0BUS	B0W2	B0W1	B0W0
			W				W			
			0				0	0	0	0
			0: Disable 1: Enable				Data bus width selection 0: 16-bit 1: 8-bit	000: 2WAIT 001: 1WAIT 010: 1WAIT + N 011: 0WAIT	100: NWAIT 101 110 111	Do not set
WAITC1	Block 1 Bus Width/ WAIT Control Register	91H (RMW prohibited)	B1E				B1BUS	B1W2	B1W1	B1W0
			W				W			
			0				0	0	0	0
			0: Disable 1: Enable				Data bus width selection 0: 16-bit 1: 8-bit	000: 2WAIT 001: 1WAIT 010: 1WAIT + N 011: 0WAIT	100: NWAIT 101 110 111	Do not set
WAITC2	Block 2 Bus Width/ WAIT Control Register	92H (RMW prohibited)	B2E	B2M			B2BUS	B2W2	B2W1	B2W0
			W				W			
			1	0			0	0	0	0
			0: Disable 1: Enable	0: 16M 1:Address specification area			Data bus width selection 0: 16-bit 1: 8-bit	000: 2WAIT 001: 1WAIT 010: 1WAIT + N 011: 0WAIT	100: NWAIT 101 110 111	Do not set
WAITC3	Block 3 Bus Width/ WAIT Control Register	93H (RMW prohibited)	B3E				B3BUS	B3W2	B3W1	B3W0
			W				W			
			0				0	0	0	0
			0: Disable 1: Enable				Data bus width selection 0: 16-bit 1: 8-bit	000: 2WAIT 001: 1WAIT 010: 1WAIT + N 011: 0WAIT	100: NWAIT 101 110 111	Do not set
WAITCEX	External Bus Width/ WAIT Control Register	9CH (RMW prohibited)					BEXBUS	BEXW2	BEXW1	BEXW0
							W			
							0	0	0	0
							Data bus width selection 0: 16-bit 1: 8-bit	000: 2WAIT 001: 1WAIT 010: 1WAIT + N 011: 0WAIT	100: NWAIT 101 110 111	Do not set
MSAR0	Memory Start Address Register 0	94H	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			Start address A23 to A16 setting							
MAMR0	Memory Address Mask Register 0	95H	V20	V19	V18	V17	V16	V15	V14 to 9	V8
			R/W							
			1	1	1	1	1	1	1	1
			Address area 0 size setting 0: Used for address comparison							
MSAR1	Memory Start Address Register 1	96H	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			Start address A23 to A16 setting							
MAMR1	Memory Address Mask Register 1	97H	V21	V20	V19	V18	V17	V16	V15 to 9	V8
			R/W							
			1	1	1	1	1	1	1	1
			Address area 1 size setting 0: Used for address comparison							

Bus width/wait controller (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
MSAR2	Memory Start Address Register 2	98H	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			Start address A23 to A16 setting							
MAMR2	Memory Address Mask Register 2	99H	V22	V21	V20	V19	V18	V17	V16	V15
			R/W							
			1	1	1	1	1	1	1	1
			Address area 2 size setting 0: Used for address comparison							
MSAR3	Memory Start Address Register 3	9AH	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			Start address A23 to A16 setting							
MAMR3	Memory Address Mask Register 3	9BH	V22	V21	V20	V19	V18	V17	V16	V15
			R/W							
			1	1	1	1	1	1	1	1
			Address area 3 size setting 0: Used for address comparison							

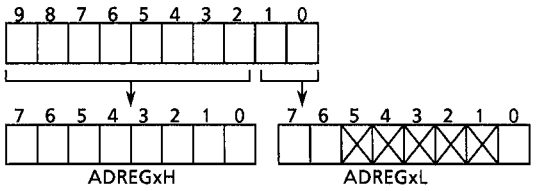
(8) AD converter control (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
ADMOD 0	AD Mode Control Register 0	5EH	EOCF	ADBF	–	–	ITM0	REPET	SCAN	ADS
			R		R/W					
			0	0	0	0	0	0	0	0
			AD conversion end flag 0: Conversion in progress 1: Conversion complete	AD conversion busy flag 0: Conversion idle 1: Conversion in progress	Note: Always fixed to 0.	Note: Always fixed to 0.	Interrupt specification in channel fixed repeat conversion mode 0: Every conversion 1: Every fourth conversion	Repeat mode specification 0: Single conversion mode 1: Repeat conversion mode	Scan mode specification 0: Conversion channel fixed mode 1: Conversion channel scan mode	AD conversion start 0: Don't Care 1: Conversion start Note: Always read as 0.

AD converter control (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
ADMOD 1	AD Mode Control Register 1	5FH	VREFON				ADTRGE	ADCH2	ADCH1	ADCH0
			R/W				R/W			
			1				0	0	0	0
			VREF application control 0:OFF 1:ON				External trigger start control 0:Enable 1:Disable	Analog input channel selection		
AD REG04L	AD Conversion Result Register 0/4 Low	60H	ADR01	ADR00						ADR0RF
			R							R
			Undefined							0
			Stores lower 2 bits of AD conversion result							AD conversion result storage flag
AD REG04H	AD Conversion Result Register 0/4 High	61H	ADR09	ADR08	ADR07	ADR06	ADR05	ADR04	ADR03	ADR02
			R							
			Undefined							
			Stores upper 8 bits of AD conversion result							
AD REG15L	AD Conversion Result Register 1/5 Low	62H	ADR11	ADR10						ADR1RF
			R							R
			Undefined							0
			Stores lower 2 bits of AD conversion result							AD conversion result storage flag
AD REG15H	AD Conversion Result Register 1/5 High	63H	ADR19	ADR18	ADR17	ADR16	ADR15	ADR14	ADR13	ADR12
			R							
			Undefined							
			Stores upper 8 bits of AD conversion result							
AD REG26L	AD Conversion Result Register 2/6 Low	64H	ADR21	ADR20						ADR2RF
			R							R
			Undefined							0
			Stores lower 2 bits of AD conversion result							AD conversion result storage flag
AD REG26H	AD Conversion Result Register 2/6 High	65H	ADR29	ADR28	ADR27	ADR26	ADR25	ADR24	ADR23	ADR22
			R							
			Undefined							
			Stores upper 8 bits of AD conversion result							
AD REG37L	AD Conversion Result Register 3/7 Low	66H	ADR31	ADR30						ADR3RF
			R							R
			Undefined							0
			Stores lower 2 bits of AD conversion result							AD conversion result storage flag
AD REG37H	AD Conversion Result Register 3/7 High	67H	ADR39	ADR38	ADR37	ADR36	ADR35	ADR34	ADR33	ADR32
			R							
			Undefined							
			Stores upper 8 bits of AD conversion result							

Channel x AD conversion result



- Bits 5 to 1 of ADREGxL are always read as 1. Bit 0 is the AD conversion result storage flag <ADRxRF>.
  When the AD conversion result is stored, the flag is set to 1. When either of the registers (ADREGxH, ADREGxL) are read, the flag is cleared to 0.

(9) Serial expansion interface control

Symbol	Name	Address	7	6	5	4	3	2	1	0
SECR	SEI Control Register	9DH (RMW prohibited)	SEIE	SEE	BOS	MSTR	CPOL	CPHA	SER1	SER0
			R/W							
			0	0	0	0	0	1	0	0
			SEI interrupt 0: Disabled 1: Enabled	SEI operation 0: Stopped 1: Operating	Bit order select 0: MSB first 1: LSB first	Mode select 0: Slave 1: Master	Clock polarity select See Fig.3.12.2, 3.12.3	Clock phase select See Fig.3.12.2, 3.12.3	SEI transfer rate select See table 3.12.1	
SESR	SEI Status Register	9EH (Compati-bility mode)	SEF	WCOL	SOVF	–	–	–	–	TMSE
			R							R/W
			0	0	0					0
			SEI transfer complete flag 1: Transfer completed	Write collision flag 1: Write collided	Overflow flag (slave) 1: Overflow occurred					SEI mode select 0: Compati-bility mode 1: Micro DMA mode
		9EH (Micro DMA mode) (RMW prohibited)	–	WCOL	SOVF	–	TSRC	TSTC	TASM	TMSE
				R			R		R/W	
				0	0		0	0	0	0
				Write collision flag 1: Write collided	Overflow flag (slave) 1: Overflow occurred		SEI receive complete flag 1: Receive completed	SEI transmit complete flag 1: transmit completed	SEI automated shift mode (master) INTSE0 mask (slave)	SEI mode select 0: Compati-bility mode 1: Micro DMA mode
SEDR	SEI Data Register	9FH	SED7	SED6	SED5	SED4	SED3	SED2	SED1	SED0
			R (receive) / W (transmit)							
			0	0	0	0	0	0	0	0

(10) CAN controller (1/5)

Symbol	Name	Address	7	6	5	4	3	2	1	0
MI0L	Message Identifier 0L	MBn* + 00H (RMW prohibited)	ID23	ID22	ID21	ID20	ID19	ID18	ID17	ID16
			R/W							
			–	–	–	–	–	–	–	–
MI0H	Message Identifier 0H	MBn* + 01H (RMW prohibited)	IDE	GAME	RFH	ID28	ID27	ID26	ID25	ID24
			R/W							
			–	–	–	–	–	–	–	–
MI1L	Message Identifier 1L	MBn* + 02H (RMW prohibited)	ID7	ID6	ID5	ID4	ID3	ID2	ID1	ID0
			R/W							
			–	–	–	–	–	–	–	–
MI1H	Message Identifier 1H	MBn* + 03H (RMW prohibited)	ID15	ID14	ID13	ID12	ID11	ID10	ID9	ID8
			R/W							
			–	–	–	–	–	–	–	–
MCFL	Message Control Field L	MBn* + 04H (RMW prohibited)				RTR	DLC3	DLC2	DLC1	DLC0
						R/W				
						–	–	–	–	–
MCFH	Message Control Field H	MBn* + 05H (RMW prohibited)								
D0	Data 0	MBn* + 06H (RMW prohibited)	D07	D06	D05	D04	D03	D02	D01	D00
			R/W							
			–	–	–	–	–	–	–	–
D1	Data 1	MBn* + 07H (RMW prohibited)	D17	D16	D15	D14	D13	D12	D11	D10
			R/W							
			–	–	–	–	–	–	–	–
D 2	Data 2	MBn* + 08H (RMW prohibited)	D27	D26	D25	D24	D23	D22	D21	D20
			R/W							
			–	–	–	–	–	–	–	–
D3	Data 3	MBn* + 09H (RMW prohibited)	D37	D36	D35	D34	D33	D32	D31	D30
			R/W							
			–	–	–	–	–	–	–	–
D4	Data 4	MBn* + 0AH (RMW prohibited)	D47	D46	D45	D44	D43	D42	D41	D40
			R/W							
			–	–	–	–	–	–	–	–
D5	Data 5	MBn* + 0BH (RMW prohibited)	D57	D56	D55	D54	D53	D52	D51	D50
			R/W							
			–	–	–	–	–	–	–	–
D6	Data 6	MBn* + 0CH (RMW prohibited)	D67	D66	D65	D64	D63	D62	D61	D60
			R/W							
			–	–	–	–	–	–	–	–
D7	Data 7	MBn* + 0DH (RMW prohibited)	D77	D76	D75	D74	D73	D72	D71	D70
			R/W							
			–	–	–	–	–	–	–	–
TSVL	Time Stamp Value L	MBn* + 0EH	TSV7	TSV6	TSV5	TSV4	TSV3	TSV2	TSV1	TSV0
			R							
			–	–	–	–	–	–	–	–
TSVH	Time Stamp Value H	MBn* + 0FH	TSV15	TSV14	TSV13	TSV12	TSV11	TSV10	TSV9	TSV8
			R							
			–	–	–	–	–	–	–	–

* MBn = 2200H + n × 10H

CAN controller (2/5)

Symbol	Name	Address	7	6	5	4	3	2	1	0
MCL	Mailbox Configu-ration Register L	2300H	MC7	MC6	MC5	MC4	MC3	MC2	MC1	MC0
			R/W							
			0	0	0	0	0	0	0	0
MCH	Mailbox Configu-ration Register H	2301H	MC15	MC14	MC13	MC12	MC11	MC10	MC9	MC8
			R/W							
			0	0	0	0	0	0	0	0
MDL	Mailbox Direction Register L	2302H	MD7	MD6	MD5	MD4	MD3	MD2	MD1	MD0
			R/W							
			0	0	0	0	0	0	0	0
MDH	Mailbox Direction Register H	2303H	MD15	MD14	MD13	MD12	MD11	MD10	MD9	MD8
			R	R/W						
			1	0	0	0	0	0	0	0
TRSL	Transmission Request Set Register L	2304H (RMW Prohibited)	TRS7	TRS6	TRS5	TRS4	TRS3	TRS2	TRS1	TRS0
			R/S							
			0	0	0	0	0	0	0	0
TRSH	Transmission Request Set Register H	2305H (RMW Prohibited)		TRS14	TRS13	TRS12	TRS11	TRS10	TRS9	TRS8
				R/S						
				0	0	0	0	0	0	0
TAL	Transmission Acknow-ledge Register L	2308H (RMW Prohibited)	TA7	TA6	TA5	TA4	TA3	TA2	TA1	TA0
			R/C							
			0	0	0	0	0	0	0	0
TAH	Transmission Acknow-ledge Register H	2309H (RMW Prohibited)		TA14	TA13	TA12	TA11	TA10	TA9	TA8
				R/C						
				0	0	0	0	0	0	0
RMPL	Receive Message Pending Register L	230CH (RMW Prohibited)	RMP7	RMP6	RMP5	RMP4	RMP3	RMP2	RMP1	RMP0
			R/C							
			0	0	0	0	0	0	0	0
RMPH	Receive Message Pending Register H	230DH (RMW Prohibited)	RMP15	RMP14	RMP13	RMP12	RMP11	RMP10	RMP9	RMP8
			R/C							
			0	0	0	0	0	0	0	0
RMLL	Receive Message Lost Register L	230EH (RMW Prohibited)	RML7	RML6	RML5	RML4	RML3	RML2	RML1	RML0
			R/C							
			0	0	0	0	0	0	0	0
RMLH	Receive Message Lost Register H	230FH (RMW Prohibited)	RML15	RML14	RML13	RML12	RML11	RML10	RML9	RML8
			R/C							
			0	0	0	0	0	0	0	0

CAN controller (3/5)

Symbol	Name	Address	7	6	5	4	3	2	1	0
LAM0L	Local Acceptance Mask Register 0L	2310H	LAM23	LAM22	LAM21	LAM20	LAM19	LAM18	LAM17	LAM16
			R/W							
			0	0	0	0	0	0	0	0
LAM0H	Local Acceptance Mask Register 0H	2311H	LAMI			LAM28	LAM27	LAM26	LAM25	LAM24
			R/W					R/W		
			0			0	0	0	0	0
LAM1L	Local Acceptance Mask Register 1L	2312H	LAM7	LAM6	LAM5	LAM4	LAM3	LAM2	LAM1	LAM0
			R/W							
			0	0	0	0	0	0	0	0
LAM1H	Local Acceptance Mask Register 1H	2313H	LAM15	LAM14	LAM13	LAM12	LAM11	LAM10	LAM9	LAM8
			R/W							
			0	0	0	0	0	0	0	0
GAM0L	Global Acceptance Mask Register 0L	2314H	GAM23	GAM22	GAM21	GAM20	GAM19	GAM18	GAM17	GAM16
			R/W							
			0	0	0	0	0	0	0	0
GAM0H	Global Acceptance Mask Register 0H	2315H	GAMI			GAM28	GAM27	GAM26	GAM25	GAM24
			R/W					R/W		
			0			0	0	0	0	0
GAM1L	Global Acceptance Mask Register 1L	2316H	GAM7	GAM6	GAM5	GAM4	GAM3	GAM2	GAM1	GAM0
			R/W							
			0	0	0	0	0	0	0	0
GAM1H	Global Acceptance Mask Register 1H	2317H	GAM15	GAM14	GAM13	GAM12	GAM11	GAM10	GAM9	GAM8
			R/W							
			0	0	0	0	0	0	0	0
MCRL	Master Control Register L	2318H	CCR	SMR	HMR	WUBA	MTOS		TSCC	SRES
			R/W						W	
			1	0	0	0	0		0	0
MCRH	Master Control Register H	2319H							TSTLB	TSTERR
									R/W	
									0	0
GSRL	Global Status Register L	231AH	CCE	SMA	HMA		TSO	BO	EP	EW
			R				R			
			1	0	0		0	0	0	0
GSRH	Global Status Register H	231BH	MsgInSlot <3:0>				RM	TM		
			R							
			1	1	1	1	0	0		
BCR1L	Bit Configuration Register 1L	231CH	BRP7	BRP6	BRP5	BRP4	BRP3	BRP2	BRP1	BRP0
			R/W							
			0	0	0	0	0	0	0	0
BCR1H	Bit Configuration Register 1H	231DH								
BCR2L	Bit Configuration Register 2L	231EH	SAM	TSEG22	TSEG21	TSEG20	TSEG13	TSEG12	TSEG11	TSEG10
			R/W							
			0	0	0	0	0	0	0	0
BCR2H	Bit Configuration Register 2H	231FH							SJW1	SJW0
									R/W	
									0	0

CAN controller (4/5)

Symbol	Name	Address	7	6	5	4	3	2	1	0
GIFL	Global Interrupt Flag L	2320H (RMW prohibited)	RFPF	WUIF	RMLIF		TSOIF	BOIF	EPIF	WLIF
			R/C							
			0	0	0		0	0	0	0
GIFH	Global Interrupt Flag H	2321H (RMW prohibited)								
GIML	Global Interrupt Mask L	2322H	RFPM	WUIM	RMLIM		TSOIM	BOIM	EPIM	WLIM
			R/W							
			0	0	0	0 Note)	0	0	0	0
GIMH	Global Interrupt Mask H	2323H								
MBTIFL	Mailbox Transmit Int. Flag L	2324H (RMW prohibited)	MBTIF7	MBTIF6	MBTIF5	MBTIF4	MBTIF3	MBTIF2	MBTIF1	MBTIF0
			R/C							
			0	0	0	0	0	0	0	0
MBTIFH	Mailbox Transmit Int. Flag H	2325H (RMW prohibited)		MBTIF14	MBTIF13	MBTIF12	MBTIF11	MBTIF10	MBTIF9	MBTIF8
			R/C							
				0	0	0	0	0	0	0
MBRIFL	Mailbox Receive Int. Flag L	2326H (RMW prohibited)	MBRIF7	MBRIF6	MBRIF5	MBRIF4	MBRIF3	MBRIF2	MBRIF1	MBRIF0
			R/C							
			0	0	0	0	0	0	0	0
MBRIFH	Mailbox Receive Int. Flag H	2327H (RMW prohibited)	MBRIF15	MBRIF14	MBRIF13	MBRIF12	MBRIF11	MBRIF10	MBRIF9	MBRIF8
			R/C							
			0	0	0	0	0	0	0	0
MBIML	Mailbox Interrupt Mask L	2328H	MBIM7	MBIM6	MBIM5	MBIM4	MBIM3	MBIM2	MBIM1	MBIM0
			R/W							
			0	0	0	0	0	0	0	0
MBIMH	Mailbox Interrupt Mask H	2329H	MBIM15	MBIM14	MBIM13	MBIM12	MBIM11	MBIM10	MBIM9	MBIM8
			R/W							
			0	0	0	0	0	0	0	0
CDRL	Change Data Request Register L	232AH	CDR7	CDR6	CDR5	CDR4	CDR3	CDR2	CDR1	CDR0
			R/W							
			0	0	0	0	0	0	0	0
CDRH	Change Data Request Register H	232BH		CDR14	CDR13	CDR12	CDR11	CDR10	CDR9	CDR8
			R/W							
				0	0	0	0	0	0	0
RFPL	Remote Frame Pending Register L	232CH (RMW prohibited)	RFP7	RFP6	RFP5	RFP4	RFP3	RFP2	RFP1	RFP0
			R/C							
			0	0	0	0	0	0	0	0
RFPH	Remote Frame Pending Register H	232DH (RMW prohibited)	RFP15	RFP14	RFP13	RFP12	RFP11	RFP10	RFP9	RFP8
			R/C							
			0	0	0	0	0	0	0	0
CECL	CAN Error Counter L	232EH (RMW prohibited)	REC7	REC6	REC5	REC4	REC3	REC2	REC1	REC0
			R/W							
			0	0	0	0	0	0	0	0
CECH	CAN Error Counter H	232FH (RMW prohibited)	TEC7	TEC6	TEC5	TEC4	TEC3	TEC2	TEC1	TEC0
			R/W							
			0	0	0	0	0	0	0	0

CAN controller (5/5)

Symbol	Name	Address	7	6	5	4	3	2	1	0
TSPL	Time Stamp Prescaler L	2330H					TSP3	TSP2	TSP1	TSP0
			R/W							
							0	0	0	0
TSPH	Time Stamp Prescaler H	2331H								
TSCL	Time Stamp Counter L	2332H (RMW prohibited)	TSC7	TSC6	TSC5	TSC4	TSC3	TSC2	TSC1	TSC0
			R/W							
			0	0	0	0	0	0	0	0
TSCH	Time Stamp Counter H	2333H (RMW prohibited)	TSC15	TSC14	TSC13	TSC12	TSC11	TSC10	TSC9	TSC8
			R/W							
			0	0	0	0	0	0	0	0

6. Diagram of Equivalent Circuit in Port Block

- Reading circuit diagrams

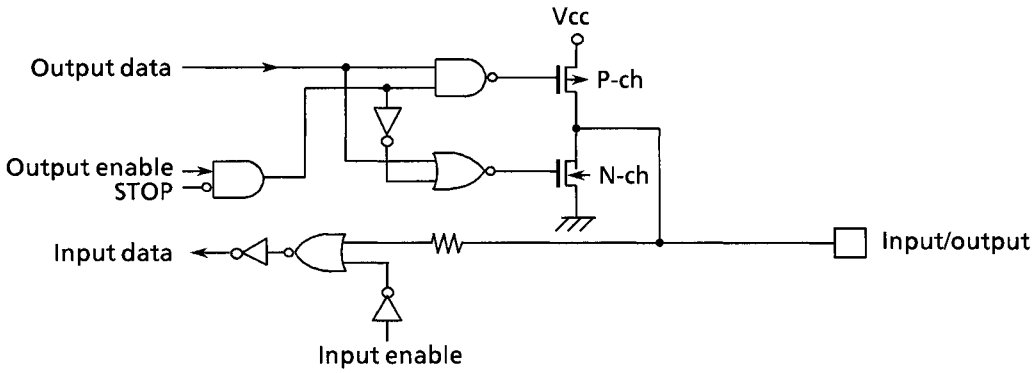
  The TMP95CU54A uses essentially the same gate symbols as the standard CMOS logic IC (74HCxxx) series.

  The following lists the special symbols.

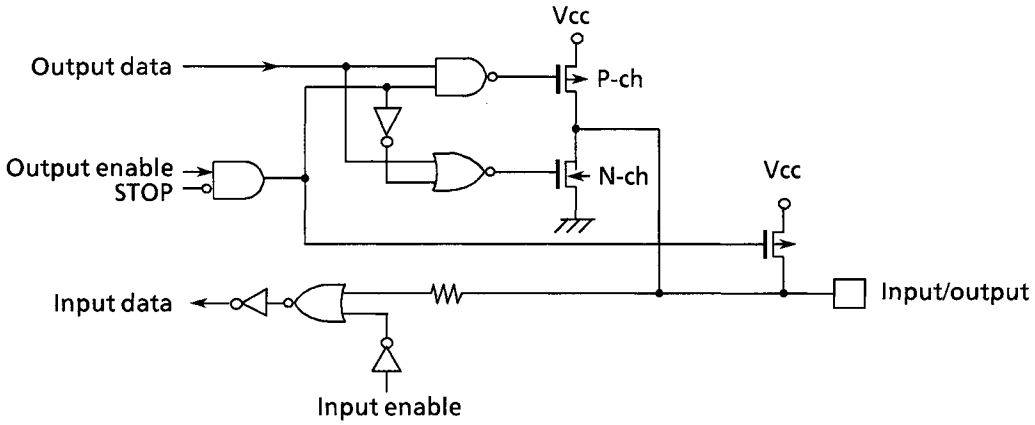
  STOP: This symbol sets the HALT mode setting register to STOP mode (WDMOD <HALTM1:0> = 0, 1). When the CPU executes the HALT instruction, STOP is active 1.
  Note that when the drive enable bit WDMOD<DRVE> is set to 1, STOP remains at 0.

- The input protection resistor operates in the range of tens to hundreds of ohms.

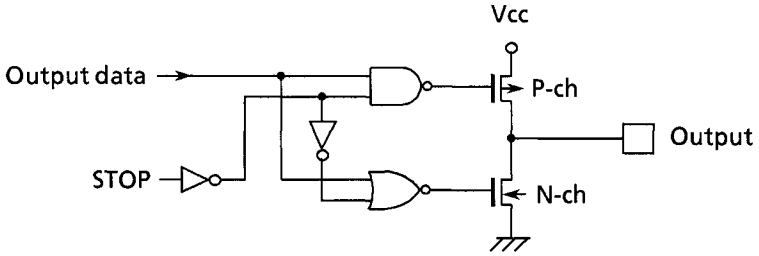
■ P0 (D0 to D7), P1 (D8 to D15), P2 (A16 to A23), P31 to P37 (A9 to A15), P4 (A0 to A7)



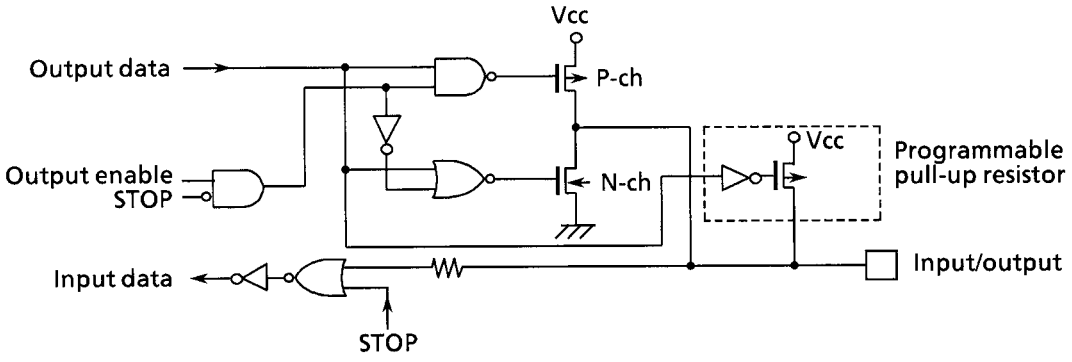
■ P30 (A8)



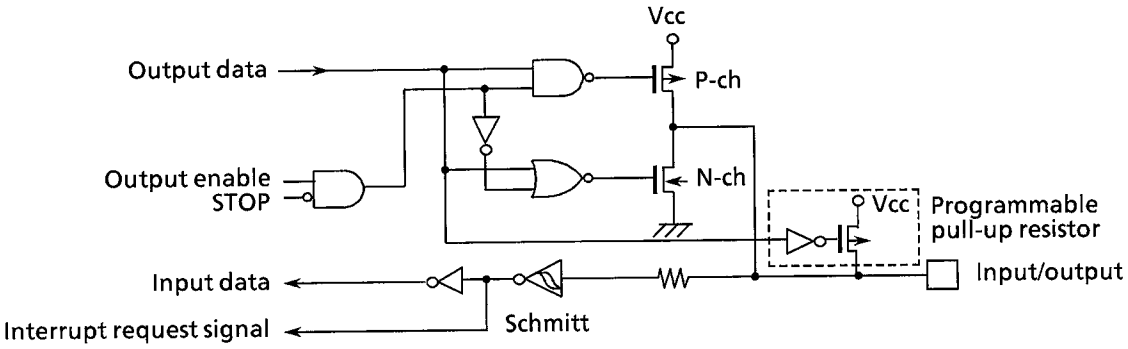
■ P50 ($\overline{RD}$), P51 ($\overline{WR}$)



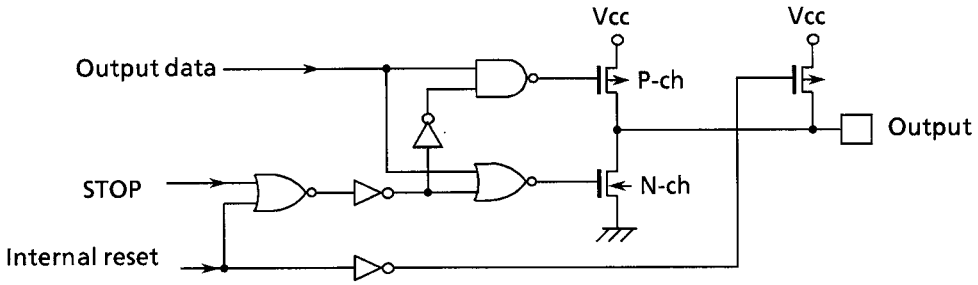
■ P52 to 55, P81, P82, P84, P85, P86, P87



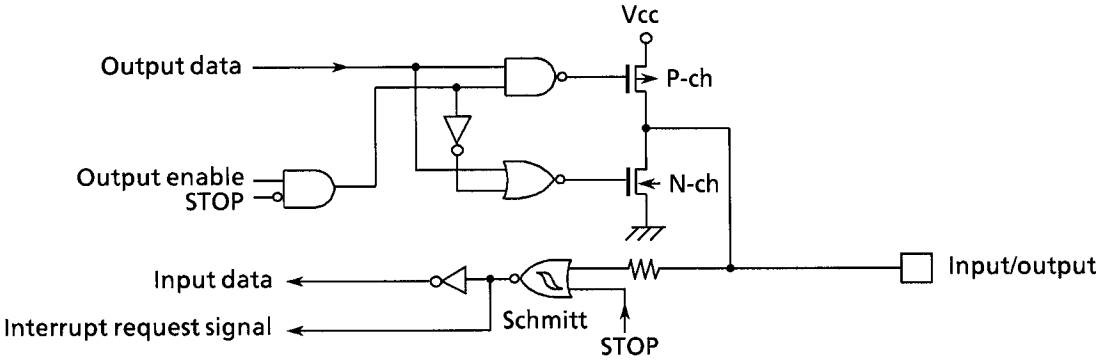
■ P56 (INT0)



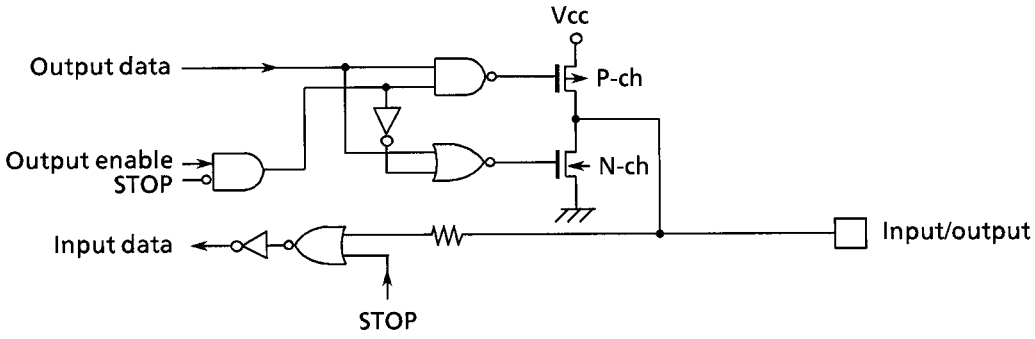
■ P57 (CLKOUT)



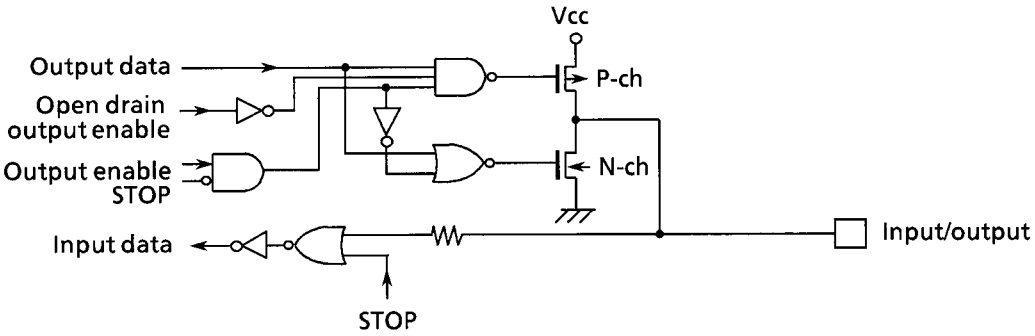
■ P70 (INT1), P72 (INT2), P73 (INT3), P75 (INT4)



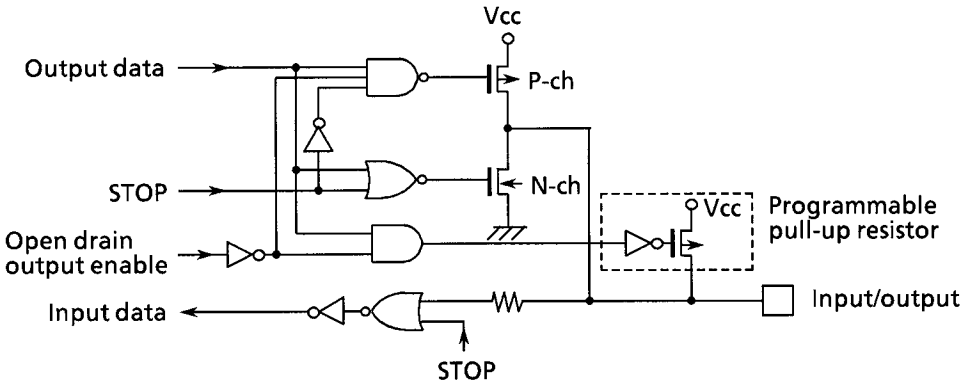
■ P60, P71, P74, P9



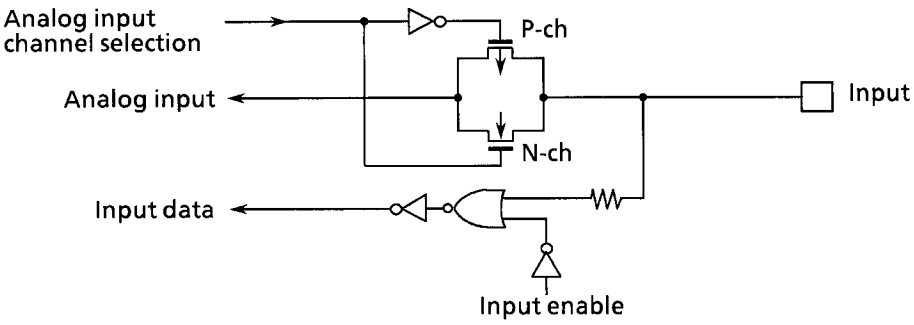
■ P61 (MOSI), P62 (MISO), P63 (SCLK)



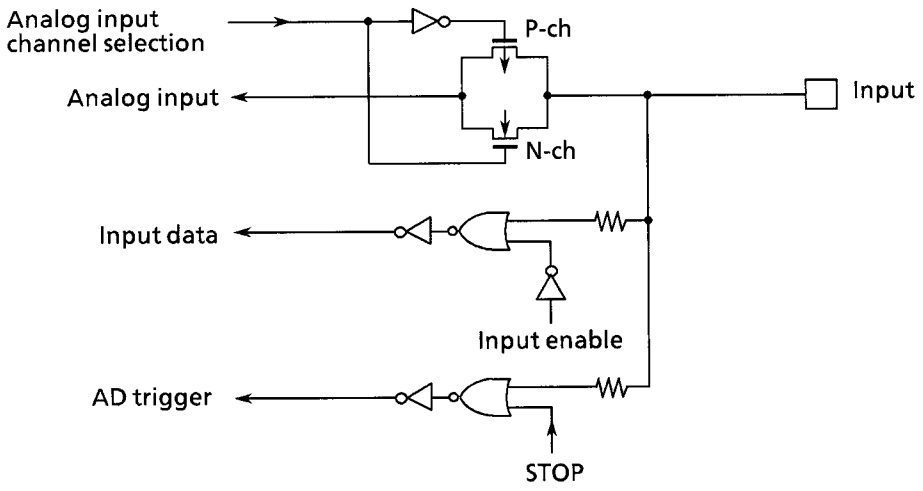
■ P80 (TxD0), P83 (TxD1)



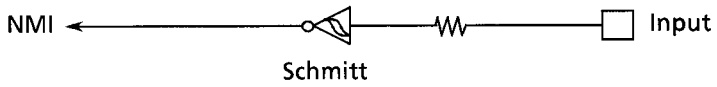
■ PA0 to 2 (AN0 to 2), PA4 to 7 (AN4 to 7)



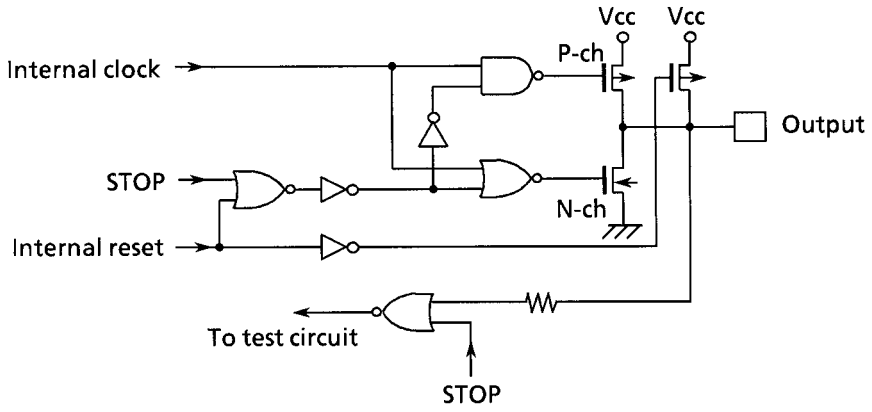
■ PA3 (AN3)



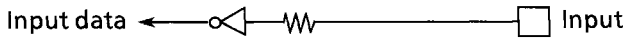
■ $\overline{\text{NMI}}$



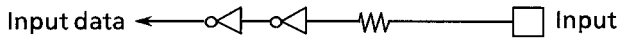
■ CLK



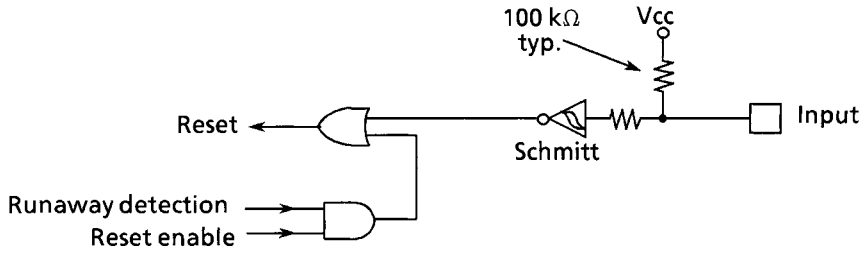
■ $\overline{EA}$



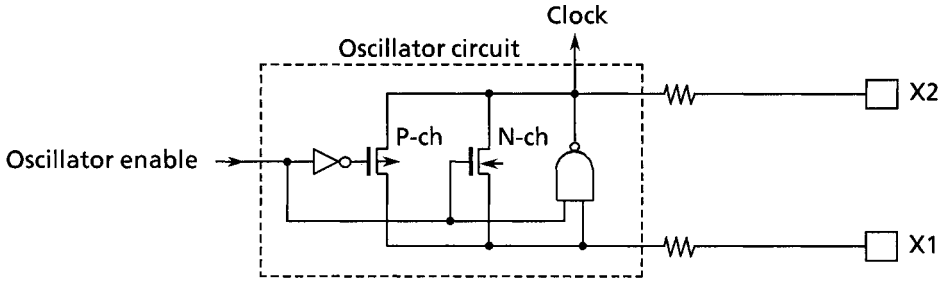
■ AM8/$\overline{16}$



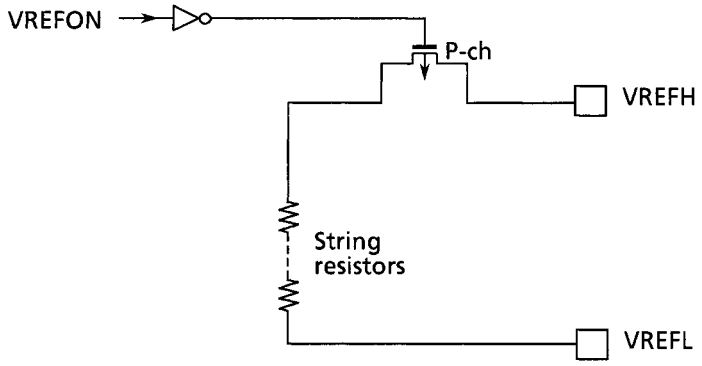
■ $\overline{RESET}$



■ X1, X2



■ VREFH, VREFL



7. Use Precautions and Restrictions

(1) Special Notations and Words

[1] Description of internal I/O registers: Register symbol<bit symbol>

Example: T8RUN<T0RUN> … The T0RUN bit of the T8RUN register

[2] Read-modify-write instructions

Instructions which tell the CPU to read the data in memory, manipulate them, then write them back to memory are called read-modify-write instructions.

Example 1) SET 3, (T8RUN) … Sets bit 3 of the T8RUN register.
Example 2) INC 1, (100H) … Adds 1 to the data at address 100H.

- TLCS-900 read-modify-write instructions

Conversion instruction

EX (mem), R

Arithmetic operations

ADD (mem), R/#	ADC (mem), R/#
SUB (mem), R/#	SBC (mem), R/#
INC #3, (mem)	DEC #3, (mem)

Logic operations

AND (mem), R/#	OR (mem), R/#
XOR (mem), R/#	

Bit manipulation

STCF #3/A, (mem)	SET #3, (mem)
RES #3, (mem)	TSET #3, (mem)
CHG #3, (mem)	

Rotate, shift

RLC (mem)	RRC (mem)
RL (mem)	RR (mem)
SLA (mem)	SRA (mem)
SLL (mem)	SRL (mem)
RLD A, (mem)	RRD A, (mem)

[3] One state

The single cycle resulting from dividing the oscillation frequency by 2 is called “one state”.

Example: At oscillation frequency 24 MHz
2/24 MHz = 83 ns = 1 state

(2) Use Precautions and Limitations

[1] $\overline{EA}$ pin, AM8/$\overline{16}$ pin

This pin is connected to the VCC pin. Do not alter the level while the pin is active.

[2] Warm-up counter

When releasing STOP mode (by interrupt, for example) in a system that uses an external oscillator, a warm-up time is required until the system clock is output. The warm-up counter operates during the warm-up time.

[3] Programmable pull-up resistor

The pull-up resistor of a port can only be set to programmable or non-programmable in input port mode. When using a port as an output port, its pull-up resistor cannot be set to programmable.

[4] Watchdog timer

As the watchdog timer is enabled after a reset, disable the watchdog timer when it is not required.

Note that during bus release, the I/O block, including the watchdog timer, still operates.

[5] CPU (Micro DMA)

Only "LDC cr, r" and "LDC r, cr" can write or read data to or from control registers (eg, transfer source register DMASx) in the CPU.

[6] As this device does not support minimum mode, do not use the MIN instruction.

[7] POP SR instruction

Please execute POP SR instruction during DI condition.

[8] Releasing the HALT mode by requesting an interruption

Usually, interrupts can release all halts status. However the interrupts (= NMI and INT0) which can release the HALT mode may not be able to do so if they are input during the period when the CPU is shifting to the HALT mode (for about 3 clock of fc) with IDLE1 or STOP mode (RUN and IDLE2 are not applicable to this case). (In this case, an interrupt request is kept on hold internally)

If another interrupt is generated after it has shifted to completely HALT mode, halt status can be released without difficulty. The priority of this interrupt is compared with that of the interrupt kept on hold internally, and the interrupt with the higher priority is handled first followed by the other interrupt.