TOSHIBA Original CMOS 16-Bit Microcontroller

TLCS-900/H Series

TMP95C001FG

TOSHIBA CORPORATION

Semiconductor Company

Preface

Thank you very much for making use of Toshiba microcomputer LSIs.
Before use this LSI, refer the section, "Points of Note and Restrictions".
Especially, take care below cautions.

CAUTION

How to release the HALT mode

Usually, interrupts can release all halts status. However, the interrupts = ($\overline{\text{NMI}}$, INT0), which can release the HALT mode may not be able to do so if they are input during the period CPU is shifting to the HALT mode (for about 3 clocks of X1) with IDLE or STOP mode. (In this case, an interrupt request is kept on hold internally.)

If another interrupt is generated after it has shifted to HALT mode completely, halt status can be released without difficultly. The priority of this interrupt is compare with that of the interrupt kept on hold internally, and the interrupt with higher priority is handled first followed by the other interrupt.

Document Change Notification

The purpose of this notification is to inform customers about the launch of the Pb-free version of the device. The introduction of a Pb-free replacement affects the datasheet. Please understand that this notification is intended as a substitute for a revision of the datasheet.

Changes to the datasheet may include the following, though not all of them may apply to this particular device.

1. Part number

   Example: TMPxxxxxxF → TMPxxxxxxFG

   All references to the previous part number were left unchanged in body text. The new part number is indicated on the prelims pages (cover page and this notification).

2. Package code and package dimensions

   Example: LQFP100-P-1414-0.50C → LQFP100-P-1414-0.50F

   All references to the previous package code and package dimensions were left unchanged in body text. The new ones are indicated on the prelims pages.

3. Addition of notes on lead solderability

   Now that the device is Pb-free, notes on lead solderability have been added.

4. RESTRICTIONS ON PRODUCT USE

   The previous (obsolete) provision might be left unchanged on page 1 of body text. A new replacement is included on the next page.

5. Publication date of the datasheet

   The publication date at the lower right corner of the prelims pages applies to the new device.

1. Part number

Previous Part Number (in Body Text)	New Part Number
TMP95C001F	TMP95C001FG

2. Package code and dimensions

Previous Package Code (in Body Text)	New Package Code
P-QFP64-1414-0.80A	QFP64-P-1414-0.80C

*: For the dimensions of the new package, see the attached Package Dimensions diagram.

3. Addition of notes on lead solderability

The following solderability test is conducted on the new device.

Solderability of lead free products

Test Parameter	Test Condition	Note
Solderability	Use of Sn-37Pb solder Bath Solder bath temperature = 230°C, Dipping time = 5 seconds The number of times = one, Use of R-type flux	Pass: Solderability rate until forming ≥ 95%
	Use of Sn-3.0Ag-0.5Cu solder bath Solder bath temperature = 245°C, Dipping time = 5 seconds The number of times = one, Use of R-type flux (use of lead free)	

4. RESTRICTIONS ON PRODUCT USE

The following replaces the "RESTRICTIONS ON PRODUCT USE" on page 1 of body text.

RESTRICTIONS ON PRODUCT USE

20070701-EN

- The information contained herein is subject to change without notice.
- TOSHIBA is continually working to improve the quality and reliability of its products. Nevertheless, semiconductor devices in general can malfunction or fail due to their inherent electrical sensitivity and vulnerability to physical stress. It is the responsibility of the buyer, when utilizing TOSHIBA products, to comply with the standards of safety in making a safe design for the entire system, and to avoid situations in which a malfunction or failure of such TOSHIBA products could cause loss of human life, bodily injury or damage to property.
  In developing your designs, please ensure that TOSHIBA products are used within specified operating ranges as set forth in the most recent TOSHIBA products specifications. Also, please keep in mind the precautions and conditions set forth in the "Handling Guide for Semiconductor Devices," or "TOSHIBA Semiconductor Reliability Handbook" etc.
- The TOSHIBA products listed in this document are intended for usage in general electronics applications (computer, personal equipment, office equipment, measuring equipment, industrial robotics, domestic appliances, etc.).These TOSHIBA products are neither intended nor warranted for usage in equipment that requires extraordinarily high quality and/or reliability or a malfunction or failure of which may cause loss of human life or bodily injury ("Unintended Usage"). Unintended Usage include atomic energy control instruments, airplane or spaceship instruments, transportation instruments, traffic signal instruments, combustion control instruments, medical instruments, all types of safety devices, etc.. Unintended Usage of TOSHIBA products listed in his document shall be made at the customer's own risk.
- The products described in this document shall not be used or embedded to any downstream products of which manufacture, use and/or sale are prohibited under any applicable laws and regulations.
- The information contained herein is presented only as a guide for the applications of our products. No responsibility is assumed by TOSHIBA for any infringements of patents or other rights of the third parties which may result from its use. No license is granted by implication or otherwise under any patents or other rights of TOSHIBA or the third parties.
- Please contact your sales representative for product-by-product details in this document regarding RoHS compatibility. Please use these products in this document in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances. Toshiba assumes no liability for damage or losses occurring as a result of noncompliance with applicable laws and regulations.
- 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.

5. Publication date of the datasheet

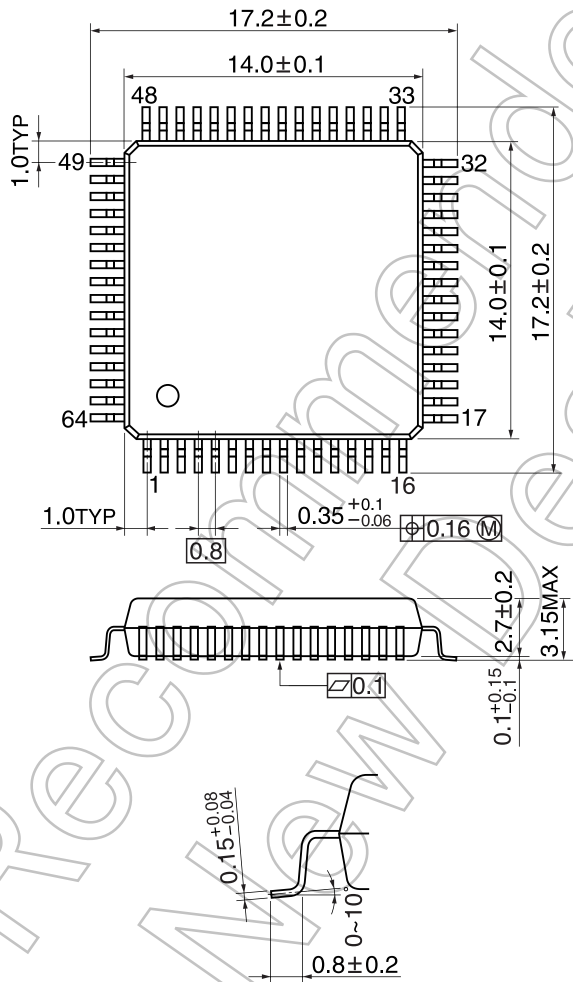
The publication date of this datasheet is printed at the lower right corner of this notification.

(Annex)

Package Dimensions

QFP64-P-1414-0.80C

Unit: mm



CMOS 16-Bit Microcontroller
TMP95C001F

1. Outline and Features

TMP95C001F is a 16-bit microcontroller of a high-speed 16-bit CPU (TLCS-900/H) core. It has only an indispensable function such as a wait controller, an interrupt controller, and etc.

TMP95C001F is presented in a 64-pin flat package. Its features are as follows.

(1) High-speed 16-bit CPU (TLCS-900/H_CPU)
- Instruction mnemonics upwardly compatible with TLCS-90/900
- 16M-byte linear address space
- General-purpose registers using register bank system
- 16-bit multiplication / division instructions, bit transfer / arithmetic instructions
- Micro DMA: four channels (640 ns / 2 bytes at 25 MHz)

(2) Minimum instruction execution time: 160 ns (at 25 MHz)

(3) Internal RAM : No
Internal ROM : No

(4) External memory expansion
- Expandable to 16 Mbytes (common to programs and data)
- External data bus width selection pin (AM8 / $\overline{16}$)
- Can use both 8- and 16-bit external buses··· dynamic bus sizing

(5) Wait controller : four blocks

(6) Interrupt function
- Interrupt sources : 20
  (Internal interrupt : 13
  External interrupt : 7)

(7) Standby function
Three HALT modes (RUN, IDLE, STOP)

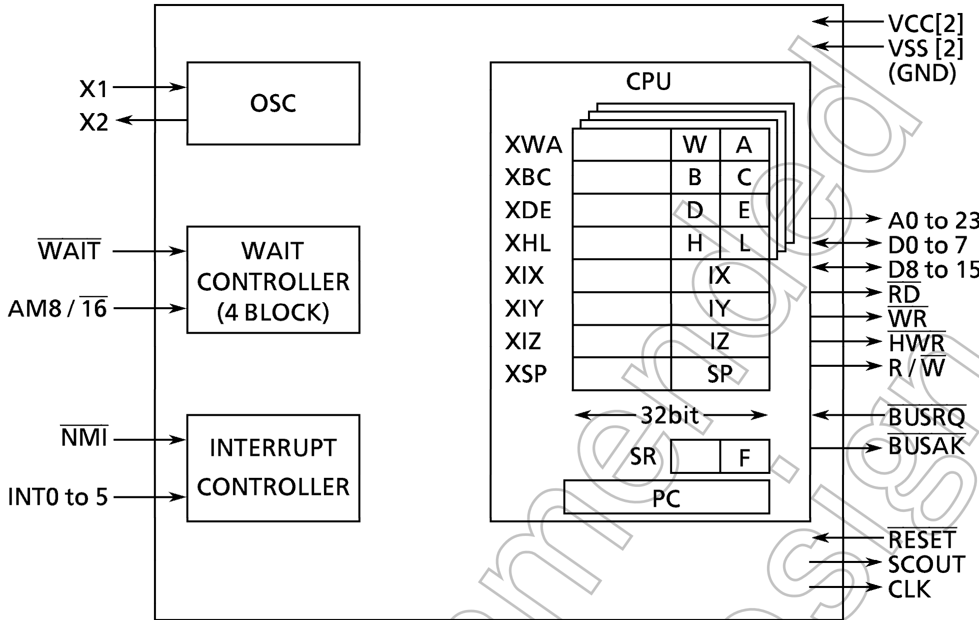


Figure 1 TMP95C001 Block Diagram

2. Pin Assignment and Functions

The assignment of input/output pins for TMP95C001F their name and outline functions are described below.

2.1 Pin Assignment

Figure 2.1 shows pin assignment of TMP95C001F.

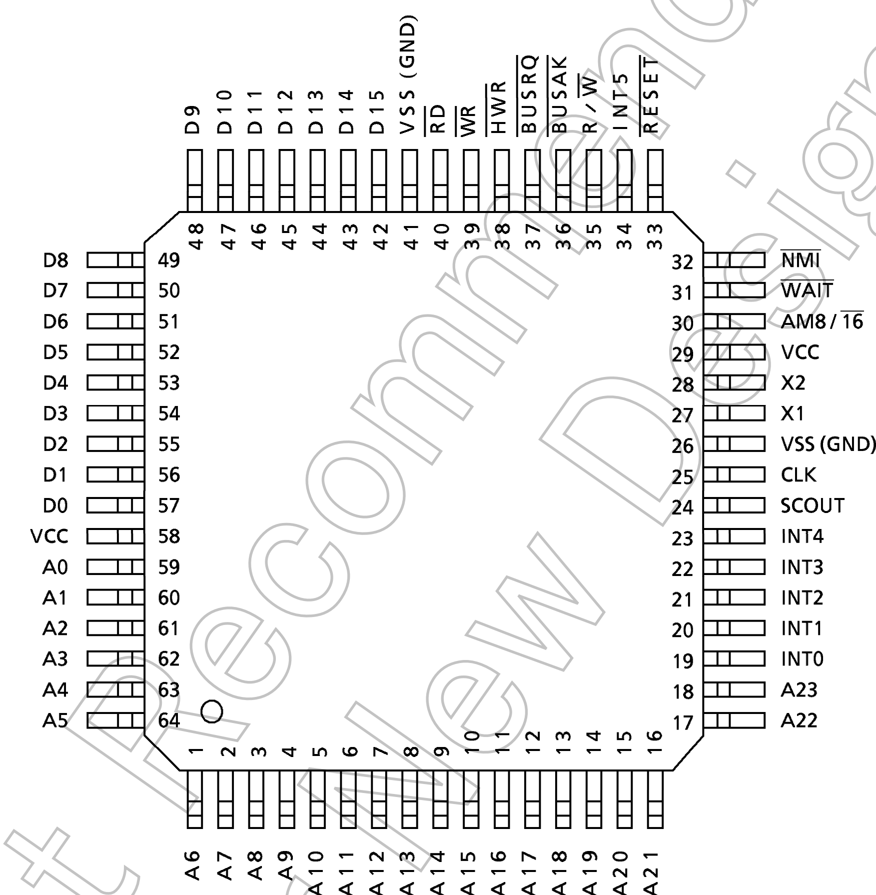


Figure 2.1 Pin Assignment (64-pin QFP)

2.2 Pin Names and Functions

Table 2.2 shows the I/O pin names and their functions.

Table 2.2 Pin Names and Functions

Pin Name	Pin Number	Input / Output	Function
D0 to D15	16	Input / Output	Data : Data bus 0 to 15
A0 to A23	24	Output	Address : Address bus 0 to 23
$\overline{\text{RD}}$	1	Output	Read : Strobe signal to read external memory Setting RSRAM mode outputs $\overline{\text{RD}}$ even when reading internal areas.
$\overline{\text{WR}}$	1	Output	Write : Strobe signal to write data of pins D0 to 7.
$\overline{\text{HWR}}$	1	Output	Upper write: Strobe signal for writing data of pins D8 to 15.
$\overline{\text{BUSRQ}}$	1	Input	Input Bus request: Signal to request external bus release.
$\overline{\text{BUSAK}}$	1	Output	Bus acknowledge: Signal to indicate external bus is released after receiving $\overline{\text{BUSRQ}}$.
R / $\overline{\text{W}}$	1	Output	Read/write: "1" indicates read or dummy cycle; "0" indicates write cycle.
SCOUT	1	Output	System clock output: Outputs system clock (external clock divided by 2).
$\overline{\text{WAIT}}$	1	Input	Wait: CPU bus wait request pin. (enabled in 1 + N or 0 + N WAIT mode).
INT0	1	Input	Interrupt request pin 0: Can be programmed for level or rising-edge detection.
INT1 to 4	4	Input	Interrupt request pin 1 to 4: Rising-edge interrupt request pin
INT5	1	Input	Interrupt request pin 5: Can be programmed for level or rising-edge detection.
$\overline{\text{NMI}}$	1	Input	Non-maskable interrupt request pin: Can be programmed for falling-edge or falling-rising-edge detection.
CLK	1	Output	Clock output: Outputs external input clock X1 divided by 4. Pulled up during reset.
AM8 / $\overline{\text{16}}$	1	Input	Address mode: External data bus width selection pin. Set to 0 when using fixed 16-bit external bus or dual 8/16-bit external bus. Set to 1 with 8-bit external bus fixed.
$\overline{\text{RESET}}$	1	Input	Reset: Initializes TMP95C001. (with pull-up)
X1 / X2	2	Input / Output	Oscillator connecting pins
VCC	2		Power supply pin (All Vcc pins should be connected with the power supply pin.)
VSS (GND)	2		Ground pin (0 V) (All Vss pins should be connected with GND (0 V).)

Note : Connect all VCC pins to power supply and all VSS pins to GND.

3. Operation

The following is a block-by-block description of the functions and basic operation of TMP95C001.

Note that the description concludes with cautions and restrictions for each block in 7, Usage Cautions and Restrictions.

3.1 CPU

TMP95C001 contains an advanced, high-speed 16-bit CPU (the TLCS-900/H__CPU). The CPU is described in the TLCS-900 CPU section in the previous chapter.

The following describes the CPU functions unique to TMP95C001 that are not described in "TLCS-900 CPU".

3.1.1 Reset Operation

Figure 3.1 (1) shows reset timing.

At TMP95C001 reset, the power supply voltage must be within the operating range and internal oscillation must be stable. Set the $\overline{\text{RESET}}$ input to 0 for at least ten system clocks (= 10 states: 0.8 μs for a 25-MHz clock).

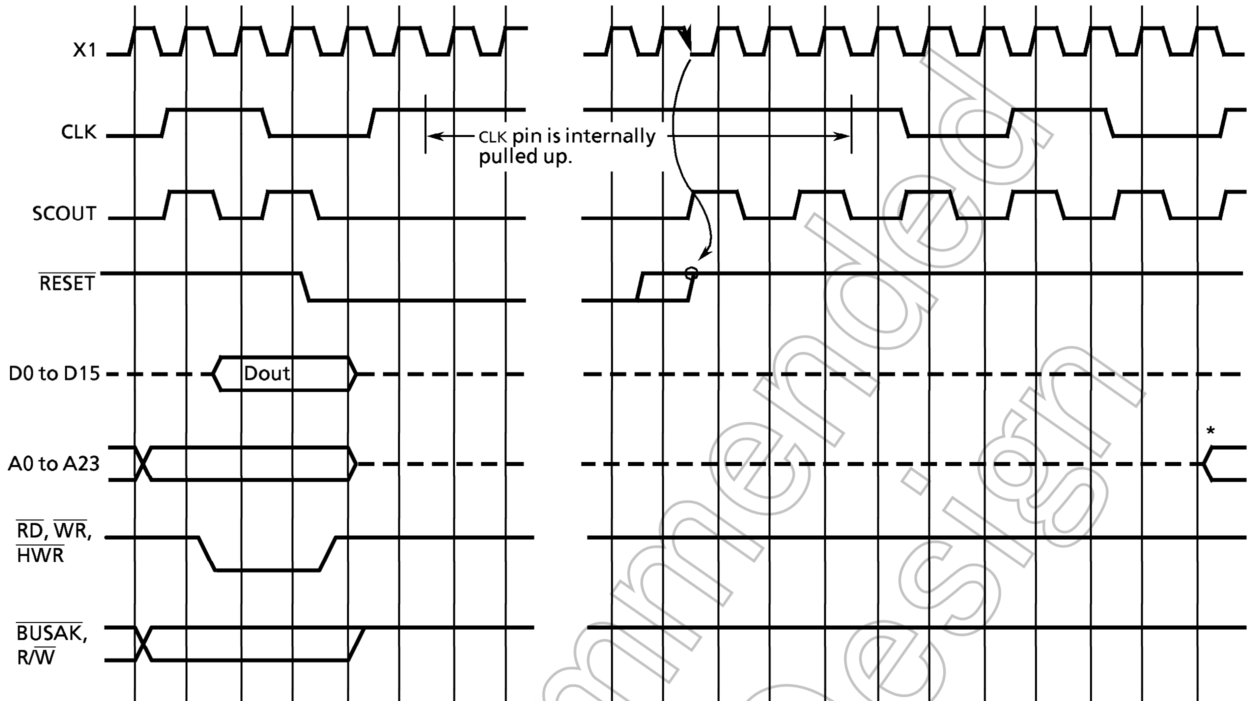
When the reset is accepted, the CPU:

- Sets the program counter (PC) to the reset vector stored at addresses FFFF00H to FFFF02H.
  PC (7 : 0) ← value at address FFFF00H
  PC (15 : 8) ← value at address FFFF01H
  PC (23 : 16) ← value at address FFFF02H
- Sets the stack pointer (XSP) to 100H
- Sets bits IFF2 to 0 of the status register (SR) to 111 (this sets the interrupt level mask register to level 7).
- Sets the MAX bit of the status register (SR) to 1 (this sets maximum mode). (Note: This product does not support minimum mode. Do not set the MAX bit to 0.)
- Clears bits RFP2 to 0 of the status register (SR) to 000 (this sets the register banks to 0).

After reset is released, the CPU begins execution from the instruction at the location specified in the PC. Other than the changes described above, reset does not alter any internal CPU registers.

When reset is accepted, processing of the internal I/O and other pins are as follows:

- Initializes the internal I/O registers as per specifications.
- Pulls up the clock pin to 1.



* A0 to A23 are output at the rising of X1 with the 10th or the 12th clock after acknowledging RESET = 1.

Figure 3.1 (1) TMP95C001 reset timing

3.1.2 External Data Bus Size Selection Pin (AM8/16)

TMP95C001 selects an external data bus size by sampling inputs to AM8/16 pin at the rising of a reset signal.

- AM8/16 = 0 (In case with 8 bit bus interlarded with 16 bit bus or fixed 16 bit bus)
  D0 to D15 function as a 16 bit data bus.
  The data bus size for external access is set by the wait control register. (Refer to "Wait control register" in section 3.5.2.).

- AM8/16 = 1 (In case with fixed 8 bit bus)
  D0 to D7 function as an 8 bit data bus.
  The values set in the wait control registers, <B0BUS>, <B1BUS>, <B2BUS>, <B3BUS> and <BEXBUS> are invalid, and it is fixed to an 8 bit data bus.
  When using in case with fixed 8 bit bus, D8 to D15 should be fixed to 1 or 0.

3.1.3 Clock Output

TMP95C001 has two clock output pins.
Setting a standby mode control register (STMOD) can control a clock output.

(1)Clock output pin (CLK)

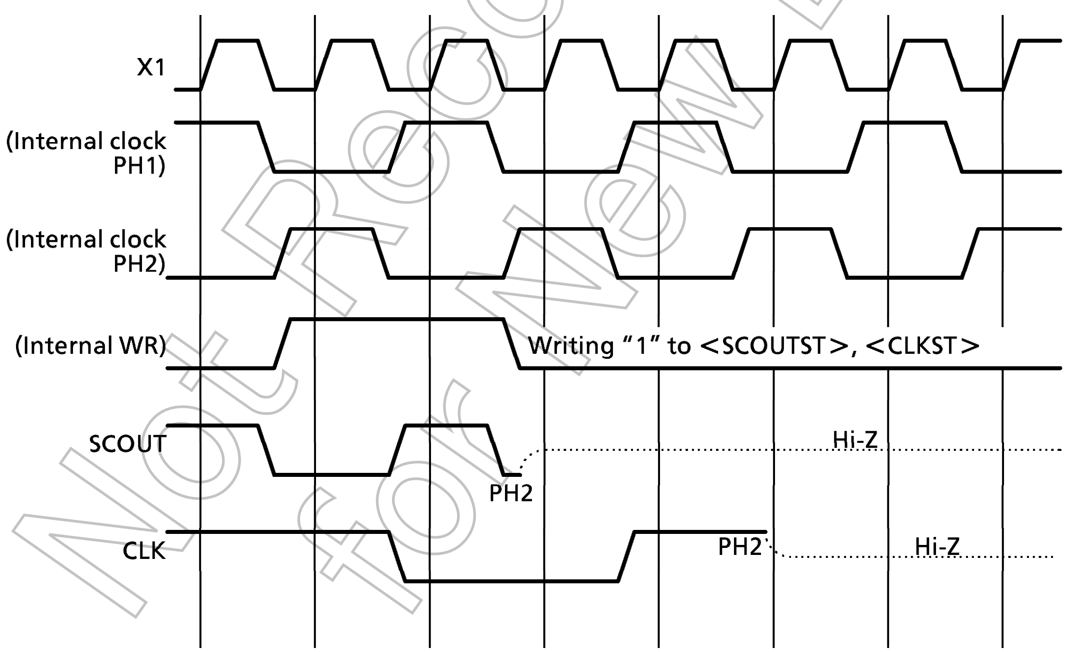
- Setting STMOD<CLKST> to “1” disables output (high-impedance).
  When CLK pins are in the high-impedance condition, the pull-up register should be needed externally because of preventing the through current flowed into an input buffer of CLK pins.
- An output starts just after reset cancel.

(2)System clock output pin (SCOUT)

- Setting STMOD<SCOUTST> to “1” disables output (high-impedance).
- An output starts just after reset cancel.

Figure 3.1 (2) shows a standby mode control register.

A timing Hi-Z (output disable) of SCOUT and CLK pins is shown as below figure.
<CLKST> and <SCOUTST> can be cleared by only a reset. These bits should not be written by “0”.



SCOUT : Hi-Z at the rising of PH2 just after writing <SCOUTST> of WR · PH1.
CLK : Hi-Z at the rising of PH2 following to PH2 and PH1 after writing to <CLKST> of WR · PH1.

3.1.4 Pseudo SRAM Support

TMP95C001 has PSRAM mode to use a pseudo SRAM externally. Using PSRAM mode outputs $\overline{RD}$ signal even when an internal area is read. Thus, an external PSRAM is refreshed. Writing "1" to the standby mode control register (STMOD) <RDE> can set PSRAM mode.

Figure 3.1(2) shows a standby mode control register.

Standby mode control register

STMOD (001EH)

		7	6	5	4	3	2	1	0
	Bit Symbol	RDE		SCOUTST	CLKST	HALTM1	HALTM0		DRVE
	Read / Write	W		R / W	R / W	R / W	R / W		R / W
	After reset	0		0	0	0	0		0
Read-modify-write is inhibited.	Function	$\overline{RD}$ pin output control 1: PSRAM mode		System clock output control 0: Enable 1: Disable	Clock output control 0: Enable 1: Disable	Halt mode setting 00: RUN 01: STOP 10: IDLE 11: Don't care			Pin control in STOP mode 1: Pins are driven in STOP mode.

Bits 3–0: Refer to "Standby function" in section 3.4.

Clock output control (bit 4)

0	Output enable
1	Output disable (high-impedance)

System clock output control (bit 5)

0	Output enable
1	Output disable (high-impedance)

$\overline{RD}$ pin output control (bit 7)

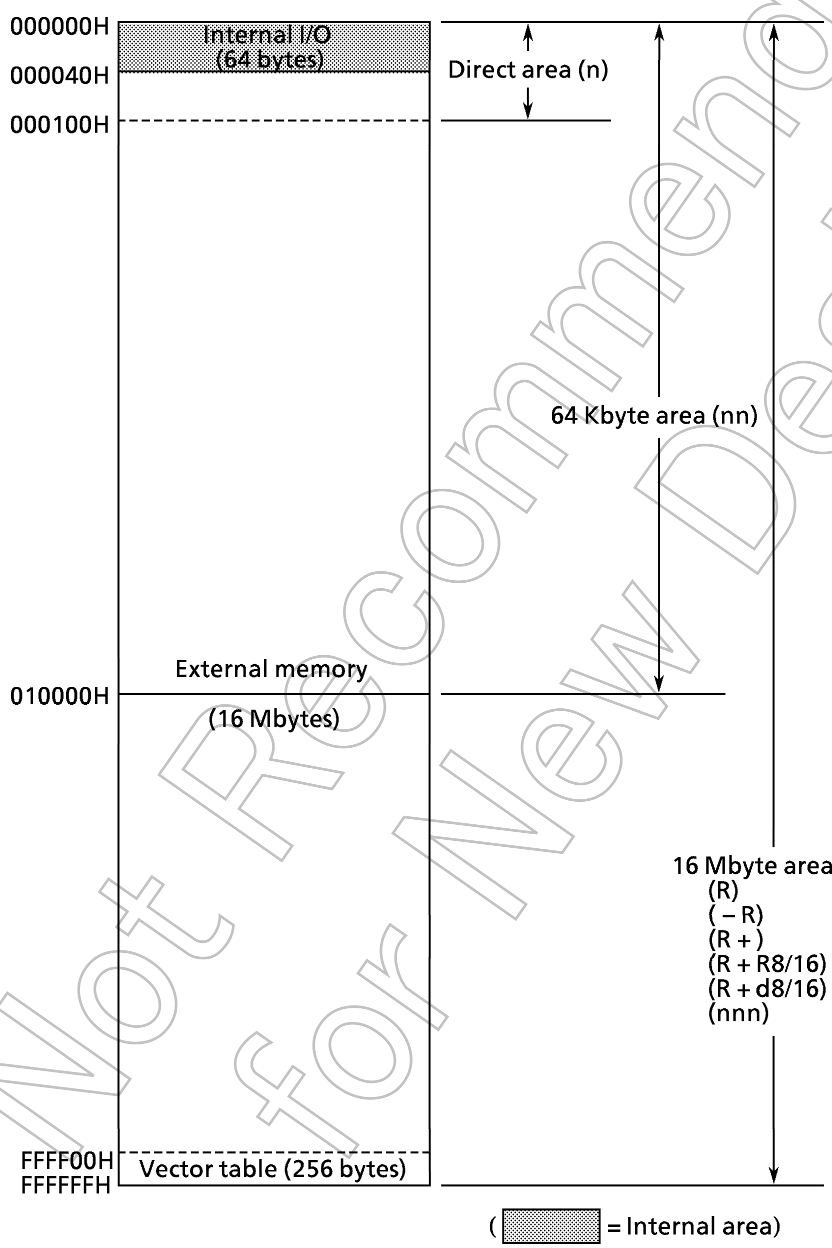
0	$\overline{RD}$ signal is output to an access area. (When an internal area is read, $\overline{RD}$ signal is not output.)
1	RSRAM mode (Even when an internal area is read, $\overline{RD}$ signal is output.)

Figure 3.1 (2) Standby mode control register

3.2 Memory Map

TMP95C001 uses an address area of 64 bytes as an internal I/O area.
This is allocated at addresses 000000H to 00003FH. The CPU can also access this internal I/O using a short instruction code according to "direct addressing mode".

Figure 3.2 shows an accessing area in the respective addressing modes for the memory map and the CPU.



Note : After reset, the stack pointer (XSP) is set to 100H.

Figure 3.2 TMP95C001 Memory Map

3.2.1 Operation at internal I/O area access

TMP95C001 uses 64 bytes of address space (0H to 3FH) as an internal I/O area. Internal I/O registers are mapped on this area.

Operation of the internal I/O area access is different from that of the other address area access about following two points.

(1) In the internal I/O area access, $\overline{RD}$, $\overline{WR}$ and $\overline{HWR}$ strobe signals are nonactive and fixed to high level.

However, in PSRAM mode set by STMOD<RDE>register, $\overline{RD}$ strobe signal becomes active also in the internal I/O area access. (See 3.1.4 Pseudo SRAM Support.)

(2) In the internal I/O area access, the number of waits becomes zero or one depending on the internal state of the CPU. This wait can't be controlled by wait controller (see 3.5 Wait Controller). When the specified address area overlaps with the internal I/O area, the operation as the internal I/O area takes priority of the specified address area.

3.3 Interrupts

TLCS-900 interrupts are controlled by the CPU interrupt mask flip-flops <IFF2 to 0> and the internal interrupt controller. Interrupts can come from a total of 20 sources :

Internal interrupts ··· 13

- Software interrupts : 8
- Illegal instructions : 1
- Interrupts from micro DMA : 4

External interrupts ··· 7

- Interrupts from external pins ($\overline{\text{NMI}}$, INT0 to INT5)

Individual interrupt vector numbers (fixed) are allocated to each interrupt source. Seven levels of priority (variable) can be allocated to maskable interrupts. The priority of non-maskable interrupts is fixed at "7" (the highest priority).

When an interrupt is generated, the interrupt controller sends the priority value of that interrupt to the CPU. If more than one interrupt is generated simultaneously, the interrupt with the highest priority (7 non-maskable interrupts is the highest) is sent to the CPU.

The CPU compares the priority value with the value of the CPU interrupt mask register <IFF2 to 0>, and accepts the interrupt if the priority is higher or equal to the value in the CPU interrupt mask register. However, software interrupts and illegal instruction interrupts generated by the CPU are processed without comparison with the IFF <2:0> value.

The value of the interrupt mask register <IFF2 to 0> can be modified using the EI instruction (EI num sets IFF <2:0> to num). For example, executing "EI 3" enables acceptance of non-maskable interrupts and maskable interrupts with a priority of 3 or higher set in the interrupt controller.

However programming EI 0 enables acceptance of maskable interrupts with a priority of 1 or greater, and non-maskable interrupts. (It operates as the same as EI 1.)

The DI instruction (sets IFF <2:0> to "7") is operationally the same as specifying "EI 7". As maskable interrupts have priorities in the range of 0 to 6, the DI instruction disables acceptance of maskable interrupts. The EI instruction is valid immediately after its execution. (With the TLCS-90, the EI instruction becomes valid only after the instruction following it is executed.)

As well as the general-purpose interrupt processing mode described above, the TLCS-900 also supports micro DMA processing mode. In micro DMA mode, the CPU transfers data automatically, thus accelerating interrupt processing such as data transfer to, or from, internal I/Os.

In addition to using an interrupt to start a micro DMA request, TMP95C001 also supports the "software start function", which start micro DMA requests by software.

Figure 3.3 (1) is a flowchart of overall interrupt processing.

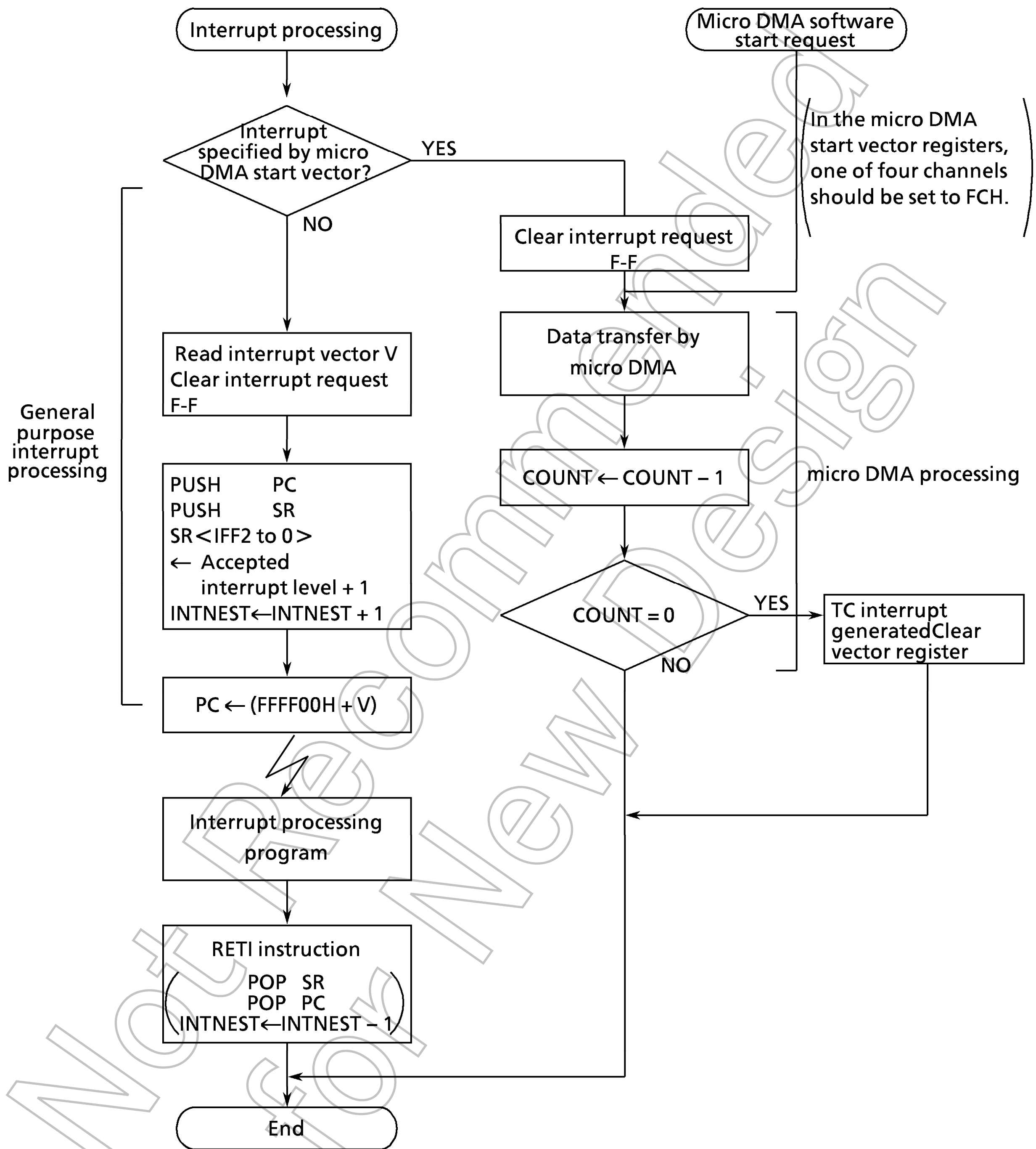


Figure 3.3 (1) Interrupt and Micro DMA Processing Flowchart

3.3.1 General-Purpose Interrupt Processing

On receiving an interrupt, the CPU operates as follows

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. When more than one interrupt with the same level is generated at the same time, the interrupt controller generates an interrupt vector in accordance with the default priority (the smaller the vector value, the higher the priority (fixed)), and clears the interrupt request.

(2) The CPU pushes the program counter (PC) and status register (SR) onto the stack (the area pointed to by XSP).

(3) The CPU sets the interrupt mask register <IFF2 to 0> value to the level of the received interrupt incremented by 1. If the received interrupt is a level 7 interrupt, the CPU does not increment the interrupt mask register but sets it 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 begins the interrupt processing routine.

Table 3.3 (1) shows the times required by this processing.

Table 3.3 (1) Interrupt Processing Time

Stack Area Bus Width	Interrupt Vecter Area Bus Width	Number of Interrupt Processing Execution States	Interrupt Processing Time at fc = 25 MHz (μs)
8	8	28	2.24
	16	24	1.92
16	8	22	1.76
	16	18	1.44

When interrupt processing is complete, the RETI instruction is executed to return processing to the main routine. Executing the RETI instruction restores the program counter (PC) and status register (SR) from the stack, and decrements interrupt nesting counter INTNEST by 1.

Non-maskable interrupts cannot be disabled by program. However, the program can enable or disable maskable interrupts, and can set priorities individually for each maskable interrupt source. (Setting the interrupt priority level to 0 (or 7) disables an interrupt request.)

The CPU accepts interrupt requests with a higher or equal priority than the value of the CPU interrupt mask register <IFF2 to 0>. On accepting an interrupt, the CPU sets the <IFF2 to 0> register to the received interrupt level incremented by 1. This means that if an interrupt is generated with a higher priority than the interrupt currently being processed, the CPU accepts the interrupt request for the higher priority interrupt and nests processing.

If a new interrupt request is generated while the CPU is accepting an interrupt and performing steps (1) to (5) described above, the CPU does not sample the new interrupt until after execution of the first instruction of the interrupt processing routine. Therefore, setting DI as the first instruction disables maskable interrupt nesting. (Note: The 900 and 900/L series sample the interrupt before executing the first instruction.) Resetting initializes the CPU mask register <IFF2 to 0> to "7". This disables maskable interrupts.

Table 3.3 (2) shows an interrupt vector and a micro DMA start vector tables for TMP95C001. The addresses FFFF00H to FFFFFFH (256 bytes) of the TMP95C001 are assigned for an interrupt vector area.

The interrupt vector area is depended on the derivative products.

Table 3.3 (2) TMP95C001 Interrupt Table

Default Priority	Type	Interrupt Request Source	Vector "V"	Vector Reference Address	HDMA Start Vector
1	Non-maskable	Reset, or SWI 0 instruction	0000H	FFFF00H	–
2		SWI 1 instruction	0004H	FFFF04H	–
3		INTUNDEF: Illegal instruction or SWI 2	0008H	FFFF08H	–
4		SWI 3 instruction	000CH	FFFF0CH	–
5		SWI 4 instruction	0010H	FFFF10H	–
6		SWI 5 instruction	0014H	FFFF14H	–
7		SWI 6 instruction	0018H	FFFF18H	–
8		SWI 7 instruction	001CH	FFFF1CH	–
9		$\overline{\text{NMI}}$ pin	0020H	FFFF20H	–
–		(reserved)	0024H	FFFF24H	–
10	Maskable	INT0 pin	0028H	FFFF28H	28H
11		INT1 pin	002CH	FFFF2CH	2CH
12		INT2 pin	0030H	FFFF30H	30H
13		INT3 pin	0034H	FFFF34H	34H
14		INT4 pin	0038H	FFFF38H	38H
15		INT5 pin	003CH	FFFF3CH	3CH
–		(reserved)	0040H	FFFF40H	–
16		INTTC0 : micro DMA completa (channel0)	0044H	FFFF44H	–
17		INTTC1 : micro DMA completa (channel1)	0048H	FFFF48H	–
18		INTTC2 : micro DMA completa (channel2)	004CH	FFFF4CH	–
19		INTTC3 : micro DMA completa (channel3)	0050H	FFFF50H	–
–		(reserved)	0054H	FFFF54H	–
to		to	to	to	to
–		(reserved)	00FCH	FFFFFCH	–
–		Software micro DMA	–	–	FCH

Setting reset or interrupt vector

① Reset vector

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

② Interrupt vector (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)

To define the reset vector as address 8100H, the NMI vector as address 9ABCH, and the INT1 vector as address 123456H:

```
ORG     8100H
LD      A, B
   :
ORG     9ABCH
LD      B, C
   :
ORG     123456H
LD      C, A
   :
ORG     0FFFF00H
DL      008100H      ; reset = 8100H

ORG     0FFFF20H
DL      009ABCH      ; NMI = 9ABCH

ORG     0FFFF2CH
DL      123456H      ; INT1 = 123456H
```

Note:

ORG and DL are assembler directives.

- ORG : For controlling location counter
- DL : For defining long word data (32 bits)

3.3.2 MicroDMA Processing

In addition to conventional interrupt processing, TMP95C001 supports micro DMA function. For interrupt requests set for micro DMA, micro DMA processing is performed at the highest priority for maskable interrupts (level 6), regardless of the actual interrupt level set for the interrupt.

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

(1) Micro DMA Operation

When an interrupt request occurs for an interrupt specified by the micro DMA start vector register, micro DMA sends the micro DMA request to the CPU with the highest priority for maskable interrupts (level 6), regardless of the actual interrupt level set for the interrupt, and starts micro DMA. The micro DMA function has four channels. This allows micro DMA to be set for up to four interrupts at the same time.

When the micro DMA is accepted, the interrupt request flip flop is cleared, data are automatically transferred from the transfer source address to the transfer destination addresses(the address are set in the control register), and the transfer count is decremented. If the decremented result is other than zero, a value of the micro DMA start vector register retains, and the micro DMA processing terminates. If the decremented result is zero, the CPU sends the micro DMA transfer end interrupt (INTTC0 to 3) to the interrupt controller, clears a value of the micro DMA start vector register to 0, disables the next micro DMA startup, and terminates the micro DMA processing.

If multiple-channel micro DMA requests occur at the same time, the priority is determined by the channel numbers, not the interrupt levels. The lower the channel number, the higher the priority. (CH0 (high) → CH3 (low))

If an interrupt request for the interrupt source used is received between the time that the micro DMA start vector is cleared and the time that it is reset, the CPU performs general-purpose processing at the specified interrupt level. Therefore, if the interrupt source is only being used for starting micro DMA (not used as an interrupt), set the interrupt level to zero.

When simultaneously using the same interrupt resource for both the micro DMA and general-purpose interrupts as described above, set the level of the interrupt source used to start micro DMA lower than the levels of all other interrupt sources. In this case, the cause of general interrupt is limited to the edge interrupt.

Example : When using external interrupts INT0 to 3 for running micro DMA 0 to 3
Set the interrupt level of INT0 to 3 to 1
Set other interrupt levels to 2 to 6

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

The transfer source and transfer destination addresses are set in 32-bit control registers. However, as only 24-bit addresses are output, the address space available to micro DMA is 16M bytes. (The upper 8 bit of 32 bit is invalid.)

Three transfer modes are supported: 1-byte transfer, 1-word transfer (= two bytes), and 4-byte transfer. For each transfer mode, it is possible to specify whether to increment, decrement, or fix source and destination addresses after transfer. These modes facilitate data transfer from I/O to memory, from memory to I/O, and from I/O to I/O. For transfer mode details, see "3.3.2 (4) Transfer Mode Register Details" later in this manual.

As a 16-bit transfer counter is used, micro DMA can perform a maximum of 65536 transfers (initializing the counter to 0000H specifies the maximum number of transfers).

The 6 interrupt sources (INT0 to INT5) with micro DMA start vectors (as listed in Table 3.3 (2)) can be used to start micro DMA processing. Together with the soft start function, this gives a total of 7 different micro DMA triggers.

Figure 3.3 (2) shows the micro DMA cycle for 1-word transfer in transfer destination address INC mode (the same apart from counter mode). (In case with an external 16 bit bus width, 0 wait and an even number of a source/destination address).

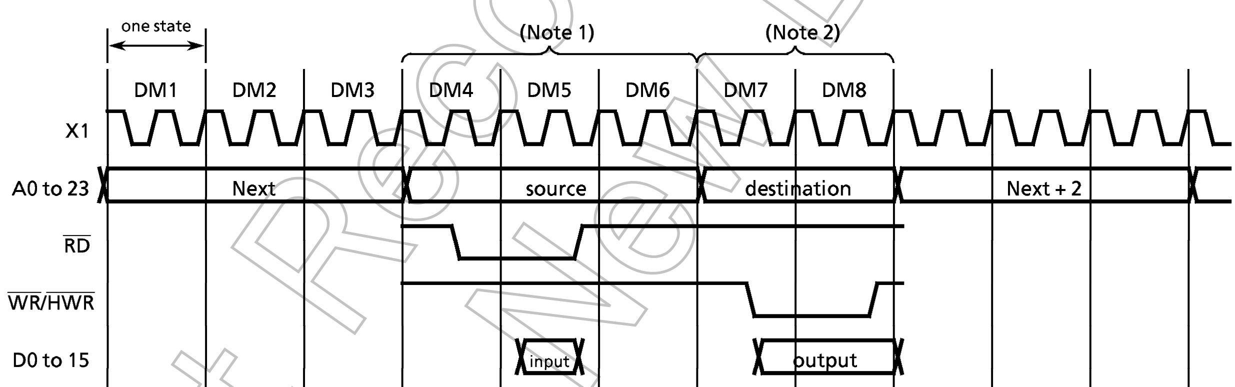


Figure 3.3 (2) Micro DMA Cycle Diagram

States 1-3 : Instruction fetch cycle (prefetches the next instruction code)
If the instruction cue buffer has three or more bytes of instruction code, the cycles are dummy cycles.

States 4-5 : Micro DMA read cycles

State 6 : Dummy cycle (address bus remains the same as in state 5)

States 6-8 : Micro DMA write cycle

Note 1 : If the source address area uses an 8-bit bus, two states are added.
If also the source address area uses a 16-bit bus and the source address is an odd-numbered address, two states are added.

Note 2 : If the destination address area uses an 8-bit bus, two states are added.
If also the destination address area uses a 16-bit bus and the destination address is an odd-numbered address, two states are added.

(2) Micro DMA Software Start Function

In addition to starting the micro DMA function by conventional interrupts, TMP95C001 includes an micro DMA software start function that starts micro DMA on the generation of the write cycle to the software DMA control register.

To trigger a software start, write the software micro DMA start vector "FCH" to the micro DMA start vector register DMA0V to 3V (memory address 26H, 27H, 28H, 29H). Next, writing data to the software DMA control register SDMACR0 to 3 (memory address 2AH, 2BH, 2CH, 2DH) (a value in the data does not effect a software start operation) causes micro DMA for the corresponding channel to run once. Writing again to the software DMA control register triggers another software start, provided the micro DMA transfer counter is set to other than "0". (It is not necessary to set the software micro DMA start vector again.)

Note that software start requests are one-shot requests and are not held over. If write cycle for the software DMA control register is generated when the software micro DMA start vector is not set, setting the software micro DMA start vector at a later time does not generate a software start. (The micro DMA start vector must be set prior to the micro DMA software start.)

(3) Micro DMA exclusive register

Figure 3.3 (3) shows the micro DMA exclusive register. This register is included in the CPU. (Refer to “Control register” in section 3.2.5 of “TLCS-900 CPU” in chapter 3.) It can be set by the LDC instruction.

In this figure, the transfer source address register represents a source address to be transferred, and the transfer destination address register represents a destination address to be transferred. These address register use only lower 24 bit and support 16M area.

The transfer count register can set an execution number of the micro DMA by 1 to 65536.

Setting the transfer mode register is referred to “Transfer mode register details” in section 3.3.2 (4).

Setting the data to the micro DMA exclusive register can be executed by only the “LDC cr,r” instruction.

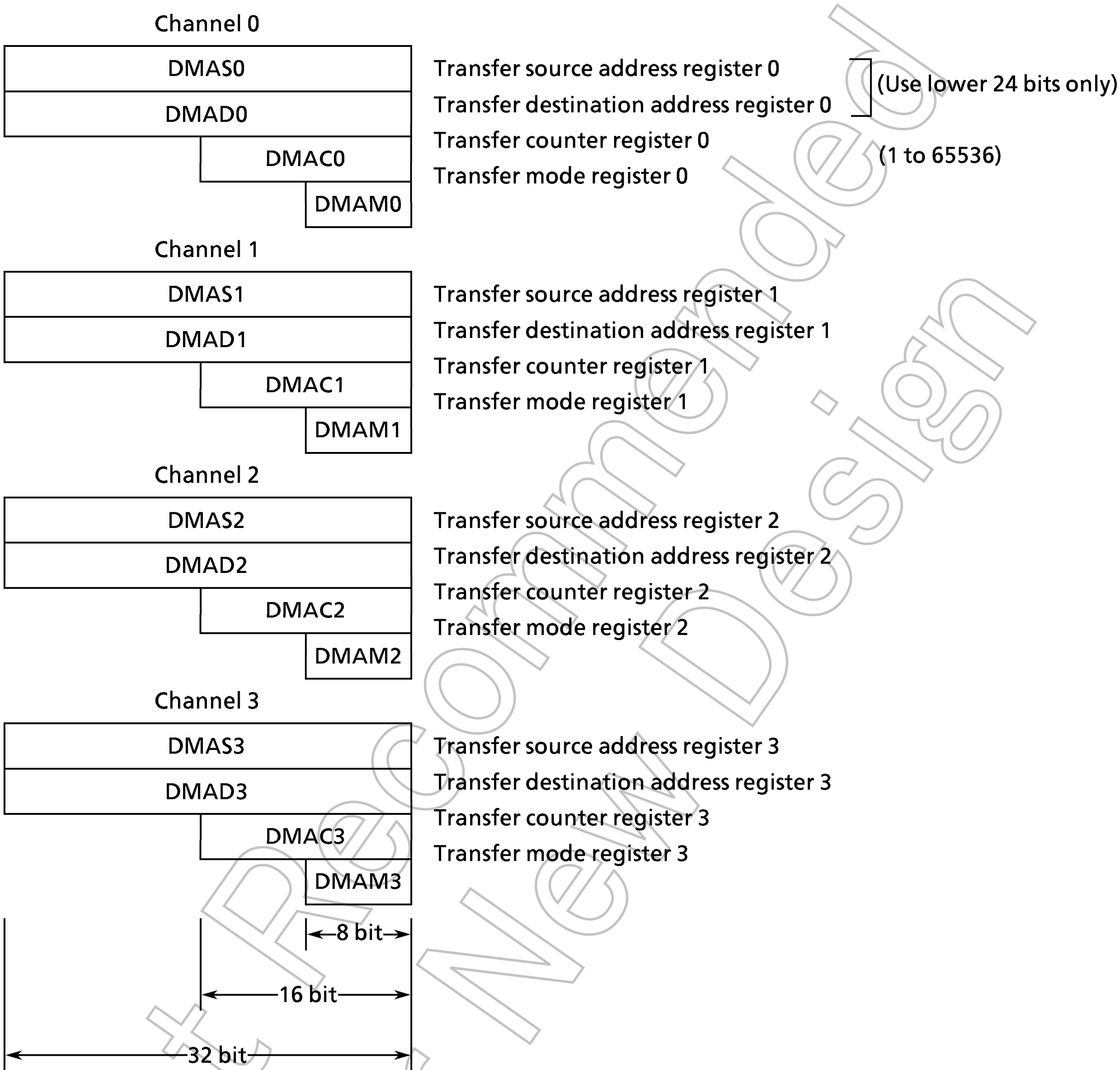


Figure 3.3 (3) Micro DMA exclusive register

(4) Transfer Mode Register Details

The micro DMA transfer mode is set by the transfer mode registers (DMAM0 to 3). Table 3.3 (3) shows the respective modes and the execution state number.

Table 3.3 (3) Micro DMA transfer mode

	← 8bit →	
DMAM0 to 3	0 0 0	Mode

Note : When setting values in this register, set the upper three bits to 0.

			Transfer byte number	Mode	Execution state number (※)	Minimum execution time at fc = 25 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 INTTC generated	8 states	640 ns
		01	word transfer		12 states	960 ns
		10	4 byte transfer			
	001	00	byte transfer	Transfer destination address DEC mode ··· For I/O to memory (DMADn –) ← (DMASn) DMACn←DMACn – 1 if DMACn = 0, then INTTC generated	8 states	640 ns
		01	word transfer		12 states	960 ns
		10	4 byte transfer			
	010	00	byte transfer	Transfer source address INC mode ··· For memory to I/O (DMADn) ← (DMASn +) DMACn←DMACn – 1 if DMACn = 0, then INTTC generated	8 states	640 ns
		01	word transfer		12 states	960 ns
		10	4 byte transfer			
	011	00	byte transfer	Transfer source address DEC mode ··· For memory to I/O (DMADn) ← (DMASn –) DMACn←DMACn – 1 if DMACn = 0, then INTTC generated	8 states	640 ns
		01	word transfer		12 states	960 ns
		10	4 byte transfer			
	100	00	byte transfer	Address fixed mode ··· For I/O to I/O (DMADn) ← (DMASn) DMACn←DMACn – 1 if DMACn = 0, then INTTC generated	8 states	640 ns
		01	word transfer		12 states	960 ns
		10	4 byte transfer			
	101	00	Counter mode ··· For interrupt count DMASn←DMASn + 1 DMACn←DMACn – 1 if DMACn = 0, then INTTC generated		5 states	400 ns

(※) In an external 16 bit bus width, 0 wait and word/4 byte transfer mode, both the source and the destination addresses should be the even numbers.

Note : n: Corresponding micro DMA channels 0 to 3
DMADn + / DMASn + : Post-increment (increment the register value after transfer)
DMADn -/ DMASn-: Post-decrement (decrement the register value after transfer)
In the above table, "I/O" refers to fixed addresses and "memory" refers to incremented or decremented addresses.
Do not use undefined codes other than the above mentioned for transfer mode registers.

3.3.3 Interrupt Controller

Figure 3.3 (4) is a block diagram of the interrupt circuits. The left half of the diagram shows the interrupt controller; the right half includes the CPU interrupt request signal circuit and the HALT release signal circuit (A halt is referred to "Standby" in section 3.4.)

Each interrupt channel (total of 11 channels : NMI, INT0 to 5 and INTTC0 to 3) in the interrupt controller has an interrupt request flip-flop (11 channels), an interrupt priority setting register (10 channels of INT0 to 5, INTTC0 to 3), and a start vector register (4 channels) for the micro DMA processing.

(1) Interrupt request flip-flop

The interrupt request flip-flop is used to latch interrupt request from peripheral devices. The channels other than NMI have the bit<IxxC> to clear the interrupt request. (Refer to "Interrupt priority setting register" in figure 3.3(5).) This flip-flop is cleared to "0" by the following operations.

- At reset
- When an interrupt is accepted, and the CPU reads the interrupt channel vector after the acceptance of interrupt
- When the CPU accepts the micro DMA request
- When the CPU executes an instruction that clears the interrupt of that channel (writes "0" in the clear bit<IxxC> of the interrupt priority setting register)

For example, to clear the interrupt request, executed the DI instruction and write "0" to the clear bit.

Ex.) Sets a register to clear INT0 interrupt request.

```
          MSB           LSB
           7 6 5 4 3 2 1 0
INTE01 ←   - - - - 0 - - -    Zero-clears the flip-flop of INT0 interrupt request.
```

Note : - ; No change

The status of the interrupt request flip-flop is detected by reading the clear bit<IxxC>. Detects whether there is an interrupt request for an interrupt channel. However the interrupt request flip-flop for NMI interrupt channel can not be read.

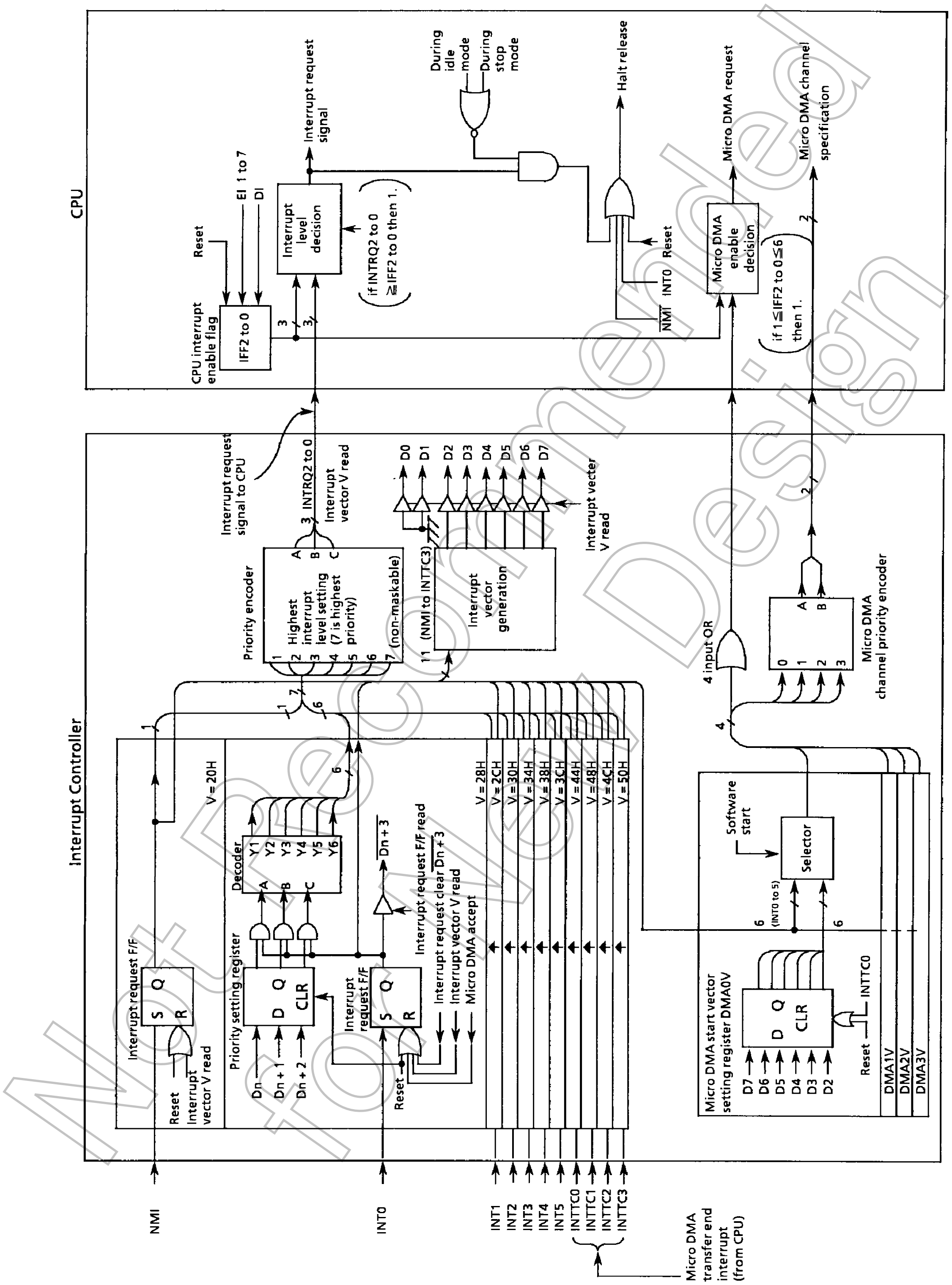


Figure 3.3 (4) Interrupt Controller Block Diagram

(2) Interrupt priority setting register

Figure 3.3(5) shows the interrupt priority setting registers. The interrupt request level setting bits<IxxM2 to 0> are provided for each 10 interrupt channels (INT0 to 5, INTTC0 to 3). Interrupt levels to be set are from 1 to 6. Writing 0 or 7 as the interrupt priority disables the corresponding interrupt request. The priority of the non-maskable interrupt ($\overline{\text{NMI}}$ pin) is fixed to 7. If interrupt requests with the same interrupt level are generated simultaneously, interrupts are accepted in accordance with the default priority.

Symbol	Address	7	6	5	4	3	2	1	0	
INTE01	20H	INT1				INT0				← Interrupt source
		I1C	I1M2	I1M1	I1M0	I0C	I0M2	I0M1	I0M0	← bit Symbol
		R/W	W			R/W	W			← Read / Write
		0	0	0	0	0	0	0	0	← Value after reset
INTE23	21H	INT3				INT2				
		I3C	I3M2	I3M1	I3M0	I2C	I2M2	I2M1	I2M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTE45	22H	INT5				INT4				
		I5C	I5M2	I5M1	I5M0	I4C	I4M2	I4M1	I4M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	
INTETC01	23H	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	24H	INTTC3				INTTC2				
		ITC3C	ITC3M2	ITC3M1	ITC3M0	ITC2C	ITC2M2	ITC2M1	ITC2M0	
		R/W	W			R/W	W			
		0	0	0	0	0	0	0	0	

Read-modify-write is inhibited.

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	No interrupt request	Clears interrupt request flag.
1	Interrupt request	----- Don't care -----

Figure 3.3 (5) Interrupt priority setting register

The interrupt controller sends the interrupt request with the highest priority among the simultaneous interrupts and its vector address to the CPU. The CPU compares the interrupt mask register <IFF2 to 0> set in the Status Register by the interrupt request signal with the priority value sent; if the latter is higher, the interrupt is accepted. Then the CPU sets a value higher than the priority value by 1 in the CPU SR <IFF2 to 0>. Interrupt requests where the priority value equals or is higher than the set value are accepted simultaneously during the previous interrupt routine.

(3) Micro DMA start vector register

The interrupt controller has the micro DMA start vector registers (4 channels). Writing the start vector of the interrupt source for the micro DMA processing (see Table 3.3 (2)), enables the corresponding interrupt to be processed by micro DMA processing. The values must be set in the micro DMA parameter registers (e.g., DMAS, DMAD, DMAC and DMAM) prior to the micro DMA processing. Figure 3.3.(6) shows the micro DMA start vector registers.

This register is used to assign the micro DMA to an interrupt source. The interrupt source whose the micro DMA start vector matches the vector value set in this register is assigned as the micro DMA start source.

When the micro DMA transfer counter value reaches 0, the interrupt controller is notified of the micro DMA transfer end interrupt (INTTC0 to 3) corresponding to the channel, the micro DMA start vector register is cleared, and the micro DMA start source of the channel is also cleared. To continue the micro DMA processing, the micro DMA start vector register must be set again within the micro DMA transfer end interrupt processing.

If the same vector is set in the micro DMA start vector registers of the multiple channels, the interrupt generated in the channel with the smaller number has a higher priority. Thus, if the same vector is set in the micro DMA start vector registers of two channels, the interrupt generated in the channel with the smaller number is processed until the micro DMA transfer end. If the micro DMA start vector or this channel is not set again, the next the micro DMA is started for the channel with the higher number. (micro DMA chaining)

Micro DMA0 start vecter register

DMA0V (0026H)
Read-modify-write is inhibited.

	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	Setting interrupt source to start the micro DMA channel 0.							

Micro DMA1 start vecter register

DMA1V (0027H)
Read-modify-write is inhibited.

	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	Setting interrupt source to start the micro DMA channel 1.							

Micro DMA2 start vecter register

DMA2V (0028H)
Read-modify-write is inhibited.

	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	Setting interrupt source to start the micro DMA channel 2.							

Micro DMA3 start vecter register

DMA3V (0029H)
Read-modify-write is inhibited.

	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	Setting interrupt source to start the micro DMA channel 3.							

Setting micro DMA start source

Micro DMA start source	Micro DMA start vector register value
INT 0 Interrupt	28H
INT 1 Interrupt	2CH
INT 2 Interrupt	30H
INT 3 Interrupt	34H
INT 4 Interrupt	38H
INT 5 Interrupt	3CH
Micro DMA software start	FCH

Figure 3.3 (6) Micro DMA start vector registers and start sources

(4) External interrupt control

Table 3.3 (4) shows setting of external interrupt pin functions.

TMP95C001 can select operating mode for $\overline{NMI}$, INT0 and INT5 pins among the external interrupt functions. (For the pulse width of external interrupt signal, refer to "Interrupt operation " in section 4.5.)

Table 3.3 (4) Setting of External Interrupt Pin Functions

Interrupt	Mode	Setting method
$\overline{NMI}$	Falling edge	IIMC<NMIREE> = 0
	Falling and rising edge	IIMC<NMIREE> = 1
INT0	Rising edge	IIMC<I0LE> = 0
	Level	IIMC<I0LE> = 1
INT1	Rising edge	―――
INT2	Rising edge	―――
INT3	Rising edge	―――
INT4	Rising edge	―――
INT5	Rising edge	IIMC<I5LE> = 0
	Level	IIMC<I5LE> = 1

Input modes of NMI, INT0 and INT5 interrupts are controlled by setting of the interrupt input mode control register, IIMC.

Figure 3.3 (7) shows the interrupt input mode control register.

Interrupt Input Mode Control Register

IIMC (002FH)		7	6	5	4	3	2	1	0
	bit Symbol						I5LE	I0LE	NMIREE
	Read/Write						W	W	W
	After reset						0	0	0
Read-modify-write is inhibited.	Function						INT5 interrupt control 0 : ↑edge 1 : level	INT0 interrupt control 0 : ↑edge 1 : level	NMI interrupt control 0 : ↓edge 1 : ↓↑edge

NMI interrupt control

0	Interrupt request generated on falling edge
1	Interrupt request generated on rising or falling edge

INT0 interrupt control

0	Rising edge detection interrupt
1	High level interrupt

INT5 interrupt control

0	Rising edge detection interrupt
1	High level interrupt

Figure 3.3 (7) Interrupt Input Mode Control Register

(5) Notes

The instruction execution unit and the bus interface unit of this CPU operate independently. Therefore, immediately before an interrupt is generated, if the CPU fetches an instruction that clears the corresponding interrupt request flag, the CPU may execute the instruction that clears the interrupt request flag between accepting and reading the interrupt vector.

To avoid the above problem, place instructions that clear interrupt request flags after a DI instruction. In the case of setting an interrupt enable again by EI instruction after the execution of clearing instruction, execute EI instruction after clearing instruction and following more than one instruction are executed. When EI instruction is placed immediately after clearing instruction, an interrupt becomes enable before interrupt request flags are cleared.

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

In addition, take care as the following interrupt modes are exceptional and demand special attention.

INT0, INT5 level mode	INT0 in level mode is not an edge-detect interrupt, so the interrupt request flip-flop function is canceled. The peripheral interrupt request bypasses the S input of the flip-flop, and acts as the Q output. Changing modes from edge to level automatically clears the interrupt request flag.
	If the CPU enters the interrupt response sequence as a result of setting INT0 from 0 to 1, INT0 must be held at 1 until the interrupt response sequence is completed. If the INT0 level mode is used to release a halt, INT0 must be held at 1 from the time INT0 changes from 0 to 1, to the time when the halt is released. (Ensure that INT0 does not go back 0 due to noise before the halt is released.)
	When switching modes from level to edge, any interrupt request flag set in level mode is not cleared. Accordingly, clear the interrupt request flag using the following sequence.
	
	DI
	LD (IIMC), 00H ; Switches from level to edge.
	LD (INTE01), 00H ; Clears interrupt request flag.
	EI
	※ INT5 (control register INTE45) needs the same operation, too.

Note : The following instructions or pin changes are equivalent to instructions that clear the interrupt request flag.

INT0, INT5 : Instructions that switch to level mode after an interrupt request is generated in edge mode.
The pin input changes from high to low after an interrupt request is generated in level mode. ("H" → "L")

3.4 Standby Function

(1) HALT mode

In TMP95C001, when the “HALT” instruction is executed, the operating mode changes RUN, IDLE, or STOP mode depending on the contents of the standby mode control registers (STMOD) <HALTM1,0>. Figure 3.4 (1) shows the standby mode control register.

Standby mode control register

STMOD (001EH)

Read-modify-write is inhibited.

	7	6	5	4	3	2	1	0
bit Symbol	RDE		SCOUTST	CLKST	HALTM1	HALTM0		DRVE
Read / Write	W		R / W					R / W
After reset	0		0	0	0	0		0
Function	$\overline{RD}$ pin output control 1 : PSRAM mode		System clock output control 0: Enable 1: Disable	Clock output control 0: Enable 1: Disable	Halt mode setting 00: RUN 01: STOP 10: IDLE 11: Don’t care			Pin control in STOP mode 1 : Pins are driven in STOP mode.

HALT mode setting

00	RUN mode(Only the CPU halts.)
01	STOP mode (All internal circuits stops.)
10	IDLE mode (Only the oscillator operates.)
11	Don’t care

I/O pin control in STOP mode

0	I/O off
1	The status in effect before entering the halt status is hold.

Figure 3.4 (1) Standby mode control register

The features of RUN, IDLE and STOP mode are as follows.

① RUN : Halts the CPU only. Power dissipation remains almost unchanged.

② IDLE : Operates only the internal oscillator, while halts all other circuits.

③ STOP : Halts all internal circuits, including the internal oscillator.

Table 3.4 (1) shows each blocks operation during halt.

Table 3.4 (1) Each Blocks and Input-output Pins Operation During Halt

Halt Mode		RUN	IDLE	STOP
STMOD <HALTM1, 0>		00	10	01
block	CPU	Halt	Halt	Halt
	Interrupt controller	Operation	Halt	Halt
	I/O function	Operation	Operation	See Table 3.4 (3)

(2) HALT release

The HALT release is executed by an interrupt request from the external interrupt pins or a reset. The halt release source is depended on the status of the interrupt mask register <IFF2 to 0> and the halt mode. For details, see Table 3.4 (2).

When the HALT state is released by INT0 and the interrupt request level is smaller than the interrupt mask register value, the CPU does not execute the interrupt processing of INT0.

The HALT state cannot be released by the micro DMA start except for INT0.

Table 3.4 (2) Halt Release Source and Halt Release Operation

Interrupt mask setting for interrupt request level			Interrupt request level ≧ Interrupt mask 〈IFF2 to 0〉			Interrupt request level ※ < Interrupt mask 〈IFF2 to 0〉		
Halt Mode			RUN	IDLE	STOP	RUN	IDLE	STOP
Source	Halt Release	Interrupt NMI*	◎	◎	×	◎	◎	×
		Interrupt INT0	◎	◎	×	○	○	×
		Interrupt INT1 to 5	◎	×	×	×	×	×
		RESET	◎	◎	◎	◎	◎	◎

◎ : After a halt is released, interrupt processing begins. (Reset initializes the LSI.)

○ : After a halt is released, processing begins from the next address following the HALT instruction.

× : Cannot be used to release a halt.

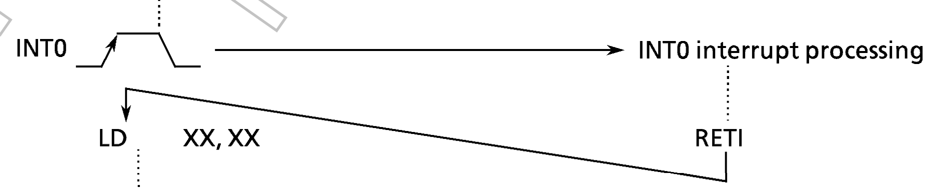
※ : Same as a case of setting interrupt mask level to 7 by DI instruction before HALT instruction is executed.

* : Halt release by NMI or RESET is not depended on the interrupt mask level.

Example of releasing halt.

On execution of the HALT instruction, the device enters standby state in RUN mode. Release halt using INT0.

```
Address
8203H   LD    (IIMC), 00H      ; Sets INT0 to rising edge detection
8206H   LD    (INTE01), 06H    ; Sets INT0 to level 6
8209H   EI    5                ; Sets CPU interrupt accept level to 5.
820BH   LD    (STMOD), 00H     ; Sets RUN mode.
820EH   HALT                   ; Halts CPU.
```




```
820FH   LD    XX, XX
```

(3) Halt Mode Operaition

① RUN mode

In RUN mode, the MCU internal system clock does not stop after the HALT instruction is executed. Only CPU instruction execution stops. Therefore, the CPU performs repeated dummy cycles until the halt state is released. In the halt state, interrupt requests are sampled on the falling edge of the CLK signal.

The halt state can be released by external interrupts (INT0 to 5, NMI) in RUN mode. When the interrupt request level of INT1 to 5 is smaller than the interrupt mask <IFF2 to 0>, the HALT state cannot be released by INT1 to 5.

Figure 3.4 (2) is the timing chart for releasing a halt in RUN mode using an interrupt.

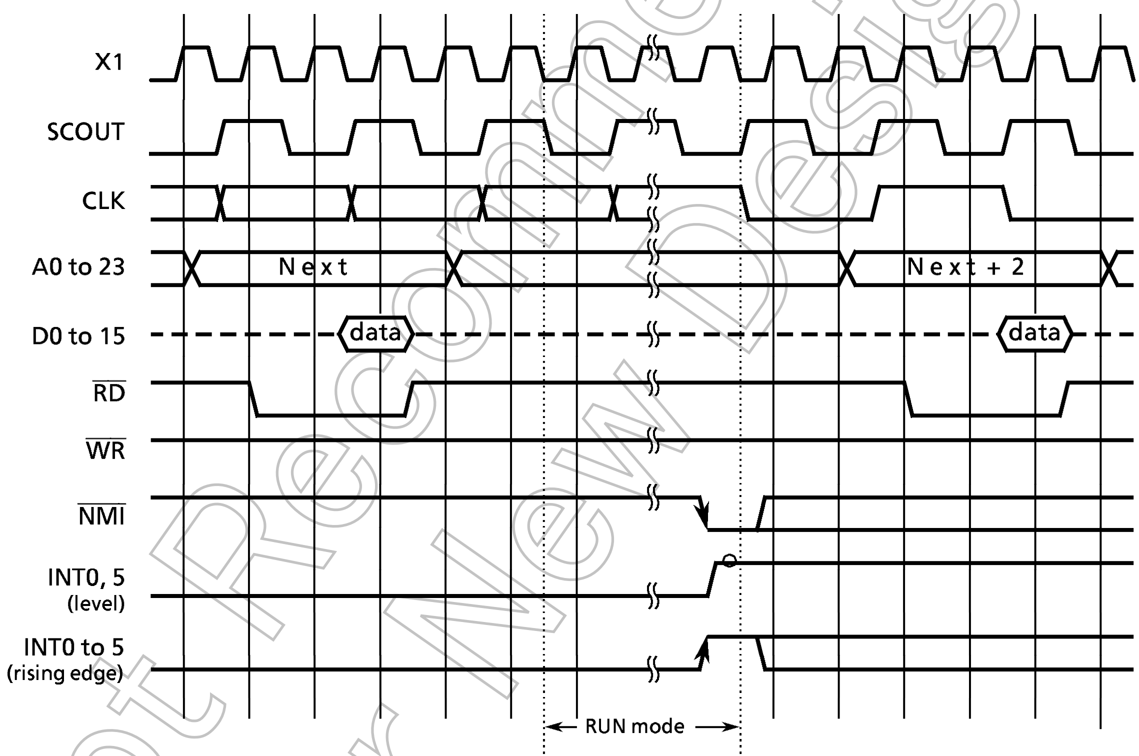


Figure 3.4 (2) Timing Chart for Releasing Halt in RUN Mode Using Interrupt

② IDLE mode

In IDLE mode, the MCU internal system clock stops. Only the internal oscillator functions. The CLK pin is fixed at “1”.

In the halt state, interrupt requests are sampled asynchronously to the system clock. The release from the halt state (operation restart), however, is synchronized with the clock.

In the IDLE mode, the HALT state can be released by only the external interrupt (NMI, INT0) and a reset.

Figure 3.4 (3) is the timing chart for releasing a halt in IDLE mode using an interrupt.

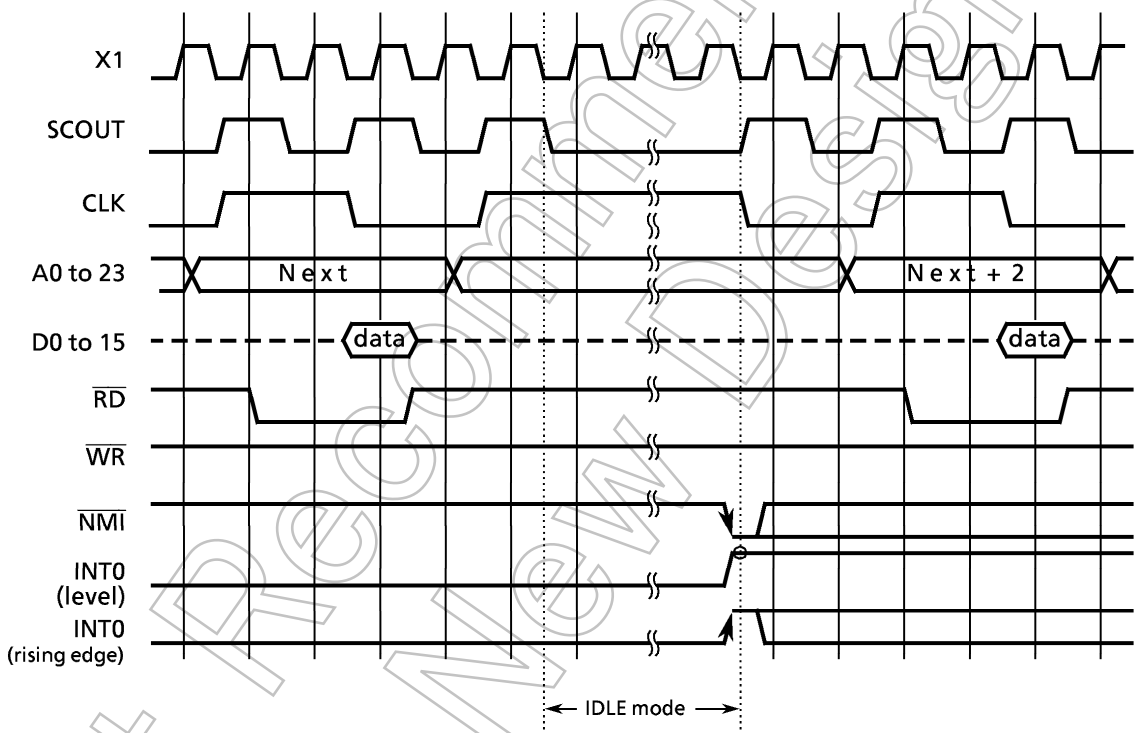


Figure 3.4 (3) Timing Chart for Releasing Halt in IDLE Mode Using Interrupt

③ STOP mode

The STOP mode is selected to stop all internal circuits including the internal oscillator. In this mode, the state of all pins is depended on the settings of STMOD<DRVE>. (For settings of STMOD<DRVE>, see Figure 3.4.(1).)

Table 3.4 (3) shows the pin states in the STOP mode.

In the STOP mode, the HALT state can be released by a reset. To release the HALT state by a reset, hold the reset input level to be 3ms or more, "0". The STMOD<DRVE> is initialized to "0" by a reset.

When the STOP mode is set to the standby state, take care not to input INT0 and $\overline{\text{NMI}}$ interrupts.

Note: Usually, interrupts can release all halts status. However, the interrupts = ($\overline{\text{NMI}}$, INT0), which can release the HALT mode may not be able to do so if they are input during the period CPU is shifting to the HALT mode (for about 3 clocks of X1) with IDLE or STOP mode. (In this case, an interrupt request is kept on hold internally.)
If another interrupt is generated after it has shifted to HALT mode completely, halt status can be released without difficultly. The priority of this interrupt is compare with that of the interrupt kept on hold internally, and the interrupt with higher priority is handled first followed by the other interrupt.

Figure 3.4 (4) shows the timing chart for releasing a halt in STOP mode using a reset.

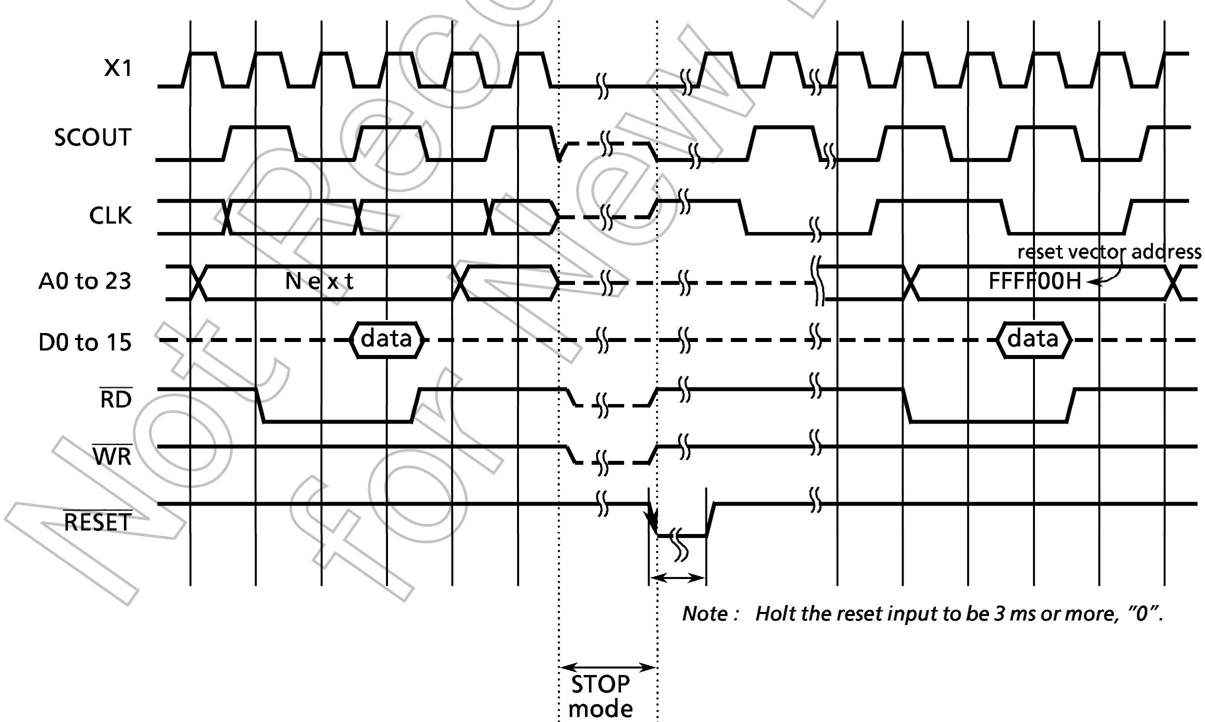


Figure 3.4 (4) Timing Chart for Releasing Halt in STOP Mode Using Reset (Case of STMOD <DRVE> = 0)

Table 3.4 (1) Pin State in STOP Mode

Pin Name	Input / Output	<DRVE> = 0	<DRVE> = 1
D0 to D15	Input / Output	Hi-Z*	Hi-Z*
A0 to A23	Output	Hi-Z	Output
$\overline{RD}$, $\overline{WR}$, $\overline{HWR}$, $\overline{BUSAK}$, R/$\overline{W}$	Output	Hi-Z	Output
SCOUT	Output	Hi-Z	"0"
$\overline{BUSRQ}$, $\overline{WAIT}$	Input	Invalid	◎
INT0	Input	◎	◎
INT1 to 5	Input	Invalid	◎
$\overline{NMI}$	Input	◎	◎
CLK	Output	Hi-Z	"1"
$\overline{RESET}$	Input	valid	valid
AM 8/$\overline{16}$	Input	◎	◎
X1	Input	Invalid	Invalid
X2	Output	"1"	"1"

Output : Maintains output states prior to a halt.
* : The input gate is disabled.
No through current, even at high impedance.
◎ : Must be driven externally.
valid : Input is valid.
Invalid : Input is invalid. As the input gate is disabled, no through current.

3.5 Wait Controller

TMP95C001 can set a variable 4-block address area (CS0 to CS3 areas), and select the data bus size and number of waits in each address area (CS0 to CS3 areas and another area) .

The CS0 to CS3 areas are specified by the memory start address registers MSAR0 to MSAR3 and memory address mask registers MAMR0 to MAMR3.

The master enable, data bus size and number of waits in each address area are specified by the wait control register, B0CS to B3CS and BEXCS.

TMP95C001 also has a bus wait request pin ($\overline{\text{WAIT}}$) and external data bus size selection pin (AM8/$\overline{16}$) as input pins to control these states. (Refer to "External data bus size selection pin" in selection 3.1.2.)

3.5.1 Address Area Specification

The CS0 to CS3 areas are specified by the memory start address registers MSAR0 to MSAR3 and memory address mask registers MAMR0 to MAMR3.

At each bus cycle, the wait controller compares the address on the bus with the address of specified areas to CS0 to CS3. If the result of the comparison is a match, this indicates an access to the specified area, and a specified operation by the wait control registers B0CS to B3CS is executed. (Refer to "Wait control register" in section 3.5.2.)

(1) Memory start address register

Figure 3.5 (1) shows the memory start address register. The memory start address registers, MSAR0 to MASR3 set the start addresses in CS0 to CS3. The higher 8 bit of the start address (A23 to A16) is set to <S23 to 16>. The lower 16 bit of the start address (A15 to A0) is always set to “0”. The start address is set to one of the 64 Kbyte intervals after 000000H. Figure 3.5.(2) shows a relationship between the start address and the start address register value.

Memory Start Address Register (CS0 to CS3 area)

MSAR0 (0034H) / MSAR1 (0036H)
MSAR2 (0038H) / MSAR3 (003AH)

	7	6	5	4	3	2	1	0
bit Symbol	S23	S22	S21	S20	S19	S18	S17	S16
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Sets start address for A23 to A16							

→ Sets CS0 to CS3 start address

Figure 3.5 (1) Memory Start Address Register

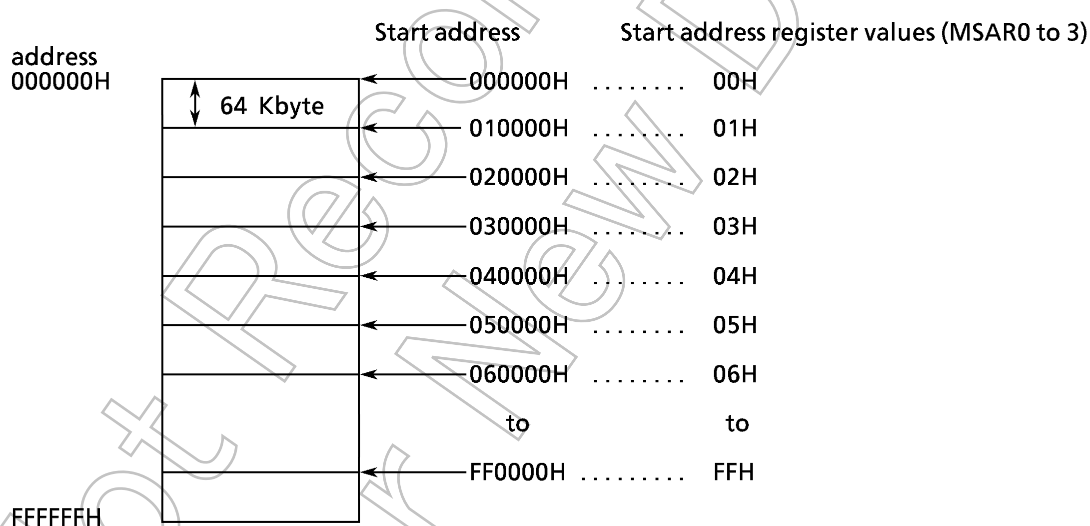


Figure 3.5 (2) Relationship between Start Address and Start Address Register Value

(2) Memory address mask register

Figure 3.5. (3) shows the memory address mask register. The memory address mask registers, MAMR0 to MAMR3 specifies the area size of CS0 to CS3 by specifying a corresponding mask to each bit of the start addresses set by the memory start address register. Whether the address corresponding to the bit written by “0” on the bus is within CS0 to CS3 area is compared.

The CS0 to CS3 areas have different address bits which can be masked by MAMR0 to MAMR3 registers. Thus the area size which can be set differs.

Memory Address Mask Register (CS0 area)

MAMR0 (0035H)

	7	6	5	4	3	2	1	0
bit Symbol	V20	V19	V18	V17	V16	V15	V14~9	V8
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Sets CS0 area size 0 : Compare address							

CS0 area size can be set from minimum 256 byte area to maximum 2 Mbyte area.

Memory Address Mask Register (CS1 area)

MAMR1 (0037H)

	7	6	5	4	3	2	1	0
bit Symbol	V21	V20	V19	V18	V17	V16	V15~9	V8
Read/Write	R/W							
After reset	1	1	1	1	1	1	1	1
Function	Sets CS1 area size 0 : Compare address							

CS1 area size can be set from minimum 256 byte area to maximum 4 Mbyte area.

Memory Address Mask Register (CS2, CS3 areas)

MAMR2 (0039H) / MAMR3 (003BH)

	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	Set CS2, CS3 area size 0 : Compare address							

CS2 and CS3 area sizes can be set from minimum 32K byte area to maximum 8 Mbyte area.

Figure 3.5 (3) Memory Address Mask Register

(3) Memory start address, Address area specification

Figure 3.5. (4) explains an example of specifying 64 Kbyte area which starts from address 010000H in the CS0 area.

"01H" corresponding to the upper 8 bit of the start address is set to MSAR0<S23 to 16>. The difference between the end address(01FFFFH) and the start address which is supposed by the area size of CS0 is calculated. As a result, bit 20 to 8 correspond to the mask value in specifying CS0 area. The area size can be specified by specifying this value to MAMR0<V20 to 8>.

In this example, "07H" is specified to MAMR0 and 64K-byte area is specified.

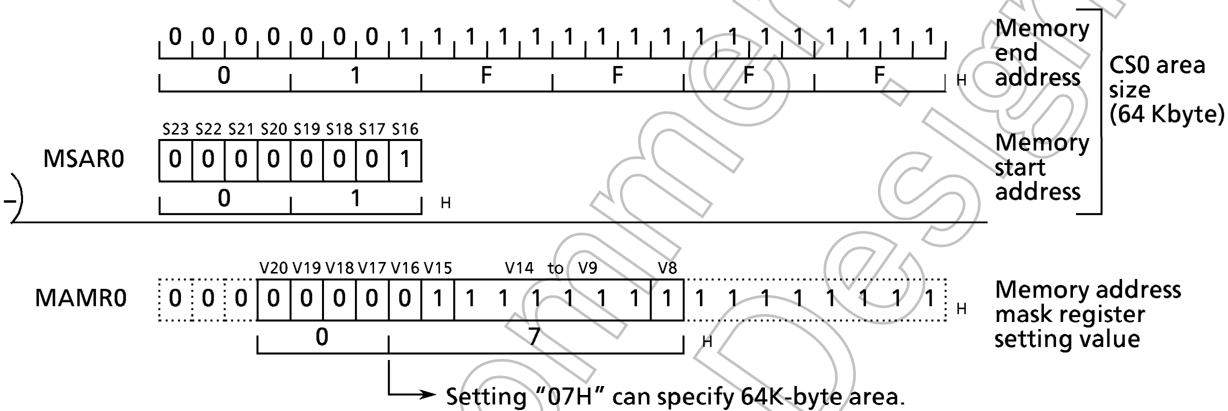


Figure 3.5 (4) Example of CS0 area setting

MSAR0 to 3 and MAMR0 to 3 are set to "FFH" after reset. B0CS<B0E>, B1CS<B1E> and B3CS<B3E> are reset to "0" after reset. CS0, CS1 and CS3 area are disabled. CS2 area is enabled at the addresses 000040H to FFFFFFH(16 Mbyte), since B2CS<B2M> is reset to "0" and B2CS<B2E> is set to "1". In addition, the addresses other than the specified CS0 to 3 areas are operated by the bus size and the number of waits specified by BEXCS. (See "Wait control register" in section 3.5.2.)

(4) Address area size specification

Table 3.5 (2) shows the relationship of CS area and the area size. (△ : The area size may not be specified by the combinations between the memory start address registers and the memory address mask registers.) When you set the area size using the combinations which may not specify the area size, set the start address by a size step which you can select from address 000000H.

If CS2 area is set to 16M area or the set address areas overlap, the one with a smaller CS number is selected.

When the set address area overlaps with the internal I/O area, the functions as the internal I/O area take priority of the set address area.

(Ex.) Sets CS0 area to 128K-byte area.

① Start address which can be set

000000H
} 128 Kbyte
020000H
} 128 Kbyte
040000H
} 128 Kbyte
060000H
⋮

In this case, all start addresses can be set.

② Start address which can not be set

000000H
} 64 Kbyte ← It the size step other than the specified size.
010000H
} 128 Kbyte
030000H
} 128 Kbyte
050000H
⋮

In this case, the area size which you select can not be set from the following start address.

Table 3.5 (2) CS area and Area size

CS area \ size [byte]	256	512	32 K	64 K	128 K	256 K	512 K	1 M	2 M	4 M	8 M
CS0	○	○	○	○	△	△	△	△	△		
CS1	○	○		○	△	△	△	△	△	△	
CS2			○	○	△	△	△	△	△	△	△
CS3			○	○	△	△	△	△	△	△	△

3.5.2 Wait Control Register

Figure 3.5 (5) shows the wait control registers. In each address area (CS0 to CS3 areas and another area), the master enable / disable, the data bus size and the number of waits are set by the wait control registers, B0CS to B3CS and BEXCS.

Wait control register

Register		7	6	5	4	3	2	1	0
B0CS (0030H)	bit Symbol	B0E				B0BUS	B0W2	B0W1	B0W0
	Read/Write	W				W	W		
	After reset	0				0	0	0	0
Read-modify-write is inhibited.	Function	0: Disable 1: Enable				Data bus size selection 0: 16 BIT 1: 8 BIT	Number of waits setting 000: 2 WAIT 001: 1 WAIT 010: 1 WAIT + N 011: 0 WAIT 100: 0 + N WAIT 101, 110, 111: Don't set		
B1CS (0031H)	bit Symbol	B1E				B1BUS	B1W2	B1W1	B1W0
	Read/Write	W				W	W		
	After reset	0				0	0	0	0
Read-modify-write is inhibited.	Function	0: Disable 1: Enable				Data bus size selection 0: 16 BIT 1: 8 BIT	Number of waits setting 000: 2 WAIT 001: 1 WAIT 010: 1 WAIT + N 011: 0 WAIT 100: 0 + N WAIT 101, 110, 111: Don't set		
B2CS (0032H)	bit Symbol	B2E	B2M			B2BUS	B2W2	B2W1	B2W0
	Read/Write	W	W			W	W		
	After reset	1	0			0	0	0	0
Read-modify-write is inhibited.	Function	0: Disable 1: Enable	CS2 area selection 0: 16M area 1: CS area			Data bus size selection 0:16 BIT 1: 8 BIT	Number of waits setting 000: 2 WAIT 001: 1 WAIT 010: 1 WAIT + N 011: 0 WAIT 100: 0 + N WAIT 101, 110, 111: Don't set		
B3CS (0033H)	bit Symbol	B3E				B3BUS	B3W2	B3W1	B3W0
	Read/Write	W				W	W		
	After reset	0				0	0	0	0
Read-modify-write is inhibited.	Function	0: Disable 1: Enable				Data bus size selection 0: 16 BIT 1: 8 BIT	Number of waits setting 000: 2 WAIT 001: 1 WAIT 010: 1 WAIT + N 011: 0 WAIT 100: 0 + N WAIT 101, 110, 111: Don't set		
BEXCS (003FH)	bit Symbol					BEXBUS	BEXW2	BEXW1	BEXW0
	Read/Write					W	W		
	After reset					0	0	0	0
Read-modify-write is inhibited.	Function					Data bus size selection 0: 16 BIT 1: 8 BIT	Number of waits setting 000: 2 WAIT 001: 1 WAIT 010: 1 WAIT + N 011: 0 WAIT 100: 0 + N WAIT 101, 110, 111: Don't set		

CS2 area selection

0	16 Mbyte area
1	Specified address area

Number of waits setting in address area (Refer to "wait control" in section 3.5.2 (3).)

Master enable bit

0	CS area disable
1	CS area enable

Data bus size selection

0	16 bit data bus
1	8 bit data bus

Figure 3.5 (5) Wait Control Registers

(1) Master Enable

Bit 7 of the wait control register <B0E>,<B1E>,<B2E>,<B3E> is the master enable/disable bit in each address area. Set the bit to “1” to enable the setting. A reset sets <B0E>, <B1E>, and <B3E> to “0” (disabled), and sets <B2E> to “1” (enabled). (Only CS2 area is enabled by a reset.)

(2) Data Bus Size Selection

Bit 3 of the wait control registers <B0BUS>,<B1BUS>,<B2BUS>,<B3BUS>, and<BEXBUS> specifies the width of the data bus. Set “0” to access memory in 16-bit data bus mode, or to “1” in 8-bit data bus mode.

Note that this bit is valid only in 16-bit bus mode (when the $AM8/\overline{16}$ pin is “0”). In 8-bit bus mode (when the $AM8/\overline{16}$ pin is “1”), memory access to all address areas uses 8-bit data bus mode, regardless of the value of bit 3. (See “External Data Bus Size Selection Pin” in section 3.1.2.)

This way of changing the data bus size depending on the address being accessed is called “dynamic bus sizing”. See Table 3.5 (3) for details of this bus operation.

Table 3.5 (3) Dynamic Bus Sizing

Operand Data Width	Operand Start Address	Memory Data Width	CPU Address	CPU Data	
				D15 to D8	D7 to D0
8 bits	2n + 0	8 bits	2n + 0	xxxxx	b7 to b0
	(even-numbered)	16 bits	2n + 0	xxxxx	b7 to b0
	2n + 1	8 bits	2n + 1	xxxxx	b7 to b0
	(odd-numbered)	16 bits	2n + 1	b7 to b0	xxxxx
16 bits	2n + 0	8 bits	2n + 0	xxxxx	b7 to b0
	(even-numbered)		2n + 1	xxxxx	b15 to b8
		16 bits	2n + 0	b15 to b8	b7 to b0
	2n + 1	8 bits	2n + 1	xxxxx	b7 to b0
	(odd-numbered)		2n + 2	xxxxx	b15 to b8
		16 bits	2n + 1	b7 to b0	xxxxx
			2n + 2	xxxxx	b15 to b8
32 bits	2n + 0	8 bits	2n + 0	xxxxx	b7 to b0
	(even-numbered)		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	8 bits	2n + 1	xxxxx	b7 to b0
	(odd-numbered)		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 : During a read, indicates that bus input data are ignored; during a write, indicates that the bus is set to high impedance and that the bus write strobe signal remains inactive.

(3) Wait control

Wait control register bits 2 to 0(<B0W2 to 0>, <B1W2 to 0>, <B2W2 to 0>, <B3W2 to 0>, <BEXW2 to 0>) area used to specify the number of waits. Combining these bits executes the following wait operations. Do not set other than the following combinations.

000 ··· 2WAIT
Inserts a 2-state wait regardless of the $\overline{\text{WAIT}}$ pin status

001 ··· 1WAIT
Inserts a 1-state wait regardless of the $\overline{\text{WAIT}}$ pin status.

010 ··· 1WAIT+N
Inserts a 1-state wait and samples the $\overline{\text{WAIT}}$ pin status.
If the pin is low, inserting the wait maintains the bus cycle until the pin goes high.

011 ··· 0WAIT
Completes the bus cycle without a wait regardless of the $\overline{\text{WAIT}}$ pin status.

100 ··· 0+NWAIT
Always samples the $\overline{\text{WAIT}}$ pin status. If the pin is low, inserting the wait maintains the bus cycle until the pin goes high.

Figure 3.5 (6) and (7) show a timing chart of 0+NWAIT setting at N=0 and 1. SCOUT pin output can be used to suppose a sampling timing of $\overline{\text{WAIT}}$ pin.

For the timing charts of other than 0+NWAIT setting, see "Standard timing, Figure 7 (1) to (5) in section 7 of TLCS-900 CPU in chapter 3.

Resetting sets these bits to "000" (2WAIT).

(4) Bus Size and Wait Control Outside $\overline{\text{CS0}}$-$\overline{\text{CS3}}$ Area

The wait control register, BEXCS controls the bus size and the number of waits when locations outside a variable 4-block address areas (CS0 to CS3 areas) are accessed. This register settings are always enabled for access to areas outside CS0 to CS3.

(5) Accessing 16M-byte Area/Address Setting Area

Setting B2CS<B2M> to "0" selects CS2 with a 16M-byte address area (000040H to FFFFFFH). Setting B2CS<B2M> to "1" selects CS2 with the address area specified by memory start address register MSAR2 and memory address mask register MAMR2, as in the case of CS0, CS1 and CS3. A reset clears this bit to "0", and 16M-byte area is selected.

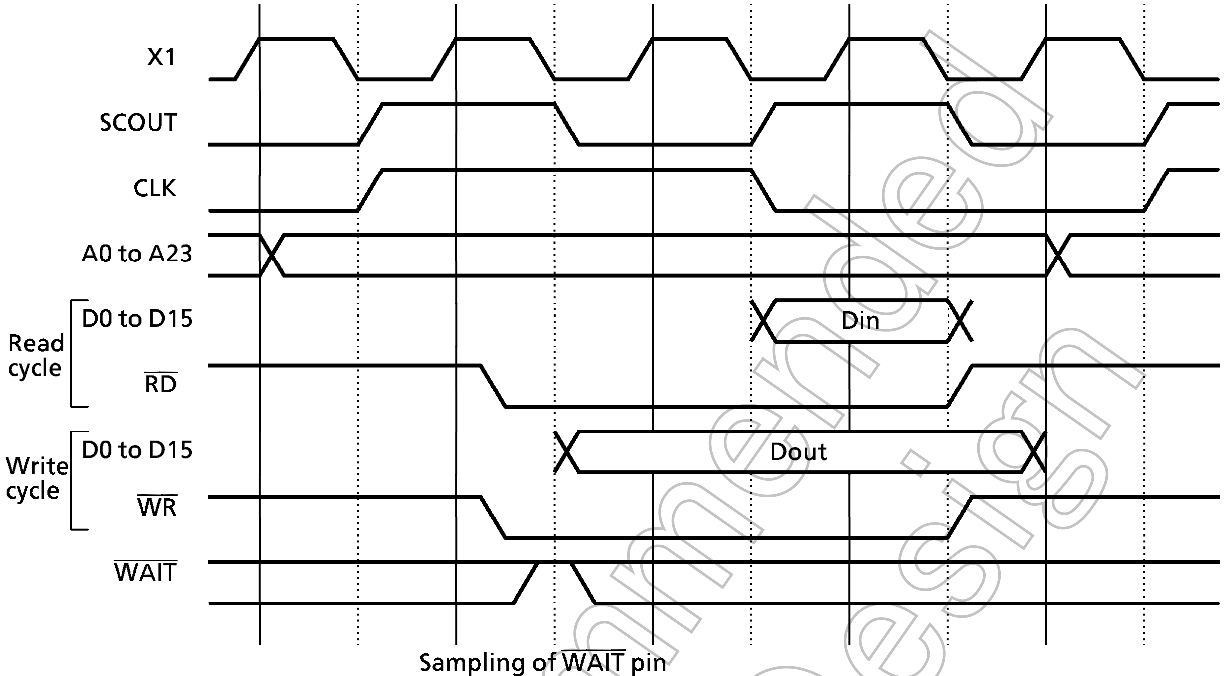


Figure 3.5 (6) Read / Write cycle in 0 + N WAIT mode (N = 0)

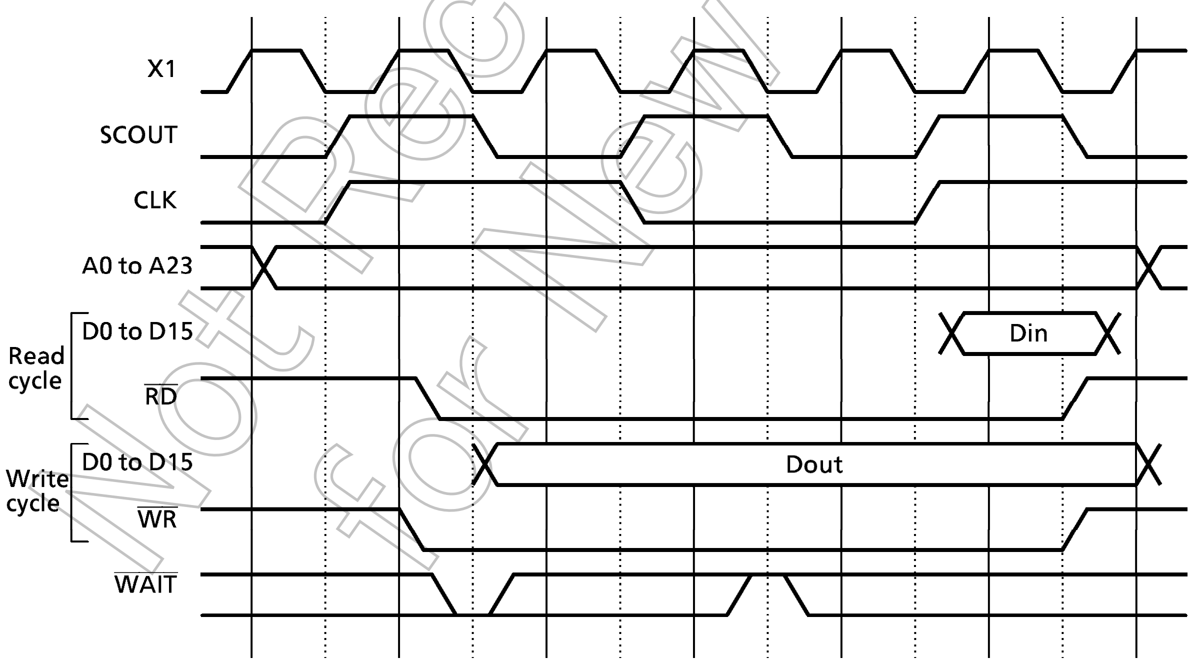


Figure 3.5 (7) Read / Write cycle in 0 + N WAIT mode (N = 1)

Wait controller setting procedure

When you set the CS0 to CS3 area, set the registers with the following procedure.

① Set the memory start address register (MSAR0 to MSAR3).
Set the CS0 to CS3 area start address.

② Set the memory address mask register (MAMR0 to MAMR3).
Set the CS0 to CS3 area size.

③ Set the control register (B0CS to B3CS).
Set the bus size, number of waits, and master enable/disable in the CS0 to CS3 area.

Example :

Set the CS0 area as 010000_H to $01FFFF_H$ (64 Kbytes), a 16-bit data bus, and zero waits:

$MSAR0 = 01_H$	……	start address 010000_H
$MAMR0 = 07_H$	…..	address area 64 Kbytes
$B0CS = 83_H$	……..	16-bit data bus, zero waits, CS0 enabled

3.6 Bus Release Function

TMP95C001 has a bus request pin ($\overline{BUSRQ}$) for releasing the bus, and a bus acknowledge pin ($\overline{BUSAK}$).

3.6.1 Operation

When “0” is input to the $\overline{BUSRQ}$ pin, TMP95C001 acknowledges a bus release request. When the current bus cycle ends, TMP95C001 sets the address bus (A23 to A0) and the bus control signals ($\overline{RD}$, $\overline{WR}$, $\overline{HWR}$, R/$\overline{W}$) simultaneously to high level, sets these signals and the output buffer for the data bus (D15 to D0) to high impedance, and sets the $\overline{BUSAK}$ pin to low level, indicating that the bus is released.

During bus release, TMP95C001 disables all access to the internal I/O registers, although the internal I/O functions are not affected.

3.6.2 Pin States at Bus Release

Table 3.6 lists pin states when the bus is released.

Table 3.6 Pin State at Bus Release

Pin Name	Pin State at Bus Release
D0 to D15	At high impedance
A0 to A23 $\overline{RD}$ $\overline{WR}$ $\overline{HWR}$ R/$\overline{W}$	At high impedance (first set to high level just before bus release)

4. Electrical Characteristics

4.1 Absolute Maximum Ratings

Parameter	Symbol	Rating	Unit
Supply voltage	V_{cc}	− 0.5 to 6.5	V
Input voltage	V_{IN}	− 0.5 to Vcc + 0.5	V
Output current (total)	Σ_{IOL}	+ 120	mA
Output current (total)	Σ_{IOH}	− 120	mA
Power dissipation (Ta = 70°C)	P_D	400	mW
Soldering temperature (10 s)	T_{SOLDER}	+ 260	°C
Storage temperature	T_{STG}	− 65 to 150	°C
Operating temperature	T_{OPR}	− 20 to 70	°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

(1) Vcc = + 5 V ± 10%, Ta = − 20 to + 70°C (fc = 8 to 25 MHz)

(Typ values are for Ta = + 25°C and Vcc = + 5 V)

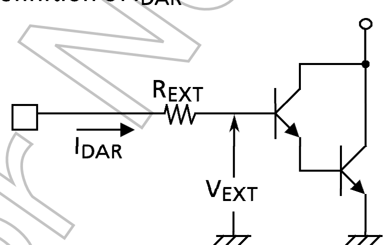
Parameter	Symbol	Test Condition	Min	Max	Unit
Input Low Voltage (D0 to 15)	V_{IL}		−0.3	0.8	V
INT1 to 5, BUSRQ, WAIT	V_{IL1}		−0.3	0.3 Vcc	V
RESET, NMI, INT0	V_{IL2}		−0.3	0.25 Vcc	V
AM8/16	V_{IL3}		−0.3	0.3	V
X1	V_{IL4}		−0.3	0.2Vcc	V
Input High Voltage (D0 to 15)	V_{IH}		2.2	Vcc + 0.3	V
INT1 to 5, BUSRQ, WAIT	V_{IH1}		0.7 Vcc	Vcc + 0.3	V
RESET, NMI, INT0	V_{IH2}		0.75 Vcc	Vcc + 0.3	V
AM8/16	V_{IH3}		Vcc − 0.3	Vcc + 0.3	V
X1	V_{IH4}		0.8Vcc	Vcc + 0.3	V
Output Low Voltage	V_{OL}	I_{OL} = 1.6 mA		0.45	V
Output High Voltage	V_{OH}	I_{OH} = − 400 μA	2.4		V
	V_{OH1}	I_{OH} = − 100 μA	0.75 Vcc		V
	V_{OH2}	I_{OH} = − 20 μA	0.9 Vcc		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	I_{LI}	0.0 ≦ Vin ≦ Vcc	0.02 (Typ)	± 5	μA
Output Leakage Current	I_{LO}	0.2 ≦ Vin ≦ Vcc − 0.2	0.05 (Typ)	± 10	μA
Operating Current (RUN)	I_{cc}	fc = 25 MHz	20 (Typ)	30	mA
IDLE			3.5 (Typ)	10	mA
STOP (Ta = − 20 to 70°C)		0.2 ≦ Vin ≦ Vcc − 0.2	0.5 (Typ)	50	μA
STOP (Ta = 0 to 50°C)		0.2 ≦ Vin ≦ Vcc − 0.2		10	μA
Power Down Voltage (at STOP)	V_{STOP}	V_{IL2} = 0.2 Vcc, V_{IH2} = 0.8 Vcc	2.0	6.0	V
RESET Pull Up Resistance	R_{RST}		50	250	kΩ
Pin Capacitance	C_{IO}	fc = 1 MHz		10	pF
Schmitt Width RESET, NMI, INT0	V_{TH}		0.4	1.0 (Typ)	V

Note: I_{DAR} is guaranteed for total of up to 8 ports.

(2) Vcc = +3 V ± 10%, Ta = −20~ +70°C (fc = 4 to 12.5 MHz)
(Typ values are for Ta = +25°C and Vcc = +3 V)

Parameter	Symbol	Test Condition	Min	Max	Unit
Input Low Voltage (D0 to 15) $\overline{\text{INT1 to 5}}$, $\overline{\text{BUSRQ}}$, $\overline{\text{WAIT}}$ $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, INT0 AM8/$\overline{16}$ X1	V_{IL} V_{IL1} V_{IL2} V_{IL3} V_{IL4}		−0.3 −0.3 −0.3 −0.3 −0.3	0.6 0.3 Vcc 0.25 Vcc 0.3 0.2Vcc	V V V V V
Input High Voltage (D0 to 15) $\overline{\text{INT1 to 5}}$, $\overline{\text{BUSRQ}}$, $\overline{\text{WAIT}}$ $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, INT0 AM8/$\overline{16}$ X1	V_{IH} V_{IH1} V_{IH2} V_{IH3} V_{IH4}		2.0 0.7 Vcc 0.75 Vcc Vcc − 0.3 0.8Vcc	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}	I_{OH} = −400 μA	2.4		V
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 (RUN) IDLE STOP (Ta = −20 to 70°C) STOP (Ta = 0 to 50°C)	I_{cc}	fc = 12.5 MHz 0.2 ≦ Vin ≦ Vcc − 0.2 0.2 ≦ Vin ≦ Vcc − 0.2	5.0 (Typ) 0.9 (Typ) 0.5 (Typ) 	9.0 1.8 50 10	mA mA μA μA
Power Down Voltage (at STOP)	V_{STOP}	V_{IL2} = 0.2 Vcc, V_{IH2} = 0.8 Vcc	2.0	6.0	V
$\overline{\text{RESET}}$ Pull Up Resistance	R_{RST}		80	500	kΩ
Pin Capacitance	C_{IO}	fc = 1 MHz		10	pF
Schmitt Width $\overline{\text{RESET}}$, $\overline{\text{NMI}}$, INT0	V_{TH}		0.4	1.0 (Typ)	V

(Reference) Definition of I_{DAR}



4.3 AC Electrical Characteristics

(1) Vcc = +5 V ±10%, Ta = −20 to +70℃

(fc = 8 MHz to 25 MHz)

No.	Parameter	Symbol	Variable		20 MHz		25 MHz		Unit
			Min	Max	Min	Max	Min	Max	
1	Oscillation cycle (= x)	t_{OSC}	40	125	50		40		ns
2	Clock pulse width	t_{CLK}	2x – 40		60		40		ns
3	A0 to A23 valid→ clock hold	t_{AK}	0.5x – 20		5		0		ns
4	Clock valid→ A0 to A23 hold	t_{KA}	1.5x – 60		5		0		ns
5	A0 to A23 valid→ RD/WR fall	t_{AC}	1.0x – 20		30		20		ns
6	RD/WR rise→ A0 to A23 hold	t_{CA}	0.5x – 20		5		0		ns
7	A0 to A23 valid→ D0 to D15 input	t_{AD}		3.5x – 35		140		105	ns
8	RD fall→ D0 to D15 input	t_{RD}		2.5x – 40		85		60	ns
9	RD Low pulse width	t_{RR}	2.5x – 40		85		60		ns
10	RD rise→ D0 to D15 hold	t_{HR}	0		0		0		ns
11	WR Low pulse width	t_{WW}	2.5x – 40		85		60		ns
12	D0 to D15 valid→ WR rise	t_{DW}	2.0x – 40		60		40		ns
13	WR rise →D0 to D15 hold	t_{WD}	0.5x – 10		15		10		ns
14	A0 to A23 valid→ WAIT input (1 WAIT + n mode)	t_{AW}		3.5x – 90		85		50	ns
	A0 to A23 valid→ WAIT input (0 WAIT + n mode)	t_{AW}		1.5x – 40		35		20	ns
15	RD/WR fall→ WAIT hold (1 WAIT + n mode)	t_{CW}	2.5x + 0		125		100		ns
	RD/WR fall→ WAIT hold (0 WAIT + n mode)	t_{CW}	0.5x + 0		25		20		ns

AC measuring conditions

- Output level : High 2.2 V / Low 0.8 V , CL = 50 pF
  (Note that for D0 to D15, A0 to A23, RD, WR, HWR,and CLK, CL = 100 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)

(2) Vcc = +3 V ±10%, Ta = −20 to +70℃

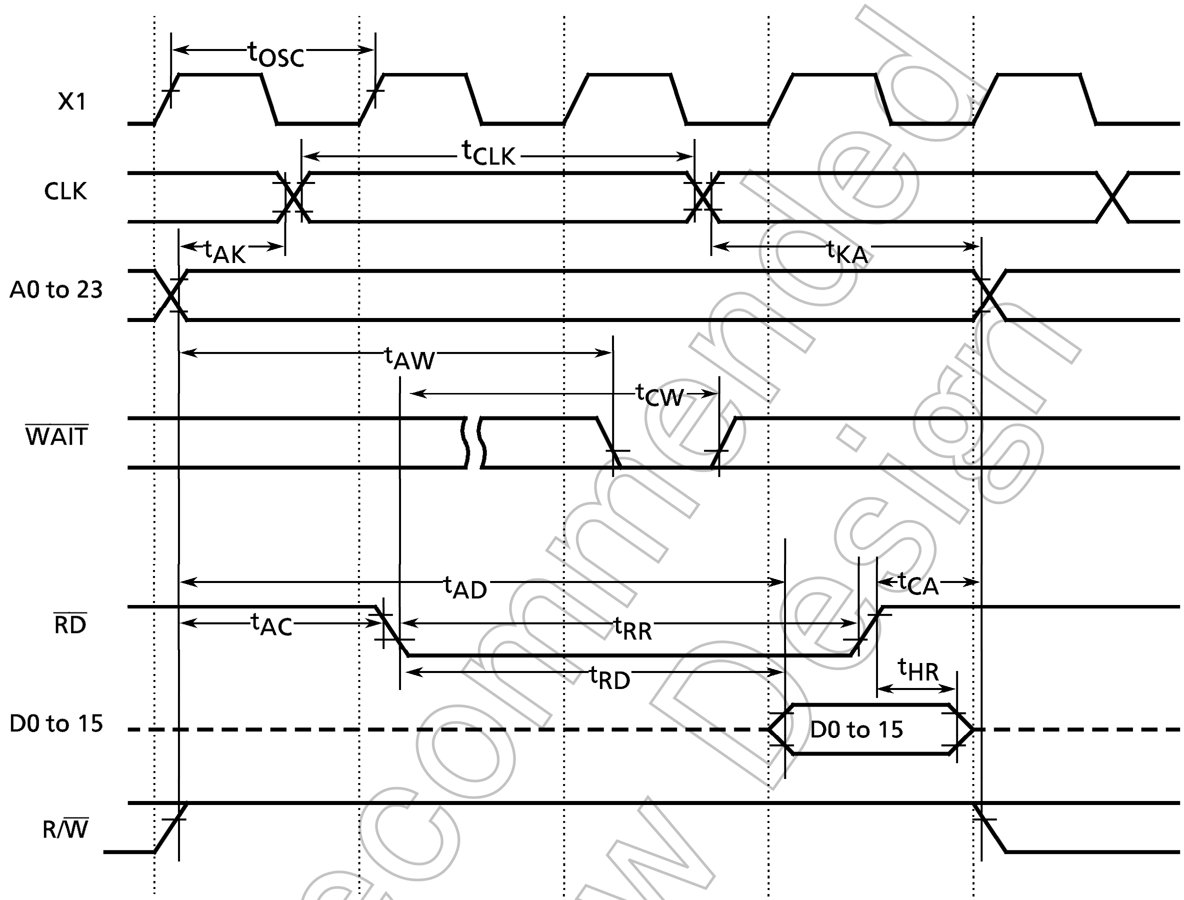
(fc = 4 MHz to 12.5 MHz)

No.	Parameter	Symbol	Variable		12.5 MHz		Unit
			Min	Max	Min	Max	
1	Oscillation cycle (= x)	t_{OSC}	80	250	80		ns
2	Clock pulse width	t_{CLK}	2x – 40		120		ns
3	A0 to A23 valid→ clock hold	t_{AK}	0.5x – 40		0		ns
4	Clock valid→ A0 to A23 hold	t_{KA}	1.5x – 80		40		ns
5	A0 to A23 valid→ RD/WR fall	t_{AC}	1.0x – 60		20		ns
6	RD/WR rise→ A0 to A23 hold	t_{CA}	0.5x – 40		0		ns
7	A0 to A23 valid→ D0 to D15 input	t_{AD}		3.5x – 125		155	ns
8	RD fall → D0 to D15 input	t_{RD}		2.5x – 115		85	ns
9	RD Low pulse width	t_{RR}	2.5x – 40		160		ns
10	RD rise→ D0 to D15 hold	t_{HR}	0		0		ns
11	WR Low pulse width	t_{WW}	2.5x – 40		160		ns
12	D0 to D15 valid→ WR rise	t_{DW}	2.0x – 60		100		ns
13	WR rise →D0 to D15 hold	t_{WD}	0.5x – 30		10		ns
14	A0 to A23 valid→ WAIT input (1 WAIT + n mode)	t_{AW}		3.5x – 130		150	ns
	A0 to A23 valid→ WAIT input (0 WAIT + n mode)	t_{AW}		1.5x – 80		40	ns
15	RD/WR fall→WAIT hold (1 WAIT + n mode)	t_{CW}	2.5x + 0		200		ns
	RD/WR fall→WAIT hold (0 WAIT + n mode)	t_{CW}	0.5x + 0		40		ns

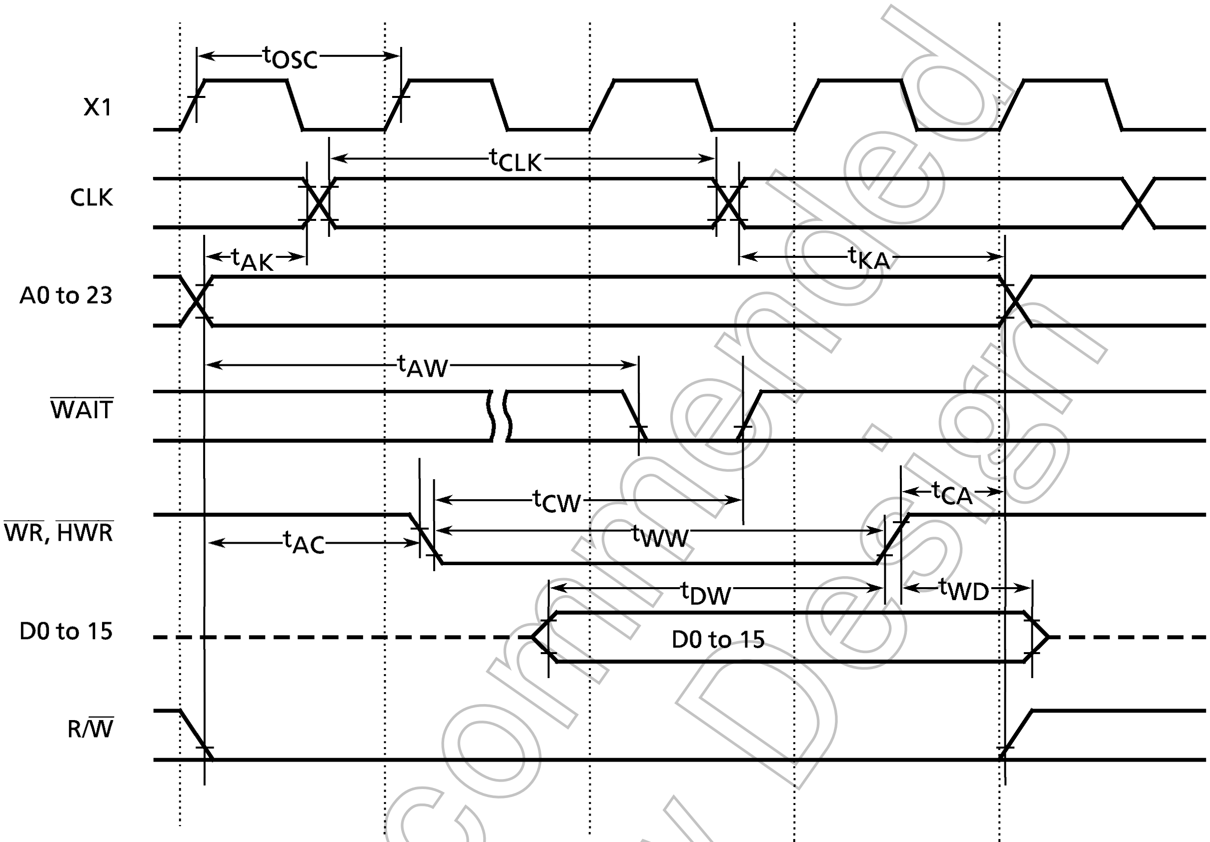
AC measuring conditions

- Output level : High 0.7 × V_{CC} / Low 0.3 × V_{CC} , CL = 50 pF
- Input level : High 0.9 × V_{CC} / Low 0.1 × Vcc

(3) Read cycle


tOSC
X1
tCLK
CLK
tAK
tKA
A0 to 23
tAW
tCW
WAIT
tAD
tCA
RD
tAC
tRR
tRD
tHR
D0 to 15
D0 to 15
R/W

(4) Write cycle



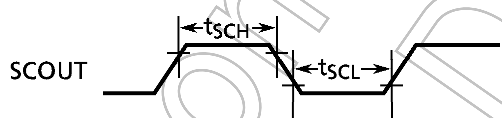
4.4 SCOUT pin AC Electrical Characteristics

Ta = − 20 to + 70°C

Parameter	Symbol	Variable Min	Variable Max	12.5 MHz Min	12.5 MHz Max	25 MHz Min	25 MHz Max
High-level pulse width VCC = + 5 V ± 10% (fc = 8 to 25 MHz)	t_{SCH}	1x − 20		60		20	
VCC = + 3 V ± 10% (fc = 4 to 12.5 MHz)		1x − 30		50		–	–
Low-level pulse width VCC = + 5 V ± 10% (fc = 8 to 25 MHz)	t_{SCL}	1x − 20		60		20	
VCC = + 3 V ± 10% (fc = 4 to 12.5 MHz)		1x − 30		50		–	–

AC measuring conditions

- Output level
  - (1) Vcc = + 5 V ± 10%
    High 2.2 V / Low 0.8 V, CL = 30pF
  - (2) Vcc = + 3 V ± 10%
    High 0.7 × Vcc / Low 0.3 × Vcc, CL = 30pF

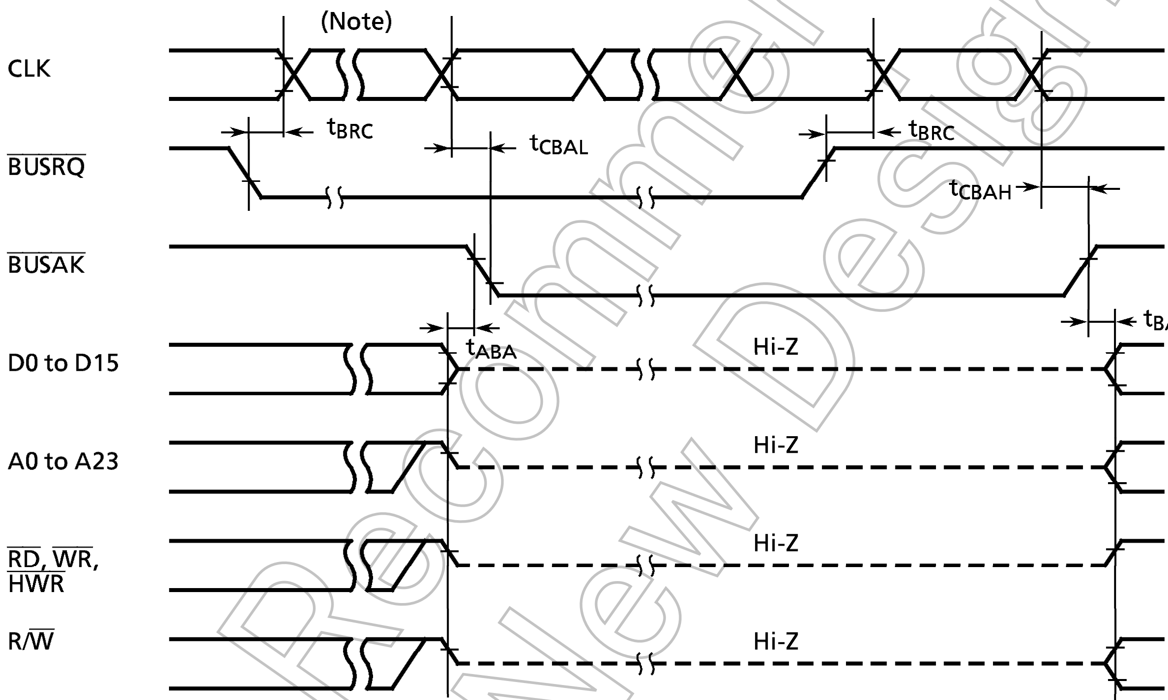


4.5 Interrupt Operation

Vcc = +5 V ± 10%, Ta = −20 to +70℃ (fc = 8 to 25 MHz)
Vcc = +3 V ± 10%, Ta = −20 to +70℃ (fc = 4 to 12.5 MHz)

Parameter	Symbol	Variable		12.5 MHz		25 MHz		Unit
		Min	Max	Min	Max	Min	Max	
$\overline{NMI}$、INT0 low-level pulse width	t_{INTAL}	4x		320		160		ns
$\overline{NMI}$、INT0 high-level pulse width	t_{INTAH}	4x		320		160		ns
INT1 to INT5 low-level pulse width	t_{INTBL}	8x + 100		740		420		ns
INT1 to INT5 high-level pulse width	t_{INTBH}	8x + 100		740		420		ns

4.6 Bus Request/Bus Acknowledge Timing



Vcc = +5 V ± 10%, Ta = −20 to +70℃ (fc = 8 to 25 MHz)
Vcc = +3 V ± 10%, Ta = −20 to +70℃ (fc = 4 to 12.5 MHz)

Parameter	Symbol	Variable		12.5 MHz		25 MHz		Unit
		Min	Max	Min	Max	Min	Max	
$\overline{BUSRQ}$ setup time for CLK	t_{BRC}	120		120		120		ns
CLK→$\overline{BUSAK}$ fall	t_{CBAL}		2.0x + 120		280		200	ns
CLK→$\overline{BUSAK}$ rise	t_{CBAH}		0.5x + 40		80		60	ns
Time from output buffer off until $\overline{BUSAK}$ fall	t_{ABA}	0	80	0	80	0	80	ns
Time from $\overline{BUSAK}$ rise until output buffer on	t_{BAA}	0	80	0	80	0	80	ns

Note: When bus release is requested with $\overline{BUSRQ}$ cleared to 0, that request cannot be granted until the previous bus cycle is terminated by a WAIT, and the WAIT is released.

5. List of Special Function Registers

(SFR ; Special Function Register)

The special function registers control the input/output functions and peripheral components. Registers are allocated to 64 bytes within the address range from 000000H to 00003FH.

The built-in registers can not be externally accessed.

(1) Interrupt control

(2) Wait control

(3) Standby mode control

Table configuration

Symbol	Name	Address	7	6	…	1	0	
								→ bit Symbol
								→ Read / Write
								→ Initial value at reset
								→ Remarks

TMP95C001 Special Function Register Address List

Address	Register Symbol	Address	Register Symbol	Address	Register Symbol	Address	Register Symbol
00000000H		10H		20H	INTE01	30H	B0CS
1H		1H		1H	INTE23	1H	B1CS
2H		2H		2H	INTE45	2H	B2CS
3H		3H		3H	INTETC01	3H	B3CS
4H		4H		4H	INTETC23	4H	MSAR0
5H		5H		5H		5H	MAMR0
6H		6H		6H	DMA0V	6H	MSAR1
7H		7H		7H	DMA1V	7H	MAMR1
8H		8H		8H	DMA2V	8H	MSAR2
9H		9H		9H	DMA3V	9H	MAMR2
AH		AH		AH	SDMACR0	AH	MSAR3
BH		BH		BH	SDMACR1	BH	MAMR3
CH		CH		CH	SDMACR2	CH	
DH		DH		DH	SDMACR3	DH	
EH		EH	STMOD	EH		EH	
FH		FH		FH	IIMC	FH	BEXCS

(1) Interrupt Control (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
INTE01	Interrupt Enable 0 / 1	20H (No RMW)	INT1				INT0			
			I1C	I1M2	I1M1	I1M0	I0C	I0M2	I0M1	I0M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTE23	Interrupt Enable 2 / 3	21H (No RMW)	INT3				INT2			
			I3C	I3M2	I3M1	I3M0	I2C	I2M2	I2M1	I2M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTE45	Interrupt Enable 4 / 5	22H (No RMW)	INT5				INT4			
			I5C	I5M2	I5M1	I5M0	I4C	I4M2	I4M1	I4M0
			R/W	W			R/W	W		
			0	0	0	0	0	0	0	0
INTETC 01	Interrupt Enable TC 0/1	23H (No RMW)	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	Interrupt Enable TC2/3	24H (No RMW)	INTTC3				INTTC2			
			ITC3C	ITC3M2	ITC3M1	ITC3M0	ITC2C	ITC2M2	ITC2M1	ITC2M0
			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.	Clears interrupt request flag.
1	Indicates interrupt request.	- - - - - Don't care - - - - -

Read / Write

R/W : Read / Write

W : Write only

No RMW : Prohibit read-modify-write. (Cannot use the EX、ADD、ADC、SUB、SBC、INC、DEC、AND、OR、XOR、STCF、RES、SET、CHG、TEST、RLC、RRC、RL、RR、SLA、SRA、SLL、SRL、RLD、or RRD instructions)。

Interrupt Control (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
IIMC	Interrupt Input Mode Contorol	2FH (No RMW)						I5LE	I0LE	NMIREE
								W	W	W
								0	0	0
								INT5 0: ↑ edge 1:level	INT0 0: ↑ edge 1:level	NMI 0: ↓ edge 1: ↓ ↑ edge
DMA0V	DMA 0 Request Vector	26H (No RMW)	DMA0V7	DMA0V6	DMA0V5	DMA0V4	DMA0V3	DMA0V2		
			W							
			0	0	0	0	0	0		
			Micro DMA0 start vector							
DMA1V	DMA 1 Request Vector	27H (No RMW)	DMA1V7	DMA1V6	DMA1V5	DMA1V4	DMA1V3	DMA1V2		
			W							
			0	0	0	0	0	0		
			Micro DMA1 start vector							
DMA2V	DMA 2 Request Vector	28H (No RMW)	DMA2V7	DMA2V6	DMA2V5	DMA2V4	DMA2V3	DMA2V2		
			W							
			0	0	0	0	0	0		
			Micro DMA2 start vector							
DMA3V	DMA 3 Request Vector	29H (No RMW)	DMA3V7	DMA3V6	DMA3V5	DMA3V4	DMA3V3	DMA3V2		
			W							
			0	0	0	0	0	0		
			Micro DMA3 start vector							

Note: Micro DMA is started by software using (2AH/2BH/2CH/2DH) write cycle of a SDMACR 0/1/2/3. (Data are invalid)

(2) Wait control (1/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
B0CS	Block 0 CS/WAIT Control Register	30H (No RMW)	B0E				B0BUS	B0W2	B0W1	B0W0
			W				W		W	
			0				0	0	0	0
			0: Disable 1: Enable				0: 16 BIT 1: 8 BIT	000: 2WAIT 001: 1WAIT 010: 1WAIT + N 011: 0WAIT		100: 0 + NWAIT 101, 110, 111: Don't set.
B1CS	Block 1 CS/WAIT Control Register	31H (No RMW)	B1E				B1BUS	B1W2	B1W1	B1W0
			W				W		W	
			0				0	0	0	0
			0: Disable 1: Enable				0: 16 BIT 1: 8 BIT	000: 2WAIT 001: 1WAIT 010: 1WAIT + N 011: 0WAIT		100: 0 + NWAIT 101, 110, 111: Don't set.
B2CS	Block 2 CS/WAIT Control Register	32H (No RMW)	B2E	B2M			B2BUS	B2W2	B2W1	B2W0
			W	W			W		W	
			1	0			0	0	0	0
			0: Disable 1: Enable	0: 16M area 1: Sets CS area			0: 16 BIT 1: 8 BIT	000: 2WAIT 001: 1WAIT 010: 1WAIT + N 011: 0WAIT		100: 0 + NWAIT 101, 110, 111: Don't set.
B3CS	Block 3 CS/WAIT Control Register	33H (No RMW)	B3E				B3BUS	B3W2	B3W1	B3W0
			W				W		W	
			0				0	0	0	0
			0: Disable 1: Enable				0: 16 BIT 1: 8 BIT	000: 2WAIT 001: 1WAIT 010: 1WAIT + N 011: 0WAIT		100:0 + NWAIT 101, 110, 111: Don't set.
BEXCS	External CS/WAIT Control Register	3FH (No RMW)					BEXBUS	BEXW2	BEXW1	BEXW0
							W		W	
							0	0	0	0
							0: 16 BIT 1: 8 BIT	000: 2WAIT 001: 1WAIT 010: 1WAIT + N 011: 0WAIT		100: 0 + NWAIT 101, 110, 111: Don't set.
MSAR0	Memory Start Address Reg. 0	34H	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			A23 to A16 Sets start address.							
MAMR0	Memory Start Address Mask Reg. 0	35H	V20	V19	V18	V17	V16	V15	V14~9	V8
			R/W							
			1	1	1	1	1	1	1	1
			Sets CS0 area size. 0 : Compare address							
MSAR1	Memory Start Address Reg. 1	36H	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			A23 to A16 Sets start address.							
MAMR1	Memory Start Address Mask Reg. 1	37H	V21	V20	V19	V18	V17	V16	V15~9	V8
			R/W							
			1	1	1	1	1	1	1	1
			Sets CS1 area size. 0 : Compare address							

Wait Control (2/2)

Symbol	Name	Address	7	6	5	4	3	2	1	0
MSAR2	Memory Start Address Reg. 2	38H	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			A23 to A16 Sets start address.							
MAMR2	Memory Start Address Mask Reg. 2	39H	V22	V21	V20	V19	V18	V17	V16	V15
			R/W							
			1	1	1	1	1	1	1	1
			Sets CS2 area size. 0 : Compare address							
MSAR3	Memory Start Address Reg. 3	3AH	S23	S22	S21	S20	S19	S18	S17	S16
			R/W							
			1	1	1	1	1	1	1	1
			A23 to A16 Sets start address.							
MAMR3	Memory Start Address Mask Reg. 3	3BH	V22	V21	V20	V19	V18	V17	V16	V15
			R/W							
			1	1	1	1	1	1	1	1
			Sets CS3 area size. 0 : Compare address							

(3) Stanby Mode Control

Symbol	Name	Address	7	6	5	4	3	2	1	0
STMOD	Stand-By-Mode Control Register	1EH (No RMW)	RDE		SCOUTST	CLKST	HALTM1	HALTM0		DRVE
			W		R/W					R/W
			0		0	0	0	0		0
			1: PSRAM mode		0: Output enable 1: Output disable (High-impedance)	0: Output enable 1: Output disable (High-impedance)	Halt mode setting 00: RUN 01: STOP 10: IDLE 11: Don't care			1: Pins are driven in STOP mode.

Note: "0" must not be written in <CLKST> and <SCOUTST>. (Only resetting can clear.)

6. Port Section Equivalent Circuit Diagram

- Reading circuit diagrams

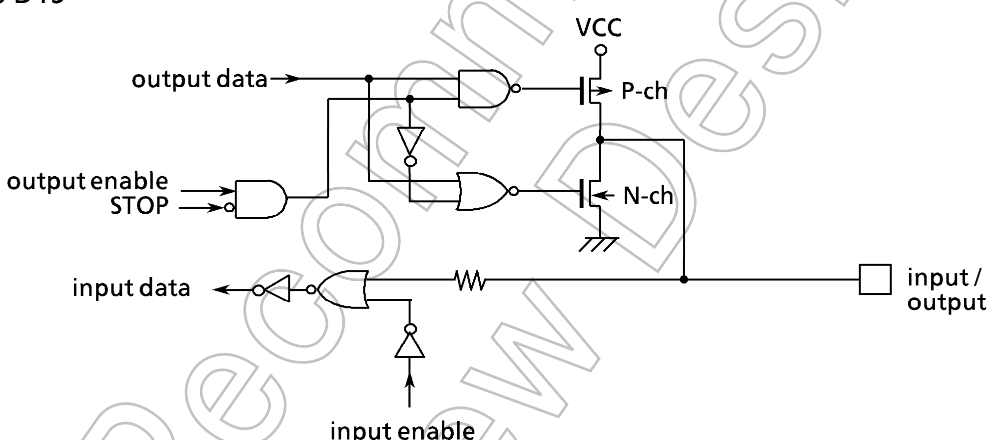
  Basically, TMP95C001 uses the same gate symbols as the standard CMOS logic IC (74HCxxx) series.

  The following is a special signal.

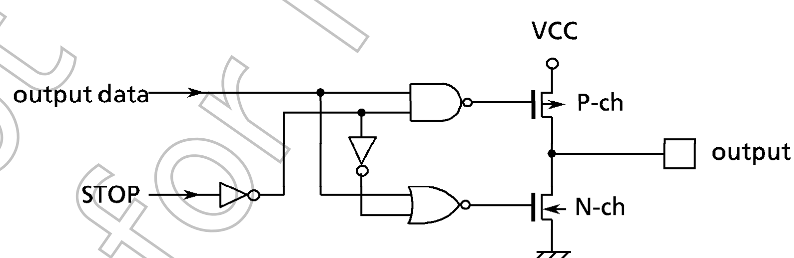
  STOP : When the hold mode register is set to STOP mode (STMOD<HALTM1,0> set to 0,1) and the CPU executes the HALT instruction, the STOP signal is set to active, "1". Note that when drive enable bit STMOD<DRVE> is set to "1", STOP remains at "0".

- The input protection resistor operates in the range of several tens to several hundreds of ohms.

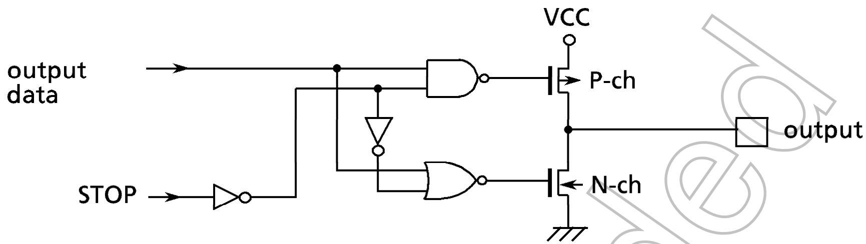
■ D0 to D15



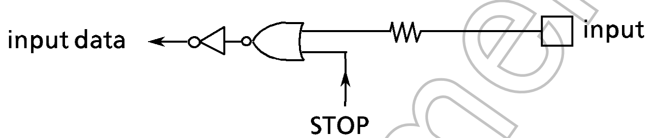
■ A0 to A23



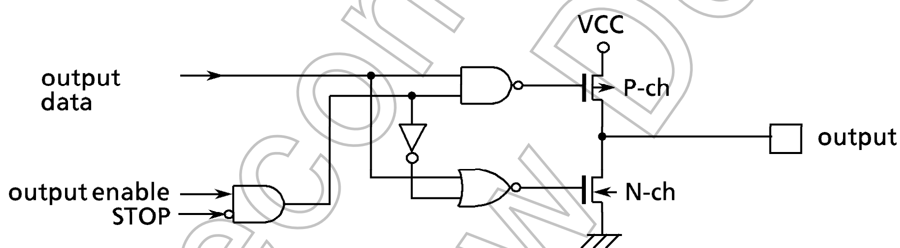
■ $\overline{RD}$, $\overline{WR}$, $\overline{HWR}$, $\overline{BUSAK}$, R/$\overline{W}$



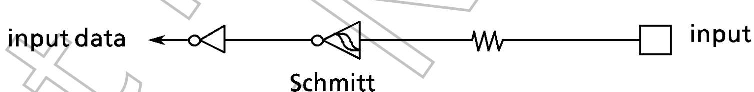
■ $\overline{BUSRQ}$, $\overline{WAIT}$



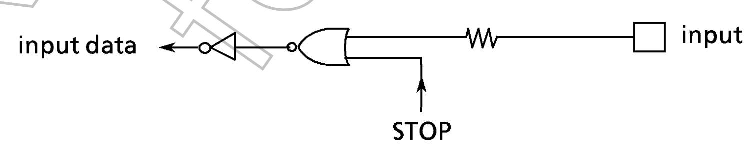
■ SCOUT



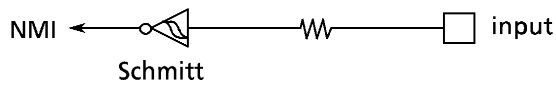
■ INT0



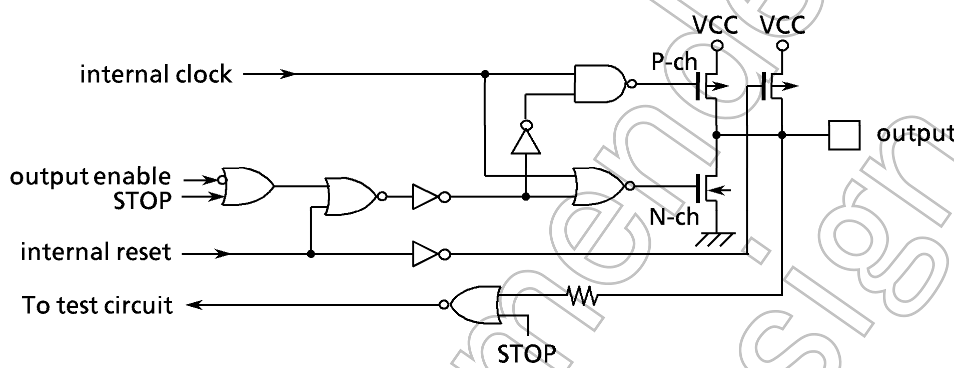
■ INT1 to 5



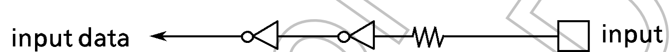
■ $\overline{\text{NMI}}$



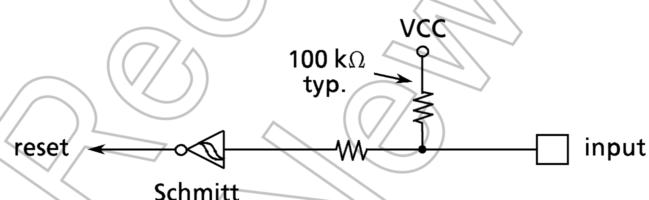
■ CLK



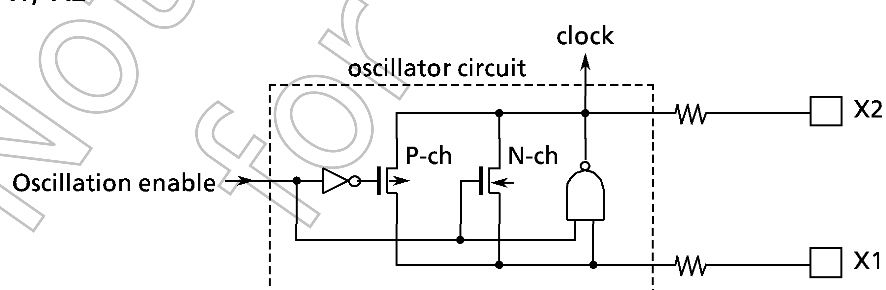
■ AM8/$\overline{16}$



■ $\overline{\text{RESET}}$



■ X1, X2



Note : The oscillation enable signal becomes nonactive "0" by execution of HALT instruction (STOP mode).

7. Cautions and Restrictions

(1) Special Notation and Terms

① Internal I/O registers: Register symbols <bit symbols>

Example: MSAR0<S23> ··· The S23 bit of the MSAR0 register

② Read-modify-write instructions

The CPU reads the data from memory, modifies them, and writes them to the same memory address.

Example 1: SET 3, (MSAR0)··· Sets bit 3 of MSAR0 register.

Example 2: INC 1, (100H) ··· Increments data at address 100H by 1.

- TLCS-900 read-modify-write instructions.

Exchange

EX	(mem), R		

Arithmetic Operations

ADD	(mem), R/#	ADC	(mem), R/#
SUB	(mem), R/#	SBC	(mem), R/#
INC	#3, (mem)	DEC	#3, (mem)

Logical Operations

AND	(mem), R/#	OR	(mem), R/#
XOR	(mem), R/#		

Bit Operations

STCF	#3/A, (mem)	RES	#3, (mem)
SET	#3/A, (mem)	CHG	#3, (mem)
TSET	#3, (mem)		

Rotate and shift

RLC	(mem)	RRC	(mem)
RL	(mem)	RR	(mem)
SLA	(mem)	SRA	(mem)
SLL	(mem)	SRL	(mem)
RLD	(mem)	RRD	(mem)

③ One state

The single cycle resulting from dividing the oscillation frequency by 2 is called "one state".

Example: At oscillation frequency 25 MHz

2/25 MHz = 80 ns = 1 state

(2) Points of Note and Restrictions

① AM8/$\overline{16}$ pin

Connect this pin to Vcc or GND pins. Do not change pin levels during operation.

② CPU (Micro DMA)

Only "LDC cr, r" and "LDC r, cr" can write or read data to or from control registers, such as the transfer source register (DMASn) in the CPU.

③ As this device does not support minimum mode, do not use the "MIN" instruction.

④ POP SR instruction

Please execute POP SR instruction during DI condition.

⑤ Releasing the HALT mode by requesting an interruption

Usually, interrupts can release all halts status. However, the interrupts = ($\overline{NMI}$, INT0), which can release the HALT mode may not be able to do so if they are input during the period CPU is shifting to the HALT mode (for about 3 clocks of X1) with IDLE or STOP mode. (In this case, an interrupt request is kept on hold internally.)

If another interrupt is generated after it has shifted to HALT mode completely, halt status can be released without difficultly. The priority of this interrupt is compare with that of the interrupt kept on hold internally, and the interrupt with higher priority is handled first followed by the other interrupt.