Part 2

TMP68301AF-12 / TMP68301AF-16
TMP68301AFR-12 / TMP68301AFR-16

1. Introduction

TMP68301A uses the 68HC000, the CMOS version of the 68000, as its core processor. It also includes peripheral circuits such as a serial interface, parallel interface, timer, interrupt controller, and address decoder.

In addition, TMP68301A can directly use 68000 development environments and software resources.

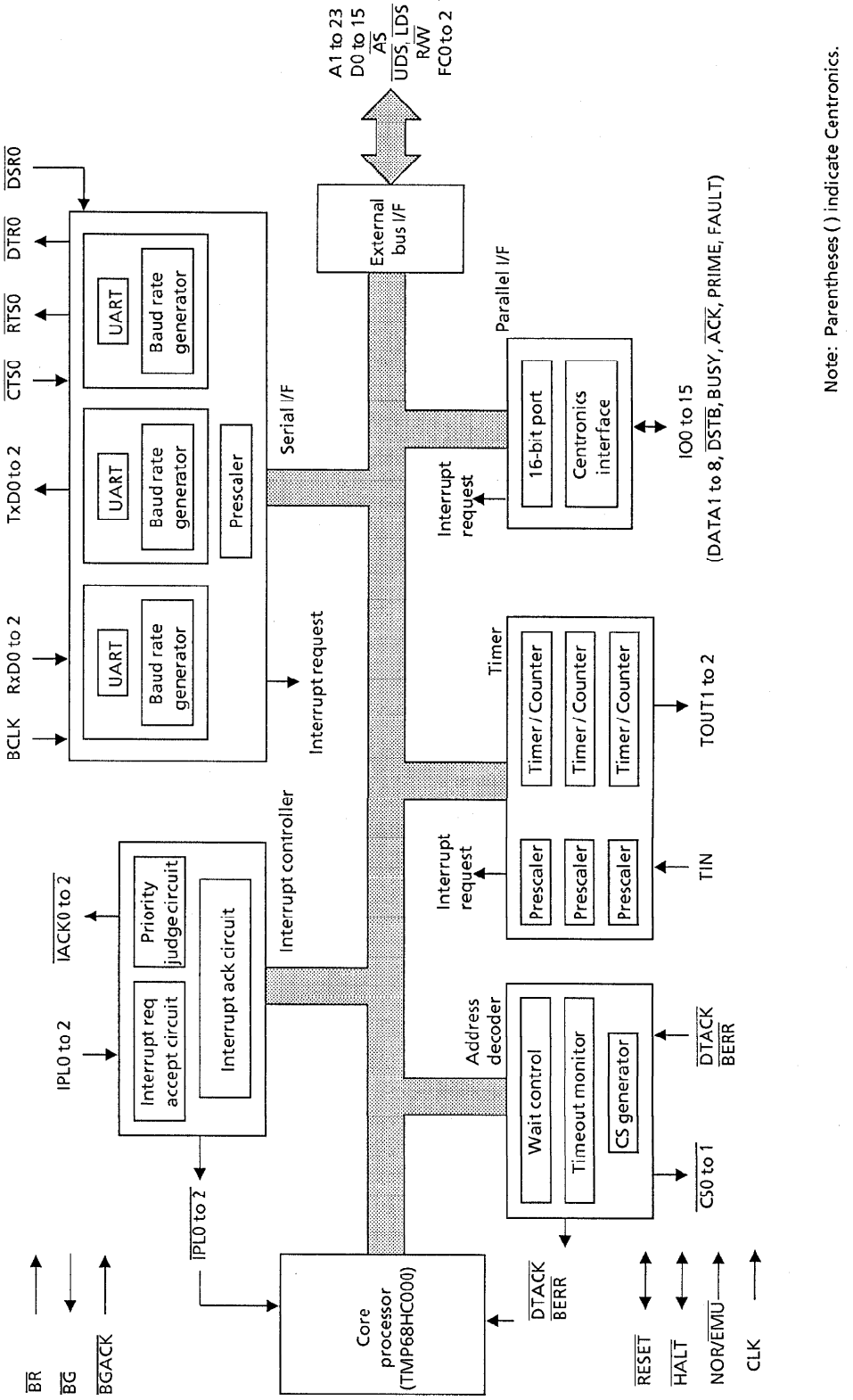
- Core processor 68HC000
- Minimum instruction execution time : 240ns (with 16.67MHz-system clock)
- 17 32-bit registers
- 16M-byte direct addressing
- 56 powerful basic instructions
- 14 addressing modes

- 3-channel asynchronous serial interface
- 16-bit parallel I/O interface (supports a Centronics interface)
- 3-channel, 16-bit timer/counter
- 10-channel interrupt controller (3 external channels, 7 internal channels; can be extended by software to 10 external channels)
- 2-channel chip-select signal output ($\overline{CS0}$, $\overline{CS1}$)
- Automatic wait state insertion
- Bus monitor function
- Low power consumption (CMOS)
- 2 types of package: 100-pin QFP and 100-pin RFP

TMP68301A has two operating modes: normal operating mode, and emulation mode that enables use of an in-circuit emulator (ICE), a 68000 development tool. 0In emulation mode, the 68HC000 core built into TMP68301A is disconnected from the bus, and the internal peripheral circuits are controlled by address, data, and control signals from the development tool.

The 68HC000 core built into TMP68301A is the same as the standard TMP68HC000, except that 8-bit peripheral device control signals E, $\overline{VPA}$, and $\overline{VMA}$ are disabled. For 68HC000 operation and instructions, refer to your TLCS-68000 Data Book.

Figure 1.1 is the TMP68301A block diagram.



Note: Parentheses ( ) indicate Centronics.

Figure 1.1 TMP68301A Block Diagram

2. Signal Description

This section briefly describes input and output signals.

The terms "assert" and "negate" appear frequently. These terms are used to avoid ambiguity when terms such as "active high" and "active low" are used. "Assert" is used to show that signals are active or true, irrespective of whether the signal is electrically high or low. "Negate" is used to show that signals are inactive or false.

2.1 Pin Assignments

Figure 2.1 shows the pin assignments.

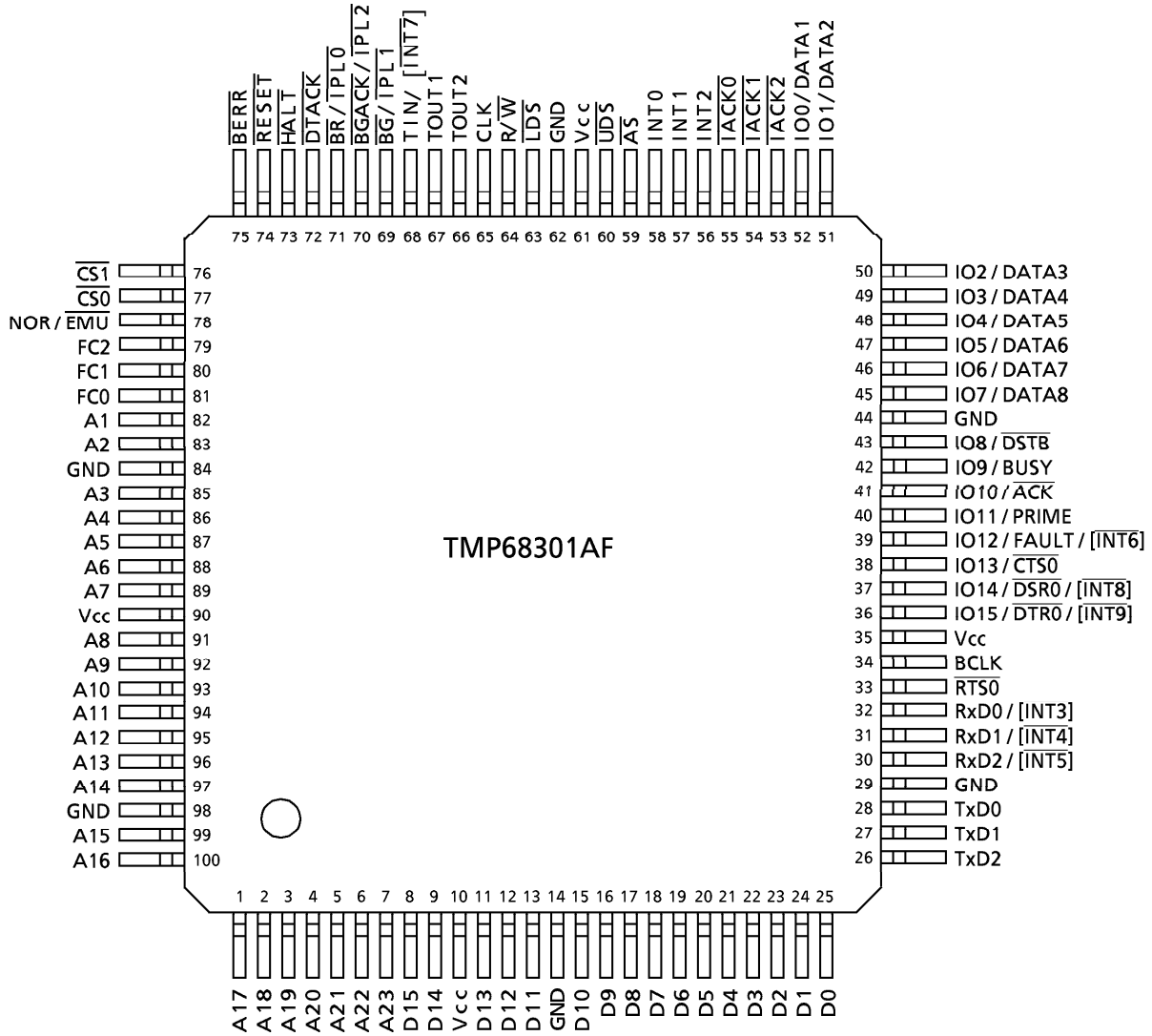


Figure 2.1 Pin Assignements (top view) (1/2)

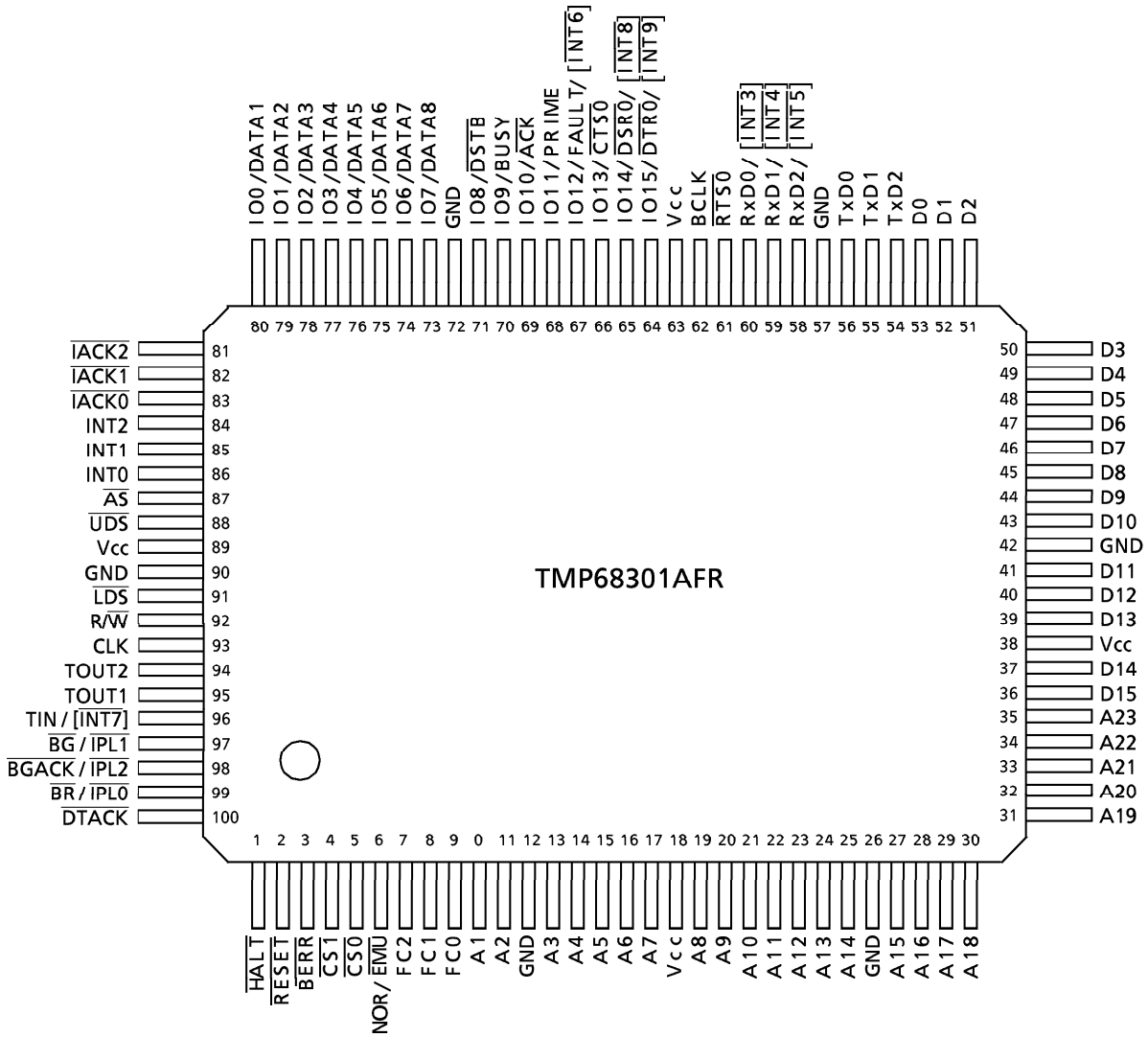


Figure 2.1 Pin Assignments (top view) (2/2)

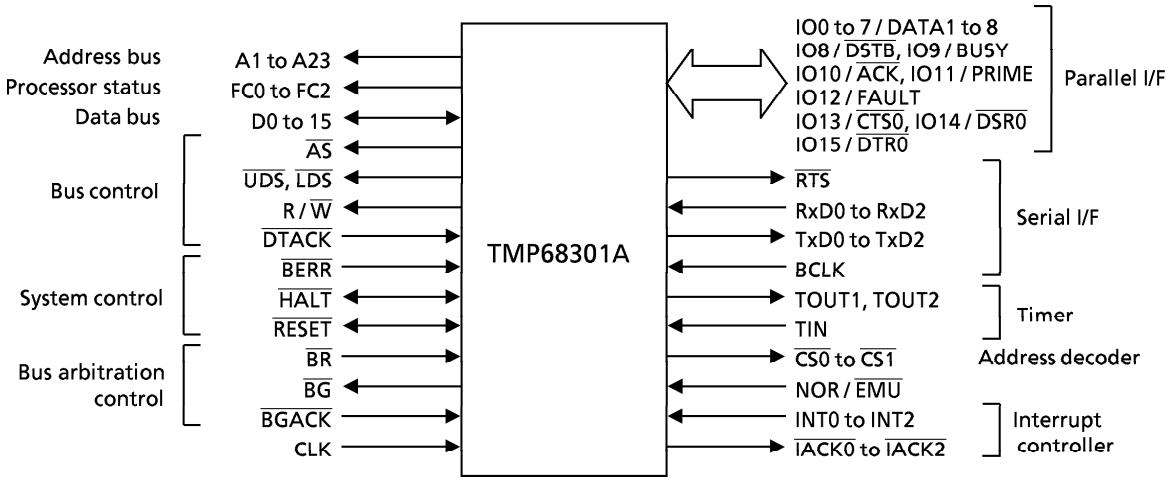


Figure 2.2 Normal Mode Input/Output Signals When Configured as Bus Master

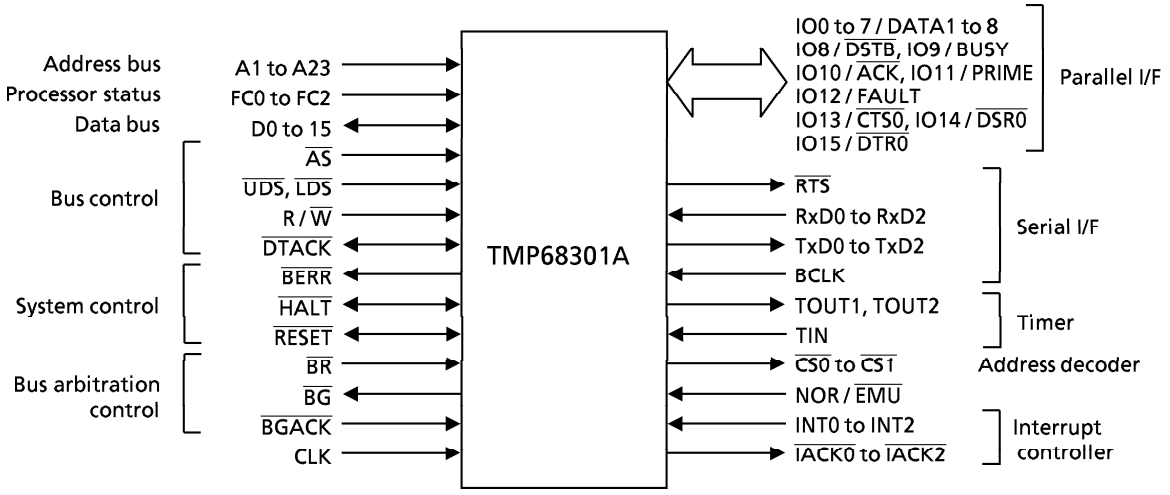


Figure 2.3 Normal Mode Input/Output Signals When Bus Mastership Released

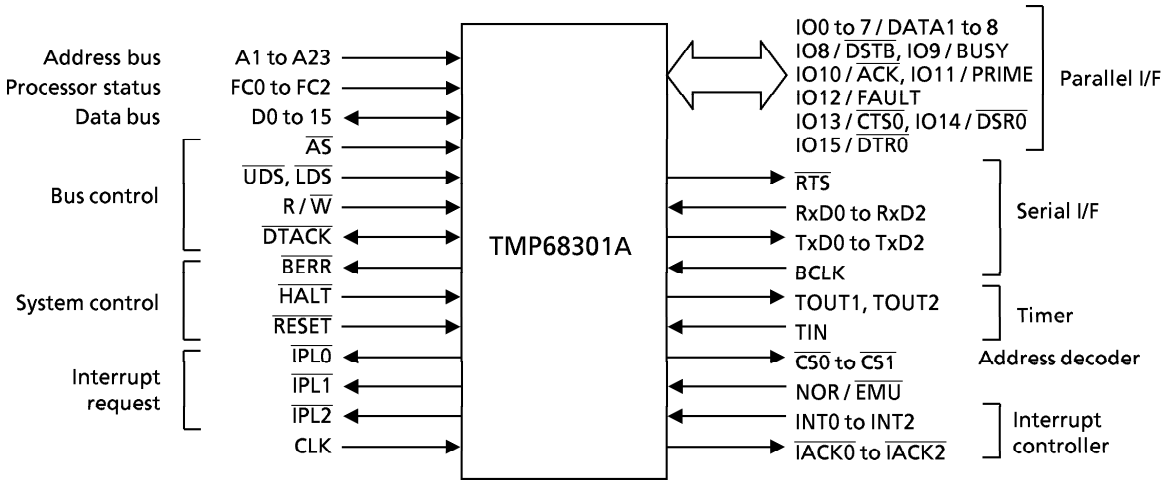


Figure 2.4 Emulation Mode Input/Output Signals

2.2 Pin Names and Functions

The following describes pin states and functions in normal and emulation modes.

NOR : Normal mode, EMU : Emulation mode,
O.D : Open drain output, O : Output, I : Input, I/O : Input/output

Signal name	Pin status		Function
	NOR	EMU	
A1 to A23	O	I	23-bit address bus can directly access 16M bytes of memory.
FC0 to FC2	O	I	These signals show the status of the processor and the current cycle type. For details, see Table 2.2.
D0 to D15	I/O	I/O	16-bit general-purpose data bus.
$\overline{AS}$	O	I	This signal indicates that there is a valid address on the address bus.
R/$\overline{W}$	O	I	This signal indicates whether the data transfer is read (high) or write (low).
$\overline{UDS}$ / $\overline{LDS}$	O	I	These signals control the data on the data bus. See Table 2.1 for details.
$\overline{DTACK}$	I	O.D	This signal indicates the end of a data transfer.
$\overline{BR}$ ($\overline{IPL0}$)	I	O	This signal is wire-ORed with all other devices that could become bus masters. The signal indicates that another device is requesting bus mastership. In emulation mode, this signal becomes the $\overline{IPL0}$ output. The $\overline{IPL}$ signal codes the priority level of a device requesting the interrupt.
$\overline{BG}$ ($\overline{IPL1}$)	O	O	This signal indicates to devices that could become bus masters that the processor will release control of the bus at the end of the current bus cycle. In emulation mode, the signal becomes the $\overline{IPL1}$ output.
$\overline{BGACK}$ ($\overline{IPL2}$)	I	O	This signal indicates that another device has become the bus master. In emulation mode, the signal becomes the $\overline{IPL2}$ output.
$\overline{BERR}$	I	O.D	This signal reports to the processor that there is a problem in the current cycle.
$\overline{RESET}$	O.D	I	In combination with $\overline{HALT}$, this signal resets the processor. If the RESET instruction is executed, this signal functions as a reset signal for external devices.
$\overline{HALT}$	O.D	I	As an input, this signal halts the processor when the current bus cycle ends. Also, the signal acts as an output when a double bus error exception occurs.
CLK	I	I	Clock input pin.
INT0, INT1, INT2	I	I	External interrupt request input.
$\overline{IACK0}$, $\overline{IACK1}$ $\overline{IACK2}$	O	O	These signals indicate the interrupt acknowledge cycles corresponding to INT0, INT1, and INT2.
NOR / $\overline{EMU}$	I	I	Normal mode/emulation mode switch signal.

NOR : Normal mode, EMU : Emulation mode,
O : Output, I : Input, I/O : Input/output

Signal name	Pin status		Function
	NOR	EMU	
RxD0 / [$\overline{INT3}$] RxD1 / [$\overline{INT4}$] RxD2 / [$\overline{INT5}$]	I	I	Serial interface data inputs. These signals also act as interrupt inputs according to the expansion interrupt register.
TxD0, TxD1, TxD2	O	O	Serial interface data output.
BCLK	I	I	Reference clock input for generating the serial interface baud rate.
$\overline{RTS0}$	O	O	$\overline{RTS}$ signal for serial interface channel 0.
TIN [$\overline{INT7}$]	I	I	Signal input to each timer channel. This signal also acts as an interrupt input according to the expansion interrupt register.
TOUT1, TOUT2	O	O	Timer channel 1 and 2 output signals.
IO0 to IO7 / DATA1 to DATA8	I/O	I/O	General-purpose I/O ports or data bus for Centronics interface.
IO8 / $\overline{DSTB}$	I/O	I/O	General-purpose I/O port or strobe signal for Centronics interface.
IO9 / BUSY	I/O	I/O	General-purpose I/O port or busy signal for Centronics interface.
IO10 / $\overline{ACK}$	I/O	I/O	General-purpose I/O port or acknowledge signal for Centronics interface.
IO11 / PRIME	I/O	I/O	General-purpose I/O port or prime signal for Centronics interface.
IO12 / FAULT [$\overline{INT6}$]	I/O	I/O	General-purpose I/O port or fault signal for Centronics interface. This signal also acts as an interrupt input according to the expansion interrupt register.
IO13 / $\overline{CTS0}$	I/O	I/O	General-purpose I/O port or $\overline{CTS}$ signal for serial interface channel 0.
IO14 / $\overline{DSR0}$ [$\overline{INT8}$]	I/O	I/O	General-purpose I/O port or $\overline{DSR}$ signal for serial interface channel 0. This signal also acts as an interrupt input according to the expansion interrupt register.
IO15 / $\overline{DTR0}$ [$\overline{INT9}$]	I/O	I/O	General-purpose I/O port or $\overline{DTR}$ signal for serial interface channel 0. This signal also acts as an interrupt input according to the expansion interrupt register.
$\overline{CS0}$, $\overline{CS1}$	O	O	These signals are used to access the memory area set by the address decoder.
Vcc	–	–	Power input pin (+ 5V).
GND	–	–	GND pin (0V).

Table 2.1 Data Bus Control By Data Strobe

$\overline{UDS}$	$\overline{LDS}$	R/$\overline{W}$	D8 to D15	D0 to D7
H	H	–	Data invalid	Data invalid
L	L	H	Valid data bits 8 to 15	Valid data bits 0 to 7
H	L	H	Data invalid	Valid data bits 0 to 7
L	H	H	Valid data bits 8 to 15	Data invalid
L	L	L	Valid data bits 8 to 15	Valid data bits 0 to 7
H	L	L	Valid data bits 0 to 7*	Valid data bits 0 to 7
L	H	L	Valid data bits 8 to 15	Valid data bits 8 to 15*

L : LOW H : HIGH
* : This condition is a result of the current implementation and may not apply in the future.

Table 2.2 Function Code Output

FC2	FC1	FC0	Cycle type	
L	L	L	*	
L	L	H	User data	
L	H	L	User program	
L	H	H	*	
H	L	L	*	
H	L	H	Supervisor data	
H	H	L	Supervisor program	
H	H	H	Interrupt acknowledge	(Note)

L : LOW H : HIGH
* : Undefined, for future use

Note : If FC0, FC1, and FC2 are pulled up, when releasing the bus, the state is the same as an interrupt acknowledge cycle, and the interrupt controller malfunctions. To prevent this, pull down one of FC0, FC1, or FC2.

2.3 Signal Summary

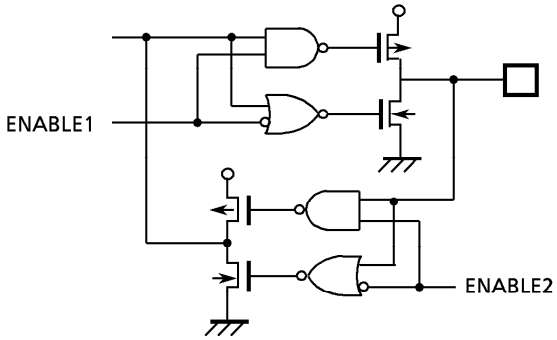
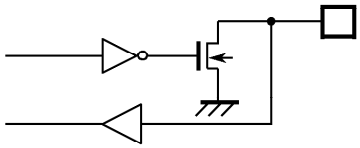
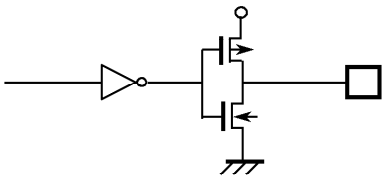
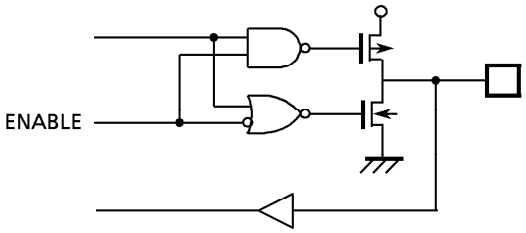
Table 2.3 Signal Summary

Signal name	Mnemonic	Input/output	Active state	Tri-state output	When $\overline{HALT}$ asserted	When $\overline{BG}$ asserted	When $\overline{RESET}$ / $\overline{HALT}$ both asserted
Address bus	A1 to A23	Out (In)	H	Y	Hi-Z	Hi-Z	Hi-Z
Data bus	D0 to D15	In/out (Out/In)	H	Y	Hi-Z	Hi-Z	Hi-Z
Address strobe	$\overline{AS}$	Out (In)	L	Y	H	Hi-Z	Hi-Z
Read / write	R/$\overline{W}$	Out (In)	H (read) L (write)	Y	H	Hi-Z	Hi-Z
Upper and lower data strobe	$\overline{UDS}$, $\overline{LDS}$	Out (In)	L	Y	H	Hi-Z	Hi-Z
Data transfer acknowledge	$\overline{DTACK}$	In (Out)	L	O.D.	–	–	–
Bus request	$\overline{BR}$ (/ $\overline{IPL0}$)	In (Out)	L	N	–	–	–
Bus grant	$\overline{BG}$ (/ $\overline{IPL1}$)	Out	L	N	H	L	H
Bus grant acknowledge	$\overline{BGACK}$ (/ $\overline{IPL2}$)	In (Out)	L	N	–	–	–
Bus error	$\overline{BERR}$	In (Out)	L	O.D.	–	–	–
Reset	$\overline{RESET}$	In/out	L	O.D.	O.D.	O.D.	L
Halt	$\overline{HALT}$	In/out	L	O.D.	L	O.D.	L
Function code	FC0, FC1, FC2	Out (In)	H	Y	–	Hi-Z	Hi-Z
Clock	CLK	In	H	–	–	–	–
I/O port	IO0 to 15	In/out	H	–	–	–	–
Timer output	TOUT1, TOUT2	Out	H	N	–	–	L
Timer input	TIN	In	L	–	–	–	–
Request to send	$\overline{RTS0}$	Out	L	N	–	–	H
Receive data	RxD0 to 2	In	H	–	–	–	–
Transmit data	TxD0 to 2	Out	H	N	–	–	H
Baud rate clock	BCLK	In	H	–	–	–	–
Mode switch	NOR / $\overline{EMU}$	In		–	–	–	–
Interrupt request	INT0 to 2	In	H	–	–	–	–
Interrupt acknowledge	$\overline{IACK0 \text{ to } 2}$	Out	L	N	–	–	H
Chip select	$\overline{CS0}$, $\overline{CS1}$	Out	L	N	–	–	H
Power input	V_{CC}	In	–	–	–	–	–
Ground	GND	In	–	–	–	–	–

The parentheses in the input / output column indicate in emulation mode.

Note : O.D. : Open drain Hi-Z : High impedance – : Optional
In : Input
Out : Output H : HIGH Y : Yes
In/out : Input/output L : LOW N : No

2.4 Pin Input / Output Circuits

Pin	Input/output circuit	Remarks
A1 to A23, D0 to D15 FC0 to FC2, $\overline{AS}$, $\overline{UDS}$, $\overline{LDS}$ R/$\overline{W}$		Tri-state output
$\overline{RESET}$, $\overline{HALT}$, $\overline{DTACK}$, $\overline{BERR}$		Sink open drain output
$\overline{BG}$ TOUT1, TOUT2 $\overline{RTS0}$ TxD0, TxD1, TxD2 $\overline{IACK0}$, $\overline{IACK1}$, $\overline{IACK2}$ $\overline{CS0}$, $\overline{CS1}$		Push/pull output
CLK TIN RxD0, RxD1, RxD2 BCLK INT0, INT1, INT2 NOR / $\overline{EMU}$		
$\overline{BR}$, $\overline{BGACK}$ IO0~IO7 IO8~IO15		

3. Address Decoder

3.1 Outline

The address decoder generates two $\overline{CS}$ (chip select) signals to select memory or external devices. The registers in the address decoder can freely select the $\overline{CS}$ memory area from the 16M bytes of address space available to the core processor. The size of the memory area can also be modified. A $\overline{DTACK}$ signal can be internally generated to allow the insertion of between 0 and 7 wait cycles in each memory area.

The address decoder can freely allocate, in boundary units of 1K byte, the internal peripheral circuit registers to the 16M bytes of address space.

Moreover, to allow the decoder to monitor the bus cycles in all memory areas, the address decoder includes a function to automatically generate $\overline{BERR}$ (bus errors) after a set timeout time.

3.2 Area Selection

The address decoder decodes the addresses output from the core processor at each bus cycle. The decoder checks for access to the two memory areas and access to the register area. When a memory area is accessed, the address decoder externally asserts the chip select signal ($\overline{CSn}$), inserts the pre-set wait cycle, and outputs the internal $\overline{DTACK}$ signal, which is generated automatically, to the core processor. Depending on circumstances, it may be possible to control the wait cycle inserted using the externally generated $\overline{DTACK}$ signal.

When a register area is accessed, the address decoder selects an associated internal peripheral circuit and transfers the data. An internal $\overline{DTACK}$ signal is automatically generated and output to the core processor. At this time, a $\overline{DTACK}$ signal from an external source is ignored.

When a register area is accessed, a valid signal is output to the external pins same as when a memory area is accessed. In the read cycle, if external memory overlaps a register area, the data from the internal peripheral circuit are valid; the data from the external source are ignored. In the write cycle, take care when external memory overlaps a register area, as the data are written to both the register area and external memory. To avoid writing to both the register and external memory areas, either ensure that the register area and the external memory do not overlap, or select the external memory using the $\overline{CSn}$ signal. If the memory area selected using the $\overline{CSn}$ signal overlaps with the register area, the access area is determined according to the area priority in Table 3.1. When the register area is accessed, the $\overline{CSn}$ signal is not asserted, thus avoiding accessing the same memory area.

Table 3.1 Area and Access Priority

Area	Priority
Register area	High
Memory area 0 (area selected by $\overline{CS0}$)	|
Memory area 1 (area selected by $\overline{CS1}$)	Low

In the interrupt acknowledge cycle (IACK cycle), the core processor starts loading the vector number. At this time, no area selection signal is generated, even when the address generated attempts to access an area. (No register area access signal or $\overline{CSn}$ signal is asserted.)

This prevents inadvertently accessing areas during the IACK cycle.

When the vector number is input from an external source, the address decoder can set the number of wait cycles for a IACK cycle independently of other areas.

Figure 3.1 outlines $\overline{CSn}$ signal generation.

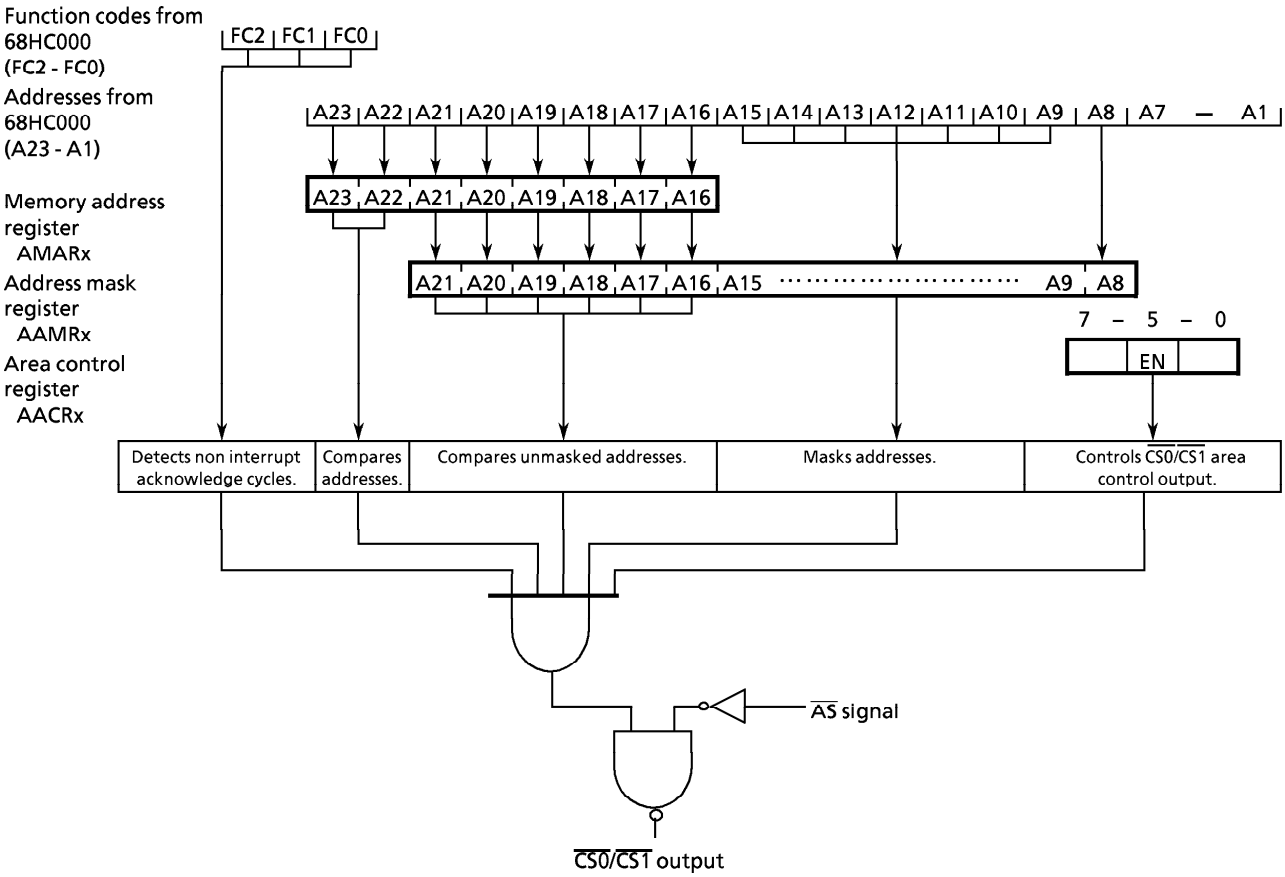


Figure 3.1 Chip Select Signal Generation

Figure 3.2 shows the output status of area selection signals for overlapping areas.

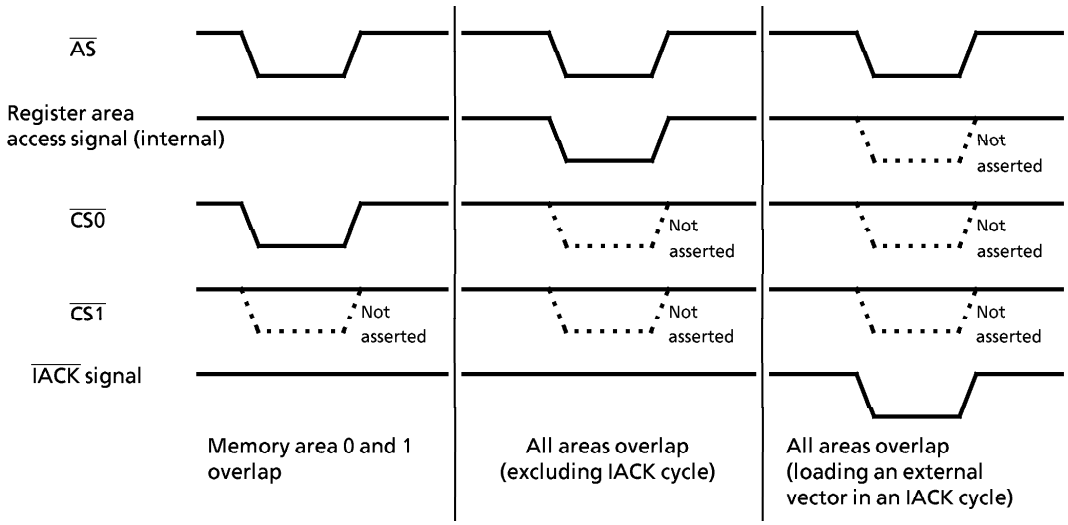


Figure 3.2 Output Status of Area Selection Signals

Figure 3.3 shows the relationship between the $\overline{CSn}$ signal and the AS signal.

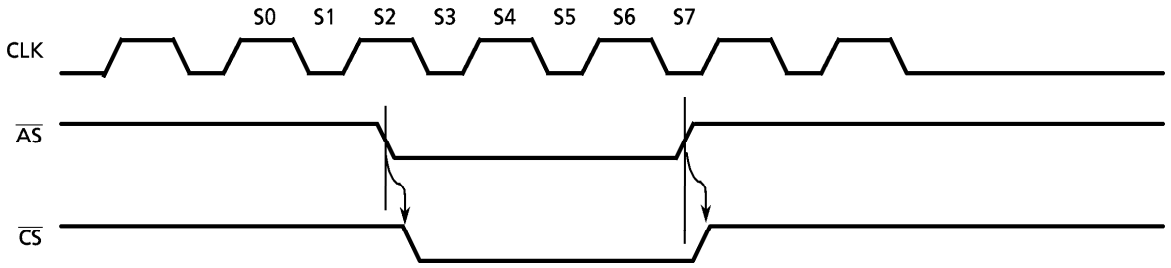


Figure 3.3 $\overline{CS}$ Signal and $\overline{AS}$ Signal Relationship

3.2.1 Memory Area

The memory area register (AMAR) specifies the start address in the memory area. Two areas can be specified in the 16M-byte address space. Every bus cycle, the address decoder compares the bus address and the value in the address register. If the values match, the address decoder determines that the memory area is accessed.

The address mask register determines the size of the memory area. Figure 3.2 shows the relationship between the size of the area and the value in the address mask register for contiguous address space.

Table 3.2 Relationship Between Memory Area Size and Address Mask Register

Memory Size / Mask Register	256	512	64K	128K	256K	512K	1M	2M	4M
For $\overline{CS0}$ (AAMR0)	00	01	03	07	0F	1F	3F	7F	FF
For $\overline{CS1}$ (AAMR1)	00	01	03	07	0F	1F	3F	7F	FF

The numeric values in the Table are hexadecimal.

The address mask register addresses for masking are determined in units of bits. Therefore, depending on the setting, non-contiguous memory areas can be specified. For example, setting the memory address register to $00 and the address mask register to $5F specifies two memory areas, $000000 - $07FFFF and $100000 - $17FFFF.

3.2.2 Register Area

The register area contains internal peripheral circuit registers, including the address decoder registers. The size of the register area is 1K byte. The start address of the register area is specified by the relocation register. Accordingly, the register area can be freely allocated at 1K-byte boundaries within the 16M-byte address space. Figure 3.4 shows actual address generation methods for the registers. The register area after reset release is allocated to $FFFC00 - $FFFFFF (last part of address space). (See Figure 3.5.)

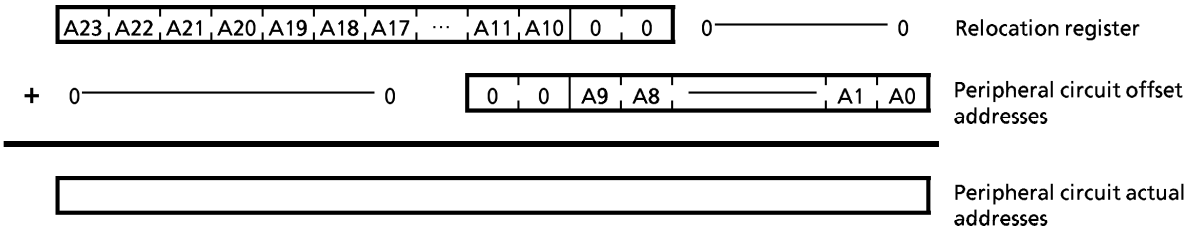


Figure 3.4 Real Register Address Generation

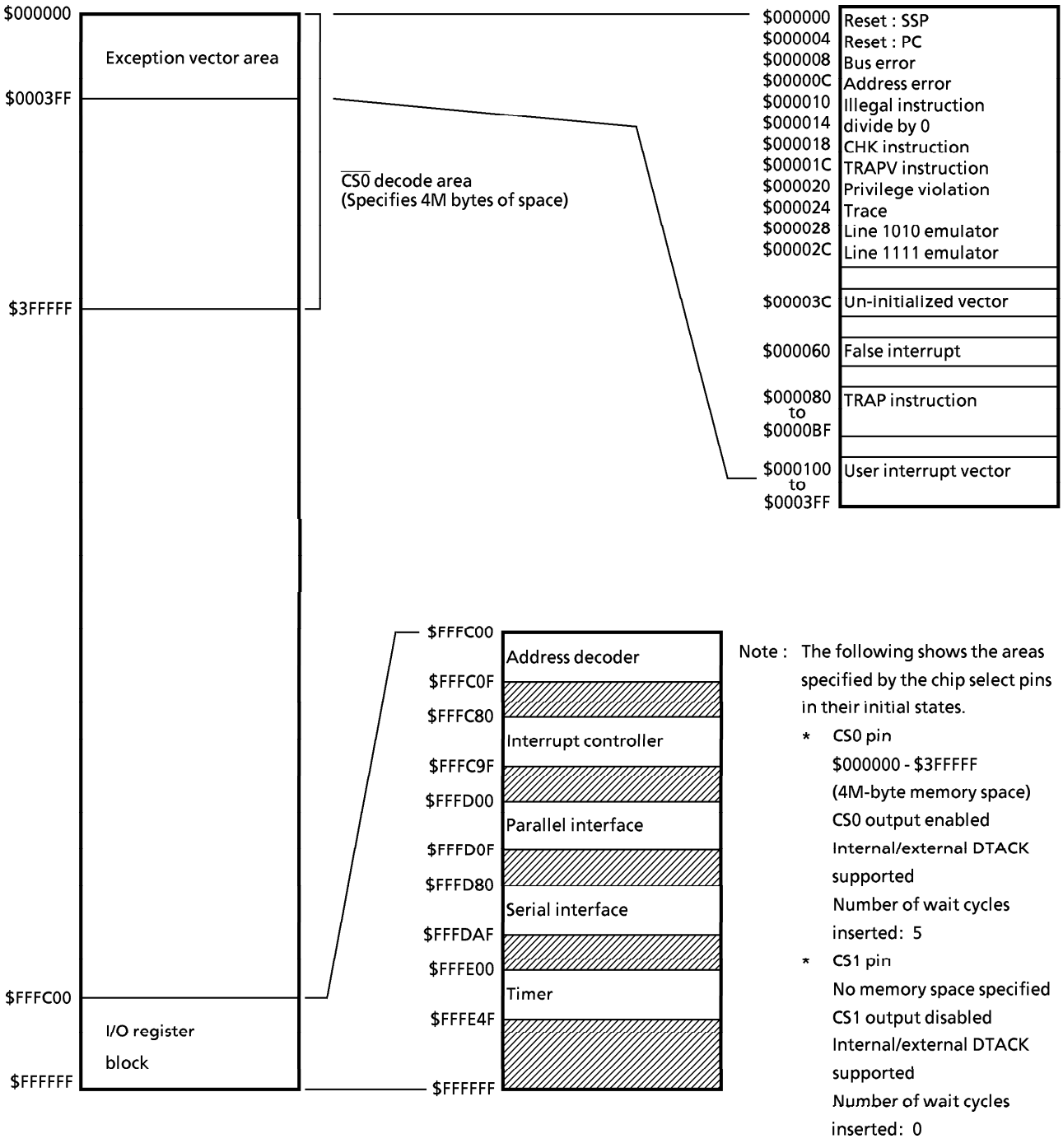
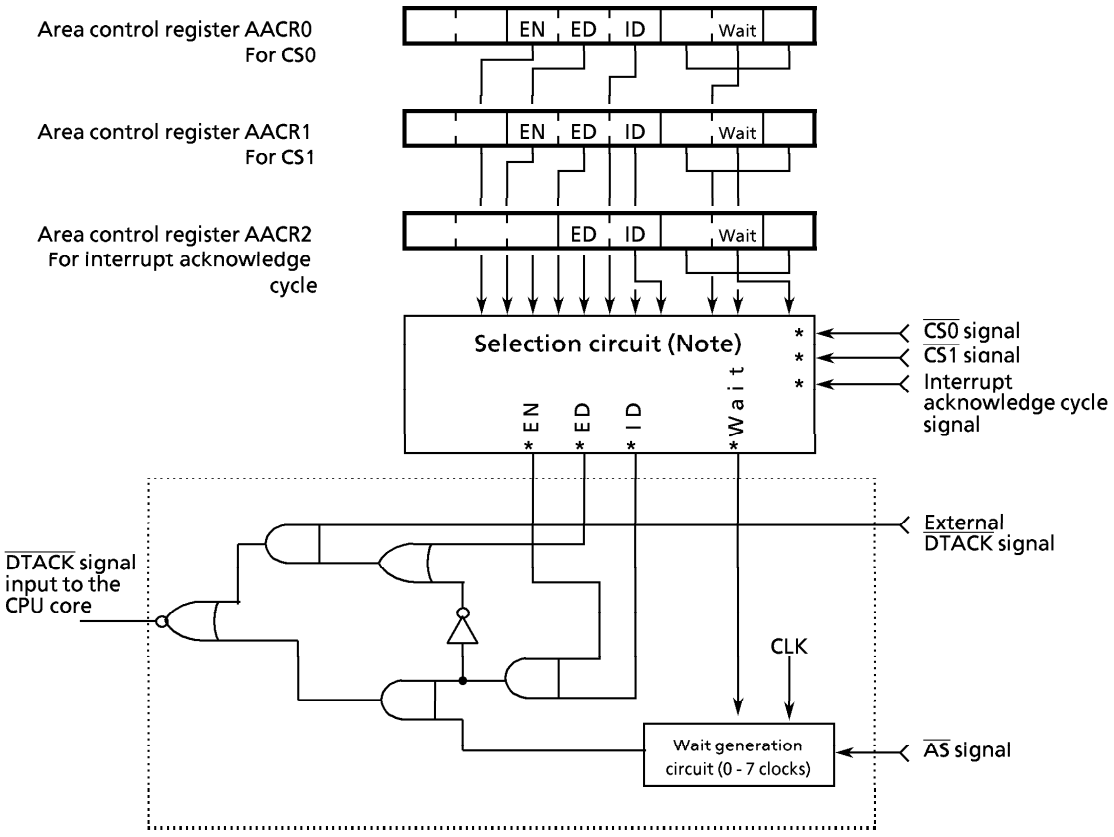


Figure 3.5 Memory Map After Reset Release

3.3 Automatic $\overline{\text{DTACK}}$ Generation

The address decoder can generate a $\overline{\text{DTACK}}$ signal for each memory area. The register area automatically generates $\overline{\text{DTACK}}$ to operate the bus with a zero wait. The register area can select whether $\overline{\text{DTACK}}$ signal generated in an external circuit or that generated in the address decoder is valid, or whether both are valid for the memory area. When both are valid, the first signal asserted is the $\overline{\text{DTACK}}$ signal output to the core processor. The address decoder can control the $\overline{\text{DTACK}}$ signal output to the core processor by inserting 0 to 7 wait cycles. To control the $\overline{\text{DTACK}}$ signal, the address decoder can also insert wait cycles in the IACK cycle (which loads external exception vectors) the same as for the memory area. Figure 3.6 illustrates the $\overline{\text{DTACK}}$ signal generation circuit.

The internal peripheral circuit registers of TMP68301A automatically generate the $\overline{\text{DTACK}}$ signal to enable access with no wait.



Note: During assertion of signals marked with an asterisk (*), the selection circuit inputs the set register values to the circuit in the dotted-line box.
During the interrupt acknowledge cycle, *EN is 1.
When internal peripheral circuit registers are accessed, *EN and *ID are set to 1 and *wait is set to 0.

Figure 3.6 $\overline{\text{DTACK}}$ Signal Generation Circuit

3.4 Bus Cycle Outline

For an address with no memory allocated in the system, the bus cycle stops without returning the $\overline{DTACK}$ signal. If the length of the bus cycle is abnormal, an address decoder function generates a $\overline{BERR}$ (bus error) signal. The length of the bus cycle until bus error generation can be selected among 32, 64, 128, or 256 clocks.

To use bus cycles above 256 clocks, an externally generated $\overline{BERR}$ signal can be validated by disabling generation of the $\overline{BERR}$ signal by the address decoder. Figure 3.7 illustrates the $\overline{BERR}$ signal generation circuit. Figure 3.8 shows the $\overline{BERR}$ signal generation timing.

This bus cycle monitoring applies to all memory areas regardless of the memory area of $\overline{CSx}$.

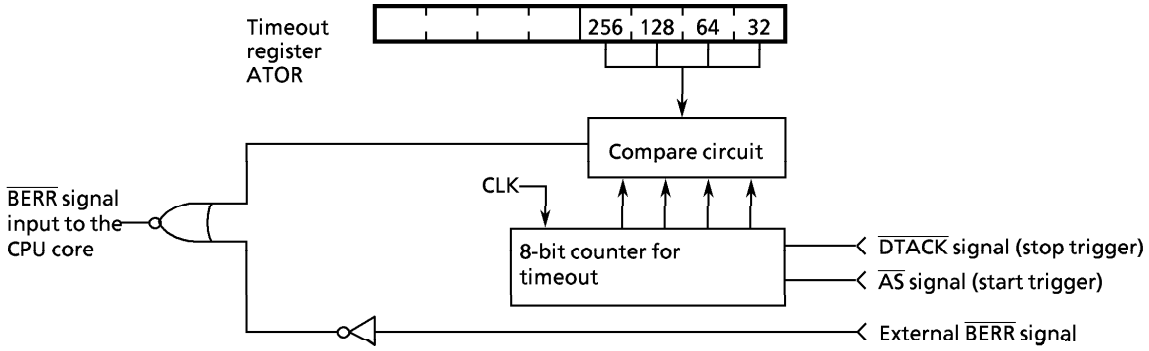


Figure 3.7 $\overline{BERR}$ Signal Generation Circuit

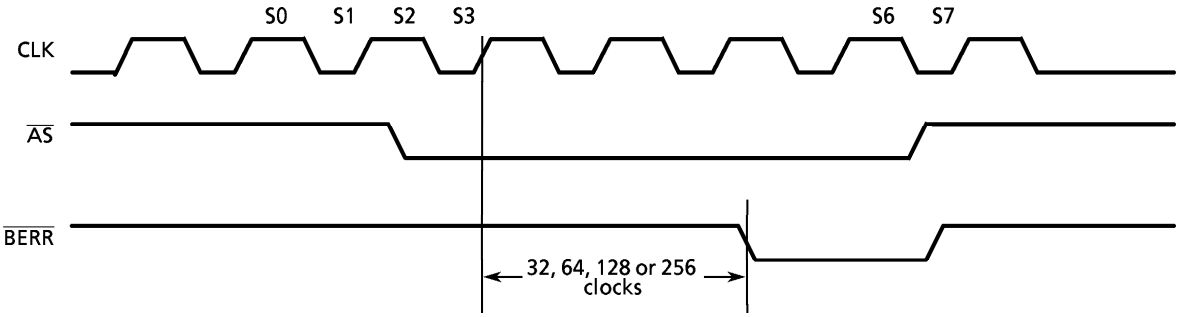


Figure 3.8 $\overline{BERR}$ Signal

3.5 Register Configuration

3.5.1 Memory Address register (AMAR0, 1)

This register specifies the start address of the memory area. Addresses are specified in upper eight bits, A23 to A16. The start address of the memory area can be set only within the boundary of the area determined by the address mask register. For example, if the size of the area is 4M bytes, only bits A23 and A22 of the memory address register are compared with the address, and four start addresses are available: $000000, $400000, $800000, and $C00000. (If the area is 256 or 512 bytes, the start address has a 64K-byte boundary because of the relationship with masked addresses.)

After reset, AMAR0 (for $\overline{CS0}$) is $00, and AMAR1 (for $\overline{CS1}$) is undefined.

Offset address	15	14	13	12	11	10	9	8	
$000	A23	A22	A21	A20	A19	A18	A17	A16	AMAR0 (for $\overline{CS0}$)
Reset	0	0	0	0	0	0	0	0	

	15	14	13	12	11	10	9	8	
$004	A23	A22	A21	A20	A19	A18	A17	A16	AMAR1 (for CS1)
	–	–	–	–	–	–	–	–	

3.5.2 Address Mask Register (AAMR0, 1)

This register sets the size of the memory area by masking the address output from the core processor. Addresses A21 to A8 can be masked.

The following table lists address mask register bits and their corresponding addresses to be masked.

Mask register bit	bit7	bit6	bit5	bit4	bit3	bit2	bit1	bit0
Address to be masked	A21	A20	A19	A18	A17	A16	A15 - A9	A8

Setting the bits to 1 masks the corresponding addresses; to 0 does not mask.

After reset, AAMR0 (for $\overline{CS0}$) is $FF, and AAMR1 (for $\overline{CS1}$) is undefined.

Offset address	7	6	5	4	3	2	1	0	
$001	M21	M20	M19	M18	M17	M16	*1	M8	AAMR0 (for $\overline{CS0}$)
Reset	1	1	1	1	1	1	1	1	

	7	6	5	4	3	2	1	0	
$005	M21	M20	M19	M18	M17	M16	*1	M8	AAMR1 (for CS1)
	–	–	–	–	–	–	–	–	

*1 : M15 - M9

3.5.3 Area Control Register (AACR0, 1, 2)

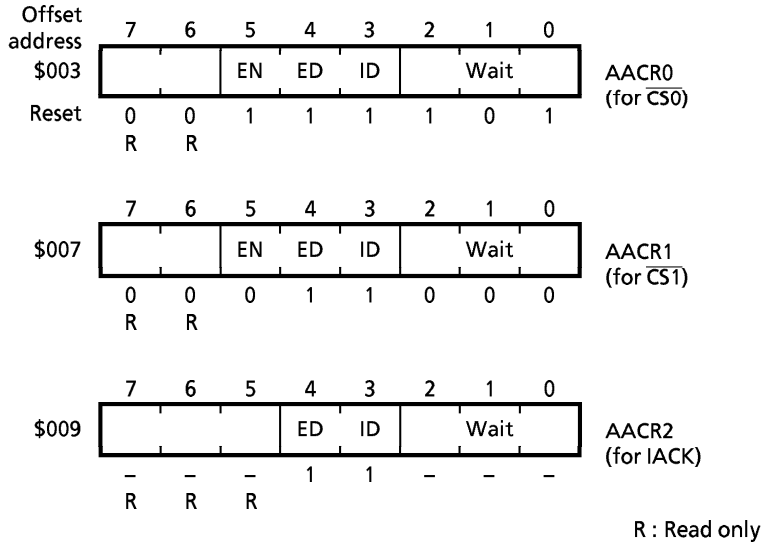
Registers AACR0 and AACR1 enable or disable memory areas in response to $\overline{CS0}$ and $\overline{CS1}$ signals, specify $\overline{DTACK}$ signal generation mode, and the number of waits to insert.

When an area is disabled, the $\overline{CS}$ signal is not asserted although the specified memory area is accessed, and the address decoder does not generate a $\overline{DTACK}$ signal. Register AACR2 specifies $\overline{DTACK}$ generation mode in the interrupt acknowledge cycle, and the number of waits to insert.

The setting in register AACR2 determines the $\overline{DTACK}$ generation mode in the IACK cycle when no vector number is automatically generated in an external interrupt. When a vector number is automatically generated due to an interrupt from an internal device or an external interrupt, a zero wait $\overline{DTACK}$ is generated regardless of the setting of register AACR2.

After a hardware reset, AACR0 is set to $3D (area enable, both external and internal $\overline{DTACK}$ enable, up to five wait clocks) and AACR1 is set to $18 (area disable, both external and internal $\overline{DTACK}$ enable, no wait clocks).

AACR2 is set to $18 (both external and internal $\overline{DTACK}$ enable, no wait clocks).



EN : Area control

0 : Area disable ($\overline{CSn}$ signal not output, no internal $\overline{DTACK}$ generated)

1 : Area enable ($\overline{CSn}$ signal output, internal $\overline{DTACK}$ generated)

ED, ID : $\overline{DTACK}$ selection

ED	ID	$\overline{DTACK}$ Used
0	0	Use prohibited (even when set, ED = 0, ID = 1)
0	1	Use internal $\overline{DTACK}$ (even when $\overline{DTACK}$ pin asserted, external $\overline{DTACK}$ ignored)
1	0	Use external $\overline{DTACK}$ (no $\overline{DTACK}$ generated by address decoder)
1	1	Use both internal and external $\overline{DTACK}$ (the first asserted is valid)

Wait : Number of waits to be inserted

When using internal $\overline{DTACK}$, set the number of waits to be inserted. Settings are from 0 to 7 clocks.

3.5.4 Timeout Register

This register specifies the time until $\overline{BERR}$ generation. The time can be selected from among 32, 64, 128, or 256 clocks. If more than one bit is set, only the bit corresponding to the longest time remains set and the other bits are cleared. If no bit is set, no $\overline{BERR}$ is generated by the address decoder. After a hardware reset, the timeout register is set to $08 (256 clocks).

Offset address	7	6	5	4	3	2	1	0	
$00B					256	128	64	32	ATOR
Reset	0 R	0 R	0 R	0 R	1	0	0	0	

R : Read only

Set value				Time until $\overline{BERR}$ generation
256	128	64	32	
0	0	0	0	No $\overline{BERR}$ generation
0	0	0	1	Generated after 32 clocks
0	0	1	X	Generated after 64 clocks
0	1	X	X	Generated after 128 clocks
1	X	X	X	Generated after 256 clocks

X : Any value

3.5.5 Relocation Register

This register specifies the start address of the internal register block. Specifiable addresses are A23 to A10. Accordingly, the register block can only specify in 1K-byte boundaries. After a hardware reset, the relocation register is set to $FFFC.

Offset address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
$00C	A23	A22	A21	A20	A19	A18	A17	A16	A15	A14	A13	A12	A11	A10			ARELR
Reset	1	1	1	1	1	1	1	1	1	1	1	1	1	1	0 R	0 R	

R : Read only

3.6 Bus Cycles Based on External Bus Master

When an external device other than the core processor becomes the bus master, the address decoder functions as if the core processor was the bus master. As a result, the external bus master can access the internal registers of the address decoder. The timing of the bus cycle generated by the external bus master must match the specifications of the bus cycle generated by the core processor. If the timing does not match these specifications, the internal register contents may be overwritten in the write cycle generated by the external bus master.

3.7 Cautions

1. While area control register 2 is for the interrupt acknowledge cycle, the settings here are valid only for reading external vector numbers by external interrupts.
2. When both the internal and external DTACK (ED=1, ID=1) are selected in the area control register, accessing an area under the conditions specified below prevents normal execution of the next cycle. This occurs with either CS0 or CS1. For example, if the conditions are satisfied in a CS1 cycle followed by a CS0 cycle, CS0 cycle becomes an abnormal cycle. In the reverse case, even if the CS0 or CS1 access is repeated and the conditions are satisfied, then the next cycle becomes an abnormal cycle.

		External wait set value							
		0	1	2	3	4	5	6	7
Internal wait set value	0	○	–	–	–	–	–	–	–
	1	×a	○	–	–	–	–	–	–
	2	×a	×	○	–	–	–	–	–
	3	○	×	×	○	–	–	–	–
	4	○	○	×	×	○	–	–	–
	5	○	○	○	×	×	○	–	–
	6	○	○	○	○	×	×	○	–
	7	○	○	○	○	○	×	×	○

○ : If the bus cycle ends at this condition, the next bus cycle receives the set number of waits and operates normally (no problem).

× : If the bus cycle ends at this condition, the next bus cycle generates the following:

- When the set number of waits for the next bus cycle is one or two and the cycle is the read cycle, this read cycle has no-wait access.
  If the read cycle is repeated, all subsequent cycles have no-wait access.
  (Because the Xa status shown in the table above continues.)
- When the set number of waits for the next bus cycle is other than one or two and the cycle is a read cycle, this read cycle has no-wait access.
  If the read cycle is repeated, only the first cycle has no-wait access. (Because when the actual number of waits is 0, the status is ○.)
- When the next bus cycle is a write cycle, the set number of waits is inserted as normal.
- When the next bus cycle is an internal peripheral circuit register read or write cycle, ends normally.

For example, if the wait setting for CS0 is 1 or 2, the wait setting for CS1 is 4, the actual cycle for CS1 ends at wait 3, and the immediate next cycle is CS0, then there is no wait. If, at this time, CS0 uses only the internal DTACK (ED = 0, ID = 1), there is still no wait. If an interrupt vector is written from an external source, the number of waits can be set in the interrupt acknowledge cycle. However, if prior to the acknowledge cycle, the bus cycle ends in an X state as shown in the table above, there is no wait.

If the above conditions are satisfied, the next cycle may not operate normally. Therefore, do not make settings which result in combinations marked with "x" in the table.

4. Interrupter Controller

4.1 Overview

The interrupt controller provides 10 interrupt channels. Seven of the channels are for internal peripheral circuits, while the other three are for external interrupts from interrupt request input pins INT0, 1, and 2. Interrupt request levels input to the core processor (the pattern of "0"s and "1"s input to the core processor IPL0, 1, and 2) can be set for each channel. Priorities can be set independently. Interrupt request input mode (input level, rising / falling edge) for external interrupts can also be set independently. In addition, the external interrupt vector number can be selected as either an internal vector number in the interrupt controller, or an externally input vector number.

If an external vector is input, the IACK output (IACK0, 1, or 2) corresponding to an external interrupt channel is asserted in accordance with the interrupt acknowledge cycle.

The auto-vectored interrupt function supported by the TMP68HC000 is not available because the TMP68301AK does not have 68000 interface signals.

Figure 4.1 shows a block diagram of the interrupt controller.

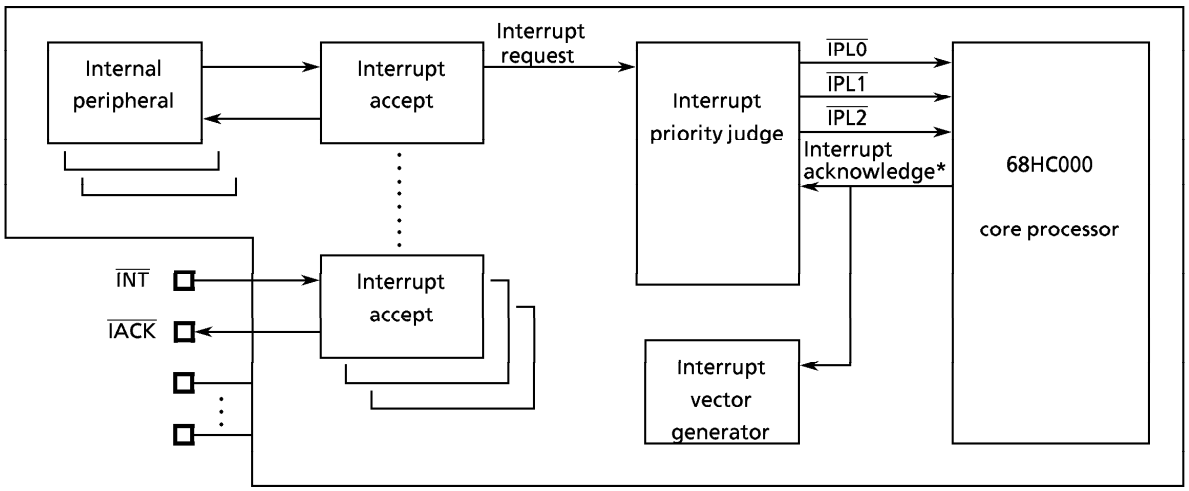


Figure 4.1 Interrupt Controller Block Diagram

4.2 Interrupt Request

When an interrupt request is present on an interrupt channel, the interrupt controller uses the internal $\overline{IPL0}$ to $\overline{IPL2}$ signals (not output to external pins in normal mode) to issue an interrupt request to the core processor at the previously set interrupt level. If requests are generated on more than one channel at the same time, the request with the highest priority level is issued.

The following input modes can be selected for external interrupt requests (interrupts using INT0, 1, or 2).

- Low-level interrupt
- High-level interrupt
- Rising-edge interrupt
- Falling-edge interrupt

Interrupt request inputs (INT0, 1, or 2) are detected on the falling edge of the system clock. Level-triggered interrupts must be asserted until the $\overline{IACK}$ output ($\overline{IACK0}$, $\overline{1}$, $\overline{2}$) for the channel corresponding to the interrupt request is asserted. Edge-triggered interrupts must be held at the same state for at least two system clocks after the edge to prevent malfunction due to noise.

Note : When switching from level mode to edge mode, the interrupt requests (pending bits) received in level mode are not cleared. To avoid this, clear any interrupt requests as follows:

1. Mask interrupt requests
2. Switch from level mode to edge
3. Clear the pending bits
4. Release the interrupt request mask

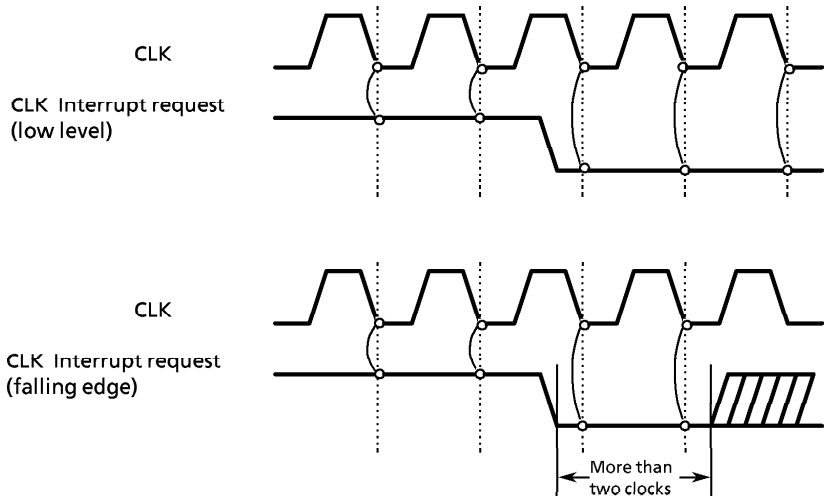


Figure 4.2 External Interrupt Request

4.3 Priority Between Channels

The interrupt controller can set the interrupt level of IPL0, 1, and 2 for interrupting the core processor using the level bit of the interrupt control register for each channel. This sets the relative priority of each channel. The following priority applies if more than one channel is set to the same interrupt level:

Priority	Channel	
High	External interrupt request	Channel 0
	Timer	Channel 0
	Serial interface	Channel 0
	Parallel interface	
	External interrupt request	Channel 1
	Timer	Channel 1
	Serial interface	Channel 1
	Serial interface	Channel 2
	Timer	Channel 2
Low	External interrupt request	Channel 2

Table 4.1 Priority of Channels Set to Same Interrupt Request Level

4.4 Interrupt Acknowledge Cycle (IACK Cycle)

In 68000 interrupt processing, if an interrupt is accepted, the interrupt acknowledge cycle (IACK cycle) is performed. The interrupt request level is released on addresses A1 to A3 is output and the corresponding interrupt vector number is read from data bus D0 to D7.

The 68301A interrupt controller has a function to automatically generate the vector number to be read by the core processor during the $\overline{\text{IACK}}$ cycle. This function allows the interrupt controller to automatically perform the above interrupt processing. Also, when an external interrupt is accepted, the interrupt controller can assert the IACKn signal corresponding to the interrupt and obtain the vector externally.

If more than one channel issues requests at the same level, an interrupt acknowledge signal is asserted for the channel with the highest priority in Table 4.1. The interrupt acknowledge signal asserted here only applies internally to the 68301A and is not output externally. However, depending on the register setting, external interrupts can be output externally as $\overline{\text{IACK}}$ signals. $\overline{\text{IACK}}$ signals are asserted using the same timing as that for $\overline{\text{AS}}$ signals. Area control register 2 in the address decoder can be set to insert a WAIT into IACK cycles. For details, see the address decoder section.

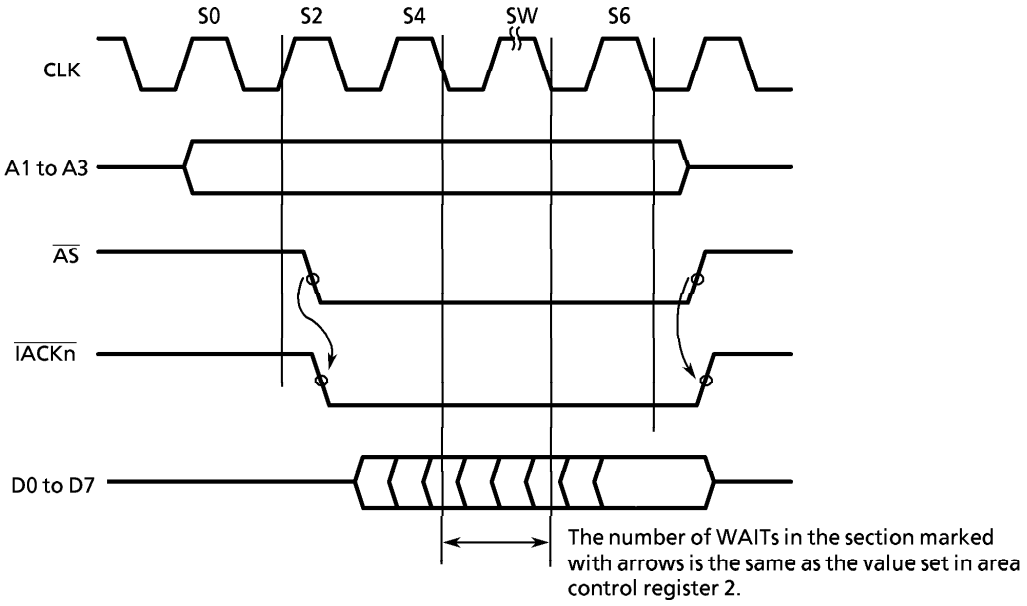


Figure 4.3 IACK Signals for External Vector Fetch by External Interrupt

4.5 Automatic Generation of Vector Numbers

The interrupt controller automatically generates vector numbers during IACK cycles and these are read by the core processor. The five lower bits of the vector are determined depending on the interrupt channel or request. The three upper bits are set using the interrupt vector number register (IVNR). See Table 4.2 for a list of vector numbers.

In the case of external interrupt requests, automatic generation or external vector input can be selected by setting the vector generation mode bit (V bit) of the interrupt control register (ICR0, 1, or 2).

Serial interface interrupt requests generate vector numbers not only corresponding to the channel generating the interrupt, but also in accordance with the cause of the interrupt.

Channel	Cause	Vector Number
External interrupt Channel 0		XXX00000
External interrupt Channel 1		XXX00001
External interrupt Channel 2		XXX00010
Timer 0 Channel 0		XXX00100
Timer 1 Channel 1		XXX00101
Timer 2 Channel 2		XXX00110
Serial interface Channel 0	Receive error, break detection	XXX01000
	Receive complete	XXX01001
	Transmit ready	XXX01010
	Interrupt source cleared while interrupt pending (Note 1)	XXX01011
Serial interface Channel 1	Receive error, break detection	XXX01100
	Receive complete	XXX01101
	Transmit ready	XXX01110
	Interrupt source cleared while interrupt pending (Note 1)	XXX01111
Serial interface Channel 2	Receive error, break detection	XXX10000
	Receive complete	XXX10001
	Transmit ready	XXX10010
	Interrupt source cleared while interrupt pending (Note1)	XXX10011
Parallel interface	Receive complete	XXX10101
	Transfer ready	XXX10100
	Status change (fault input at transmit) of external device	XXX10110
	Interrupt cause cleared while interrupt pending (Note 1)	XXX10111
Other	Default vector (Note 2)	XXX11111

XXX : Set by the three upper bits of the IVNR.

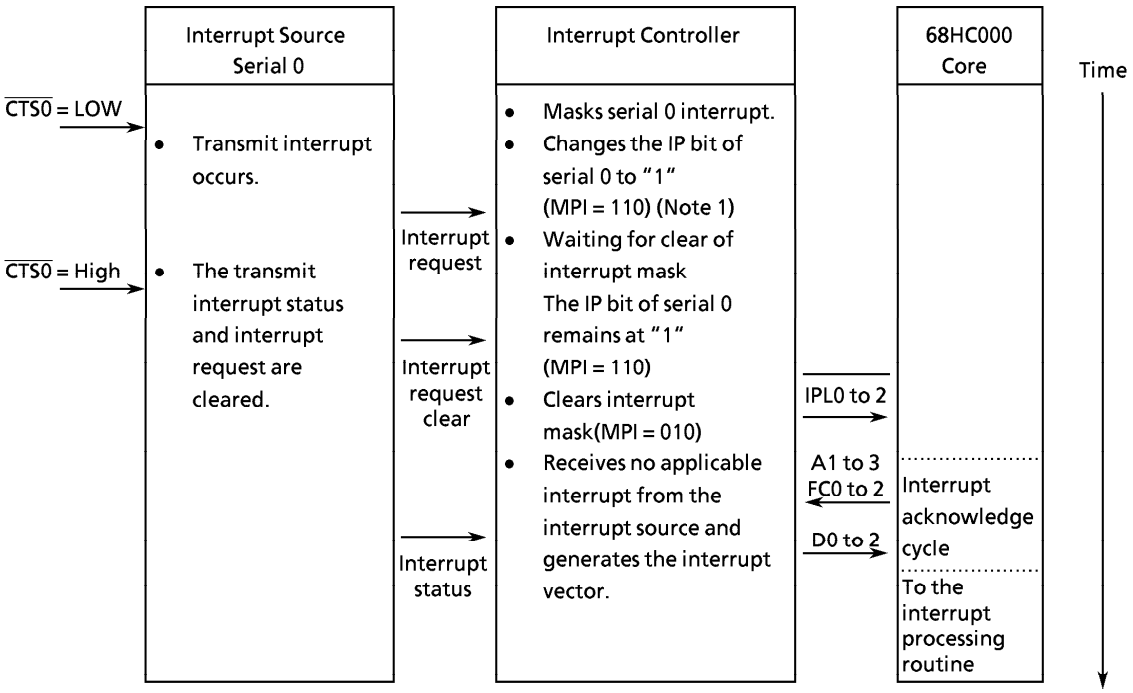
Table 4.2 Vector Number List

Note1: This vector number is generated when the cause of the interrupt becomes unknown if the interrupt source is cleared before the interrupt acknowledge is returned for a pending interrupt.

Note2: If the CPU accepts an interrupt, the IACK cycle starts after the following instruction is completed. However, if that instruction masks the interrupt, the cause becomes masked and the vector is fixed at "11111" because the vector cannot now be generated by the IACK cycle.

4.5.1 Interrupt Source Cleared While Interrupt Pending

This interrupt is generated under the following conditions. The following description is based on the generation of a serial 0 interrupt.



When an interrupt is generated by the interrupt source, an interrupt request is passed to the interrupt controller. As a result, the interrupt controller sets the relevant IP bit to "1". If the relevant interrupt is masked, the interrupt controller waits for the interrupt request to be issued to the 68HC000 core. (Note 1: MPI represents the bit status of the mask register, pending register, and in-service register for the generated interrupt). During this period, the interrupt condition may be cleared at the interrupt source. (The above example shows when the $\overline{CTS0}$ input changes to high in the transmit interrupt state). Even if the interrupt cause is cleared at the interrupt source, because interrupt control is still pending, the interrupt controller sends an interrupt request to the 68HC000 core when the interrupt mask is cleared. This starts the interrupt sequence and generates the interrupt vector. Because the vector reflects the state of the interrupt source, "no applicable interrupt" indicates that the interrupt cause was cleared at the interrupt source while the interrupt was pending, as mentioned previously. For serial interfaces, these kinds of interrupts occur under the following conditions:

(a) The interrupt is masked by the serial mode register (SMRn) while it is pending.

(b) The $\overline{CTS0}$ input changes to high while a transmit ready interrupt is pending. (TxRDY is set to "0" when $\overline{CTS0}$ changes to high, clearing the transmit interrupt cause.)

(c) In the serial command register (SCMRn), TxEN is set to "0" while a transmit ready interrupt is pending, or RxEN is set to "0" while a receive complete interrupt is pending.

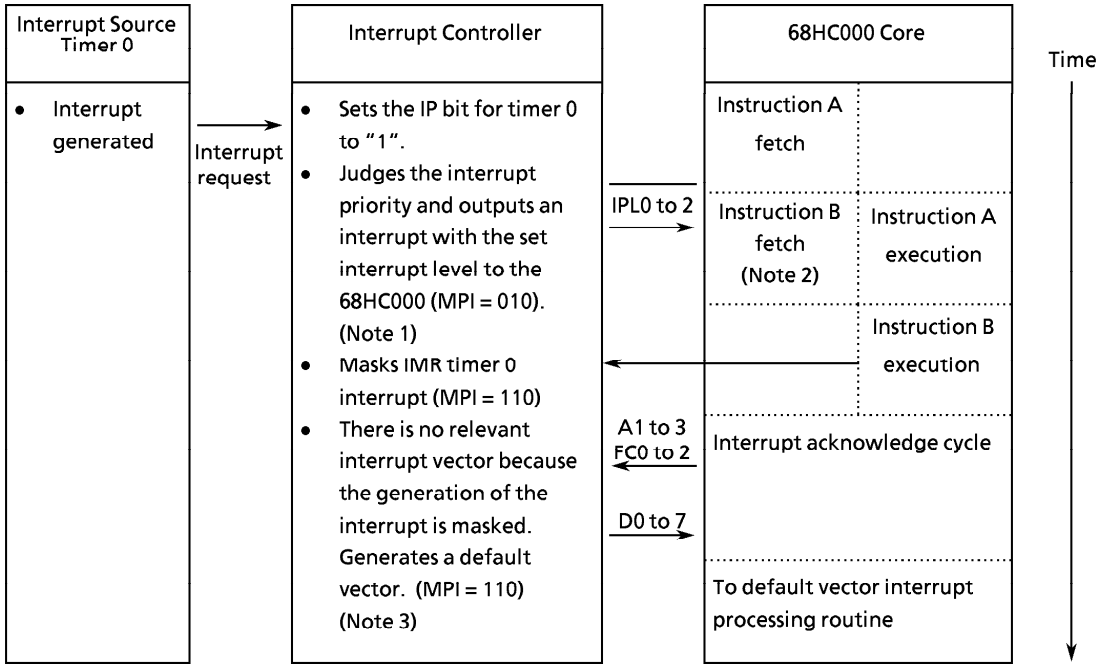
(d) The receive buffer is read by the error interrupt processing routine.

(If both ERINTM and RxINTM of the serial mode register [SMRn] are "0" and both interrupt masks are cleared, both error interrupt and receive interrupt are generated when an error occurs. Since the priority of the error interrupt is higher, the error interrupt processing routine is executed first. If the serial data register [SDRn] is read by this processing routine, the pending receive interrupt is cleared).

These interrupts occur due to software processing problems as described above. Therefore, check your software and make necessary modification so that these interrupts will not occur.

4.5.2 Default Vector Interrupts

Default vector interrupts occur under the following conditions. The following description is based on the generation of a timer 0 interrupt.



When an interrupt is generated, an interrupt request is passed to the interrupt controller. As a result, the interrupt controller sets the relevant IP bit to "1", checks that the interrupt is the highest priority interrupt, and outputs an interrupt with the set interrupt level to the 68HC000 core ($\overline{IPL0}$ to $\overline{2}$ output). The internal status of the interrupt controller at this time is as follows: mask bit = "1" (M=1), pending bit = "1" (P=1), and in-service bit = "0" (I=0). (Note 1: MPI represents the bit status of the mask register, pending register, and in-service register for the generated interrupt).

When the 68HC000 core accepts the $\overline{IPL0}$ to $\overline{2}$ interrupts, it attempts to jump to the interrupt sequence operations. However the interrupt sequence is delayed until instruction B fetched by the 68HC000 core is executed. If instruction B masks the generated interrupt (Note 2), the instruction is executed and the interrupt request by the interrupt controller is cancelled. Subsequently, even if the 68HC000 core starts the interrupt acknowledge cycle, the interrupt controller does not have a corresponding interrupt and so generates a default vector indicating that there is no relevant interrupt and ends the interrupt acknowledge cycle.

Even if a default vector interrupt is generated, the interrupt generated remains in the interrupt controller in the pending state (Note 3). Accordingly, after clearing the interrupt mask, a normal interrupt sequence can be performed.

If you simply wish to return to the main processing and disable vector generation when this default interrupt occurs, insert an instruction before the interrupt mask instruction (instruction B), to set the 68HC000 core interrupt level to the highest level. This prevents the 68HC000 core from receiving interrupts, apart from interrupt level 7, thus preventing initiation of the interrupt sequence midway through masking the interrupt. However, if the level of the interrupt generated is 7, the 68HC000 core cannot disable the interrupt request and the default vector is generated. In this case, perform default vector processing.

4.6 Interrupt Status

The interrupt status for each channel is represented by the mask register (IMR), pending register (IPR), and in-service register (IISR) bits corresponding to the channel. The meaning of these bits are described below. The set (setting the bit to "1") and reset (setting the bit to "0") methods vary according to a bit.

Mask Bit (M)	Control Bit for masking interrupt requests
1	Masks the interrupt request.
0	Unmasks the interrupt request.
set	Set by hardware reset or by writing "1" by software.
reset	Set to "0" by software.

Pending bit (P)	Indicates an interrupt request occurred and is pending (waiting for interrupt processing).
1	An interrupt request occurred and is pending.
0	No interrupt request occurred
set	An interrupt request occurred (this bit cannot be set to "1" by software).
reset	Reset by hardware reset or by writing "0" by software when the interrupt request is accepted by the core processor. (See Notes)

In-service bit (I)	Indicates that an interrupt request has been accepted by the core processor.
1	Indicates that an interrupt request has been accepted.
0	Indicates that no interrupt request has been accepted.
set	An interrupt request is accepted by the core processor (this bit cannot be set to "1" by software).
reset	Hardware reset Set to "0" by software

Note1 : The function to clear pending bits by software is used when initializing the whole system or when pending bits are set by unnecessary interrupts. However, if pending bits set according to interrupts generated by internal peripheral circuits are cleared to "0", interrupts will no longer occur. This is because the pending bit is only set when the interrupt request from the interrupt source changes from "0" to "1" (when an interrupt is generated). To re-enable interrupts after the pending bit has been cleared, the interrupt source must also be operated as follows.

Timer

First clear the interrupt request bit (INT bit) of the timer control register (TCRn) to "0", then set to "1".

Serial Interface

First set the interrupt mask bit (INTM bit) of the serial control register (TCRn) to "1", then clear to "0". If, at that time, an interrupt request is present due to an interrupt source in a channel, the corresponding IP bit is set immediately after the INTM bit is cleared to "0". To avoid this, disable interrupt requests for the channel (for example, by setting an interrupt mask for each channel, by reading the receive data, or by performing an error reset) before performing the above procedure.

Parallel Interface

With a transmit ready interrupt, set the transmit interrupt mask bit (bit IM1) of the parallel command register (PCR) to 1, then clear the bit to 0.

With a receive complete interrupt, set the receive complete interrupt mask bit (bit IM3) of the PCR to 1, then clear the bit to 0. At a fault interrupt, set the fault interrupt mask bit (bit IM2) of the PCR to 1, then clear the bit to 0.

External Interrupt

For edge mode interrupts, the IP bit is set the next time the interrupt edge is input. For level mode interrupts, the IP bit is set the next time the interrupt input is asserted.

Note2: For level mode interrupts, clearing the IP bit by software requires first negating the interrupt input. The IP bit cannot be cleared by software while the interrupt input is still asserted.
When clearing the pending bit by software, write "1" to all bits except for the bit to be cleared. Writing "1" to the pending bit will not affect operation but enables interrupts to be generated even though the pending bit is mistakenly cleared.

The interrupt states for the mask bit (M), pending bit (P), and in-service bit (I) values are shown in Table 4.4.

M	P	I	
0	0	0	No interrupt request
0	0	1	During interrupt processing routine
0	1	0	Interrupt request generated
0	1	1	Another interrupt request is generated during an interrupt processing routine.
1	0	0	No interrupt request
1	0	1	Interrupt is masked during an interrupt processing routine
1	1	0	Interrupt request generated while interrupt is being masked
1	1	1	Interrupt request generated after masking interrupt during an interrupt processing routine.

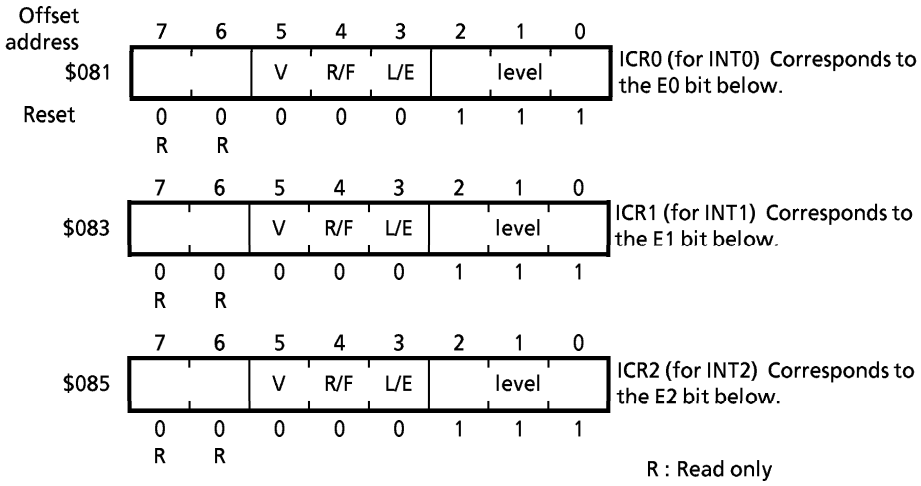
Table 4.3 Interrupt Status

4.7 Register Configuration

4.7.1 Interrupt Controller Registers 0, 1, and 2 (ICR0, 1, and 2)

These registers control the external interrupt inputs (INT0, 1, and 2). The registers set the interrupt level and select external vector input or automatic generation of the vector number set for input mode.

After a hardware reset, ICR0, 1, and 2 are all initialized to $07 (vector number from external source, falling edge mode, interrupt request level 7). These registers can only be written to when the interrupt is masked by the interrupt mask register (IMR).



V : Vector number automatic generation control

0 : Reads vector number from an external source instead of automatic generation of vector number

1 : Vector number automatic generation

R/F, L/E : Request input mode for external interrupts

R/F	L/E	Interrupt Request Input Mode
0	0	Falling edge
1	0	Rising edge
0	1	Low level
1	1	High level

Level : Interrupt request level

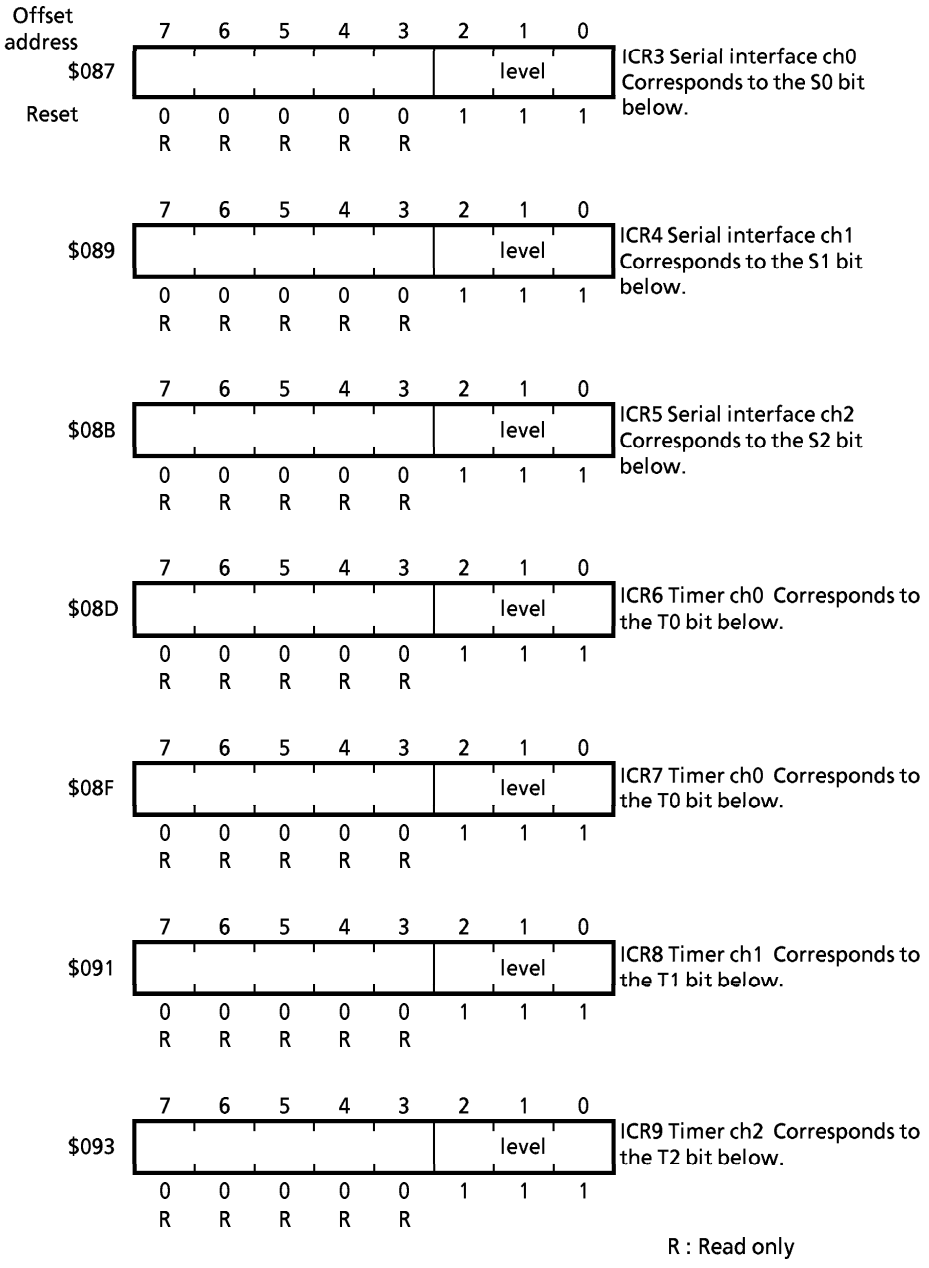
0 to 7 : Indicate the interrupt request level corresponding to $\overline{IPL0}$, $\overline{1}$, and $\overline{2}$ to be input to the core processor. For example, $\overline{IPL2}=\overline{1}$, $\overline{IPL1}=\overline{0}$, and $\overline{IPL0}=\overline{0}$ indicate request level 3. The following table shows the correspondence between IPLx and interrupt levels.

Interrupt level	$\overline{IPL2}$	$\overline{IPL1}$	$\overline{IPL0}$
0	1	1	1
1	1	1	0
2	1	0	1
3	1	0	0
4	0	1	1
5	0	1	0
6	0	0	1
7	0	0	0

4.7.2 Interrupt Control Registers 3 to 5, 7 to 9 (ICR3 to 5, 7 to 9)

These are the interrupt control registers for interrupts from internal peripheral circuits.

After a hardware reset, ICR3 to 5 and 7 to 9 are all initialized to $07 (interrupt request level 7). These registers can be written to only when the interrupt is masked by the interrupt mask register (IMR).



Level : Interrupt request level

0 to 7 : Indicate the interrupt request level corresponding to IPL0, 1, and 2 to be input to the core processor.

4.7.3 Interrupt Mask Register (IMR)

This register sets the interrupt mask for a channel. A "1" masks the interrupt (ignore interrupt). A "0" unmasks the interrupt (allow interrupt). If masking an interrupt during operation, set the channels corresponding to the bits to be modified to a state whereby, even if an interrupt is generated, it will not be accepted by the core processor. (For example, set the interrupt level in the core processor status register to 7). This is necessary because, if an interrupt occurs during masking, the interrupt to be masked may be generated due to the timing mismatch.

After a hardware reset, this register is initialized to $07F7 (all interrupt channels masked).

Offset address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
$094						T2	T1	T0	P	S2	S1	S0		E2	E1	E0	IMR
Reset	0	0	0	0	0	1	1	1	1	1	1	1	0	1	1	1	
	R	R	R	R	R								R				

T2 : Timer channel 2
T1 : Timer channel 1
T0 : Timer channel 0
P : Parallel interface
S2 : Serial interface channel 2
S1 : Serial interface channel 1
S0 : Serial interface channel 0
E2 : External interrupt channel 2
E1 : External interrupt channel 1
E0 : External interrupt channel 0
P : Parallel interface (only the case that IO12 is used for $\overline{INT6}$ (External interrupt))

R : Read only

Note : Bit 7 is normally set to "1".

4.7.4 Interrupt Pending Register (IPR)

This register indicates whether there is an interrupt request and that the interrupt request is not yet accepted by the core processor. A "1" indicates that there is an interrupt request that has not yet been accepted by the core processor. A "0" indicates that there is no interrupt request.

Each bit is automatically cleared when the request is accepted by the core processor. Clearing a bit using software cancels the interrupt request. However, to enable subsequent interrupts, the interrupt source must be cleared.

After a hardware reset, this register is initialized to $0000 (no interrupt requests).

Offset address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
$096						T2	T1	T0	P	S2	S1	S0		E2	E1	E0	IPR
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	R	R	R	R	R								R				

R : Read only

4.7.5 Interrupt In-Service Register (IISR)

This register indicates whether or not an interrupt request has been accepted by the core processor. A "1" indicates that a request has been accepted. A "0" indicates that a request has not been accepted.

While operating with the register set to "1" does not affect operation, clear each bit during the interrupt processing routine. (These bits are not cleared automatically).

After a hardware reset, the register is initialized to $0000 (no interrupt requests accepted).

Offset address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
$098						T2	T1	T0	P	S2	S1	S0		E2	E1	E0	IISR
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	R	R	R	R	R								R				

R : Read only

4.7.6 Interrupt Vector Number Register (IVNR)

This register specifies the three upper bits of a vector number. The five lower bits of a vector number are determined by the interrupt channel or the interrupt source.

After a hardware reset, the register is initialized to $00.

Note : After a reset is released, the 68HC000 core interrupt vectors overlap the internal peripheral circuit interrupt vectors. Therefore, set the value of IVNR to $40 or more by software.

Offset address	7	6	5	4	3	2	1	0	
$09B	Vector								IVNR
Reset	0	0	0	0	0	0	0	0	
				R	R	R	R	R	

R : Read only

Vector : Three upper bits of the vector number

4.8 Interrupt Expansion Function

In the TMP68301A, interrupt channels assigned to unused peripheral circuits can be used as external interrupt inputs by setting the expansion interrupt register. Thus, a maximum of seven channels can be used for external interrupt inputs in addition to the three standard external interrupt channels.

The interrupt input pins are assigned as follows:

Peripheral circuit interrupt channel	Interrupt input pin	Interrupt input name
Serial interface ch0	RxD0	$\overline{INT3}$
Serial interface ch1	RxD1	$\overline{INT4}$
Serial interface ch2	RxD2	$\overline{INT5}$
Parallel interface	IO12 / $\overline{FAULT}$	$\overline{INT6}$ (Note)
Timer ch0	TIN	$\overline{INT7}$
Timer ch1	IO14 / $\overline{DSR0}$	$\overline{INT8}$ (Note)
Timer ch2	IO15 / $\overline{DTR0}$	$\overline{INT9}$ (Note)

Note : When used as interrupt inputs, the pins must be first set to input in the parallel direction register.

Only falling-edge input mode is available for expanded external interrupts. However, like the standard external interrupt inputs, the state must be held for at least two clocks after the falling edge.

Multiple falling edges may occur before the processor accepts an interrupt request. The processor treats these edges as if they were the same interrupt request.

There is no IACK output signal corresponding to the expanded external interrupts. Therefore, the vector number is generated automatically by the interrupt controller because the vector number cannot be input from an external source during the interrupt acknowledge cycle. Vector numbers generated at this time are as follows.

Interrupt channel	Interrupt input name	Vector number
Serial interface ch0	$\overline{INT3}$	XXX010**
Serial interface ch1	$\overline{INT4}$	XXX011**
Serial interface ch2	$\overline{INT5}$	XXX100**
Parallel interface	$\overline{INT6}$	XXX101**
Timer ch0	$\overline{INT7}$	XXX00100
Timer ch1	$\overline{INT8}$	XXX00101
Timer ch2	$\overline{INT9}$	XXX00110

XXX : The three upper bits specified by the vector number register

** : The two lower bits of the interrupt vector number assigned to the serial interface depend on the status of the standard interrupt channels. Accordingly, when using these interrupts, set the same destination address in all four. The same applies to $\overline{INT6}$.

4.8.1 Expansion Interrupt Register (IEIR)

This register controls the switching of interrupt channels between internal peripheral circuits and external interrupt inputs. A "1" indicates external interrupt input. A "0" indicates interrupts from the internal peripheral circuits (external interrupt input disabled).

After a hardware reset, this register is initialized to $00 (all channels set to internal peripheral circuit interrupts).

Offset address	7	6	5	4	3	2	1	0	
$09D		T2	T1	T0	P	S2	S1	S0	IEIR
Reset	0	0	0	0	0	0	0	0	

T2 : Timer ch2
T1 : Timer ch1
T0 : Timer ch0
P : Parallel interface
S2 : Serial interface ch2
S1 : Serial interface ch1
S0 : Serial interface ch0

Note1: Even if external interrupt input is enabled, the internal peripheral circuits can perform their original functions. However, the pins assigned as interrupt inputs (for example, the serial interface RxD) cannot be used for their original functions.

The peripheral circuits can function but cannot generate interrupts if the interrupt channels are used for interrupt inputs.

5. Serial Interface

5.1 Outline

The serial interface consists of three independent channels, which support full-duplex universal asynchronous receiver / transmitter (UART). Both the receive and transmit modules use double buffers and the same data format and baud rate. However, the receive and transmit modules are functionally independent. The data format used is a standard mark/space format. The data format consists of one start bit, between 5 and 8 data bits, and one or two stop bits. The serial interface also supports parity bit processing. The serial interface includes only one prescaler, and one baud rate generator for each channel to generate the baud rate clocks. Thus, independent baud rate clocks are available for each channel. The system clock (CLK) or external clock (input from the BCLK pin) can be selected as the divider clock. Error detect (parity error, overrun error, framing error) is supported. Break detect and break transmit is also supported. If an interrupt cause (receive complete, transmit ready, error occurred, break detected) is generated in any channel, an interrupt request is forwarded to the core processor via the interrupt controller. A separate vector number can be generated in the interrupt acknowledge cycle specifically for the channel and interrupt cause. Channel 0 supports modem control signals.

Figure 5.1 is the serial interface block diagram. Figure 5.2 is a detailed channel diagram.

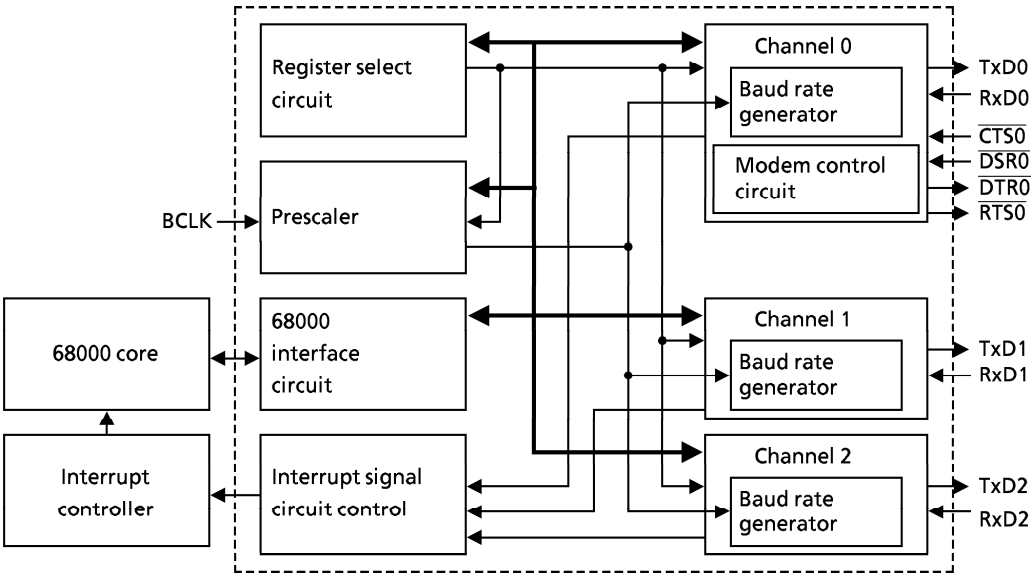


Figure 5.1 Serial Interface Block Diagram

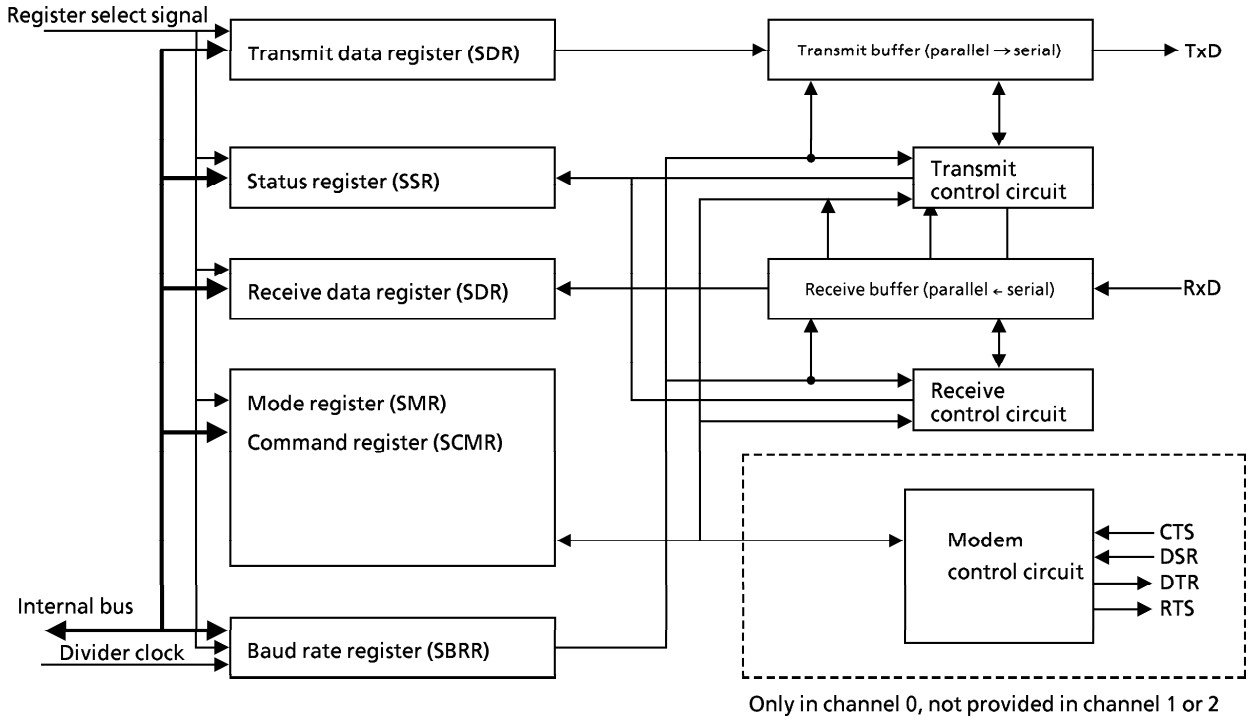


Figure 5.2 Channel Detail Diagram

Note: In the following description, the data register (SDR) is called the transmit data register at transmission, and the receive data register at reception. Although the transmit data register and the receive data register share the same register address, they are independent of each other. At read, the receive data register is accessed. At write, the transmit data register is accessed.

5.2 Communications

5.2.1 Data Format

Figure 5.3 shows the data frame formats input through the serial interface from the receive data input pins (RxD0, RxD1, and RxD2), and output to the transmit data output pins (TxD0, TxD1, and TxD2).

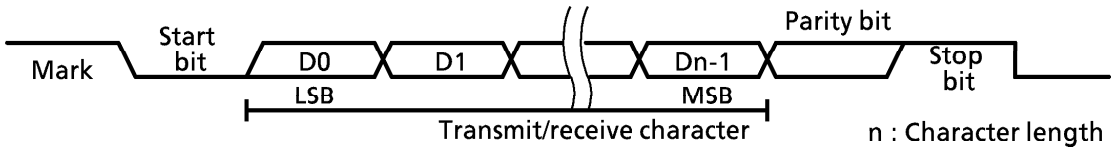


Figure 5.3 Data Frame

① When no data are transmitted or received, the data lines are at level 1 (mark state).
② The transmit (when sending) of the start bit (level 0) or detect (when receiving) of the start bit indicates the start of the data frame.
③ The transmit/receive character immediately follow the start bit, beginning with the least significant bit. The serial mode register (SMR) setting determines the character length n (from five to eight bits).
④ Setting the parity bit in the serial mode register (SMR) sends the parity bit at transmit, and checks the parity after receiving the parity bit at receive.
⑤ The stop bit (level 1) indicates the end of the data frame. At transmit, one or two stop bits are sent in accordance with the serial mode register setting. At receive, regardless of the serial mode register setting, only one bit is identified as the stop bit. The stop bit is the bit following the most significant data bit (last data bit received; when parity bit receive is not set) or the bit following the parity bit (when parity bit receive is set). If the stop bit is at 0 level, a framing error is generated.
⑥ A break consists of two consecutive frames where all data bits, the parity bit, and the stop bits, are at 0 level.

5.2.2 Baud Rate Clock Generation

The serial interface uses an 8-bit prescaler and an 8-bit baud rate generator to divide the system clock (CLK) or external input clock (BCLK) and thereby generate a baud rate clock. Figure 5.4 is the baud rate clock generation circuit block diagram.

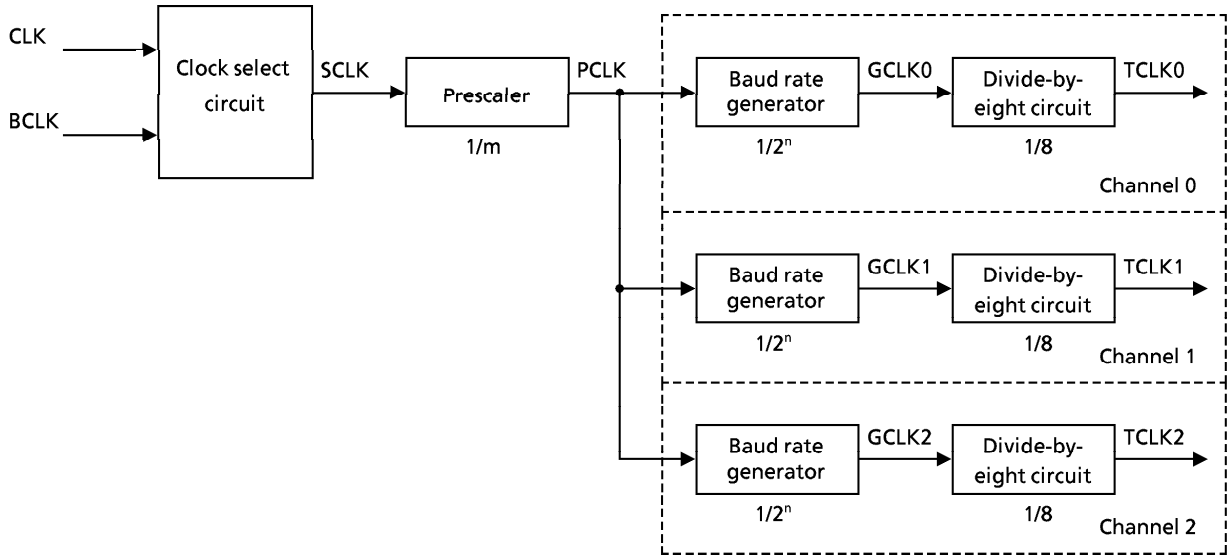


Figure 5.4 Baud Rate Clock Generation Circuit Block Diagram

The input clock select circuit selects either the system clock (CLK) or the external input clock (BCLK). The selected clock (SCLK) is divided by the prescaler by from 1/1 to 1/256 (PCLK), and the PCLK is divided by the baud rate generator by $1/2^n$ (n = 0 - 7) times (GCLK). The GCLK is used as the fundamental clock for serial interface transmit/receive. The GCLK is further divided by eight times (TCLK) for transmitting data frames. One prescaler is built into the serial interface. One baud rate generator is built into each channel.

5.2.3 Data Transmit

When a transmit character is written into the transmit data register (SDR), the serial interface resets the TxE (transmit data register buffer empty) bit of the status register (SSR) to 0 and checks whether transmit with the TxEN (transmit ready) bit of the command register (SCMR) is in progress. When $\overline{\text{CTS0}}$ is used in channel 0, the serial interface also checks pin IO13/$\overline{\text{CTS0}}$. When $\overline{\text{CTS0}}$ is not used, if TxEN = 1 and transmit is not in progress, transmit begins with the transmit character loaded from the transmit data register to the transmit buffer. When $\overline{\text{CTS0}}$ is used, if TxEN = 1, IO13/$\overline{\text{CTS0}}$ = 0, and transmit is not in progress, transmit begins in the same way. The transmit is synchronous with TCLK, which is generated by the baud rate generator. Figure 5.5 shows TLCK and the data frame detection timing.

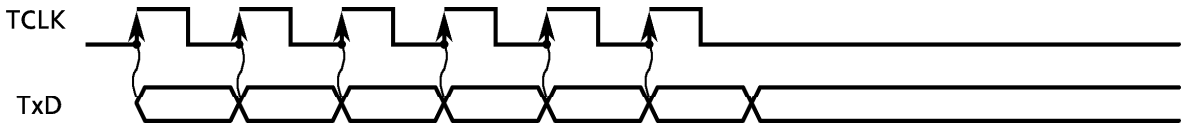


Figure 5.5 Data Frame Transmit Timing

When the serial interface starts transmit, the interface transmits the start bit and sets the TxRDY (transmit ready) bit of the status register (SSR) to 1. As the transmit character is loaded from the transmit data register to the transmit buffer at that point, the next transmit character can be loaded in the transmit data register in advance. When the next transmit character is written in the transmit data register, it is retained in that register until transmission of the current transmit character is complete. After transmitting the start bit, the serial interface transmits the number of data bits specified by the CL1 - CL0 (character length) bits of the mode register (SMR), transmits the parity bit when the mode register PE (parity enable) bit is set to 1, and transmits from the transmit output pin (TxD) the number of stop bits specified by the mode register ST (stop bit length) bit.

If the transmit data register contains a next transmit character, the serial interface transmits the start bit and the next character after transmitting the stop bit of the current data frame. Once the serial interface has begun transmit, it continues transmit of the current data character even if the status of the IO13/CTS0 pin or the TxEN bit change and ends transmit with the stop bit.

If the SBRK (break transmit) bit of the mode register is set to 1, the serial interface sends a break until SBRK is reset to 0 regardless of TxEN, $\overline{\text{CTS0}}$, or whether there is a transmit currently in progress.

5.2.4 Data Receive

The receive data input pin (RxD) is normally set to 1 (mark status). The serial interface samples the input at the GCLK generated by the baud rate generator. When it detects 0 level, it regards the 0 level as the beginning of the start bit. If the serial interface detects the 0 level three more consecutive times, it determines that the first 0 level is the beginning of a valid start bit. Then the serial interface determines the center point of subsequent receive characters, and samples the receive characters. Figure 5.6 shows how the start bit and the center point of receive characters are determined.

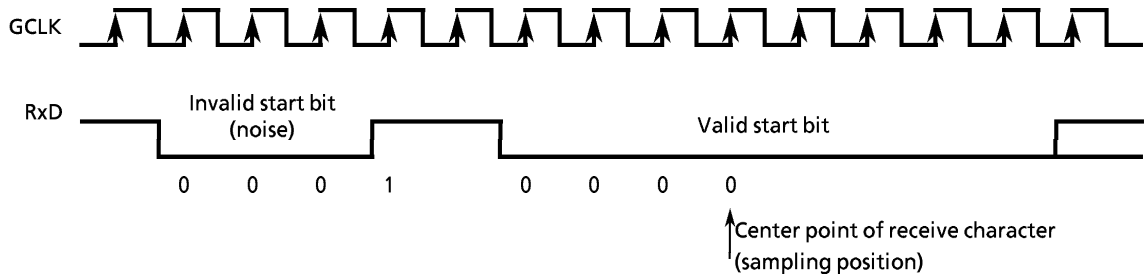


Figure 5.6 How to Determine Start Bits and Center Point of Receive Characters

If the mode register PEN (parity enable) bit is set to 1 and parity bit receive is specified, the serial interface samples the parity bit and compares the parity with that generated by the receive character. If the compared parities do not match, the serial interface sets the PE (parity error) bit of the status register to 1.

For stop bits, the serial interface samples the first stop bit irrespective of the stop bit length specified by the mode register ST (stop bit length) bit. If the result is not 1, it sets the FE bit (framing error) bit in the status register to 1.

The receive data bits are sampled sequentially and saved in the receive buffer. When the programmed number of data bits specified by CL1 - CL0 (character length) of the mode register have been received, the data bits are transferred from the receive buffer to the receive data register (SDR). When the character length is 5, 6 or 7, the upper bits in the receive data register (SDR) are unused. However, these bits are reset to 0 at transfer to the receive data register.

The RxRDY (receive ready) bit of the status register is set to 1 when a character is transferred from the receive buffer to the receive data register (SDR). At this time, the receive data register (SDR) value can be read, even while the next character is being received. If receive of the next character completes before the receive character is read, the new character is transferred from the receive buffer to the receive data register (SDR), overwriting the previous character. With this, the previous character is lost, and the status register OE (overrun error) bit is set to 1.

When two or more data frames are received where the receive characters and, if specified, the parity bit and stop bits, are all 0, the serial interface assumes a break and sets the RBRK (break detect) bit of the status register to 1.

If all error bits (status register bits OE, FE, PE, and RBRK) are set to 1, these settings are retained. Setting the control register ERS (error reset) bit to 1 clears the error bits to 0. Since the error bits remain cleared while the ERS bit is set to 1, subsequent errors cannot be detected. Clearing the ERS bit to 0 sets the error bits in accordance with the status of the receive data frame.

Parity errors and overrun errors do not affect receive, which continues as if no error had occurred. If a break is not detected and a framing error is generated, the serial interface continues receive as if a stop bit was actually present and detects the next start bit. If a break is detected, the start bit detection is suspended until the receive data input pin (RxD) is set once to 1.

5.3 Interrupt Control

5.3.1 Interrupt Causes

The serial interface supports the following four interrupt causes.

a) Transmit ready interrupt

At data transmit, the transmit data register can be loaded with the next transmit character.

TxRDY (transmit ready) = 1

b) Receive complete interrupt

At data receive, the receive character is transferred from the receive buffer to the receive data register (SDR); therefore the receive data register is ready to be read.

RxRDY (receive ready) = 1 and RxEN (receive enabled) = 1

c) Error occurred interrupt

A parity error (PE), overrun error (OE), or framing error (FE) is generated

(PE = 1, or OE or FE = 1) and RxEN = 1

d) Break detected interrupt

A break is detectedRBRK (break detected) = 1 and RxEN = 1

5.3.2 Interrupt Mask

Interrupts in the entire serial interface can be masked by the control register INTM (interrupt mask) bit. Each interrupt cause can be masked by the mode register RxINTM (receive complete interrupt mask) bit, the ERINTM (error occurred interrupt mask) bit, or the TxINTM (transmit ready interrupt mask) bit. Note that break detected interrupts are masked by the RxINTM bit. Moreover, bits S2 to S0 (serial interface channel 2 to 0 interrupt mask) of the interrupt control interrupt mask register (IMT) can mask interrupts for each channel.

5.3.3 Interrupt Generation Conditions

The interrupt generation conditions determined by the interrupt cause and mask are as follows.

Note n : Indicates the channel number n = 0, 1, 2

\+ : Indicates logical OR

× : Indicates logical AND

Channel n interrupts = (Sn = 0) × (INTM = 0) ×
{(transmit ready interrupt) + (receive complete interrupt) + (error occurred interrupt) + (break detected interrupt)}

Transmit ready interrupt = (TxINTM = 0) × (TxRDY = 1)

Receive complete interrupt = (RxINTM = 0) × (RxEN = 1) × (RxRDY = 1)

Error occurred interrupt = (RxEN = 1) × [(ERINTM = 0) × {(PE = 1) + (OE = 1) + (FE = 1)}]

Break detected interrupt = (RxINTM = 0) × (RBRK = 1)

5.3.4 Vector Number Generation

TMP68301A automatically generates vector numbers in the acknowledge cycle depending on the interrupt channels and causes. For details of vector numbers, see Table 4.2 in Chapter 4 Interrupt Controller.

The vector number register (IVNR) determines the upper three bits (7 to 5) of vector numbers generated in the acknowledge cycle. Bits 4 to 2 are determined by the channel number and generated by the interrupt controller. Bits 1 to 0 are determined by the interrupt cause and generated by the serial interface. (Figure 5.7)

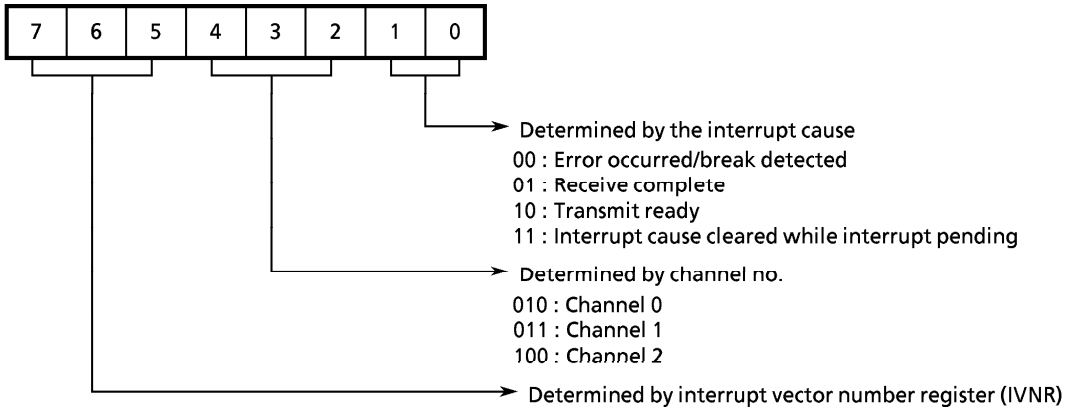


Figure 5.7 Serial Interface Vector Number Generation Method

When an interrupt is sent from the serial interface to the interrupt controller, the interrupt controller holds the interrupt and generates an interrupt request to the core processor. Interrupts pending (corresponding channel bits of the interrupt controller pending register are set to 1) due to masking by bits S2 to S0 (serial interface channel 2 to 0 interrupt mask) of the interrupt mask register (IMR) are accepted after the mask is released; interrupts pending due to execution of another interrupt with a higher priority are accepted after the higher-priority interrupt is fully processed.

When the interrupt is masked by the serial control register INTM bit, or the serial mode register RxINTM bit, ERINTM bit, or TxINTM bit, interrupt generation conditions are not satisfied and no interrupt request is sent to the interrupt controller. The serial interface does not retain masked interrupts. Therefore, if the interrupt cause is lost before releasing the masks of the control register INTM bit, or the mode register RxINTM bit, ERINTM bit, or TxINTM bit, the serial interface acts as if no interrupt is generated.

If conditions for the interrupt are no longer satisfied, the serial interface stops outputting an interrupt request to the interrupt controller. However, the interrupt controller holds the interrupt request and therefore continues to output the interrupt request to the core processor. If the pending bit of the channel corresponding to the interrupt pending register is not cleared (interrupt cancelled) before the core processor receives the interrupt request, the serial interface generates an interrupt acknowledge. When vector numbers are generated in the interrupt acknowledge cycle, numbers 7 to 2 are generated because the interrupt controller holds the interrupt request. However, as the interrupt generation conditions are no longer satisfied, the serial interface generates a vector number indicating "interrupt cause cleared while interrupt pending" using bits 1 to 0 of the vector number used to specify the interrupt cause. Figure 5.8 shows the relationship between the interrupt control circuit in the serial interface and the interrupt controller.

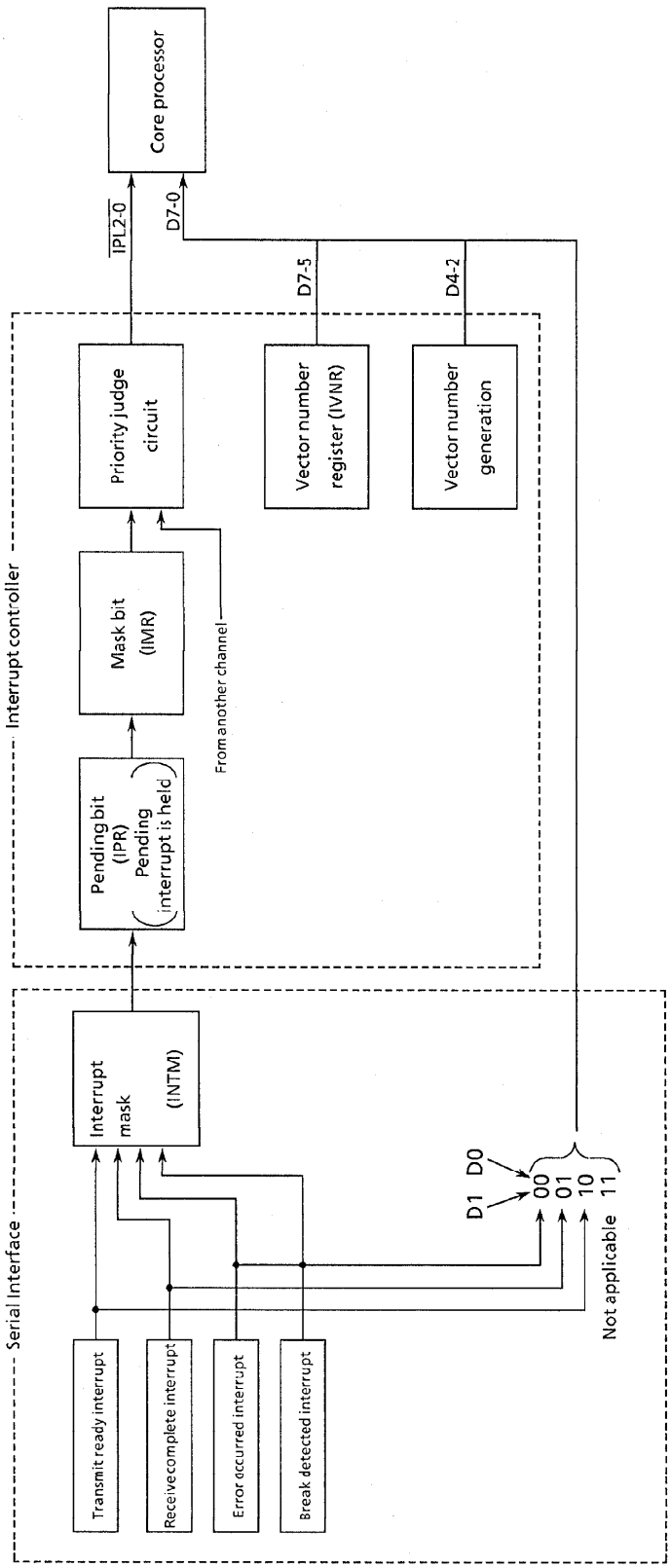


Figure 5.8 Relationship Between Interrupt Control Circuit and Interrupt Controller

5.3.5 Cautions

a) Break detected interrupts are masked by the RxINTM (receive interrupt mask) bit, but create an error occurred interrupt vector in the interrupt acknowledge cycle.

b) The TxRDY bit is still held in the transmit ready interrupt acknowledge cycle.

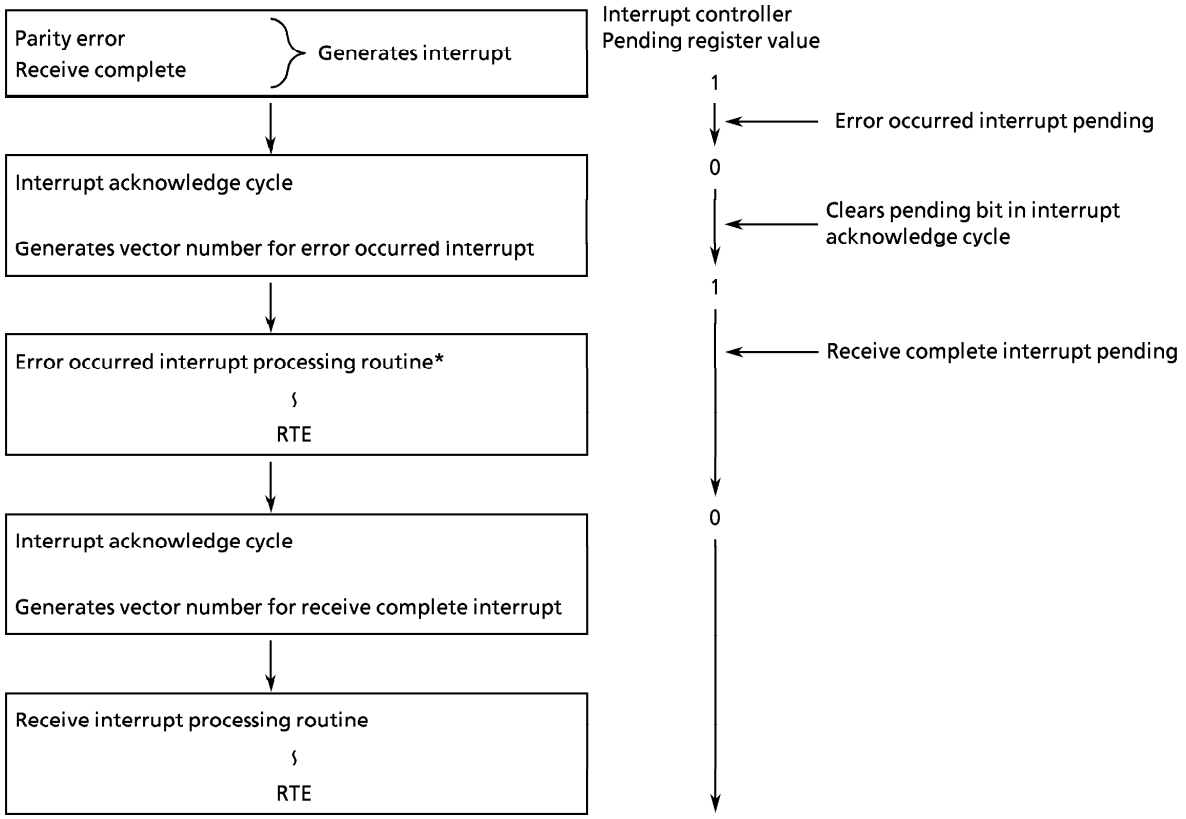
c) The TxRDY bit is initialized by a hardware reset (the RESET and HALT pins are simultaneously asserted). Therefore, the transmit ready interrupt can be used for the first data transmit after reset.

d) When interrupt request conditions are satisfied once, an interrupt processing is generated only once. That is, even if the transmit ready interrupt processing routine ends without the transmit data being written to the transmit data register (SDR), an interrupt will not be regenerated to jump to the transmit ready interrupt processing routine. If more than one error occurs simultaneously, the routine jumps only once to the error occurred interrupt processing routine.

e) The following is the interrupt processing priority when more than one interrupt request is generated simultaneously.

Error occurred interrupt	High
Receive complete interrupt	↓
Transmit ready interrupt	Low

Figure 5.9 shows an interrupt processing routine example when an error occurred interrupt and a receive complete interrupt are generated simultaneously.



Note * : When a receive error occurs, an error occurred interrupt and a receive complete interrupt are generated simultaneously. Reading the receive data register (SDR) in the error occurred interrupt processing routine resets RxRDY to 0 and prevents the receive complete interrupt generation conditions from being satisfied. Therefore, a vector number indicating "interrupt cause cleared while interrupt pending" is generated in the next interrupt acknowledge cycle.

Figure 5.9 Multiple Interrupt Processing Example

5.4 Initialization of Serial Interface

To initialize serial interface, set the RES (software reset) bit of the control register to 1. The serial interface is also initialized if a hardware reset is executed (the RESET and HALT pins are simultaneously asserted) because the RES bit is set to 1. The initialized status is held until the RES bit is reset to 0. In the initial state, transmit, receive, and interrupts are all disabled. The generation of superfluous operations can thus be avoided until the serial interface setup is complete.

Table 5.1 lists the initial register values.

Register	7	6	5	4	3	2	1	0
Serial control register (SCR)	CKSE		RES					INTM
	1	–	1	–	–	–	–	1
Serial command register (SCMR0-2)			RTS *1	ERS	SBRK	R_XEN	DTR *1	T_XEN
	–	–	0	1	0	0	0	0
Serial status register (SSR0-2)	DSR *1	RBRK	FE	OE	PE	T_XE	R_XRDY	T_XRDY
	Depending on external pin status	0	0	0	0	1	0	0

– : Undefined
*1 : channel 0 only

Table 5.1 Serial Interface Register Initial Values

5.5 Register Description

5.5.1 Serial Control Register (SCR)

This register is used to make the serial interface basic settings.
Set the registers after resetting the RES bit to 0.

Offset address $18F	7	6	5	4	3	2	1	0	
	CKSE		RES					INTM	SCR
Reset	1	0	1	0	0	0	0	1	
		R		R	R	R	R		R : Read only

Note: For future expansion, set undefined bits to 0.

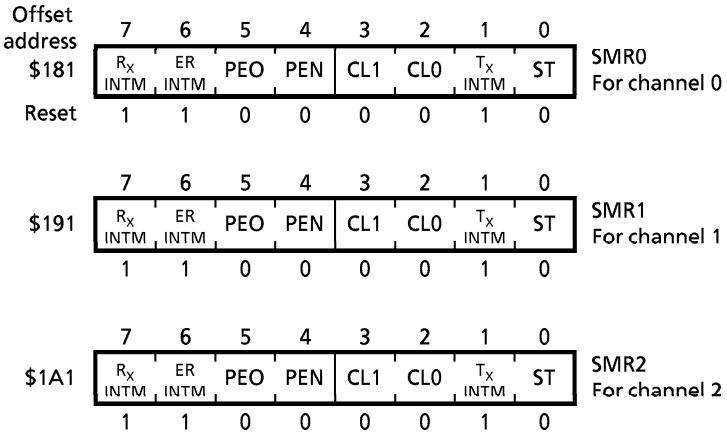
CKSE: Divider clock select
- 0 : External input clock (BCLK)
- 1 : System clock (CLK)

RES : Software reset
- 0 : Reset cansel
- 1 : Reset

INTM: Interrupt mask
- 0 : Enable interrupt signal generation
- 1 : Disable interrupt signal generation

5.5.2 Serial Mode Registers 0 - 2 (SMR0 - 2)

These registers, one per channel, specify the character structure and the interrupt generation masks.



RxINTM : Receive complete interrupt mask and break detect interrupt mask *2
- 0 : Interrupt valid
- 1 : Interrupt masked

ERINTM : Error occurred interrupt mask *2
- 0 : Interrupt valid
- 1 : Interrupt masked

PEO : Parity select *1
- 0 : Even parity
- 1 : Odd parity

PEN : Parity control
- 0 : No parity
- 1 : Parity

CL1-CL0 : Character length

CL1	CL0	Character length
0	0	5-bit character
0	1	6-bit character
1	0	7-bit character
1	1	8-bit character

TxINTM : Transmit ready interrupt mask
- 0 : Interrupt valid
- 1 : Interrupt masked

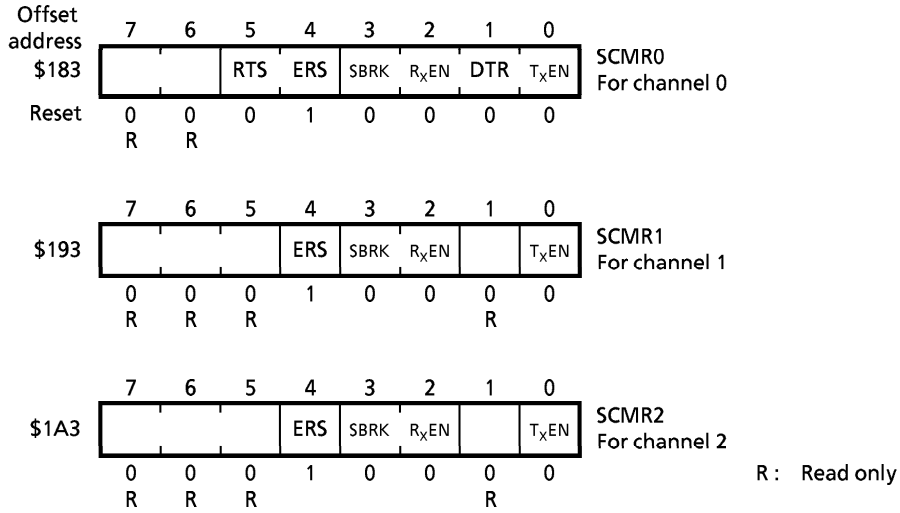
ST : Stop bit length
- 0 : One stop bit
- 1 : Two stop bits

*1 : Shows parity generation;
Even parity means that the number of 1's in the transmit/receive character including the parity bit is even.
Odd parity means that the number of 1's in the transmit/receive character including the parity bit is odd.

*2 : ERINTM is the interrupt mask for framing errors, overrun errors, and parity errors. RxINTM is the interrupt mask for break detect.

5.5.3 Serial Command Registers 0 - 2 (SCMR0 - 2)

These registers, one per channel, are for transmit/receive control.



Note: For future expansion, set undefined bits to 0.

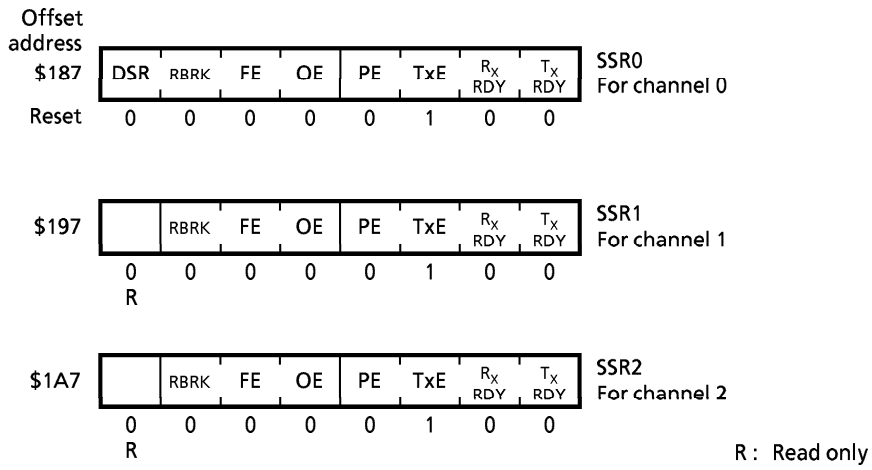
RTS : RTS0 pin output control (channel 0 only)
0 : High output from $\overline{RTS0}$ pin
1 : Low output from $\overline{RTS0}$ pin
ERS : Error reset *2
0 : No operation
1 : Reset PE, OE, FE, and RBRK bits
SBRK : Break transmit
0 : No break transmit
1 : Break transmit
RxEN : Receive enable
0 : Receive disable
1 : Receive enable
DTR : DTR pin output control (channel 0 only) *1
0 : High output from pin $\overline{DTR0}$
1 : Low output from pin $\overline{DTR0}$
TxEN : Transmit enable
0 : Transmit disable
1 : Transmit enable

*1 : As pin IO13/$\overline{DTR0}$ multiplexes with the parallel interface I/O port, the pin is used for $\overline{DTR0}$ output only when bit PF0 (pin function 0) of the parallel control register is set to 1.

*2 : While bit ERS is set to 1, the error bits (PE, OE, FE, RBRK) are continuously reset. Resetting ERS to 0 sets the error bits according to the status of the receive data frame. Therefore, when resuming error detection after resetting error bits, reset the ERS bit to 0 again.

5.5.4 Serial Status Registers 0 - 2 (SSR0 - 2)

These registers, one per channel, indicate the channel status.



DSR : DSR pin input sense (channel 0 only) *2
- 0 : IO14/DSR0 pin input is high.
- 1 : IO14/DSR0 pin input is low.

RBRK : Break detect
- 0 : Does not detect break
- 1 : Detects break

FE : Framing error
- 0 : Does not generate framing error
- 1 : Generates framing error

OE : Overrun error
- 0 : Does not generate overrun error.
- 1 : Generates overrun error.

PE : Parity error
- 0 : Does not generate parity error.
- 1 : Generates parity error.

TxE: Transmit data empty
- 0 : The transmit data register (SDR) contains a transmit character, or a transmit is in progress (transmit character is in transmit buffer).
- 1 : The transmit data register (SDR) is empty and no transmit is in progress (no transmit character in transmit buffer).

RxRDY : Receive ready
- 0 : Receive not ready
- 1 : Receive ready (the receive data register (SDR) contains a receive character, the receive data register can be read)

TxRDY : Transmit ready *1
- 0 : Transmit not ready
- 1 : Transmit ready (transmit character can be written to transmit data register (SDR))

*1 : TxRDY is set to 1 when [{transmit data register empty} x (CTS = 0) x (TxEN = 1)]. When channels 1, and 2, and 0 do not use CTS0, CTS0 is treated as CTS0 = 0 without checking CTS0 pin input. CTS0 multiplexes with the parallel interface I/O port: IO13/$\overline{\text{CTS0}}$. Thus, the pin is used for $\overline{\text{CTS0}}$ input only when bit PF0 (pin function 0) of the parallel control register (PCR) is set to 1.

*2 : Pin IO14/DSR0 multiplexes with the parallel interface I/O port. Thus, the pin is used for DSR0 input only when PF0 of the parallel control register is set to 1.

5.5.5 Serial Prescaler Register (SPR)

The prescaler divides the input clock (system clock or BCLK pin input) by the divider ratio specified in this register.

The values to be set in this register are from 0 to 255.

Offset address	7	6	5	4	3	2	1	0	
$18D	P7	P6	P5	P4	P3	P2	P1	P0	SPR
Reset	–	–	–	–	–	–	–	–	

Set value	Divider ratio
0	× 1/256
1	× 1
2~255	× 1/2~ × 1/255

5.5.6 Baud Rate Register 0 - 2 (SBRR0 - 2)

The baud rate generator further divides, by the divider ratio specified in this register, the clock (PCLK) divided by the prescaler. Each channel has a baud rate register. More than one bit cannot be set simultaneously. The clock obtained (GCLK) by dividing PCLK with the baud rate register setting is used as the serial interface transmit/receive fundamental clock. The clock actually used to shift out data bits and perform sampling (TCLK) is obtained by dividing the GCLK by eight. Table 5.2 shows the baud rate register settings and the divider ratios with the baud rate generator (GCLK/PCLK).

Offset address	7	6	5	4	3	2	1	0	
$185	B7	B6	B5	B4	B3	B2	B1	B0	SBRR0 For channel 0
	–	–	–	–	–	–	–	–	
	7	6	5	4	3	2	1	0	
$195	B7	B6	B5	B4	B3	B2	B1	B0	SBRR1 For channel 1
	–	–	–	–	–	–	–	–	
	7	6	5	4	3	2	1	0	
$1A5	B7	B6	B5	B4	B3	B2	B1	B0	SBRR2 For channel 2
	–	–	–	–	–	–	–	–	

Baud rate register values								Divider ratio	
B7	B6	B5	B4	B3	B2	B1	B0	10 Decimal	
1	0	0	0	0	0	0	0	128	1/128
0	1	0	0	0	0	0	0	64	1/64
0	0	1	0	0	0	0	0	32	1/32
0	0	0	1	0	0	0	0	16	1/16
0	0	0	0	1	0	0	0	8	1/8
0	0	0	0	0	1	0	0	4	1/4
0	0	0	0	0	0	1	0	2	1/2
0	0	0	0	0	0	0	1	1	1

Table 5.2 Divider Ratios by Baud Rate Generator

Channel n baud rate set value is defined by the following equation:

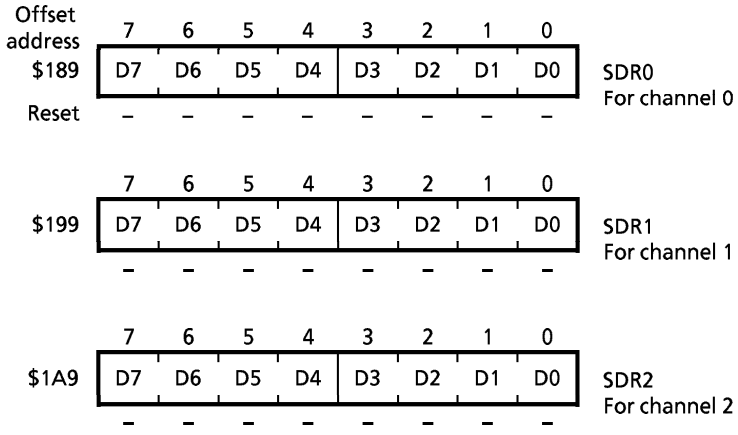
$$TCLKn = CLK \div PR \div BRRn \div 8$$

n : Channel number (n = 0~2)
TCLKn : Channel n baud rate (bps)
CLK : Input clock (system clock or BCLK pin input) frequency (Hz)
PR : Prescaler (SPR) set value (PR = 1 - 256 However, PR = 256 means value is 0)
BRRn : Channel n baud rate register (SBRRn) set value (BRRn = 1 - 128)

Table 5.3 shows baud rate setting examples.

5.5.7 Serial Data Registers 0 - 2 (SDR0 - 2)

These registers, one per channel, store transmit data and receive data. Although the transmit data register and the receive data register share the same address, the two registers are independent. The receive data register is accessed at read. The transmit data register is accessed at write.



SCLK = 16.67MHz

Baud Rate (bps)	BRR ( ) Hexadecimal	PR ( ) Hexadecimal	Error	BRR ( ) Hexadecimal	PR ( ) Hexadecimal	Error
38.4K	1 ($01)	54 ($36)	+ 0.48%	2 ($02)	27 ($1B)	+ 0.48%
19.2K	1 ($01)	109 ($6D)	– 0.44%	4 ($04)	27 ($1B)	+ 0.48%
14.4K	1 ($01)	145 ($91)	– 0.20%	16 ($10)	9 ($09)	+ 0.48%
9600	1 ($01)	217 ($D9)	+ 0.02%	8 ($08)	27 ($1B)	+ 0.48%
4800	2 ($02)	217 ($D9)	+ 0.02%	16 ($10)	27 ($1B)	+ 0.48%
2400	4 ($04)	217 ($D9)	+ 0.02%	32 ($20)	27 ($1B)	+ 0.48%
1200	8 ($08)	217 ($D9)	+ 0.02%	64 ($40)	27 ($1B)	+ 0.48%
600	16 ($10)	217 ($D9)	+ 0.02%	128 ($80)	27 ($1B)	+ 0.48%

SCLK = 16.0MHz

Baud Rate (bps)	BRR ( ) Hexadecimal	PR ( ) Hexadecimal	Error	BRR ( ) Hexadecimal	PR ( ) Hexadecimal	Error
38.4K	1 ($01)	52 ($34)	+ 0.16%	2 ($02)	26 ($1A)	+ 0.16%
19.2K	1 ($01)	104 ($68)	+ 0.16%	4 ($04)	26 ($1A)	+ 0.16%
14.4K	1 ($01)	139 ($8B)	– 0.08%	4 ($04)	35 ($23)	– 0.79%
9600	1 ($01)	208 ($D0)	+ 0.16%	8 ($08)	26 ($1A)	+ 0.16%
4800	2 ($02)	208 ($D0)	+ 0.16%	16 ($10)	26 ($1A)	+ 0.16%
2400	4 ($04)	208 ($D0)	+ 0.16%	32 ($20)	26 ($1A)	+ 0.16%
1200	8 ($08)	208 ($D0)	+ 0.16%	64 ($40)	26 ($1A)	+ 0.16%
600	16 ($10)	208 ($D0)	+ 0.16%	128 ($80)	26 ($1A)	+ 0.16%

SCLK = 12.5MHz

Baud Rate (bps)	BRR ( ) Hexadecimal	PR ( ) Hexadecimal	Error	BRR ( ) Hexadecimal	PR ( ) Hexadecimal	Error
38.4K	1 ($01)	41 ($29)	– 0.76%	2 ($02)	20 ($14)	+ 1.73%
19.2K	1 ($01)	81 ($51)	+ 0.47%	4 ($04)	20 ($14)	+ 1.73%
14.4K	1 ($01)	109 ($6D)	– 0.45%	4 ($04)	27 ($1B)	+ 0.47%
9600	1 ($01)	163 ($A3)	– 0.15%	8 ($08)	20 ($14)	+ 1.73%
4800	2 ($02)	163 ($A3)	– 0.15%	16 ($10)	20 ($14)	+ 1.73%
2400	4 ($04)	163 ($A3)	– 0.15%	32 ($20)	20 ($14)	+ 1.73%
1200	8 ($08)	163 ($A3)	– 0.15%	64 ($40)	20 ($14)	+ 1.73%
600	16 ($10)	163 ($A3)	– 0.15%	128 ($80)	20 ($14)	+ 1.73%

Figure 5.3 Baud Rate Setting Examples (1)

SCLK = 8.0MHz

Baud Rate (bps)	BRR ( ) Hexadecimal	PR ( ) Hexadecimal	Error	BRR ( ) Hexadecimal	PR ( ) Hexadecimal	Error
38.4K	1 ($01)	26 ($1A)	+0.16%	2 ($02)	13 ($0D)	+0.16%
19.2K	1 ($01)	52 ($34)	+0.16%	4 ($04)	13 ($0D)	+0.16%
14.4K	1 ($01)	69 ($45)	+0.64%	4 ($04)	17 ($11)	+2.12%
9600	1 ($01)	104 ($68)	+0.16%	8 ($08)	13 ($0D)	+0.16%
4800	2 ($02)	104 ($68)	+0.16%	16 ($10)	13 ($0D)	+0.16%
2400	4 ($04)	104 ($68)	+0.16%	32 ($20)	13 ($0D)	+0.16%
1200	8 ($08)	104 ($68)	+0.16%	64 ($40)	13 ($0D)	+0.16%
600	16 ($10)	104 ($68)	+0.16%	128 ($80)	13 ($0D)	+0.16%

SCLK = 7.3728MHz

Baud Rate (bps)	BRR ( ) Hexadecimal	PR ( ) Hexadecimal	Error	BRR ( ) Hexadecimal	PR ( ) Hexadecimal	Error
38.4K	1 ($01)	24 ($18)	0%	2 ($02)	12 ($0C)	0%
19.2K	1 ($01)	48 ($30)	0%	4 ($04)	12 ($0C)	0%
14.4K	1 ($01)	64 ($40)	0%	4 ($04)	16 ($10)	0%
9600	1 ($01)	96 ($60)	0%	8 ($08)	12 ($0C)	0%
4800	1 ($01)	192 ($C0)	0%	16 ($10)	12 ($0C)	0%
2400	2 ($02)	192 ($C0)	0%	32 ($20)	12 ($0C)	0%
1200	4 ($04)	192 ($C0)	0%	64 ($40)	12 ($0C)	0%
600	8 ($08)	192 ($C0)	0%	128 ($80)	12 ($0C)	0%

SCLK = 1.8432MHz

Baud Rate (bps)	BRR ( ) Hexadecimal	PR ( ) Hexadecimal	Error	BRR ( ) Hexadecimal	PR ( ) Hexadecimal	Error
38.4K	1 ($01)	6 ($06)	0%	2 ($02)	3 ($03)	0%
19.2K	1 ($01)	12 ($0C)	0%	4 ($04)	3 ($03)	0%
14.4K	1 ($01)	16 ($10)	0%	4 ($04)	4 ($04)	0%
9600	1 ($01)	24 ($18)	0%	8 ($08)	3 ($03)	0%
4800	1 ($01)	48 ($30)	0%	16 ($10)	3 ($03)	0%
2400	1 ($01)	96 ($60)	0%	32 ($20)	3 ($03)	0%
1200	1 ($01)	192 ($C0)	0%	64 ($40)	3 ($03)	0%
600	2 ($01)	192 ($C0)	0%	128 ($80)	3 ($03)	0%

Figure 5.3 Baud Rate Setting Examples (2)

6. Parallel Interface

6.1 Outline

This parallel interface uses a general-purpose, 16-bit I/O port where input or output can be specified for each bit. Switching modes changes the interface to Centronics.

With the Centronics interface, the 16-bit data bus becomes an 8-bit data bus with up to five control signal lines. At Centronics transmit, when the parallel interface receives 8-bit data from the MPU, the parallel interface generates an external data strobe at a period previously programmed in an internal register. At Centronics receive, 8-bit data are stored internally at the external data strobe, and the parallel interface returns a BUSY signal.

The parallel interface generates three interrupts at Centronics transmit/receive.

6.2 Operating Mode

The parallel interface has three operating modes. Use the internal control register to select one of these modes. Each mode can select one of several pin functions. Some modes can use some of the parallel interface I/O port pins as a serial interface or as a Centronics response control signal.

Mode 0	16-bit bus (input/output operation)
Mode 1	8-bit bus (output operation) + control signal (Centronics output)
Mode 2	8-bit bus (input operation) + control signal (Centronics input)

6.2.1 Control Signal Automatic Generation Function

The parallel interface can generate signals at any timing when DSTB and ACK Centronics control signals are programmed in the register. Two internal flags, XBUSY and BUFFER–FULL, are provided for the parallel interface. Thus, the parallel interface can communicate with external devices of any operating speed.

6.2.2 Interrupts

The parallel interface generates the following three interrupts.

At transmit ready	(in mode 1)
At receive complete	(in mode 2)
At external device status change	(in mode 1: at FAULT input change)

6.2.3 Mode 0 Operation

In mode 0, the port operates as a 16-bit I/O port. Each parallel interface bit can be specified as either input or output using the direction register. The input pins pass data received from an external source through the internal buffer and set them in the data register. The output pins pass data from the data register through the internal buffer and output them to external devices.

6.2.4 Mode 1 Operation

Mode 1 operates as the Centronics output.

When the parallel interface receives data from the core processor, it stores the data in the internal buffer and simultaneously generates $\overline{DSTB}$ at the timing specified in parameter registers 1 and 2. The interface then receives a negated BUSY signal returned by the external device, and generates an interrupt. To compensate for the delay of the external BUSY signal, an internal XBUSY flag is provided. When data are stored in the internal buffer, the XBUSY flag is set. When an external $\overline{ACK}$ signal is received, the flag is reset. The mode register selects the timing of the interrupt generation from either the external BUSY signal falling edge, or the clearance of the internal XBUSY flag.

When PRIME and FAULT functions are added to the pins, PRIME is assigned to an output pin; FAULT, to an input pin. They are used for the printer initialization signal and the printer abnormality detection signal, respectively.

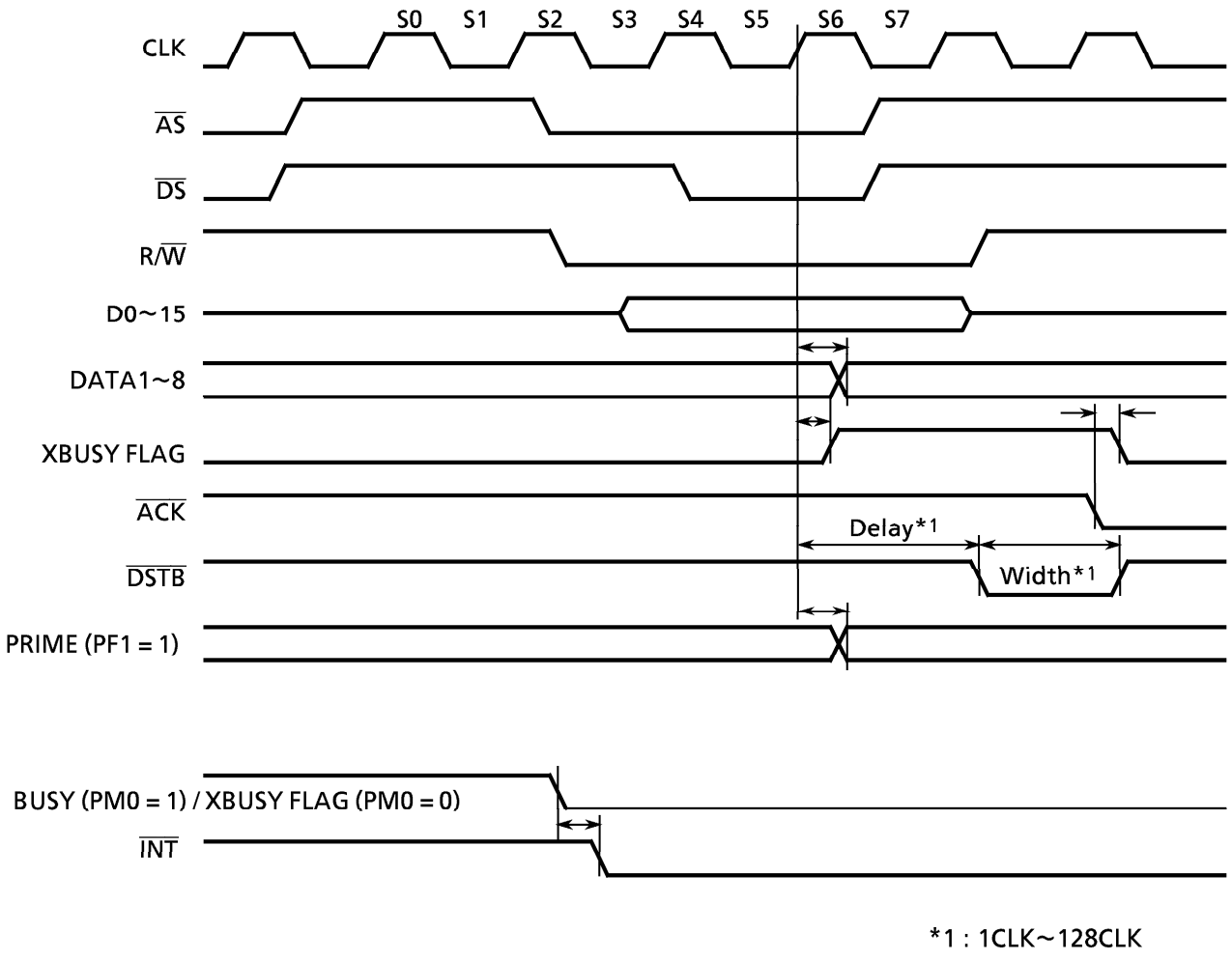


Figure 6.2 Centronics Output Timing

6.2.5 Mode 2 Operation

Mode 2 operates as the Centronics input.

When the parallel interface receives a data strobe from an external source, it stores the data in the internal buffer and simultaneously generates an interrupt and an external BUSY signal. Then, with a read or dummy write from the core processor, the parallel interface returns an $\overline{ACK}$ signal externally with the timing specified in parameter registers 1 and 2. To show that data have been received, an internal BUFFER–FULL flag is provided. When data are stored in the internal buffer, this flag is set. The BUFFER–FULL flag is reset when an external $\overline{ACK}$ signal is generated. The $\overline{ACK}$ signal ready timing can be included in the BUSY signal ready timing by setting the mode register.

When PRIME and FAULT functions are added to the pins, PRIME is assigned to an input pin; FAULT, to an output pin.

Note: Reading the lower byte of the parallel data register when not receiving data generates a receive interrupt. Therefore, access the data register only after receive.
Accessing the upper byte only does not generate an interrupt.

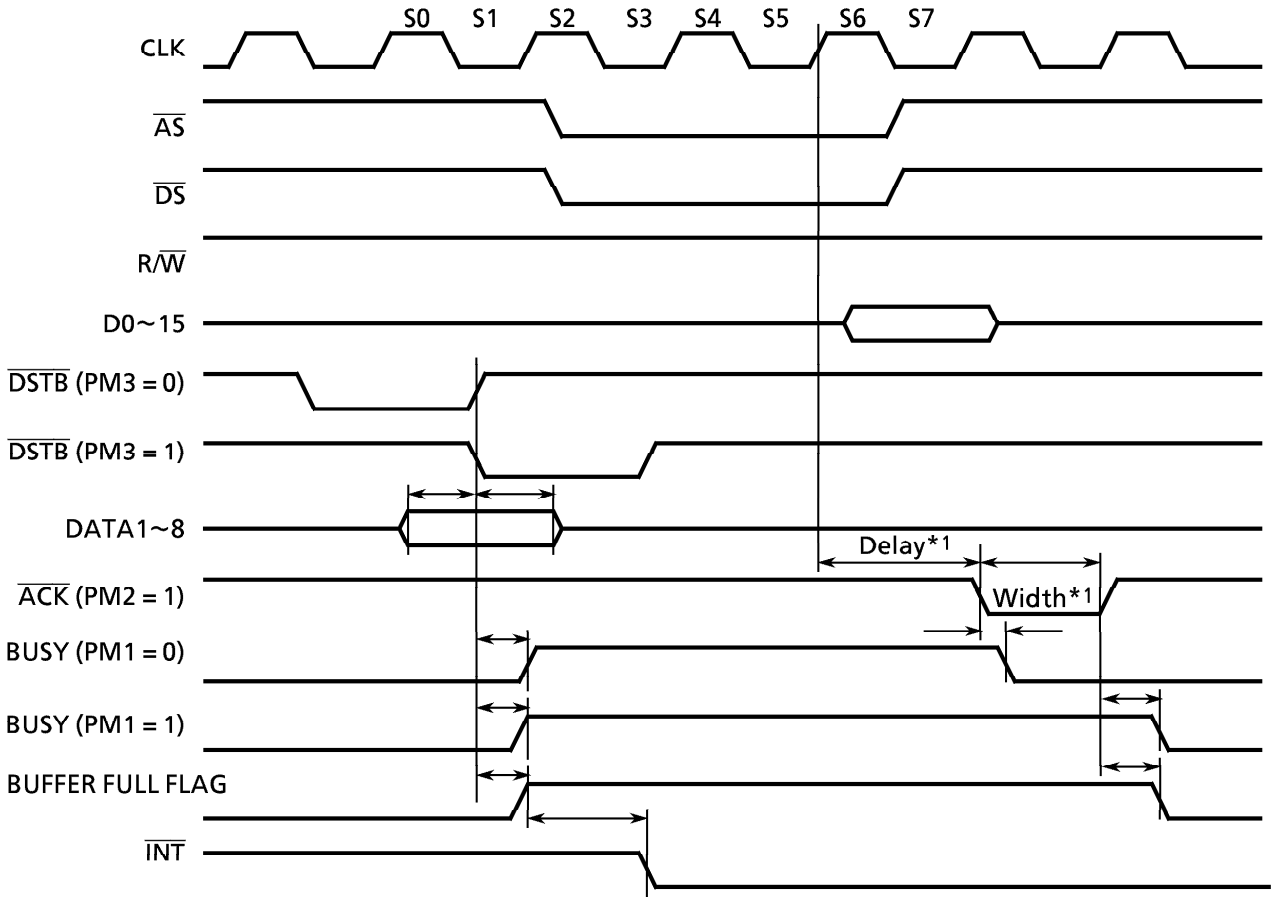
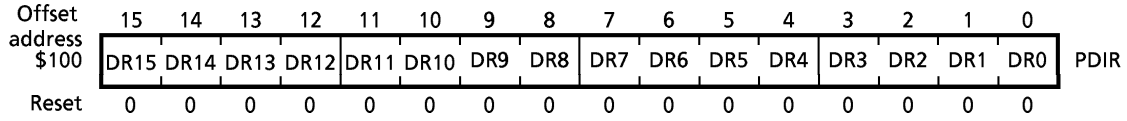


Figure 6.3 Centronics Input Timing

6.3 Register Configuration

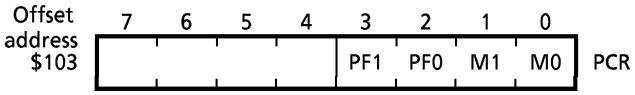
6.3.1 Parallel Direction Register

This register programs each bit of the 16-bit port for input or output. 1 specifies output; 0, input.



6.3.2 Parallel Control Register

This register selects operating modes and pin functions. M0 and M1 specify operating modes. PF0 and PF1 specify pin functions. The following diagram shows the relationship between operating modes and pin functions.



Mode 0 (M0 = 0, M1 = 0)		Mode 1 (M0 = 1, M1 = 0)				Mode 2 (M0 = X, M1 = 1)			
PF1 = X		PF1 = 0		PF1 = 1		PF1 = 0		PF1 = 1	
PF0 = 0	PF0 = 1	PF0 = 0	PF0 = 1	PF0 = 0	PF0 = 1	PF0 = 0	PF0 = 1	PF0 = 0	PF0 = 1
I/O 0	I/O 0	DATA1	DATA1	DATA1	DATA1	DATA1	DATA1	DATA1	DATA1
I/O 1	I/O 1	DATA2	DATA2	DATA2	DATA2	DATA2	DATA2	DATA2	DATA2
I/O 2	I/O 2	DATA3	DATA3	DATA3	DATA3	DATA3	DATA3	DATA3	DATA3
I/O 3	I/O 3	DATA4	DATA4	DATA4	DATA4	DATA4	DATA4	DATA4	DATA4
I/O 4	I/O 4	DATA5	DATA5	DATA5	DATA5	DATA5	DATA5	DATA5	DATA5
I/O 5	I/O 5	DATA6	DATA6	DATA6	DATA6	DATA6	DATA6	DATA6	DATA6
I/O 6	I/O 6	DATA7	DATA7	DATA7	DATA7	DATA7	DATA7	DATA7	DATA7
I/O 7	I/O 7	DATA8	DATA8	DATA8	DATA8	DATA8	DATA8	DATA8	DATA8
I/O 8	I/O 8	$\overline{\text{DSTB}}$	$\overline{\text{DSTB}}$	$\overline{\text{DSTB}}$	$\overline{\text{DSTB}}$	$\overline{\text{DSTB}}$	$\overline{\text{DSTB}}$	$\overline{\text{DSTB}}$	$\overline{\text{DSTB}}$
I/O 9	I/O 9	BUSY	BUSY	BUSY	BUSY	BUSY	BUSY	BUSY	BUSY
I/O 10	I/O 10	$\overline{\text{ACK}}$	$\overline{\text{ACK}}$	$\overline{\text{ACK}}$	$\overline{\text{ACK}}$	$\overline{\text{ACK}}$	$\overline{\text{ACK}}$	$\overline{\text{ACK}}$	$\overline{\text{ACK}}$
I/O 11	I/O 11	I/O 11	I/O 11	PRIME	PRIME	I/O 11	I/O 11	PRIME	PRIME
I/O 12	I/O 12	I/O 12	I/O 12	FAULT	FAULT	I/O 12	I/O 12	FAULT	FAULT
I/O 13	$\overline{\text{CTS0}}$	I/O 13	$\overline{\text{CTS0}}$	I/O 13	$\overline{\text{CTS0}}$	I/O 13	$\overline{\text{CTS0}}$	I/O 13	$\overline{\text{CTS0}}$
I/O 14	$\overline{\text{DSR0}}$	I/O 14	$\overline{\text{DSR0}}$	I/O 14	$\overline{\text{DSR0}}$	I/O 14	$\overline{\text{DSR0}}$	I/O 14	$\overline{\text{DSR0}}$
I/O 15	$\overline{\text{DTR0}}$	I/O 15	$\overline{\text{DTR0}}$	I/O 15	$\overline{\text{DTR0}}$	I/O 15	$\overline{\text{DTR0}}$	I/O 15	$\overline{\text{DTR0}}$

6.3.3 Parallel Status Register

This register stores the control signals used in mode 1 and 2 Centronics operation. Note that S0, S1, S4, and S5 are valid only when PF1 of the control register is set to 1. S0 to S3 store the control signals for mode 1; S4 to S6, the control signals for mode 2. When an interrupt is generated in mode 1 or 2, the interrupt flag (IF) is set to 1. If becomes into 0 after the interrupt acknowledge cycle. The IF remains set to 1 until a reset is input.

All bits of the register can be read, but only the IF bit can be written. (However, 0 cannot be set before the interrupt acknowledge cycle.)

Offset address	7	6	5	4	3	2	1	0	
$105	IF	S6	S5	S4	S3	S2	S1	S0	PSR
Reset	0	1 R	0 R	0 R	0 R	0 R	0 R	0 R	R : Read only

IF : Interrupt flag (for modes 1 and 2 only)
- 0 : No interrupt
- 1 : Interrupt

S6: BUFFER – FULL flag (for mode 2 only)
- 0 : Buffer empty
- 1 : Buffer full

S5: PRIME input status (in mode 2 only)

S4: FAULT output status (in mode 2 only)

S3: XBUSY flag (in mode 1 only)

S2: BUSY input status (in mode 1 only)

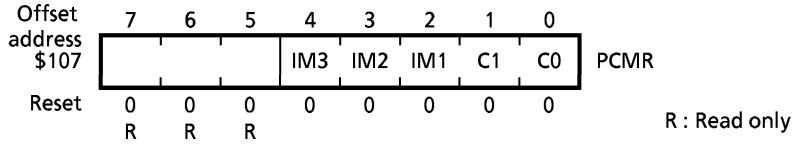
S1: PRIME output status (in mode 1 only)

S0: FAULT input status (in mode 1 only)

6.3.4 Parallel Command Register

The parallel command register sets the status of the output pins in modes 1 and 2, and enables or enables/disables each of the three interrupts. The parallel interface ignores and does not retain an interrupt request when the requested interrupt is disabled. To postpone an interrupt, use the mask register in the interrupt controller.

C0, IM1, and IM2 are valid in mode 1. C1 and IM3 are valid in mode 2. However, C0, C1, and IM2 are valid only when PF1 of the control register is set to 1.



IM3: Receive complete interrupt mask flag (valid only in mode 2)
- 0 : Enables receive complete interrupt.
- 1 : Masks receive complete interrupt.

IM2: External device status change interrupt mask flag (valid only in mode 1)
- 0 : Enables external device status change interrupt.
- 1 : Masks external device status change interrupt.

IM1: Transmit ready interrupt mask flag (valid only in mode 1)
- 0 : Enables transmit ready interrupt.
- 1 : Masks transmit ready interrupt.

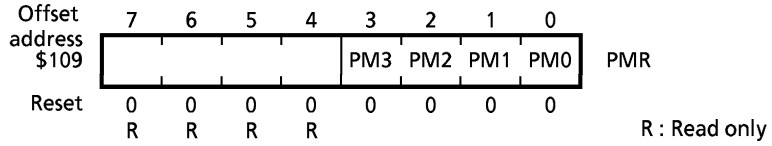
C1 : FAULT output control (valid only in mode 2)
- 0 : Sets FAULT output level to 0.
- 1 : Sets FAULT output level to 1.

C0 : PRIME output control (valid only in mode 1)
- 0 : Set PRIME output level to 0.
- 1 : Set PRIME output level to 1.

Note 1: The PRIME output level changes when data are written to the parallel data register. (Figure 6.2)

6.3.5 Parallel Mode Register

This register defines the relationship between the BUSY signal and the interrupt, and the relationship between the BUSY signal and the $\overline{ACK}$ signal. In mode 1, PM0 is valid. In mode 2, PM1, PM2, and PM3 are valid.



PM3 : $\overline{DSTB}$ edge definition
- 0 : Loads data at $\overline{DSTB}$ rising edge.
- 1 : Loads data at $\overline{DSTB}$ falling edge.

PM2 : $\overline{ACK}$ generation control
- 0 : A dummy write resets BUSY and generates the $\overline{ACK}$ signal.
  (Using this option, assertion of $\overline{ACK}$ can be delayed by software control.)
- 1 : A read resets BUSY and generates the $\overline{ACK}$ signal.

PM1 : BUSY signal definition
- 0 : $\overline{ACK}$ not included in BUSY signal.
- 1 : $\overline{ACK}$ included in BUSY signal.

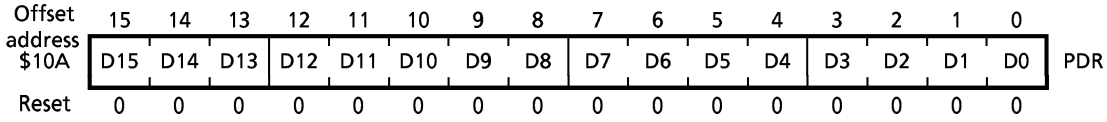
PM0 : Transfer interrupt definition
- 0 : Generates a transfer ready interrupt when XBUSY flag is cleared.
- 1 : Generates a transfer ready interrupt at the falling edge of the external BUSY signal.

6.3.6 Parallel Data Register

This register reads or writes data input/output from the I/O ports.

When a port is in input mode or during Centronics receive, data from an external source received by the receive buffer are immediately passed to this register. Note that any data written to this register at this time are ignored.

When a port is in output mode or during Centronics transmit, data written to the corresponding bit in this register are output externally. If the register is read then, the data currently output to the port will be read.



6.3.7 Parallel Parameter Register 1

This register programs the delay time required for the Centronics interface to automatically generate $\overline{DSTB}$ and $\overline{ACK}$ control signals. The register setting automatically generates the $\overline{DSTB}$ and $\overline{ACK}$ delay time. The register indicates the number of $\overline{DSTB}$ delay clocks in mode 1, and the number of $\overline{ACK}$ delay clocks in mode 2. The equation for calculating the delay time is as follows.

$$Delay = DT6 \times 64 + DT5 \times 32 + DT4 \times 16 + DT3 \times 8 + DT2 \times 4 + DT1 \times 2 + DT0$$

Offset address $10D	7	6	5	4	3	2	1	0	
		DT6	DT5	DT4	DT3	DT2	DT1	DT0	PPR1
Reset	0 R	0	0	0	0	0	0	0	R : Read only

6.3.8 Parallel Parameter Register 2

This register programs the delay time required for the Centronics interface to automatically generate $\overline{DSTB}$ and $\overline{ACK}$ control signals. The register setting automatically generates the $\overline{DSTB}$ and $\overline{ACK}$ signal width. The register specifies the width of $\overline{DSTB}$ in clocks in mode 1, and the width of $\overline{ACK}$ in clocks in mode 2. The equation for calculating the width is as follows. This register is used together with parallel parameter register 1.

$$Width = DW6 \times 64 + DW5 \times 32 + DW4 \times 16 + DW3 \times 8 + DW2 \times 4 + DW1 \times 2 + DW0$$

Offset address $10F	7	6	5	4	3	2	1	0	
		DW6	DW5	DW4	DW3	DW2	DW1	DW0	PPR2
Reset	0 R	0	0	0	0	0	0	0	R : Read only

7. Timer

7.1 Outline

The timer consists of three independent channels, each with a 16-bit counter. Each channel has an 8-bit prescaler (valid only when the system clock is used). Each channel generates interrupt requests. The timer can cascade-connect the three channels for long counts.

Three independent channels each with a built-in 16-bit counter
Count data are readable
Generates interrupts for each channel
Software/hardware trigger
Programmable operating modes
Outputs any duty comparison waveform (channels 1 and 2 only)
Outputs one-shot pulse (channels 1 and 2 only)
Event count
Max. 8MHz count (at 16 MHz)

Figures 7.1 and 7.2 show the timer outline.

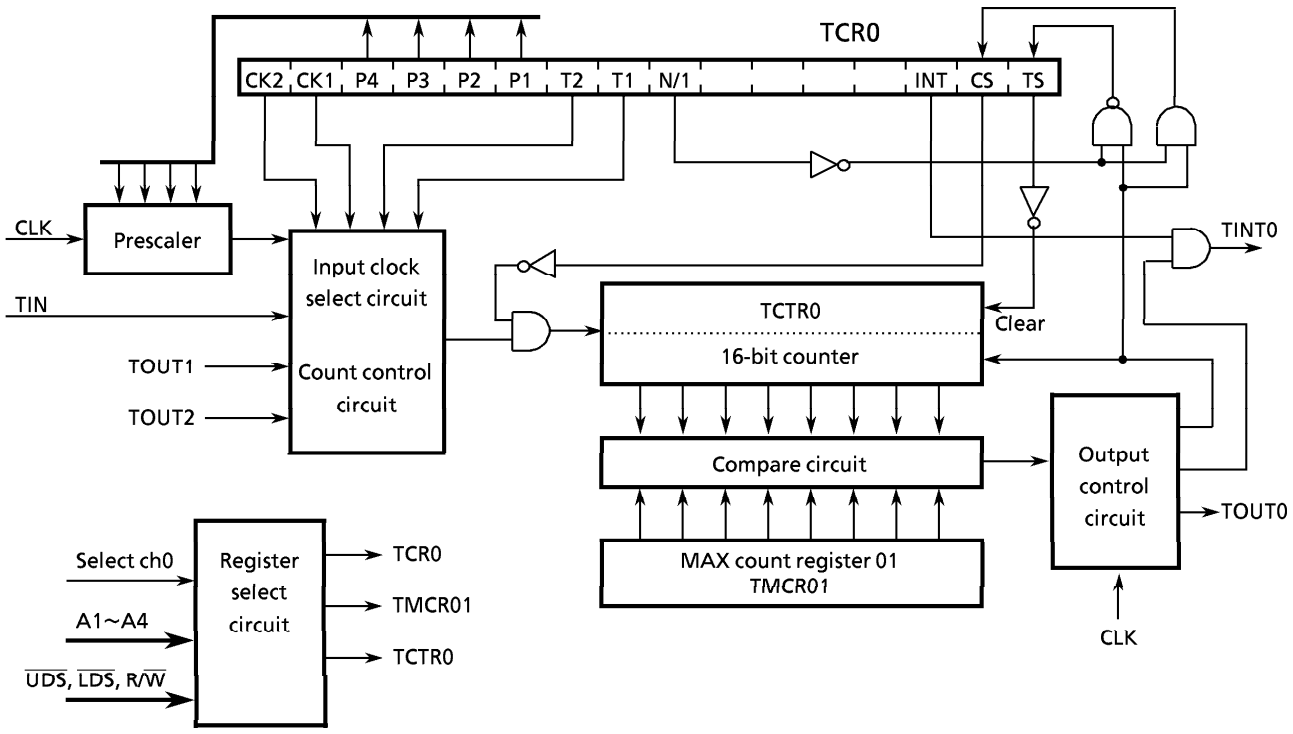


Figure 7.1 Timer Channel 0 Outline

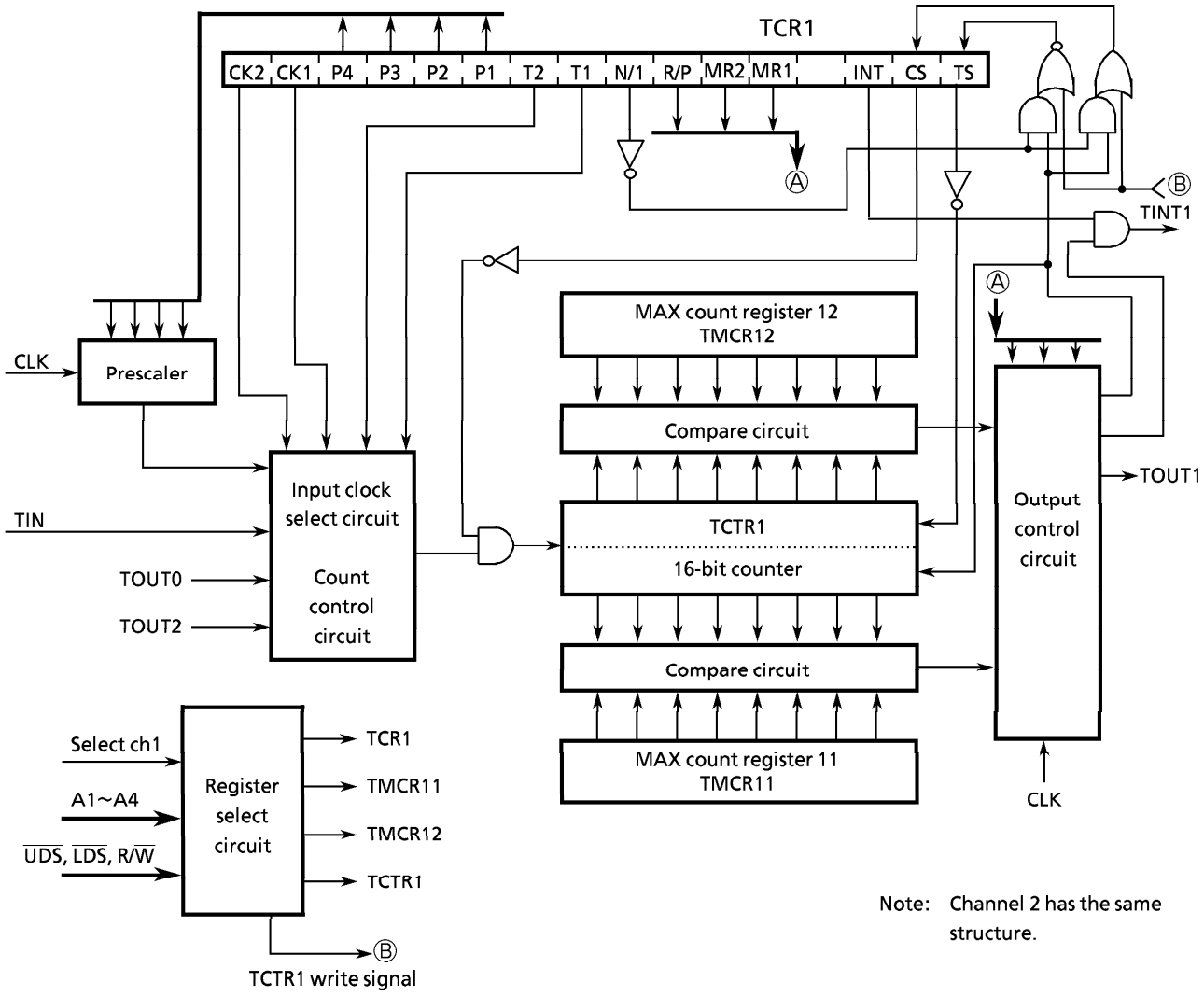


Figure 7.2 Timer Channel 1 Outline

7.2 Operational Description

Each channel internal counter is a 16-bit upcounter. Immediately the count value set in the 16-bit counter reaches the value set in the MAX count register, the channel generates a count match signal that clears the count value. The channel can generate an interrupt request signal within the channel.

7.2.1 Channel 0

Channel 0 has three registers: a (16-bit) control register, a (16-bit) count register, and a (16-bit) MAX count register. The internal counter is a 16-bit upcounter. Immediately the count value set in the 16-bit counter reaches the value set in the MAX count register, the channel generates a count match signal that clears the count value. The channel can generate an interrupt request signal within the channel. This channel can receive external clocks or external signals from external input pins (shared by channels 1 and 2), and can count up external events.

7.2.2 Channels 1 and 2

Channels 1 and 2 have four registers: MAX count register 1 and 2, a control register, and a count register. The internal counter is a 16-bit upcounter. Immediately the count value set in the 16-bit counter reaches the value set in the MAX count register, the channel generates a count match signal that clears the count value. The channel can generate an interrupt request signal within the channel. These channels have external input pins (shared by channel 0) and external output pins, and can count up external events and output any waveform.

7.2.3 Setting and Modifying Maximum Counts, Reading and Initializing Counts

The value written in the MAX count register specifies the maximum count.

Changing the MAX count register sets and modifies the maximum count. (The register can be rewritten during operation.)

The count value can be read from the count register during a count.

While the count register is normally used for reading only, the register can be used to initialize the timer to $0000 by writing any value to it. (channel 1, 2 only)

7.3 8-Bit Prescaler

The 8-bit prescaler divides the system clock by 2, 4, 8, 16, 32, 64, 128, or 256 to obtain a count clock. Use the control register to set the divider ratio.

7.4 Interrupt Generation

The control register controls interrupt generation mode.

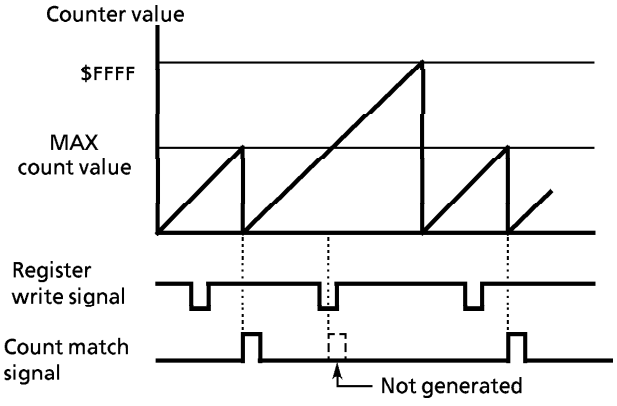
The timer ignores and does not store the count match signal during modes in which interrupt request signals are not generated. Therefore, even though modes are switched, an interrupt is not generated.

Interrupt generation timing: an interrupt request signal is generated when a count match signal is generated.

7.5 Register Reading/Writing

Reading a register during a count operation does not affect the count operation or timer output signal or interrupts. Note the following when writing to a register during a count operation.

- When the timer count a clock obtained by dividing the system clock by 2 using the 8-bit prescaler is used for timer count, or when a TIN input clock half the frequency of the system clock is used for timer count, the count match signal may not be generated and the counter value may reach $FFFF if the register write signal and the count match signal (timer count value and MAX count register value comparison match signal) are generated simultaneously in the same channel.



These conditions occur when a match signal and write signal occur simultaneously in the same channel. For example, the above situation occurs when to a channel 0-related register is written to at the same time as a channel 0 match signal is generated. However, this situation does not occur when to a channel 1 register is written to at the same time a channel 0 match signal is generated.

To avoid this occurring, set the 8-bit prescaler divider ratio to 4 or higher. When using the TIN input clock, set the clock to 1/4 the system clock frequency or lower. When using a clock obtained by dividing the system clock by two, increase the interrupt priority level by program so that data are written to the register immediately after the count match signal is generated rather than at the time of the next match signal generation.

7.6 Timer Operating Mode

7.6.1 TIN Pin Function

The external input pin (TIN) can be used for inputting the external clock, and trigger and control signals.

Signals input from TIN are all sampled at the rising edge of the system clock. Clocks and pulses input from TIN must have a pulse width of one system clock cycle or more.

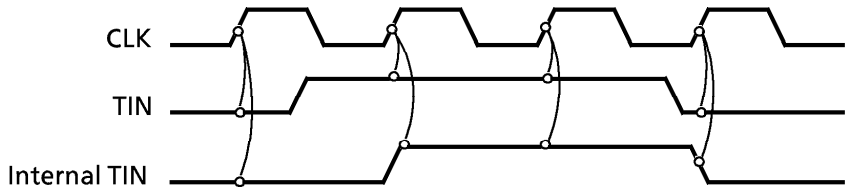


Figure 7.3 TIN Timing

7.6.1.1 External Clock

The external clock (TIN) can be used as the count clock.

7.6.1.2 Count Control Signal

Count control functions include the count start function and count wait function. These are selected by the count control register.

Count start function:

This function inputs a trigger signal from the external input pin (TIN). The count starts when the trigger signal is set to low (falling edge). (Any input after the second input is ignored.)

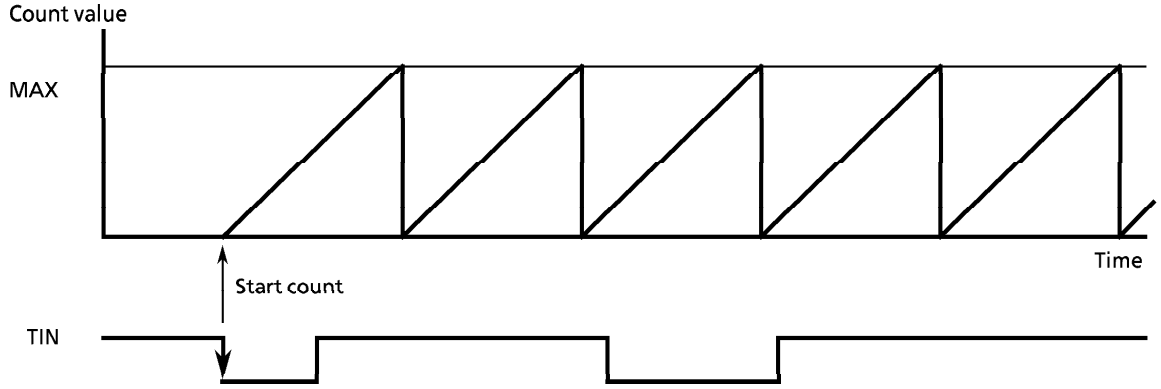


Figure 7.4 Count Start Function

7.7 Register Configuration

7.7.1 Timer Count Registers

The count values are read from these registers (TCTR1, TCTR2, and TCTR3). Only the upper byte or the lower byte can be read, too. While the timer count registers are normally read-only, writing any value to TCTR1 and TCTR2 initializes them to $0000. Writing any value to TCTR0 does not initialize it. Writing any value to TCTR0, TCTR1 and TCTR2 sets the timer control register CS bit to 1 and the TS bit to 0, clears the counter, and sets the count operation to the wait state.

Offset address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
$20C	C15	C14	C13	C12	C11	C10	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0	TCTR0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	
$22C	C15	C14	C13	C12	C11	C10	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0	TCTR1
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	
$24C	C15	C14	C13	C12	C11	C10	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0	TCTR2
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	

R : Read only

7.7.2 MAX Count Registers

The value written to the MAX count registers specifies the maximum count.

Offset address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
$204	M15	M14	M13	M12	M11	M10	M9	M8	M7	M6	M5	M4	M3	M2	M1	M0	TMCR01 MAX1
Reset	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	For ch0
$224	M15	M14	M13	M12	M11	M10	M9	M8	M7	M6	M5	M4	M3	M2	M1	M0	TMCR11 MAX1
	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	For ch1
$244	M15	M14	M13	M12	M11	M10	M9	M8	M7	M6	M5	M4	M3	M2	M1	M0	TMCR21 MAX1
	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	For ch2
$228	M15	M14	M13	M12	M11	M10	M9	M8	M7	M6	M5	M4	M3	M2	M1	M0	TMCR12 MAX2
	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	For ch1
$248	M15	M14	M13	M12	M11	M10	M9	M8	M7	M6	M5	M4	M3	M2	M1	M0	TMCR22 MAX2
	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	For ch2

7.7 Register Configuration

7.7.1 Timer Count Registers

The count values are read from these registers (TCTR1, TCTR2, and TCTR3). Only the upper byte or the lower byte can be read, too. While the timer count registers are normally read-only, writing any value to TCTR1 and TCTR2 initializes them to $0000. Writing any value to TCTR0 does not initialize it. Writing any value to TCTR0, TCTR1 and TCTR2 sets the timer control register CS bit to 1 and the TS bit to 0, clears the counter, and sets the count operation to the wait state.

Offset address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
$20C	C15	C14	C13	C12	C11	C10	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0	TCTR0
Reset	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	
$22C	C15	C14	C13	C12	C11	C10	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0	TCTR1
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	
$24C	C15	C14	C13	C12	C11	C10	C9	C8	C7	C6	C5	C4	C3	C2	C1	C0	TCTR2
	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	
	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	R	

R : Read only

7.7.2 MAX Count Registers

The value written to the MAX count registers specifies the maximum count.

Offset address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
$204	M15	M14	M13	M12	M11	M10	M9	M8	M7	M6	M5	M4	M3	M2	M1	M0	TMCR01 MAX1
Reset	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	For ch0
$224	M15	M14	M13	M12	M11	M10	M9	M8	M7	M6	M5	M4	M3	M2	M1	M0	TMCR11 MAX1
	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	For ch1
$244	M15	M14	M13	M12	M11	M10	M9	M8	M7	M6	M5	M4	M3	M2	M1	M0	TMCR21 MAX1
	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	For ch2
$228	M15	M14	M13	M12	M11	M10	M9	M8	M7	M6	M5	M4	M3	M2	M1	M0	TMCR12 MAX2
	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	For ch1
$248	M15	M14	M13	M12	M11	M10	M9	M8	M7	M6	M5	M4	M3	M2	M1	M0	TMCR22 MAX2
	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	–	For ch2

7.7.3 Timer Control Registers

The control registers control the count operation. The registers consist of the following bits.

Offset address	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
$200	CK2	CK1	P4	P3	P2	P1	T2	T1	N/1					INT	CS	TS	TCR0 For channel 0
Reset	0	0	0	0	0	0	0	0	0	1 R	0 R	1 R	0 R	0	1	0	

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
$220	CK2	CK1	P4	P3	P2	P1	T2	T1	N/1	R/P	MR2	MR1		INT	CS	TS	TCR1 For channel 1
	0	0	0	0	0	0	0	0	0	0	0	1	0 R	0	1	0	

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0	
$240	CK2	CK1	P4	P3	P2	P1	T2	T1	N/1	R/P	MR2	MR1		INT	CS	TS	TCR2 For channel 2
	0	0	0	0	0	0	0	0	0	0	0	1	0 R	0	1	0	

CK2, CK1 : Set the input clock for the counter.

CK2	CK1	Input clock
0	0	System clock (CLK)
0	1	External clock (TIN pin)
1	0	For cascade connection Channel 0 : Channel 1 counter output Channel 1 : Channel 0 counter output Channel 2 : Channel 0 counter output
1	1	For cascade connection Channel 0 : Channel 2 counter output Channel 1 : Channel 2 counter output Channel 2 : Channel 1 counter output

Note: Other timer outputs 1 and 2 are for cascade connection.
P4, P3, P2, P1: Set prescaler divider ratio.
Valid only when the system clock (CLK) is used as the input clock. (Not required if other input clock is used.)

P4	P3	P2	P1	Divider Ratio
0	0	0	1	1/2
0	0	1	0	1/4
0	0	1	1	1/8
0	1	0	0	1/16
0	1	0	1	1/32
0	1	1	0	1/64
0	1	1	1	1/128
1	X	X	X	1/256

X : Any value

T2, T1: Count mode selection

T2	T1	Function select
0	0	When an external clock is used or when count control is not used
1	0	Uses the count start function. (When repeatedly using this function, first change the setting to another value then return to this setting.)
1	1	Uses the count wait function.

N/1 : Repeat specification
0 : Ends count operation after one cycle. (One shot)
When the MAX count value is reached, the CS bit is set to 1 and the TS bit to 0, ending the count operation.
1 : Repeats count operation. (Repeat)

R/P : Output signal control (channels 1 and 2 only)
0 : Generates a pulse as the output signal. (pulse)
1 : Inverts the output level. (level inversion)

MR2, MR1 : MAX count register setting

MR2	MR1	MAX count register select
0	1	MAX1
1	0	MAX2
1	1	MAX1 and MAX2 alternately

INT : Interrupt request bit
0 : No interrupt request
1 : Interrupt request signal

CS : Count clock input control
0 : Inputs count clock to counter. (Starts count operation.)
1 : Does not input count clock to counter. (Stops count operation.)

TS : Timer operation setting
0 : Clears counter.
1 : Releases clear state.

Note : As channel 0 does not have a waveform output function, the channel 0 control register has no functions based on R/P, MR2, or MR1.

Count Wait Function

This function inputs the control signal from the external input pin (TIN). The count is performed while the control signal is low. When the control signal is high, the count is stopped. When the control signal goes low again, the count resumes. (If the counter is not cleared, the count continues from the previous value.)

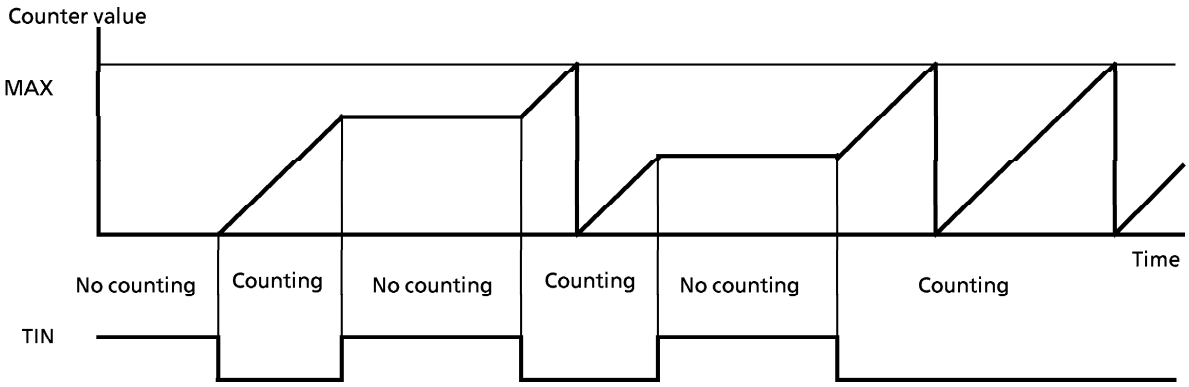


Figure 7.5 Count Wait Function

7.6.2 Count Clock Selection

The maximum count frequency is 8MHz. Therefore, use an external clock of 8MHz or less. As the prescaler divides the frequency by two or more, the maximum frequency is never exceeded when the system clock is used.

The count clock can be either the system clock, an external clock, or the output of another counter. Select the count clock when the counter is stopped.

7.6.3 TOUT Pin Function

The external output pin (TOUT) is used to output a pulse or square wave. The device can be set to either generate a pulse or invert the output level at a count match. By using two MAX counter registers, any square wave can be generated.

The signals output from TOUT (pulse generation or output level inversion timing) are synchronized with the rising edge of the system clock. If a pulse is generated, the pin, which is normally low, goes high for the duration of one clock at a count match. TOUT is low after a reset.

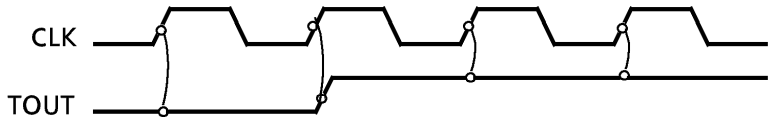


Figure 7.4 TOUT Timing

7.6.4 MAX Count Register Setting

Each channel has two MAX registers (henceforth, MAX1 and MAX2). Either register can be used individually or both registers can be used alternately. Control register settings specify how the MAX count registers are used.

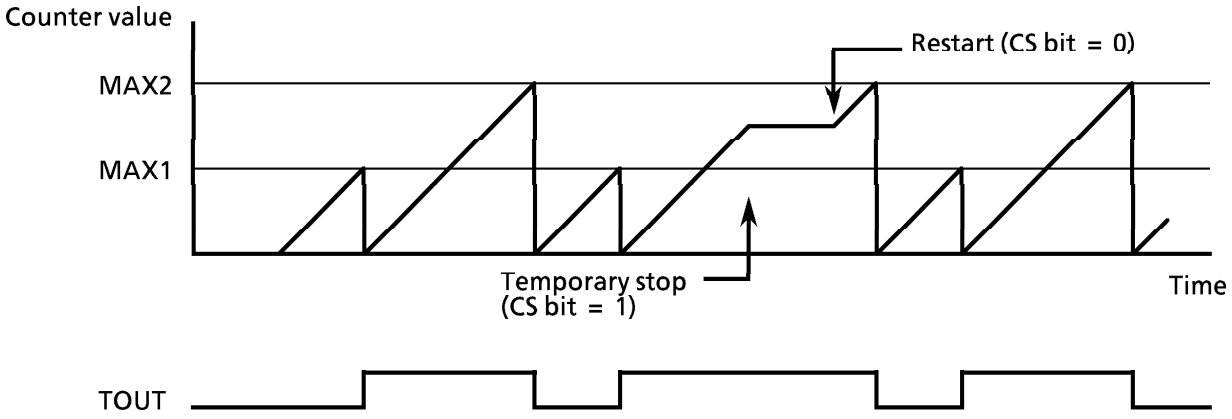
7.8 TOUT Initialization Function

With TMP68301A, use of timer output state or MAX count register is initialized by software.

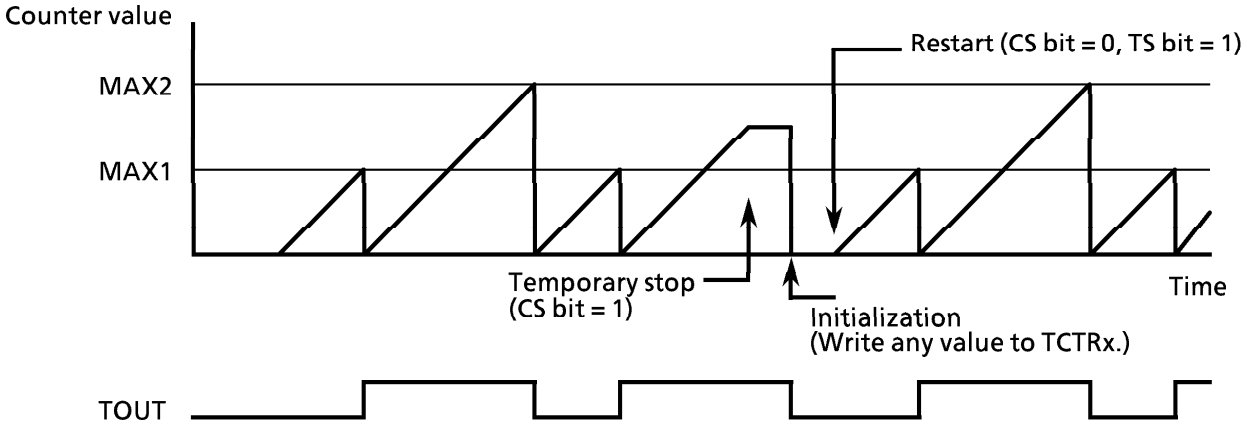
Initialize by writing any value to the timer count register. (Writing any value has no effect other than initialization.) After initialization, the timer output is low and operation is set to start from MAX count register 1. Channels 1 and 2 can be initialized separately.

When the count operation is temporarily stopped (by setting the CS bit of the timer count register to 1) then restarted (by setting the CS bit to 0), TMP68301A stores the TOUT state and the MAX count register being used. Therefore, counting restarts from the stopped state. (Operation example 1)

When the TMP68301A initialization function temporarily stops the count operation and performs initialization, the function sets TOUT to low and selects MAX count register 1 as the initial state, as shown in operation example 2.



Timer Operation Example 1



2Timer Operation Example 2

8. Internal Peripheral Circuit Register Map

8.1 Register map (1)

○ Address decoder

Offset address	15 ---------8	7 --------- 0	Offset address
$000	AMAR0	AAMR0	$001
$002		AACR0	$003
$004	AMAR1	AAMR1	$005
$006		AACR1	$007
$008		AACR2	$009
$00A		ATOR	$00B
$00C	ARELR		$00D
$00E			$00F

○ Interrupt controller

Offset address	15 ---------8	7 --------- 0	Offset address
$080		ICR0	$081
$082		ICR1	$083
$084		ICR2	$085
$086		ICR3	$087
$088		ICR4	$089
$08A		ICR5	$08B
$08C		ICR6	$08D
$08E		ICR7	$08F
$090		ICR8	$091
$092		ICR9	$093
$094	IMR		$095
$096	IPR		$097
$098	IISR		$099
$09A		IVNR	$09B
$09C		IEIR	$09D
$09E			$09F

○ Parallel interface

Offset address	15 ---------8	7 --------- 0	Offset address
$100	PDIR		$101
$102		PCR	$103
$104		PSR	$105
$106		PCMR	$107
$108		PMR	$109
$10A	PDR		$10B
$10C		PPR1	$10D
$10E		PPR2	$10F

○ Serial interface

Offset address	15 ---------8	7 --------- 0	Offset address
$180		SMR0	$181
$182		SCMR0	$183
$184		SBRR0	$185
$186		SSR0	$187
$188		SDR0	$189
$18A			$18B
$18C		SPR	$18D
$18E		SCR	$18F
$190		SMR1	$191
$192		SCMR1	$193
$194		SBRR1	$195
$196		SSR1	$197
$198		SDR1	$199
$19A			$19B
$19C			$19D
$19E			$19F
$1A0		SMR2	$1A1
$1A2		SCMR2	$1A3
$1A4		SBRR2	$1A5
$1A6		SSR2	$1A7
$1A8		SDR2	$1A9
$1AA			$1AB
$1AC			$1AD
$1AE			$1AF

○ 16-bit timer

Offset address	15 ---------8	7 --------- 0	Offset address
$200		TCR0	$201
$202			$203
$204		TMCR01	$205
$206			$207
$208			$209
$20A			$20B
$20C		TCTR0	$20D
$20E			$20F
$210			$211
$212			$213
$214			$215
$216			$217
$218			$219
$21A			$21B
$21C			$21D
$21E			$21F
$220		TCR1	$221
$222			$223
$224		TMCR11	$225
$226			$227
$228		TMCR12	$229
$22A			$22B
$22C		TCTR1	$22D
$22E			$22F
$230			$231
$232			$233
$234			$235
$236			$237
$238			$239
$23A			$23B
$23C			$23D
$23E			$23F
$240		TCR2	$241
$242			$243
$244		TMCR21	$245
$246			$247
$248		TMCR22	$249
$24A			$24B
$24C		TCTR2	$24D
$24E			$24F

8.2 Register map (2)

○ Address Decoder

Symbol	Name	Offset address	Data bus Upper bytes : 15 - 8 (even-numbered addresses) Lower bytes : 7 - 0 (odd-numbered addresses)							
			15/7	14/6	13/5	12/4	11/3	10/2	9/1	8/0
AMAR0	Memory Address Register 0	$000	A23	A22	A21	A20	A19	A18	A17	A16
			0	0	0	0	0	0	0	0
	For CS0		R/W							
AAMR0	Address Mask Register 0	$001	M21	M20	M19	M18	M17	M16	M15-M9	M8
			1	1	1	1	1	1	1	1
	For CS0		R/W							
AACR0	Area Control Register 0	$003			EN	ED	ID	WAIT		
			0	0	1	1	1	1	0	1
	For CS0		R		R/W					
AMAR1	Memory Address Register 1	$004	A23	A22	A21	A20	A19	A18	A17	A16
			–	–	–	–	–	–	–	–
	For CS1		R/W							
AAMR1	Address Mask Register 1	$005	M21	M20	M19	M18	M17	M16	M15-M9	M8
			–	–	–	–	–	–	–	–
	For CS1		R/W							
AACR1	Area Control Register 1	$007			EN	ED	ID	WAIT		
			0	0	0	1	1	0	0	0
	For CS1		R		R/W					
AACR2	Area Control Register 2	$009				ED	ID	WAIT		
			0	0	0	1	1	0	0	0
	For IACK cycle		R			R/W				
ATOR	Time Out Register	$00B					256	128	64	32
			0	0	0	0	1	0	0	0
	For BERR generation		R				R/W			
ARELR	Relocaton Register	$00C	A23	A22	A21	A20	A19	A18	A17	A16
			1	1	1	1	1	1	1	1
			R/W							
		$00D	A15	A14	A13	A12	A11	A10		
			1	1	1	1	1	1	0	0
	For internal registers		R/W						R	

○ Interrupt controller (1)

Symbol	Name	Offset address	Data bus Upper bytes : 15 - 8 (even-numbered addresses) Lower bytes : 7 - 0 (odd-numbered addresses)							
			15/7	14/6	13/5	12/4	11/3	10/2	9/1	8/0
ICR0	Interrupt Control Register 0 For external interrupt (INT0)	$081			V	R/F	L/E	Level		
			0	0	0	0	0	1	1	1
			R		R/W					
ICR1	Interrupt Control Register 1 For external interrupt (INT1)	$083			V	R/F	L/E	Level		
			0	0	0	0	0	1	1	1
			R		R/W					
ICR2	Interrupt Control Register 2 For external interrupt (INT2)	$085			V	R/F	L/E	Level		
			0	0	0	0	0	1	1	1
			R		R/W					
ICR3	Interrupt Control Register 3 For serial ch0 (INT3)	$087						Level		
			0	0	0	0	0	1	1	1
			R					R/W		
ICR4	Interrupt Control Register 4 For serial ch1 (INT4)	$089						Level		
			0	0	0	0	0	1	1	1
			R					R/W		
ICR5	Interrupt Control Register 5 For timer ch2 (INT5)	$08B						Level		
			0	0	0	0	0	1	1	1
			R					R/W		
ICR6	Interrupt Control Register 6 For parallel (INT6)	$08D						Level		
			0	0	0	0	0	1	1	1
			R					R/W		
ICR7	Interrupt Control Register 7 For timer ch0 (INT7)	$08F						Level		
			0	0	0	0	0	1	1	1
			R					R/W		
ICR8	Interrupt Control Register 8 For timer ch1 (INT8)	$091						Level		
			0	0	0	0	0	1	1	1
			R					R/W		
ICR9	Interrupt Control Register 9 For timer ch2 (INT9)	$093						Level		
			0	0	0	0	0	1	1	1
			R					R/W		

9. Electrical Characteristics

This section describes the electrical characteristics and timings of TMP68301.

9.1 Maximum Ratings

Parameter	Symbol	Rating	Unit
		TMP68301A	
Power supply voltage	Vcc	– 0.3 to + 6.5	V
Input voltage	Vin	– 0.3 to + 6.5	V
Operating temperature	Ta	0 to + 70	°C
Storage temperature	Tstg	– 55 to + 150	°C

While this device includes input protection circuits against high-voltage static electricity or damage from electric fields, avoid using a voltage higher than the maximum rating. Connect input pins not in use to GND or VCC.

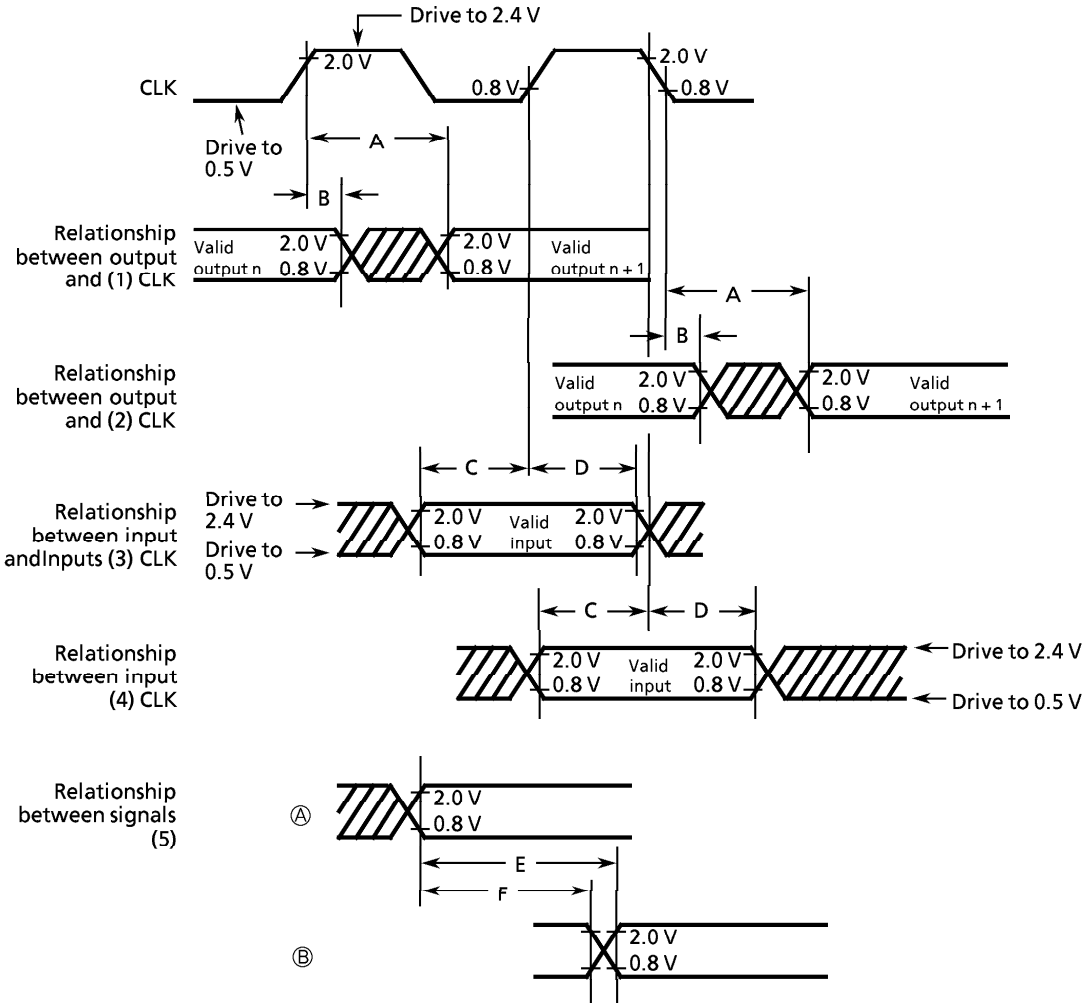
9.2 DC Characteristics

(GND = 0 V, Ta = 0 to 70 °C)

Parameter		Symbol	V_{CC} = 5.0 V ± 5 %		Unit
			Min	Max	
Supply voltage		V_{CC}	4.75	5.25	V
High-level input voltage	Except CLK CLK	V_{IH}	2.0 2.2	V_{CC} V_{CC}	V
Low-level input voltage	Except CLK CLK	V_{IL}	GND-0.3 GND-0.3	0.8 0.6	V
Input leakage current (5.25 V)	$\overline{BERR}$, $\overline{BGACK}$, $\overline{BR}$, $\overline{DTACK}$, CLK, $\overline{HALT}$, $\overline{RESET}$, NOR / $\overline{EMU}$, TIN, INT0 to INT2, RxD0 to RxD2	I_{IN}	– – –	2.5 2.5 20	μA
Tri-state (off state) input current	(2.4 V/0.4 V) $\overline{AS}$, A1 to A23, D0 to D15, FC0 to FC2, $\overline{LDS}$, $\overline{UDS}$, R / $\overline{W}$, IO0 to IO15	I_{TSI}	 – – –	 20 20 20	μA
High-level output voltage (IOH = – 400 μA)	$\overline{AS}$, A1 to A23, D0 to D15, $\overline{BG}$, FC0 to FC2, $\overline{LDS}$, $\overline{UDS}$, R / $\overline{W}$, IO0 to IO15, TOUT1, TOUT2, CS0, CS1, $\overline{RTS0}$, INT0 to INT2, TxD0 to TxD2	V_{OH}	– V_{CC}-0.75 V_{CC}-0.75 V_{CC}-0.75 V_{CC}-0.75	 – – – –	V
Low-level output voltage (IOL = 1.6 mA) (IOL = 3.2 mA) (IOL = 1.6 mA) (IOL = 5.3 mA)	 $\overline{HALT}$ A1 to A23, $\overline{BG}$, FC0 to FC2 $\overline{RESET}$ $\overline{AS}$, D0 to D15, $\overline{LDS}$, $\overline{UDS}$, R / $\overline{W}$, IO0 to IO15, TOUT1, TOUT2, CS0, CS1, $\overline{RTS0}$, $\overline{DTACK}$, $\overline{BERR}$, INT0 to INT2, TxD0 to TxD2	V_{OL}	 – – – –	 0.5 0.5 0.5 0.5	V
Current dissipation	f = 12.5 MHz f = 16.67 MHz	I_D	– –	90 100	mA
Power dissipation	f = 12.5 MHz f = 16.67 MHz	P_D	– –	0.473 0.525	W
Input capacitance (Vin = 0 V, Ta = 25 °C : Frequency = 1 MHz)*		C_{IN}	–	20.0	pF
Load capacitance	$\overline{HALT}$ Other pins	C_L	– –	70 130	pF

* : Input capacitance is periodically sampled rather than 100 % tested.

AC Electrical Characteristics (VCC = 5.0 V ± 5 %)



A : Maximum output delay

B : Minimum output hold time

C : Minimum input setup time

D : Minimum input hold time

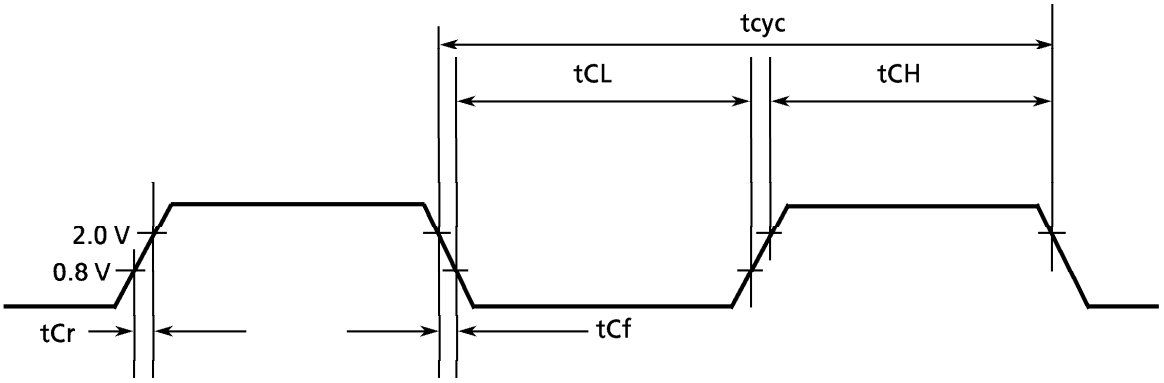
E : Signal Ⓐ valid - signal Ⓑ valid time

F : Signal Ⓐ valid - signal Ⓑ invalid time

9.3 AC Electrical Characteristics - Clock Timings

(VCC = 5.0 V ± 5 % See Figure 9.1. GND = 0 V, Ta = 0 to 70 °C)

Parameter	Symbol	12.5 MHz		16.67 MHz		Unit
		Min.	Max.	Min.	Max.	
Operating frequency	f	4.0	12.5	8.0	16.67	MHz
Cycle time	tcyc	80	250	60	125	ns
Clock pulse width	tCL tCH	35 35	125 125	27 27	62.5 62.5	ns
Rise and fall times	tCr tCf	– –	5 5	– –	5 5	ns



Note: Unless otherwise specified, timings are measured between a low voltage of 0.8 V and a high voltage of 2.0 V. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 V and 2.0 V.

Figure 9.1 Clock Input Timing

9.4 AC Electrical Characteristics - Read and Write Cycles (1/4)

(V_{CC} = 5.0 V ± 5 %, GND = 0 V, Ta = 0 to 70 °C; See figures 9.2 and 9.3.)

Number	Parameter	Symbol	12.5 MHz		16.67 MHz		Unit
			Min.	Max.	Min.	Max.	
1	Clock cycle	tCYC	80	250	60	125	ns
2	Clock width at low	tCL	35	125	27	62.5	ns
3	Clock width at high	tCH	35	125	27	62.5	ns
4	Clock fall time	tCf	–	5	–	5	ns
5	Clock rise time	tCr	–	5	–	5	ns
6	Clock low to address valid	tCLAV	–	50	–	50	ns
6A	Clock high to FC valid	tCHFCV	–	45	–	45	ns
7	Clock high to address, data bus high impedance (maximum)	tCHADZ	–	60	–	50	ns
8	Clock high to address, FC invalid (minimum)	tCHAFI	0	–	0	–	ns
9[1]	Clock high to $\overline{AS}$, $\overline{DS}$ low	tCHSL	3	40	3	40	ns
11[2]	Address valid to $\overline{AS}$, $\overline{DS}$ low (read)/Address valid to $\overline{AS}$ low (write)	tAVSL	15	–	15	–	ns
11A[2]	FC valid to $\overline{AS}$, $\overline{DS}$ low (read)/FC valid to $\overline{AS}$ low (write)	tFCVSL	60	–	30	–	ns
12[1]	Clock low to $\overline{AS}$, $\overline{DS}$ high	tCLSH	–	40	–	40	ns
13[2]	$\overline{AS}$, $\overline{DS}$ high to address/FC invalid	tSHAFI	20	–	10	–	ns
14[2]	$\overline{AS}$, $\overline{DS}$ width low (read)/$\overline{AS}$ width low (write)	tSL	160	–	120	–	ns
14A[2]	$\overline{DS}$ width low (write)	tDSL	80	–	60	–	ns
15[2]	$\overline{AS}$, $\overline{DS}$ width high	tSH	65	–	60	–	ns

9.4 AC Electrical Characteristics - Read and Write Cycles (2/4)

(V_{CC} = 5.0 V ± 5 %, GND = 0 V, Ta = 0 to 70 °C; See figures 9.2 and 9.3)

Number	Parameter	Symbol	12.5 MHz		16.67 MHz		Unit
			Min.	Max.	Min.	Max.	
16	Clock high to control bus high impedance	tCHCZ	–	60	–	50	ns
17[2]	$\overline{AS}$, $\overline{DS}$ high to R/$\overline{W}$ high (read)	tSHRH	20	–	10	–	ns
18[1]	Clock high to R/$\overline{W}$ high	tCHRH	0	40	0	40	ns
20[1]	Clock to R/$\overline{W}$ low (write)	tCHRL	0	40	0	40	ns
20A[2,6]	$\overline{AS}$ low to R/$\overline{W}$ valid (write)	tASRV	–	10	–	10	ns
21[2]	Address valid to R/$\overline{W}$ low (write), FC valid to R/$\overline{W}$ low (write)	tAVRL	0	–	0	–	ns
21A[2]	FC valid to R/$\overline{W}$ low (write)	tFCVRL	30	–	20	–	ns
22[2]	R/$\overline{W}$ low to $\overline{DS}$ low (write)	tRLSL	30	–	20	–	ns
23	Clock low to data out valid (write)	tCLDO	–	50	–	50	ns
25[2]	$\overline{AS}$, $\overline{DS}$ high to data out invalid (write)	tSHDOI	20	–	15	–	ns
26[2]	Data out valid to $\overline{DS}$ low (write)	tDOSL	20	–	15	–	ns
27[5]	Data in to clock low (setup time on read)	tDICL	10	–	7	–	ns
28[2]	$\overline{AS}$, $\overline{DS}$ high to $\overline{DTACK}$ high	tSHDAH	0	150	0	110	ns
29	$\overline{AS}$, $\overline{DS}$ negate to data invalid (hold time for read)	tSHDII	0	–	0	–	ns

9.4 AC Electrical Characteristics - Read and Write Cycles (3/4)

(V_{CC} = 5.0 V ± 5 %, GND = 0 V, Ta = 0 to 70 °C; See figures 9.2 and 9.3.)

Number	Parameter	Symbol	12.5 MHz		16.67 MHz		Unit
					Min.	Max.	
30	$\overline{AS}$, $\overline{DS}$ high to $\overline{BERR}$ high	tSHBEH	0	–	0	–	ns
31[2,5]	$\overline{DTACK}$ low setup time (input data)	tDALDI	–	50	–	40	ns
32	$\overline{HALT}$ and $\overline{RESET}$ input transition	tRHr, f	0	200	0	150	ns
33	Clock high to $\overline{BG}$ low	tCHGL	–	40	–	40	ns
34	Clock high to $\overline{BG}$ high	tCHGH	–	40	–	40	ns
35	$\overline{BR}$ low to $\overline{BG}$ low	tBRLGL	1.5	3.5	1.5	3.5	Clk. Per.
36[7]	$\overline{BR}$ high to $\overline{BG}$ high	tBRHGH	1.5	3.5	1.5	3.5	Clk. Per.
37	$\overline{BGACK}$ low to $\overline{BG}$ high	tGALGH	1.5	3.5	1.5	3.5	Clk. Per.
37A[8]	$\overline{BGACK}$ low to $\overline{BR}$ high	tGALBRH	20	1.5 Clocks	10	1.5 Clocks	ns
38	$\overline{BG}$ low to control, address, data bus, high impedance ($\overline{AS}$ high)	tGLZ	–	60	–	50	ns
39	$\overline{BG}$ width high	tGH	1.5	–	1.5	–	Clk. Per.

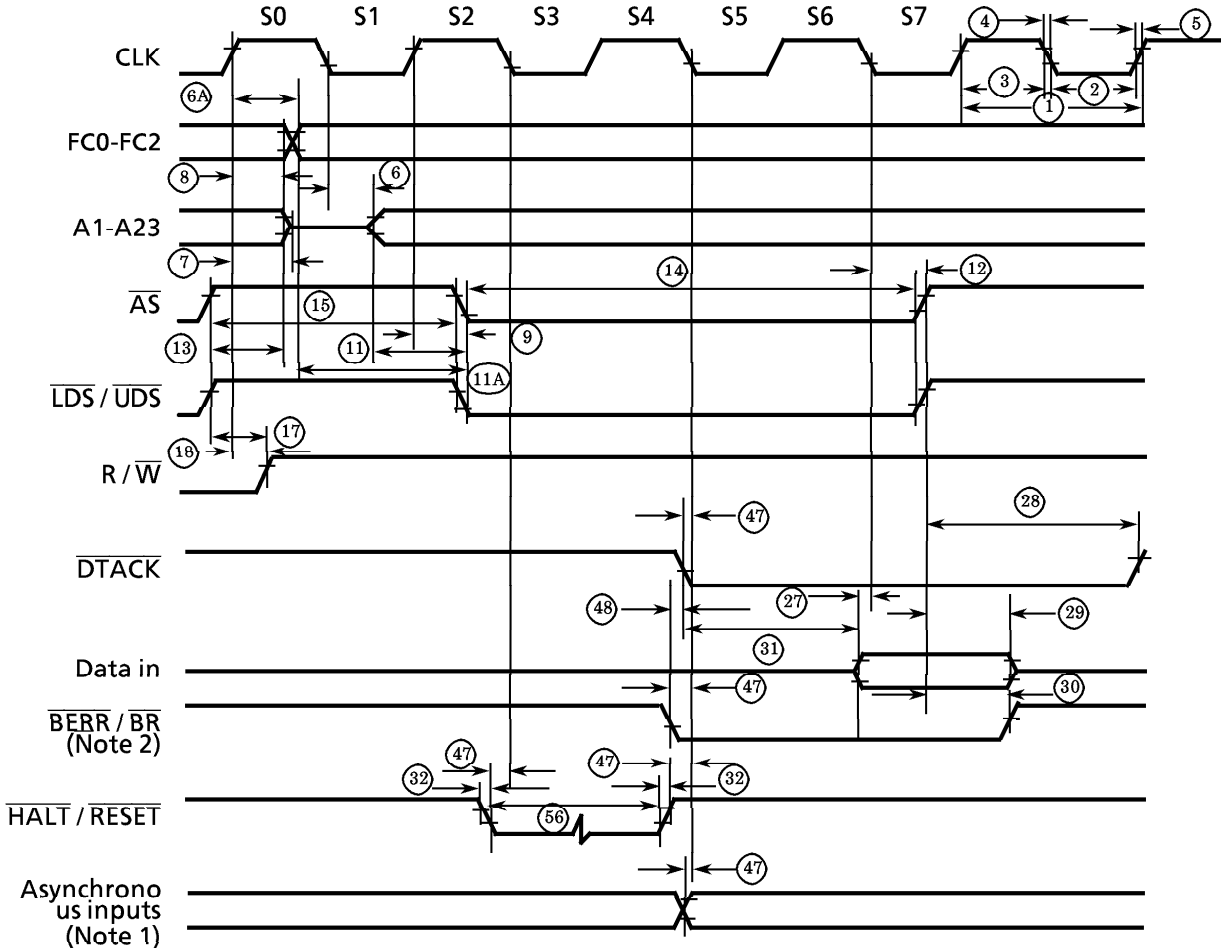
9.4 AC Electrical Characteristics - Read and Write Cycles (4/4)

(V_{CC} = 5.0 V ± 5 %, GND = 0 V, Ta = 0 to 70 °C; See figures 9.2 and 9.3.)

Number	Parameter	Symbol	12.5 MHz		16.67 MHz		Unit
			Min.	Max.	Min.	Max.	
46	$\overline{BGACK}$ width low	tGAL	1.5	–	1.5		Clk. Per.
47[5]	Asynchronous input setup time	tASI	10	–	10	–	ns
48[2,3]	$\overline{BERR}$ low to $\overline{DTACK}$ low	tBELDAL	20	–	10	–	ns
53	Clock high to data out invalid	tCHDOI	0	–	0	–	ns
55	R/$\overline{W}$ low to data bus drive	tRLDBD	10	–	0	–	ns
56[4]	$\overline{HALT}$ / $\overline{RESET}$ pulse width	tHRPW	10	–	10	–	Clk. Per.
57	$\overline{BGACK}$ high to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ drive	tGASD	1.5	–	1.5	–	Clk. Per.
58[7]	$\overline{BR}$ high to control bus drive	tRHSD	1.5	–	1.5	–	Clk. Per.

1. For a loading capacitance of less than or equal to 50 picofarads, subtract five nanoseconds from the value given in the maximum columns.
2. Actual value depends on clock cycle.
3. If #47 is satisfied for both $\overline{DTACK}$ and $\overline{BERR}$, #48 may be 0 nanoseconds.
4. At power on, hold $\overline{RESET}$ and $\overline{HALT}$ into low status for 1 to 100 microseconds to stabilize MPU circuits. See #56 for the minimum reset times following power on.
5. If the asynchronous setup time (#47) requirements are satisfied, the $\overline{DTACK}$ low-to-data setup time (#31) requirement can be ignored. The data must only satisfy the data-in to clock-low setup time (#27) requirement for the following cycle.
6. When $\overline{AS}$ and R/$\overline{W}$ are equally loaded (±20 %), subtract ten nanoseconds (8-12.5 MHz) or five nanoseconds (16.67 MHz) from the tASRV maximum value.
7. The processor will negate $\overline{BG}$ and begin driving the bus again if external arbitration logic negates $\overline{BR}$ before asserting $\overline{BGACK}$.
8. To guarantee proper operation, the minimum value must be met. If the maximum value is exceeded, $\overline{BG}$ may be reasserted.

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Functional descriptions and their related device operation diagrams are given elsewhere.

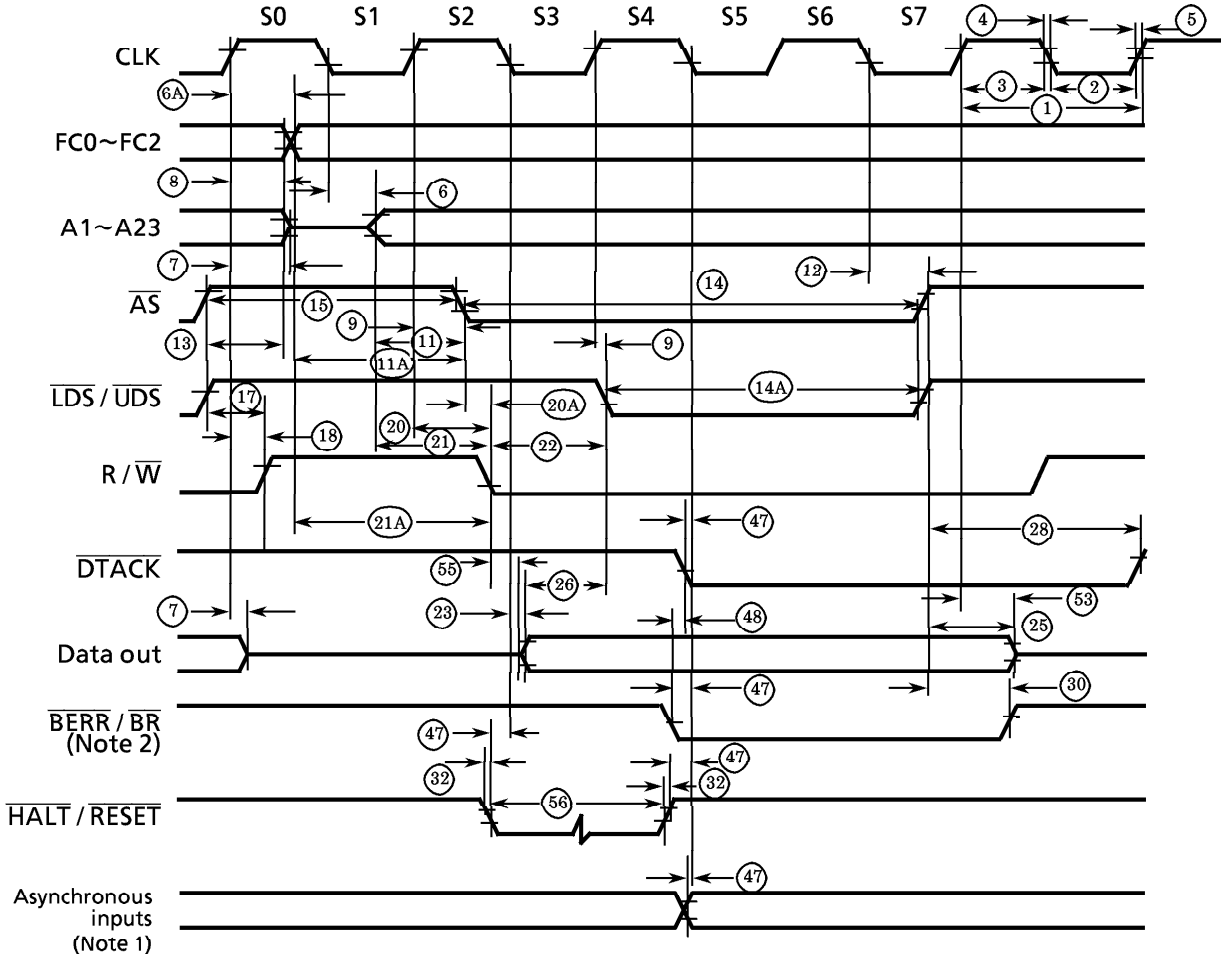


Notes :

1. Asynchronous input $\overline{\text{BGACK}}$ is detected at the clock's falling edge.
2. It is necessary to assert at this timing only if $\overline{\text{BR}}$ is recognized at the end of this bus cycle.
3. Unless otherwise specified, timings are measured between a low voltage of 0.8 V and a high voltage of 2.0 V. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 V and 2.0 V.

Figure 9.2 Read Cycle Timing Diagram

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Functional descriptions and their related device operation diagrams are given elsewhere.



Notes:

1. Unless otherwise specified, timings are measured between a low voltage of 0.8 V and a high voltage of 2.0 V. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 V and 2.0 V.
2. Because of loading variations, R/$\overline{W}$ may become valid after $\overline{AS}$ even though both are asserted at the rising edge of S2 (#20A).

Figure 9.3 Write Cycle Timing Diagram

9.4 AC Electrical Characteristics - Read and Write Cycles (1/4)

(V_{CC} = 5.0 V ± 5 %, GND = 0 V, Ta = 0 to 70 °C; See figures 9.2 and 9.3.)

Number	Parameter	Symbol	12.5 MHz		16.67 MHz		Unit
			Min.	Max.	Min.	Max.	
1	Clock cycle	tCYC	80	250	60	125	ns
2	Clock width at low	tCL	35	125	27	62.5	ns
3	Clock width at high	tCH	35	125	27	62.5	ns
4	Clock fall time	tCf	–	5	–	5	ns
5	Clock rise time	tCr	–	5	–	5	ns
6	Clock low to address valid	tCLAV	–	50	–	50	ns
6A	Clock high to FC valid	tCHFCV	–	45	–	45	ns
7	Clock high to address, data bus high impedance (maximum)	tCHADZ	–	60	–	50	ns
8	Clock high to address, FC invalid (minimum)	tCHAFI	0	–	0	–	ns
9[1]	Clock high to $\overline{AS}$, $\overline{DS}$ low	tCHSL	3	40	3	40	ns
11[2]	Address valid to $\overline{AS}$, $\overline{DS}$ low (read)/Address valid to $\overline{AS}$ low (write)	tAVSL	15	–	15	–	ns
11A[2]	FC valid to $\overline{AS}$, $\overline{DS}$ low (read)/FC valid to $\overline{AS}$ low (write)	tFCVSL	60	–	30	–	ns
12[1]	Clock low to $\overline{AS}$, $\overline{DS}$ high	tCLSH	–	40	–	40	ns
13[2]	$\overline{AS}$, $\overline{DS}$ high to address/FC invalid	tSHAFI	20	–	10	–	ns
14[2]	$\overline{AS}$, $\overline{DS}$ width low (read)/$\overline{AS}$ width low (write)	tSL	160	–	120	–	ns
14A[2]	$\overline{DS}$ width low (write)	tDSL	80	–	60	–	ns
15[2]	$\overline{AS}$, $\overline{DS}$ width high	tSH	65	–	60	–	ns

9.4 AC Electrical Characteristics - Read and Write Cycles (2/4)

(V_{CC} = 5.0 V ± 5 %, GND = 0 V, Ta = 0 to 70 °C; See figures 9.2 and 9.3)

Number	Parameter	Symbol	12.5 MHz		16.67 MHz		Unit
			Min.	Max.	Min.	Max.	
16	Clock high to control bus high impedance	tCHCZ	–	60	–	50	ns
17[2]	$\overline{AS}$, $\overline{DS}$ high to R/$\overline{W}$ high (read)	tSHRH	20	–	10	–	ns
18[1]	Clock high to R/$\overline{W}$ high	tCHRH	0	40	0	40	ns
20[1]	Clock to R/$\overline{W}$ low (write)	tCHRL	0	40	0	40	ns
20A[2,6]	$\overline{AS}$ low to R/$\overline{W}$ valid (write)	tASRV	–	10	–	10	ns
21[2]	Address valid to R/$\overline{W}$ low (write), FC valid to R/$\overline{W}$ low (write)	tAVRL	0	–	0	–	ns
21A[2]	FC valid to R/$\overline{W}$ low (write)	tFCVRL	30	–	20	–	ns
22[2]	R/$\overline{W}$ low to $\overline{DS}$ low (write)	tRLSL	30	–	20	–	ns
23	Clock low to data out valid (write)	tCLDO	–	50	–	50	ns
25[2]	$\overline{AS}$, $\overline{DS}$ high to data out invalid (write)	tSHDOI	20	–	15	–	ns
26[2]	Data out valid to $\overline{DS}$ low (write)	tDOSL	20	–	15	–	ns
27[5]	Data in to clock low (setup time on read)	tDICL	10	–	7	–	ns
28[2]	$\overline{AS}$, $\overline{DS}$ high to $\overline{DTACK}$ high	tSHDAH	0	150	0	110	ns
29	$\overline{AS}$, $\overline{DS}$ negate to data invalid (hold time for read)	tSHDII	0	–	0	–	ns

9.4 AC Electrical Characteristics - Read and Write Cycles (3/4)

(V_{CC} = 5.0 V ± 5 %, GND = 0 V, Ta = 0 to 70 °C; See figures 9.2 and 9.3.)

Number	Parameter	Symbol	12.5 MHz		16.67 MHz		Unit
					Min.	Max.	
30	$\overline{AS}$, $\overline{DS}$ high to $\overline{BERR}$ high	tSHBEH	0	–	0	–	ns
31[2,5]	$\overline{DTACK}$ low setup time (input data)	tDALDI	–	50	–	40	ns
32	$\overline{HALT}$ and $\overline{RESET}$ input transition	tRHr, f	0	200	0	150	ns
33	Clock high to $\overline{BG}$ low	tCHGL	–	40	–	40	ns
34	Clock high to $\overline{BG}$ high	tCHGH	–	40	–	40	ns
35	$\overline{BR}$ low to $\overline{BG}$ low	tBRLGL	1.5	3.5	1.5	3.5	Clk. Per.
36[7]	$\overline{BR}$ high to $\overline{BG}$ high	tBRHGH	1.5	3.5	1.5	3.5	Clk. Per.
37	$\overline{BGACK}$ low to $\overline{BG}$ high	tGALGH	1.5	3.5	1.5	3.5	Clk. Per.
37A[8]	$\overline{BGACK}$ low to $\overline{BR}$ high	tGALBRH	20	1.5 Clocks	10	1.5 Clocks	ns
38	$\overline{BG}$ low to control, address, data bus, high impedance ($\overline{AS}$ high)	tGLZ	–	60	–	50	ns
39	$\overline{BG}$ width high	tGH	1.5	–	1.5	–	Clk. Per.

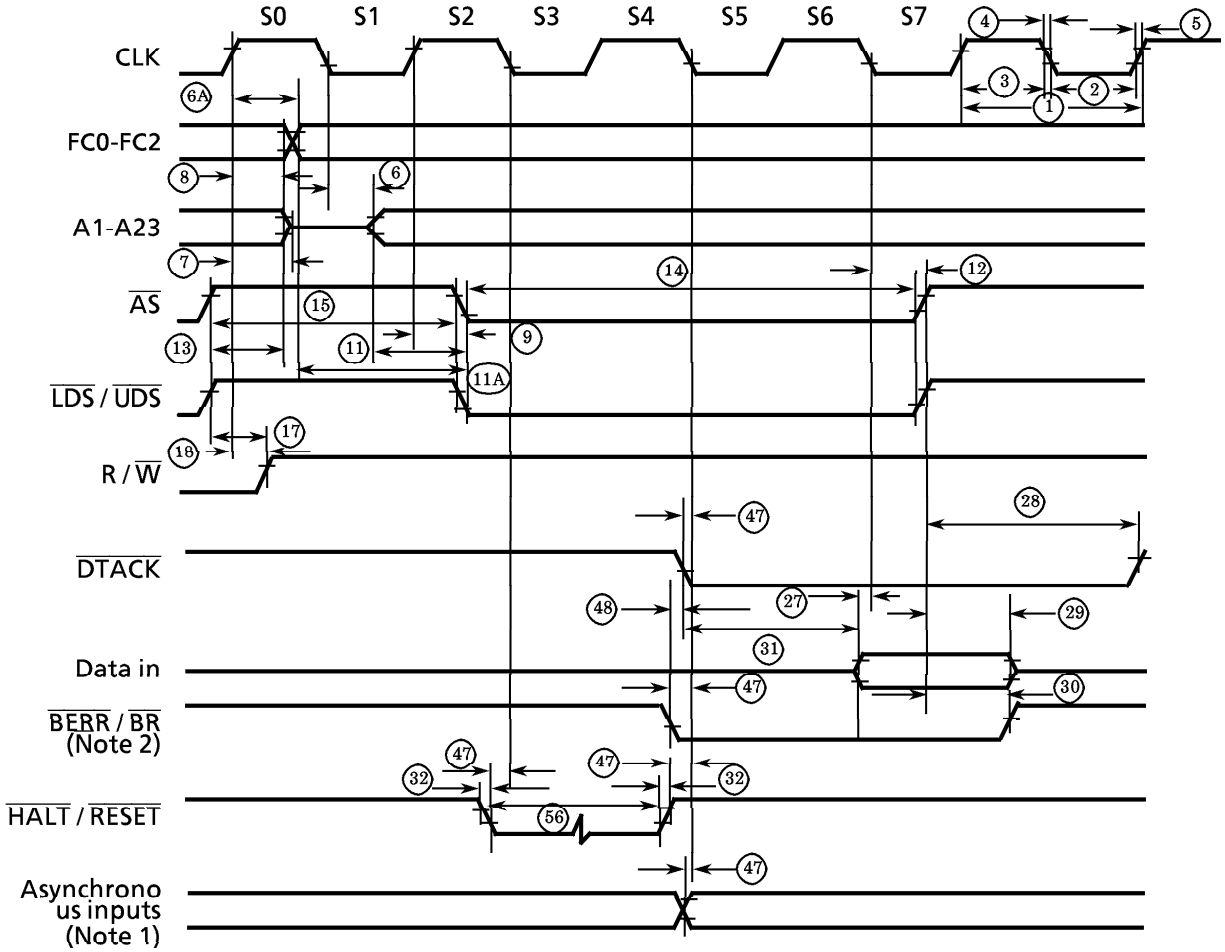
9.4 AC Electrical Characteristics - Read and Write Cycles (4/4)

(V_{CC} = 5.0 V ± 5 %, GND = 0 V, Ta = 0 to 70 °C; See figures 9.2 and 9.3.)

Number	Parameter	Symbol	12.5 MHz		16.67 MHz		Unit
			Min.	Max.	Min.	Max.	
46	$\overline{BGACK}$ width low	tGAL	1.5	–	1.5		Clk. Per.
47[5]	Asynchronous input setup time	tASI	10	–	10	–	ns
48[2,3]	$\overline{BERR}$ low to $\overline{DTACK}$ low	tBELDAL	20	–	10	–	ns
53	Clock high to data out invalid	tCHDOI	0	–	0	–	ns
55	R/$\overline{W}$ low to data bus drive	tRLDBD	10	–	0	–	ns
56[4]	$\overline{HALT}$ / $\overline{RESET}$ pulse width	tHRPW	10	–	10	–	Clk. Per.
57	$\overline{BGACK}$ high to $\overline{AS}$, $\overline{DS}$, R/$\overline{W}$ drive	tGASD	1.5	–	1.5	–	Clk. Per.
58[7]	$\overline{BR}$ high to control bus drive	tRHSD	1.5	–	1.5	–	Clk. Per.

1. For a loading capacitance of less than or equal to 50 picofarads, subtract five nanoseconds from the value given in the maximum columns.
2. Actual value depends on clock cycle.
3. If #47 is satisfied for both $\overline{DTACK}$ and $\overline{BERR}$, #48 may be 0 nanoseconds.
4. At power on, hold $\overline{RESET}$ and $\overline{HALT}$ into low status for 1 to 100 microseconds to stabilize MPU circuits. See #56 for the minimum reset times following power on.
5. If the asynchronous setup time (#47) requirements are satisfied, the $\overline{DTACK}$ low-to-data setup time (#31) requirement can be ignored. The data must only satisfy the data-in to clock-low setup time (#27) requirement for the following cycle.
6. When $\overline{AS}$ and R/$\overline{W}$ are equally loaded (±20 %), subtract ten nanoseconds (8-12.5 MHz) or five nanoseconds (16.67 MHz) from the tASRV maximum value.
7. The processor will negate $\overline{BG}$ and begin driving the bus again if external arbitration logic negates $\overline{BR}$ before asserting $\overline{BGACK}$.
8. To guarantee proper operation, the minimum value must be met. If the maximum value is exceeded, $\overline{BG}$ may be reasserted.

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Functional descriptions and their related device operation diagrams are given elsewhere.

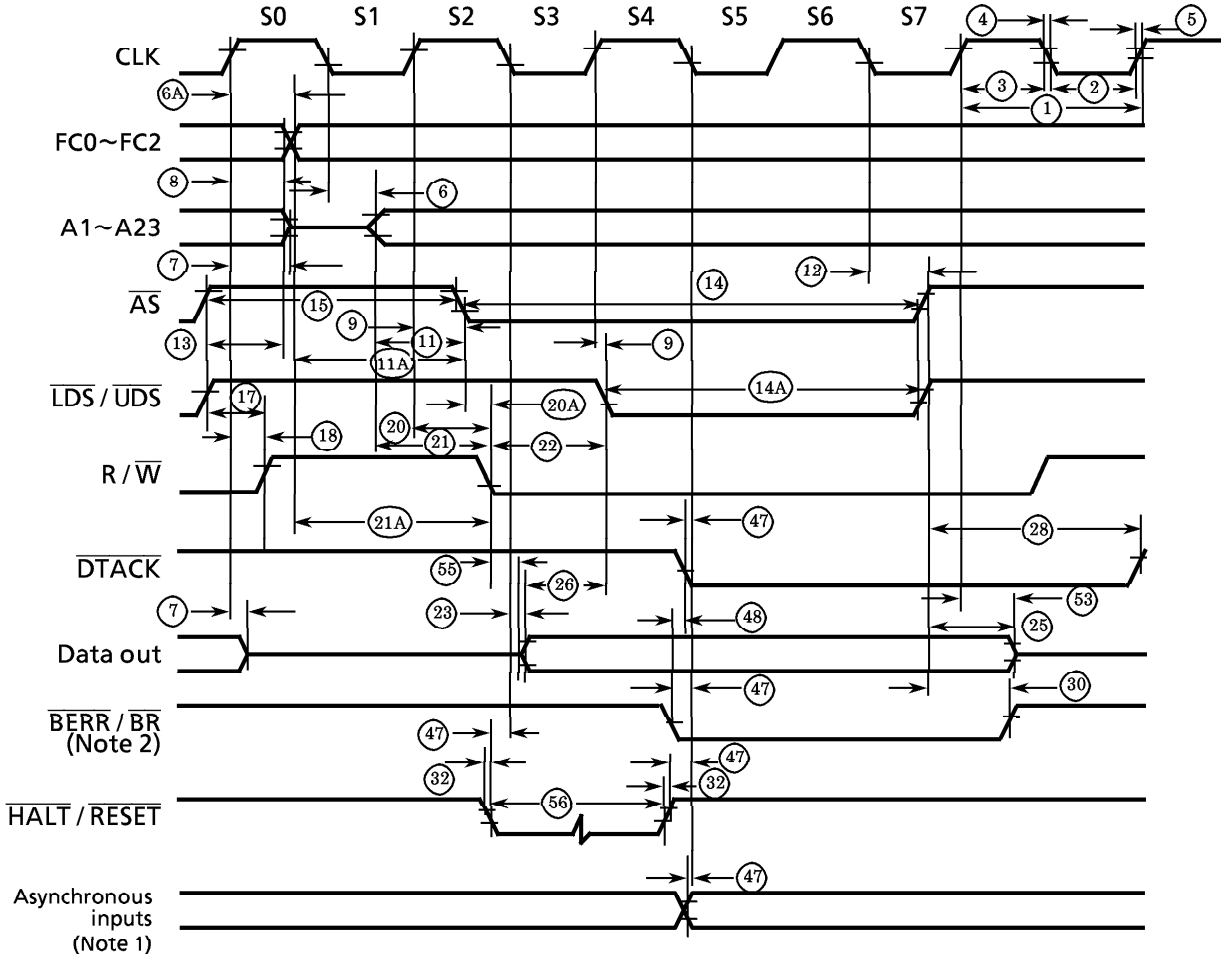


Notes :

1. Asynchronous input $\overline{\text{BGACK}}$ is detected at the clock's falling edge.
2. It is necessary to assert at this timing only if $\overline{\text{BR}}$ is recognized at the end of this bus cycle.
3. Unless otherwise specified, timings are measured between a low voltage of 0.8 V and a high voltage of 2.0 V. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 V and 2.0 V.

Figure 9.2 Read Cycle Timing Diagram

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Functional descriptions and their related device operation diagrams are given elsewhere.



Notes:

1. Unless otherwise specified, timings are measured between a low voltage of 0.8 V and a high voltage of 2.0 V. The voltage swing through this range should start outside and pass through the range such that the rise or fall will be linear between 0.8 V and 2.0 V.
2. Because of loading variations, R/$\overline{W}$ may become valid after $\overline{AS}$ even though both are asserted at the rising edge of S2 (#20A).

Figure 9.3 Write Cycle Timing Diagram

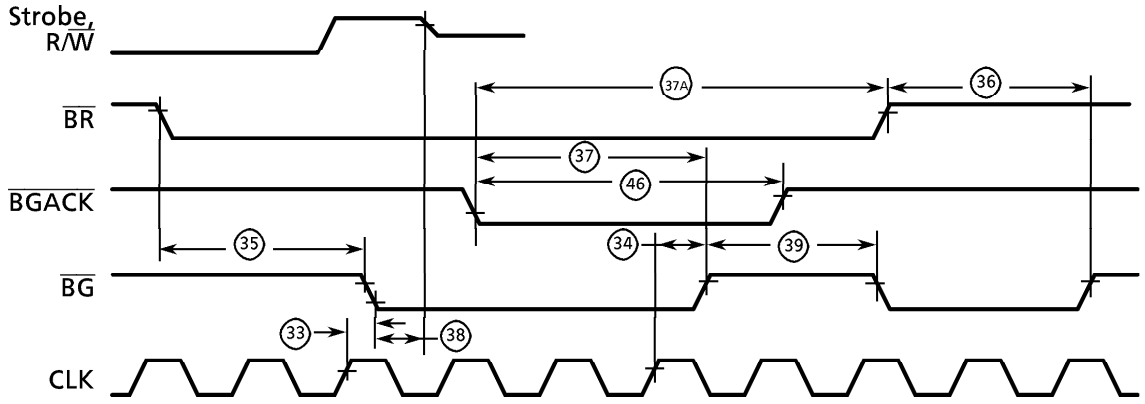
9.5 AC Electrical Characteristics - Bus Arbitration

(V_{CC} = 5.0 V ± 5 %, GND = 0 V, Ta = 0 to 70 °C; Figure 9.4)

Number	Parameter	Symbol	12.5 MHz		16.67 MHz		Unit
			Min.	Max.	Min.	Max.	
7	Clock high to address, data bus high impedance	tCHADZ	–	60	–	50	ns
16	Clock high to control bus high impedance	tCHCZ	–	60	–	50	ns
33	Clock high to $\overline{BG}$ low	tCHGL	–	40	–	40	ns
34	Clock high to $\overline{BG}$ high	tCHGH	–	40	–	40	ns
35	$\overline{BR}$ low to $\overline{BG}$ low	tBRLGL	1.5	3.5	1.5	3.5	Clk. Per.
36[1]	$\overline{BR}$ high to $\overline{BG}$ high	tBKHGH	1.5	3.5	1.5	3.5	Clk. Per.
37	$\overline{BGACK}$ low to BG high	tGALGH	1.5	3.5	1.5	3.5	Clk. Per.
37A[2]	$\overline{BGACK}$ low to BR high	tGALBRH	20	1.5 Clocks	10	1.5 Clocks	ns
38	$\overline{BG}$ low to control, address, data bus high impedance ($\overline{AS}$ high)	tGLZ	–	60	–		ns
39	$\overline{BG}$ width high	tGH	1.5	–	1.5		Clk. Per.
46	$\overline{BGACK}$ width low	tGAL	1.5	–	1.5		Clk. Per.
47	Asynchronous input setup time	tASI	10	–	5		ns
57	$\overline{BGACK}$ high to control bus drive	tGABD	1.5	–	1.5		Clk. Per.
58[1]	$\overline{BG}$ high to control bus drive	tGHBD	1.5	–	1.5		Clk. Per.

1. The processor will negate $\overline{BG}$ and begin driving the bus again if external arbitration logic negates $\overline{BR}$ before asserting $\overline{BGACK}$.
2. To guarantee proper operation, the minimum value must be met. If the maximum value is exceeded, $\overline{BG}$ may be reasserted.

These waveforms should only be referenced in regard to the edge-to-edge measurement of the timing specifications. They are not intended as a functional description of the input and output signals. Functional descriptions and their related device operation diagrams are given elsewhere.



Note: Asynchronous inputs $\overline{BERR}$, $\overline{BGACK}$, $\overline{BR}$, and $\overline{DTACK}$ are detected at the clock's falling edge.

Figure 9.4 Bus Arbitration Timing Diagram

When the external bus master accesses the registers of internal devices, addresses, data, and control signals must be input in accordance with the read/write cycle timing.

9.6 AC Electrical Characteristics - Peripherals

(V_{CC} = 5.0 V ± 5 %, GND = 0 V, Ta = 0 to 70 °C; See figures 9.5 - 9.11)

Number	Parameter	Symbol	12.5 MHz		16.67 MHz		Unit
			Min.	Max.	Min.	Max.	
47	Asynchronous input setup time	tASI	10	—	10	—	ns
101	Clock to $\overline{CS}$, $\overline{IACK}$	tCDS	—	50	—	50	ns
102	Clock high to TOUT	tCHTO	—	40	—	40	ns
103	BCLK cycle	tBCYC	125	—	125	—	ns
104	BCLK width low	tBCL	55	—	55	—	ns
105	BCLK width high	tBCH	55	—	55	—	ns
106	BCLK rise time	tBCr	—	10	—	10	ns
107	BCLK fall time	tBCf	—	10	—	10	ns
108	$\overline{LDS}$ high to $\overline{DTR}$, $\overline{RTS}$	tDSMC	—	140	—	140	ns
109	$\overline{DSR}$ to $\overline{LDS}$ low	tMCDS	50	—	50	—	ns
110	$\overline{DS}$ high to I/O output	tDSIO	—	60	—	60	ns
111	I/O data in to CLK low (setup time on I/O input)	tIOsCL	50	—	50	—	ns
112	I/O data in to clock low (hold time on I/O input)	tIOhCL	50	—	50	—	ns
113	DATA delay time	tDDA	—	60	—	60	ns
114	Clock high to $\overline{DSTB}$	tCHST	—	60	—	60	ns
115	PRIME delay time	tDPR	—	60	—	60	ns
116	DATA setup time for $\overline{DSTB}$	tDAsST	20	—	20	—	ns
117	DATA hold time for $\overline{DSTB}$	tDAhST	60	—	60	—	ns
118	$\overline{DSTB}$ width low	tSTL	70	—	70	—	ns
119	BUSY delay time	tDBY	—	100	—	100	ns
120	FAULT delay time	tDFA	—	100	—	100	ns
121	From CLK high to $\overline{ACK}$	tCHAC	—	60	—	60	ns
122	From $\overline{ACK}$ low to BUSY low	tACLBY	—	60	—	60	ns
123	From $\overline{ACK}$ high to BUSY low	tACHBY	—	60	—	60	ns

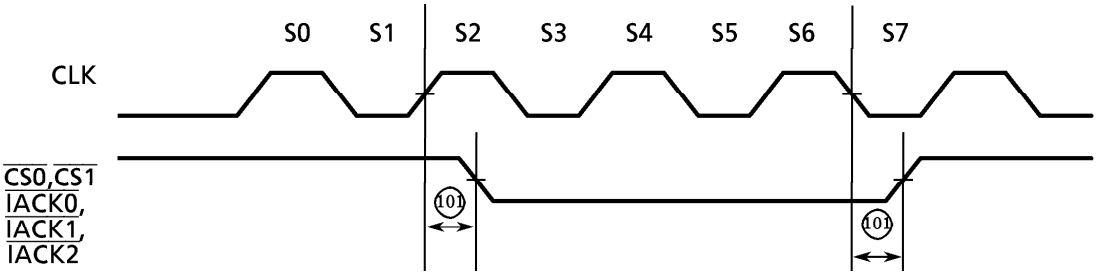


Figure 9.5 $\overline{CS}$, $\overline{IACK}$ Timing Diagram

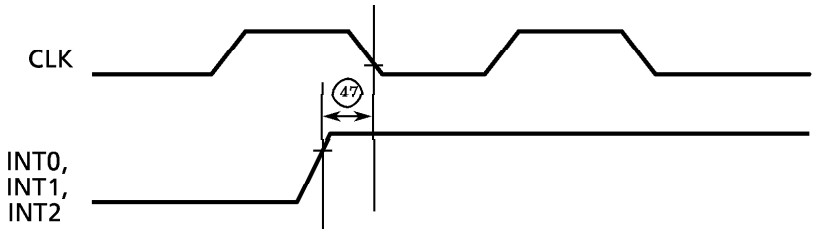


Figure 9.6 Interrupt Request Timing Diagram

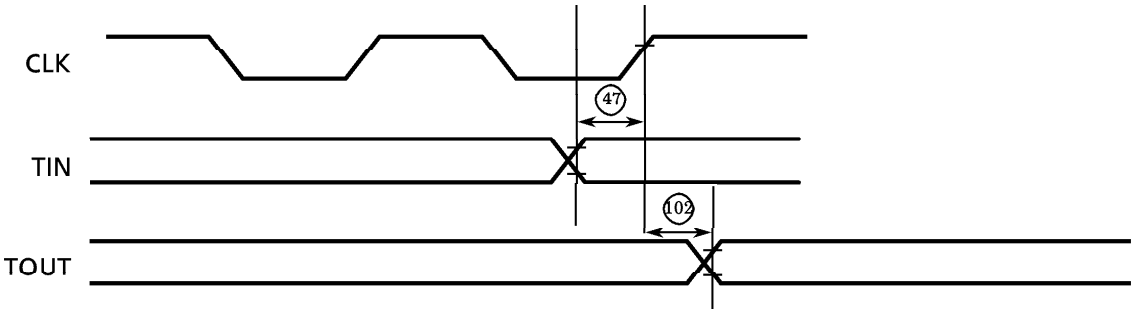


Figure 9.7 Timer Input/Output Timing Diagram

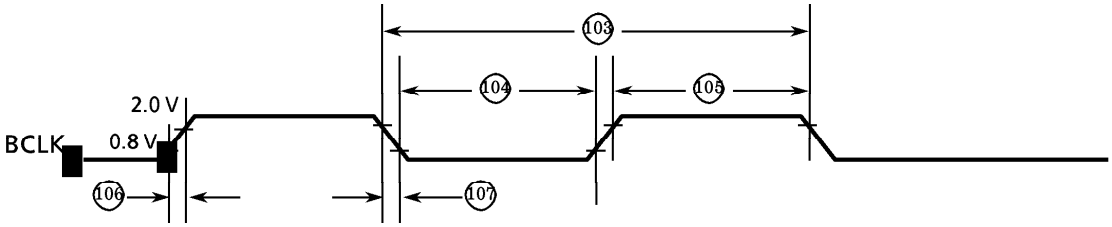


Figure 9.8 Baud Rate Clock Timing Diagram

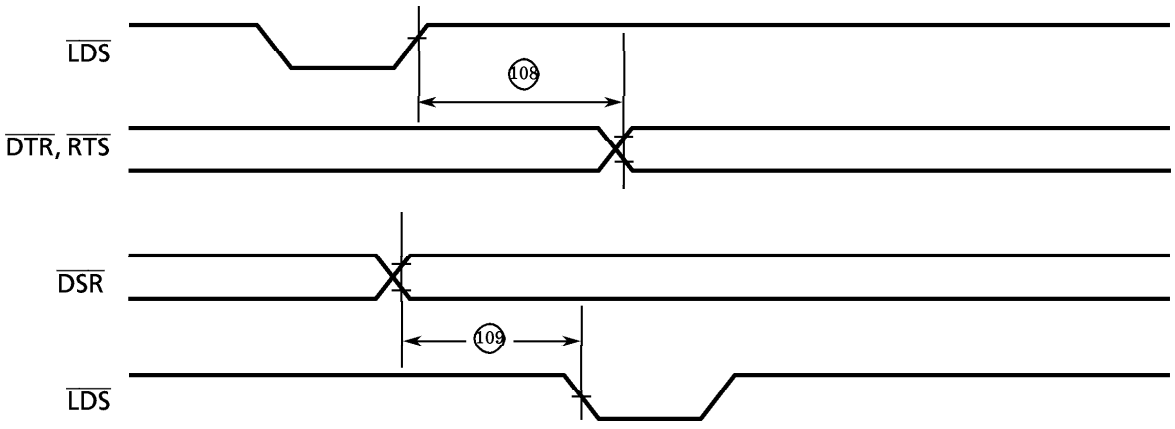


Figure 9.9 Serial Port Timing Diagram

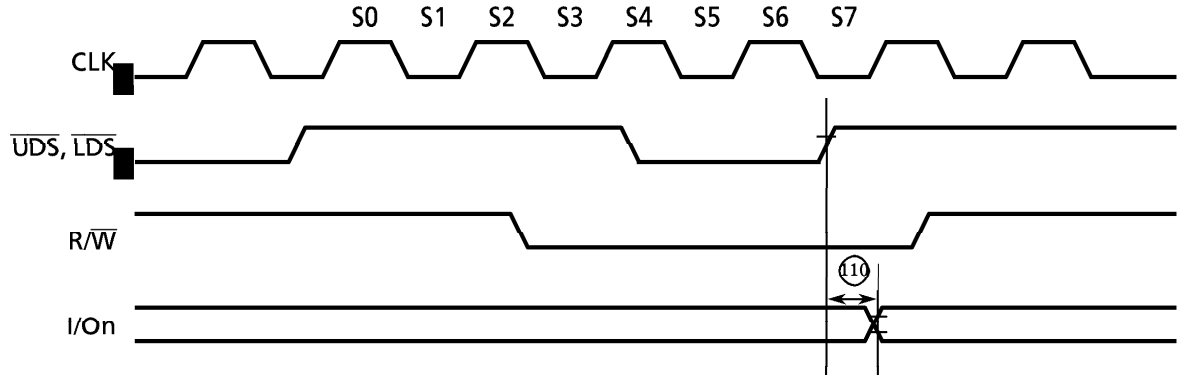


Figure 9.10 I/O Port Output Timing Diagram

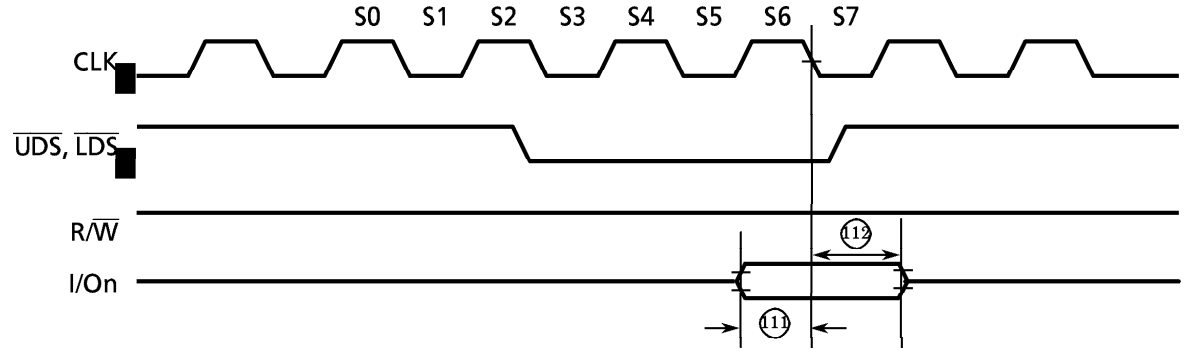


Figure 9.11 I/O Port Input Timing Diagram

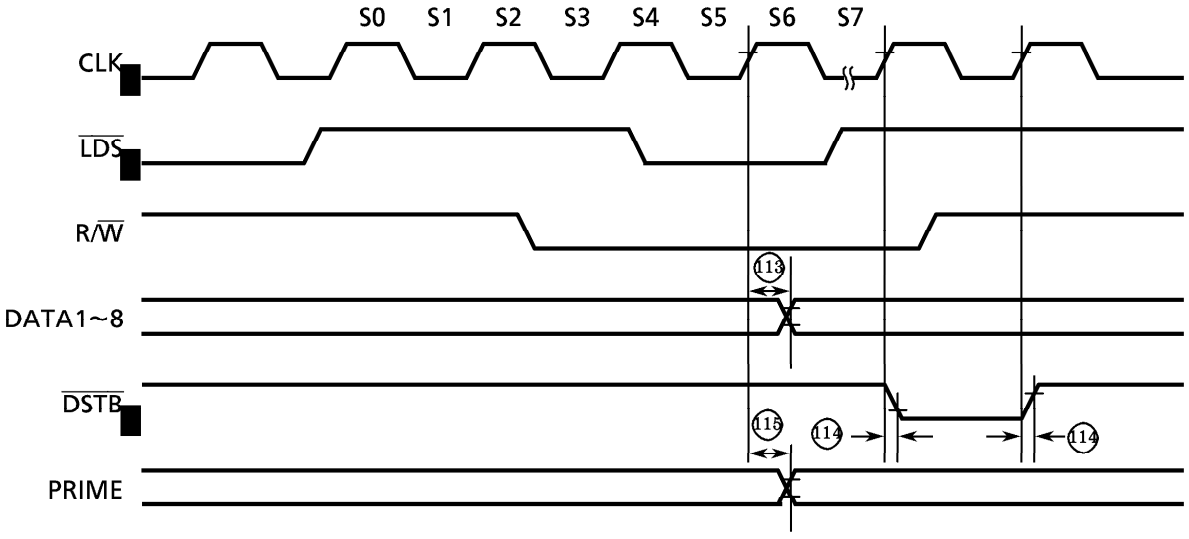


Figure 9.12 Centronics Output Timing Diagram (Mode 1)

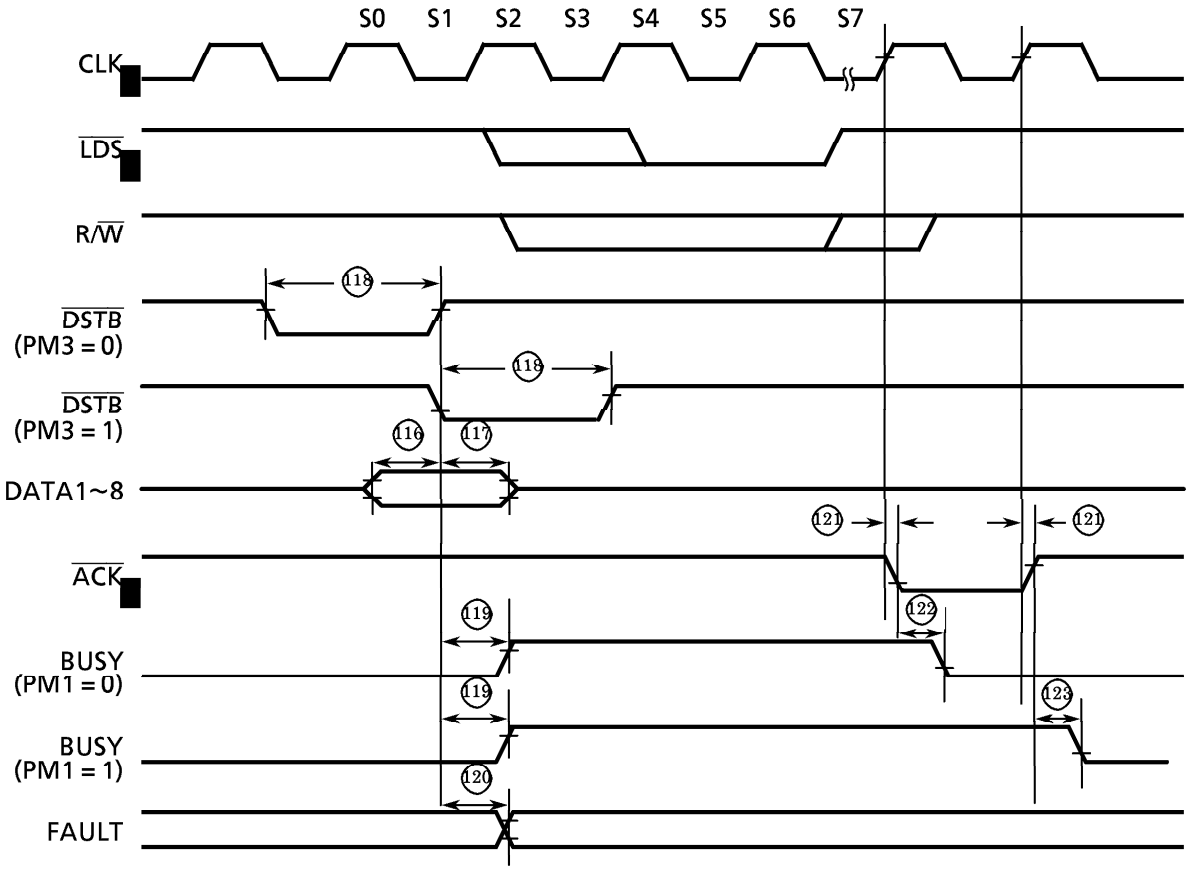


Figure 9.13 Centronics Input Timing Diagram (Mode 2)

10. Development Environment

10.1 Emulation mode

Set the NOR/$\overline{EMU}$ pin to low to enter emulation mode (henceforth, EMU mode). In EMU mode, the TMP68301A CPU is completely disconnected and only peripheral devices are active. In EMU mode, the emulator controls the peripheral devices.

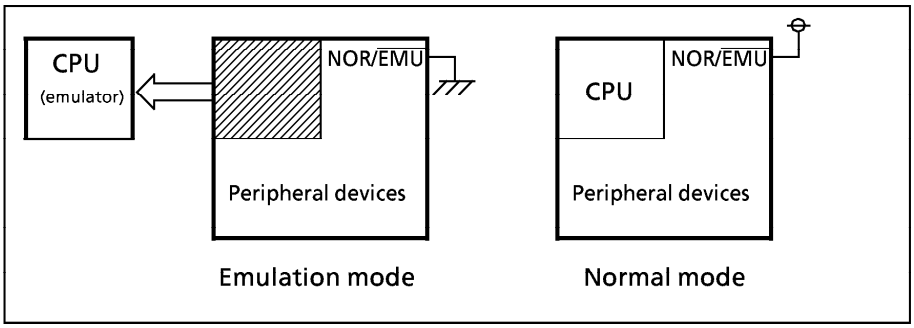


Figure 10.1 Outline of Normal and Emulation Modes

10.1.1 Pin Functions

(1) NOR/$\overline{EMU}$: Fixed to high in normal mode. Fixed to low in $\overline{EMU}$ mode.

(2) $\overline{BR}$/$\overline{IPL0}$
$\overline{BG}$/$\overline{IPL1}$
$\overline{BGACK}$/$\overline{IPL2}$: In normal mode, these pins ($\overline{BR}$, $\overline{BG}$, $\overline{BGACK}$) perform bus arbitration control. In EMU mode, $\overline{IPL0}$ to $\overline{IPL2}$ are output signals. As these pin signals operate differently for different modes, use jumpers to change pin operation, as shown in figures 10.2 and 10.3. (See *1 in the figures)

10.2 Connecting to General-Purpose 68000 Emulator

In principle, TMP68301A emulation can be performed using any commercially available general-purpose 68000 emulator. Note, however, the following points:

(1) As TMP68301A does not have E, $\overline{VPA}$, and $\overline{VMA}$ signals for controlling 8-bit peripheral devices, the $\overline{VPA}$ input to the emulator must be pulled-up.

(2) The 68000 interrupt vector automatic generation function cannot be used because TMP68301A does not have a $\overline{VPA}$ signal. However, TMP68301A has a similar function in its internal interrupt controller. For interrupt requests from external devices, use the interrupt input pins INT0 to INT2 connected to the interrupt controller.

(3) The $\overline{DTACK}$ and $\overline{BERR}$ signals are open drain outputs in EMU mode. Therefore, the $\overline{DTACK}$ and $\overline{BERR}$ signals generated on the target board must also be open drain outputs and pull-up resistors must be provided. At *2 in the figures, high resistance can result in slow rise times and cause operational problems. Therefore, use resistances of about 1K ohms.

(4) The following problems can occur due to a slow rise time for the $\overline{AS}$ signal output from the general-purpose 68000 emulator.
- If another interrupt occurs while an interrupt is pending, the pending bit is cleared and the interrupt is not subsequently accepted.
- If the $\overline{AS}$ signal setup time (*1 in Figure 10.4) does not satisfy the specification, the operation of the $\overline{DTACK}$ automatic generation function may be impaired and $\overline{DTACK}$ may not be asserted. (If $\overline{DTACK}$ is not asserted, the bus cycle does not complete and a bus error is generated.) (Figure 10.4)

For some 68000 emulators, the $\overline{AS}$ signal is always too slow. This problem can be solved, for example, by changing the relative phases of the clocks to the emulator and to 68301A, as shown in Figure 10.5.

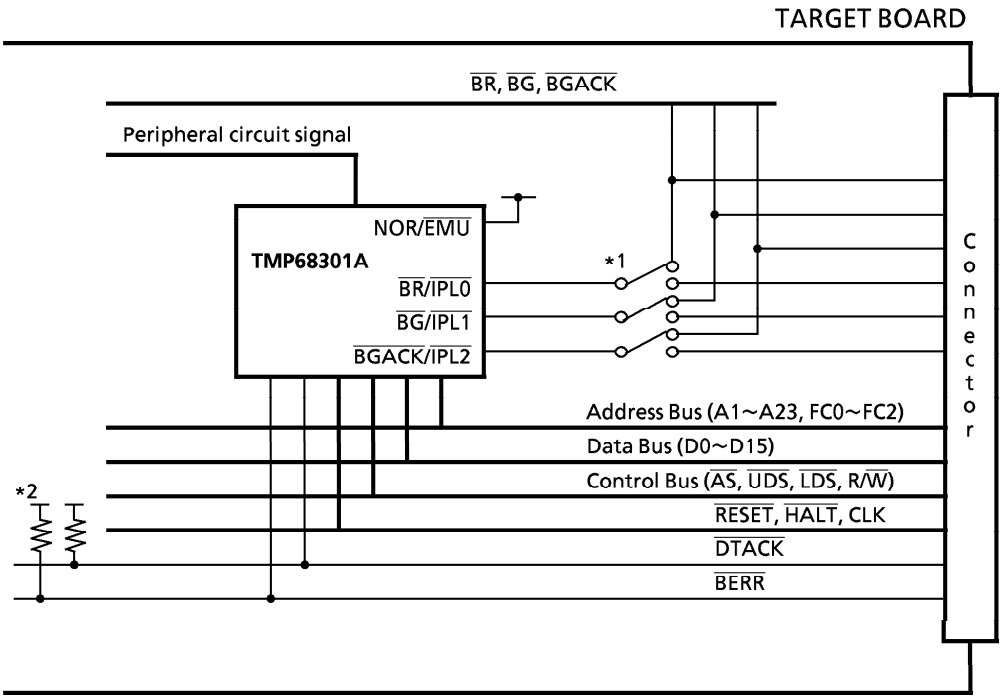


Figure 10.2 Wiring Diagram in Normal Mode

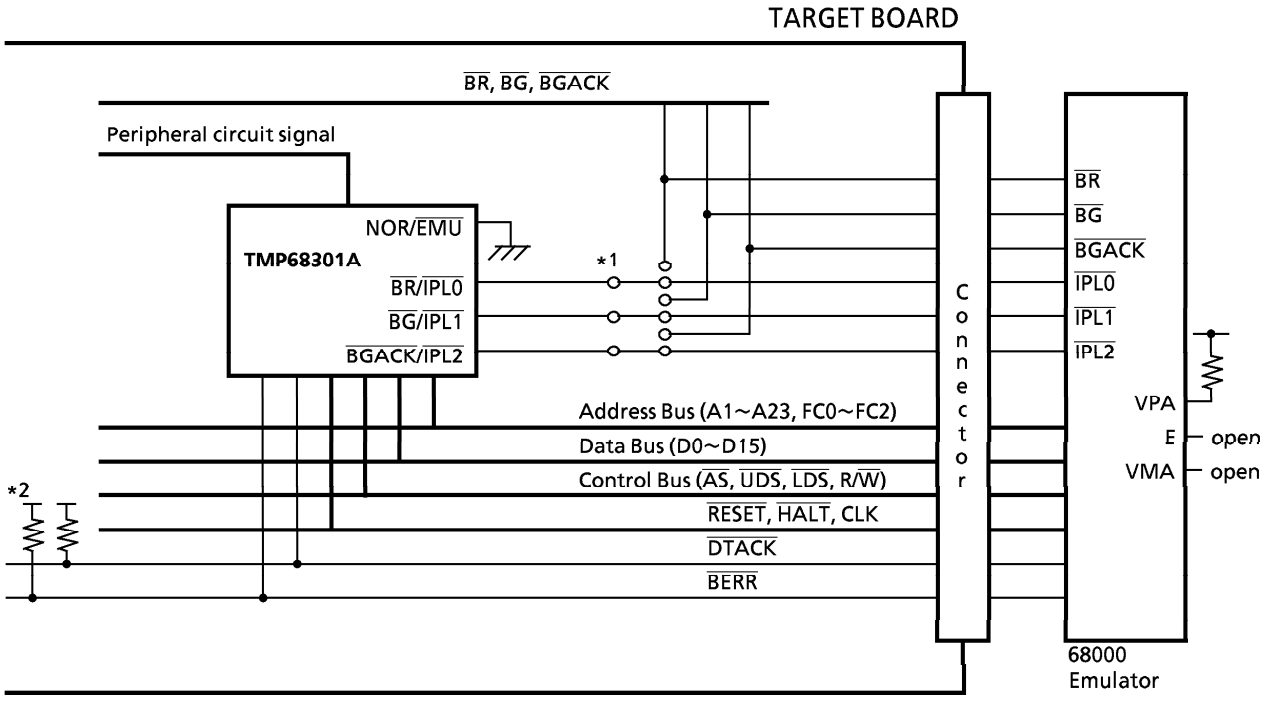


Figure 10.3 Connection to Emulator in EMU Mode

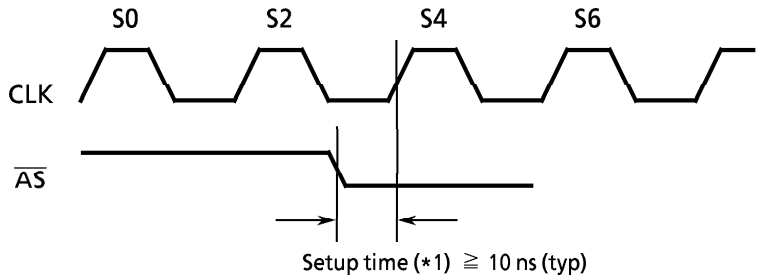


Figure 10.4 $\overline{AS}$ Signal Setup Time during Emulation

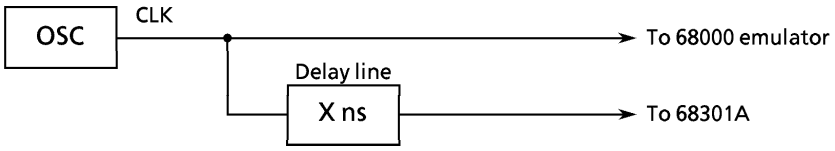


Figure 10.5 Clock Adjustments for Emulation