HD64570 SCA

Serial Communications Adaptor

User's Manual

HITACHI

ADE-602-035B
Rev. 3.0
August 28, 1998
Hitachi Company or Division

Cautions

1. Hitachi neither warrants nor grants licenses of any rights of Hitachi's or any third party's patent, copyright, trademark, or other intellectual property rights for information contained in this document. Hitachi bears no responsibility for problems that may arise with third party's rights, including intellectual property rights, in connection with use of the information contained in this document.

2. Products and product specifications may be subject to change without notice. Confirm that you have received the latest product standards or specifications before final design, purchase or use.

3. Hitachi makes every attempt to ensure that its products are of high quality and reliability. However, contact Hitachi's sales office before using the product in an application that demands especially high quality and reliability or where its failure or malfunction may directly threaten human life or cause risk of bodily injury, such as aerospace, aeronautics, nuclear power, combustion control, transportation, traffic, safety equipment or medical equipment for life support.

4. Design your application so that the product is used within the ranges guaranteed by Hitachi particularly for maximum rating, operating supply voltage range, heat radiation characteristics, installation conditions and other characteristics. Hitachi bears no responsibility for failure or damage when used beyond the guaranteed ranges. Even within the guaranteed ranges, consider normally foreseeable failure rates or failure modes in semiconductor devices and employ systemic measures such as fail-safes, so that the equipment incorporating Hitachi product does not cause bodily injury, fire or other consequential damage due to operation of the Hitachi product.

5. This product is not designed to be radiation resistant.

6. No one is permitted to reproduce or duplicate, in any form, the whole or part of this document without written approval from Hitachi.

7. Contact Hitachi's sales office for any questions regarding this document or Hitachi semiconductor products.

Preface

The HD64570 serial communications adaptor (SCA) peripheral chip enables a host microprocessor to perform asynchronous, byte-synchronous, or bit-synchronous serial communication. Its two full-duplex, multiprotocol serial channels support a wide variety of protocols, including frame relay, LAPB, LAPD, bisync, and ™DDCMP. Its built-in direct memory access controller (DMAC) is equipped with a 32-stage FIFO and can execute chained-block transfers. Due to its DMAC and 16-bit bus interface, the SCA supports serial data transfer rates up to 12 Mbits/s without monopolizing the bus, even in full-duplex communication. Other on-chip features of the SCA, including four types of MPU interfaces, a bus arbiter, timers, and an interrupt controller, provide added functionality in a wide range of applications, such as frame relay exchanges/system multiplexes, private branch exchanges, computer networks, workstations, ISDN terminals, and facsimile.

™DDCMP is a trademark of Digital Equipment Corporation.

Changes in the Revised Edition

The following tables list the main differences between this revision and the previous edition (ADE-602-035A). (2nd edition) The changes are marked with stars (*) in the text.

Changes in the Specifications

Specifications	Modifications
WTR SCA added	A chip with Wide Temperature Range (–40 to 85°C) has been added to the product lineup.

Changes in the Text

Page	Title	Modifications
1, 2, 3	1.2 Features, Table 3.1 Maximum data transfer rate supply voltage product lineup	A chip with Wide Temperature Range (–40 to 85°C) has been added to the product lineup.

How to Use This Manual

This user's manual describes the functions, performance, and usage of the HD64570 serial communications adaptor (SCA) as a general-purpose communications control chip.

This manual consists of eleven chapters and an appendix.

— **Section 1 Overview**

This section outlines the functions, performance, and internal structure of the SCA.

— **Section 2 Pin Arrangements and Functions**

This section shows the pin arrangements and lists the functions of each pin.

— **Section 3 System Controller**

This section describes the operating modes of the chip (reset mode, normal operating mode, system stop mode), the bus arbiter functions, and the bus interface needed for interfacing with a host MPU.

— **Section 4 Interrupt Controller**

This section describes the SCA's interrupt control logic, on-chip interrupt sources, vector output (fixed and modified vectoring), and acknowledge cycle.

— **Section 5 MSCI (Multiprotocol Serial Communication Interface)**

This section gives a general description of the asynchronous, byte-synchronous, and bit-synchronous communication protocols supported by the built-in multiprotocol serial communication interfaces (MSCI channels 0 and 1) that implement the main functions of the SCA, and shows the register settings for various communication functions.

— **Section 6 DMAC (DMA Controller)**

This section explains the single- and chained-block transfer modes supported by the built-in direct memory access controller (DMAC channels 0 to 3). Internal register functions and settings are also described.

— **Section 7 Timers**

This section explains operating modes and register settings of the built-in timers (channels 0 to 3), which generate internal interrupts.

— **Section 8 Wait Controller**

This section describes the built-in wait controller, which inserts wait states during memory access to one of three physical address spaces. Details about the associated internal registers and WAIT pin are also provided.

— **Section 9 Application Examples**

This section shows examples of software routines and circuits in some typical applications of the SCA.

— **Section 10 Electrical Specifications**

This section lists the SCA's electrical characteristics (absolute maximum ratings, DC characteristics, AC characteristics) and provides timing diagrams.

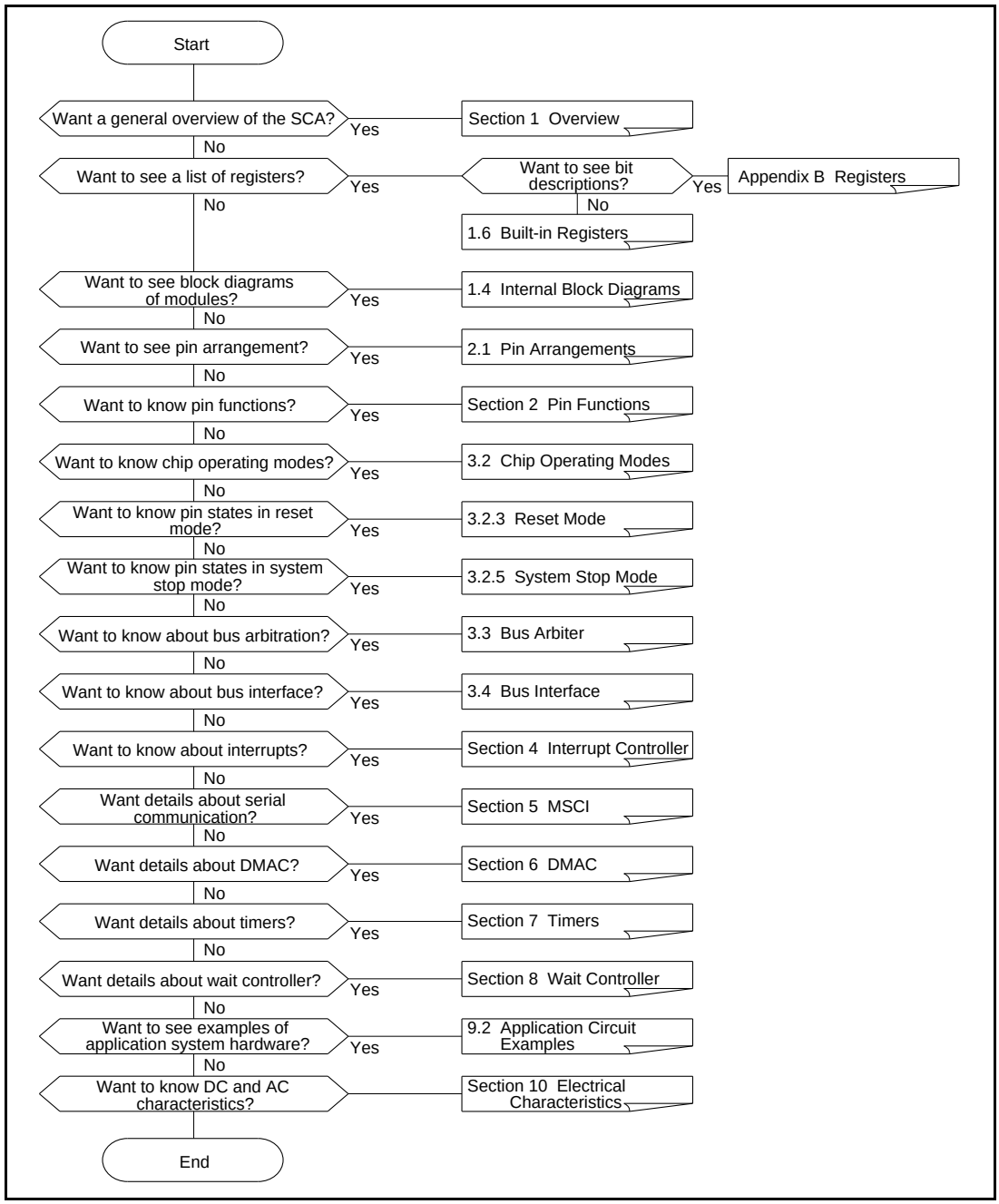
— **Section 11 Package Dimensions**

This section shows the dimensions of the SCA package.

— **Appendices Descriptor and Register Tables**

These tables list the names and bits of the descriptors used in DMA transfer, and the names, addresses, and bits of all SCA registers.

Use the following flowchart as a guide to reading this manual.



Contents

HITACHI

Section 1 Overview

1.1 Overview

The HD64570 serial communications adaptor (SCA) converts parallel data to serial data for communication with other devices. Its two independent, full-duplex transceivers support both synchronous (bit-synchronous or byte-synchronous) and asynchronous communication. Extensive protocol functions are provided.

The SCA chip provides FIFO transmit and receive buffers with 32 stages each and a four-channel direct memory access controller (DMAC) with chained-block transfer facilities, enabling high-speed transfer of data between SCA and memory. A built-in bus arbiter and 16-bit bus interface support high-performance system designs.

1.2 Features

- Data transfer rate: 50 bits/s to 7.1 Mbits/s (f = 10 MHz),
  50 bits/s to 12 Mbits/s (f = 16.7 MHz)
  50 bits/s to 5.7 Mbits/s (f = 8 MHz)*
- Protocol support
  - Asynchronous (ASYNC): 5 to 8 bits + parity
  - Byte synchronous (COP): bisync, X.21, DDCMP, etc.
  - Bit synchronous (BOP): frame relay, HDLC, ™SDLC, X.25 link level (LAPB), LAPD etc.
- Highly efficient data transfer: two 32-byte FIFOs (transmit/receive) per channel
- Error detection: parity (asynchronous)
  CRC-16, CRC-CCITT (byte- and bit-synchronous)

™SDLC is a trademark of International Business Machine.

Transmission codes: NRZ, NRZI, FM0, FM1, Manchester
Operating modes: normal operating mode (full-duplex), auto echo, local loop back
DMA transfer: on-chip DMAC with four channels and chained-block transfer capability
Address space: 16 Mbytes
Bus interface: connects to 64180-, 8086-, and 68000-system 8-/16-bit MPU buses
Timers: time-out detection, etc.
Power supply: 5 V ± 10% (–20 to 75°C) for 10-MHz chip,
5 V ± 5% (0 to 70°C) for 16.7-MHz chip
5 V ± 10% (–40 to 85°C) for 8-MHz chip*

1.3 Basic Functions

Table 1.1 Major Functions of the SCA

Item		Specification
MSCI (multiprotocol serial communication interface)	Maximum data transfer rate	7.1 Mbits/s (f = 10 MHz) 12 Mbits/s (f = 16.7 MHz) 5.7 Mbits/s (f = 8 MHz)*
	Number of channels	2
	Operating modes	Normal operating mode (full duplex) Auto echo mode Local loop back mode
	Protocol functions	Asynchronous: 5 to 8 bits, parity (odd, even, or none) Byte synchronous: mono-sync, bi-sync, or external synchronization Bit synchronous: HDLC mode
	Error detection	Five types (parity error, framing error, CRC error, overrun error, underrun error)
	Transmission codes	Five types (NRZ, NRZI, FM0, FM1, Manchester)
	FIFO	Transmit—32 bytes/receive—32 bytes
	Clock sources	Internal clock sources: 1. On-chip baud rate generators can generate arbitrary baud rates (independent transmit/ receive baud rate generators are provided for each channel) 2. On-chip digital PLL (independent ADPLLs for each receive channel) External clock
	Modem control	$\overline{\text{CTS}}$, $\overline{\text{RTS}}$, $\overline{\text{DCD}}$
	ADPLL (Advanced digital PLL)	On-chip advanced digital PLLs: 1. For extraction of clock signals 2. For suppressing noise in received data and clock signals
	Baud rate generator	On-chip (independent transmit and receive baud rate generators for each channel)
Bus interface	MPU modes	Four externally selectable modes: 1. 8086-system 16-bit MPU 2. 64180-system 8-bit MPU 3. 68000-system 16-bit MPUI 4. 68000-system 16-bit MPUII
	Data bus width	8 or 16 bits
	Address bus width	24 bits

Table 1.1 Major Functions of the SCA (cont)

Item		Specification
DMAC (direct memory access controller)	Number of channels	4
	Transfer modes	DMA transfer between memory and on-chip MSCI: 1. Single block transfer (asynchronous, byte-synchronous, bit-synchronous modes) 2. Chained-block transfer (bit-synchronous mode)
	Minimum transfer cycle	3 clocks
Timers	Number of channels	4
	Counter length	16 bits
Interrupt controller	Acknowledge cycle	Programmable: 1. Nonacknowledge cycle 2. Single acknowledge cycle 3. Double acknowledge cycle
	Vector output mode	Programmable: 1. Fixed vector output mode 2. Modified vector output mode
Wait state controller		On-chip (register programmable, or external line control)
Bus arbiter		On-chip (can be daisy-chained)
Low-power mode		System stop mode supported
Maximum system clock rate		10 MHz or 16.7 MHz
Signal level		TTL-compatible
Supply voltage		+5 V ± 10% (–20 to 75°C) for 10-MHz chip +5 V ± 5% (0 to 70°C) for 16.7-MHz chip +5 V ± 10% (–40 to 85°C) for 8-MHz chip*
Process		CMOS
Package		CP-84: 84-pin QFJ (PLCC) (quad flat j-leaded package (plastic leaded chip carrier)) FP-88: 88-pin QFP (quad flat package)

Product lineup

Type	Product Number	Maximum Operating Frequency	Operating Voltage	Package
SCA	HD64570CP	10 MHz	+5 V ± 10% (–20 to 75°C)	CP-84 (84-pin QFJ (PLCC))
	HD64570F			FP-88 (88-pin QFP)
High Speed SCA	HD64570CP16	16.7 MHz	+5 V ±5% (0 to 70°C)	CP-84 (84-pin QFJ (PLCC))
	HD64570F16			FP-88 (88-pin QFP)
WTR SCA	HD64570CP8I	8 MHz	+5 V ±10% (–40 to 85°C)	CP-84 (84-pin QFJ (PLCC))*
	HD64570F8I			FP-88 (88-pin QFP)*

1.4 Block Diagram

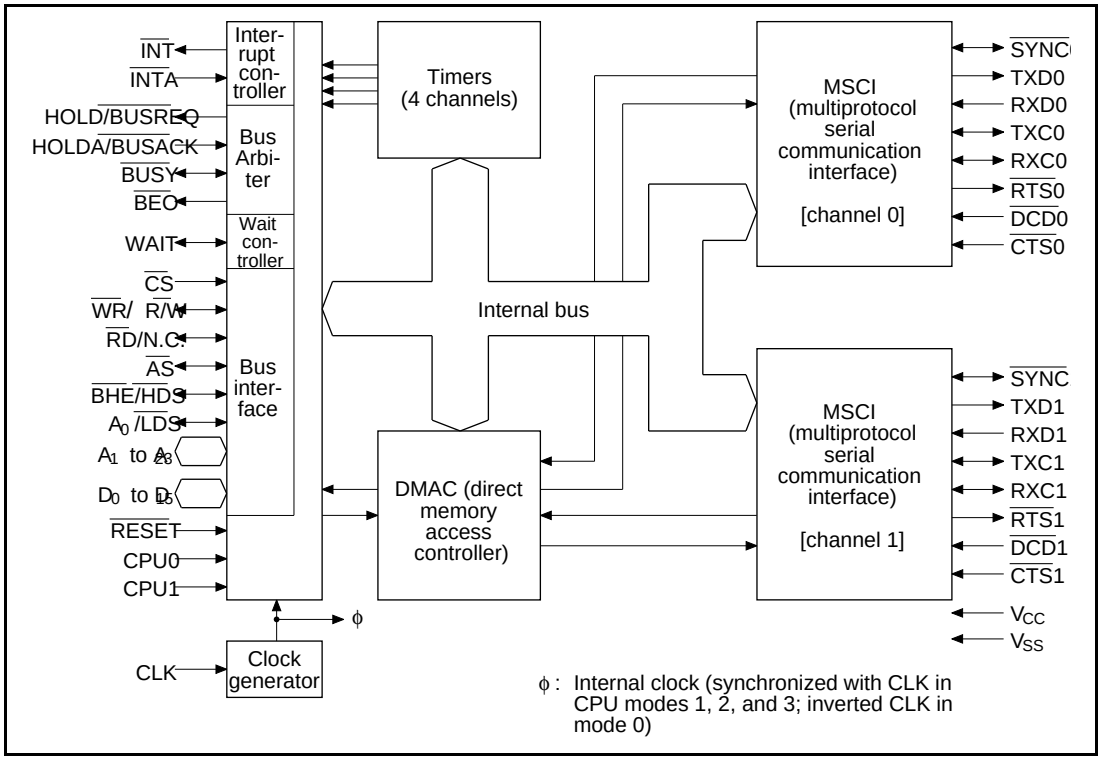


Figure 1.1 Block Diagram of SCA

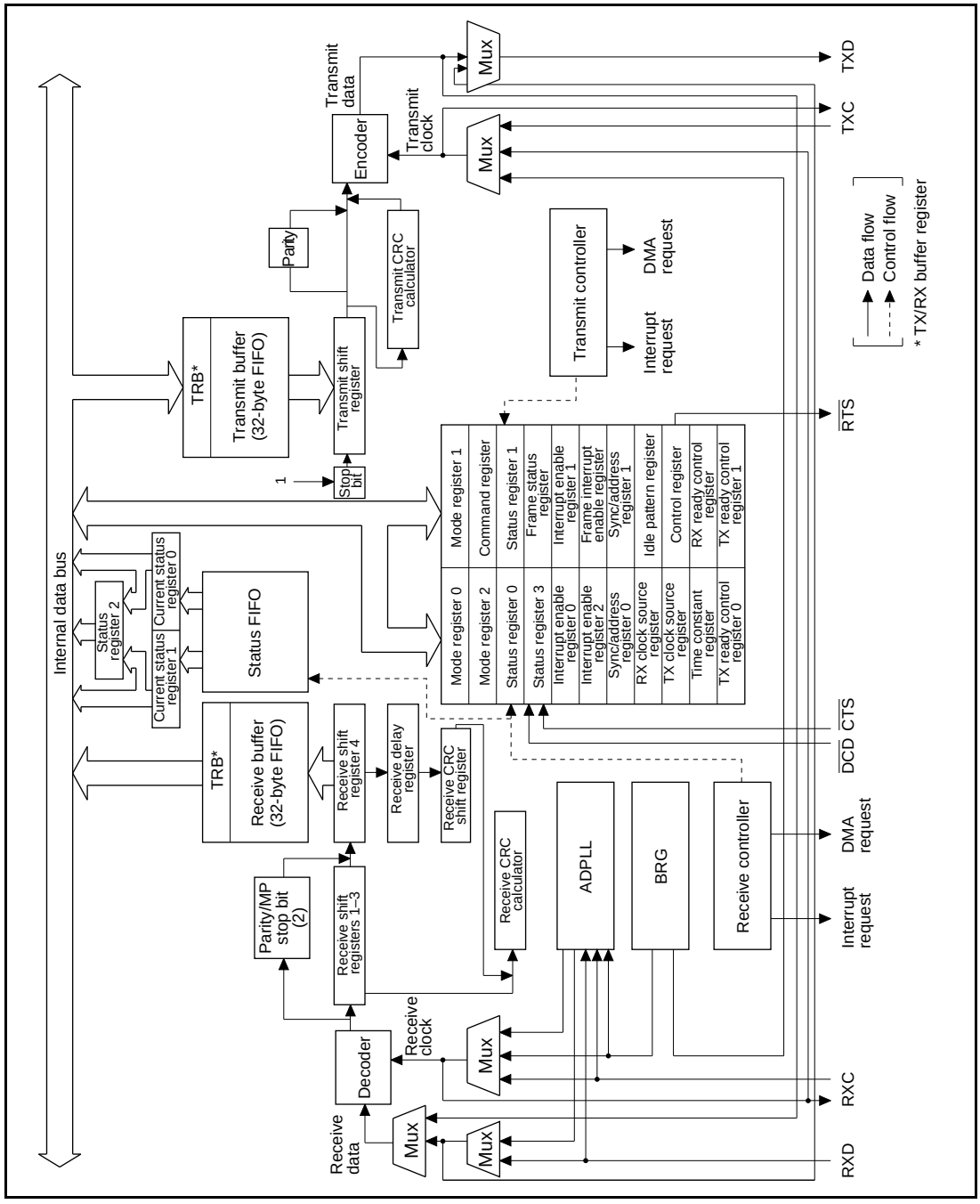


Figure 1.2 MSCI Block Diagram

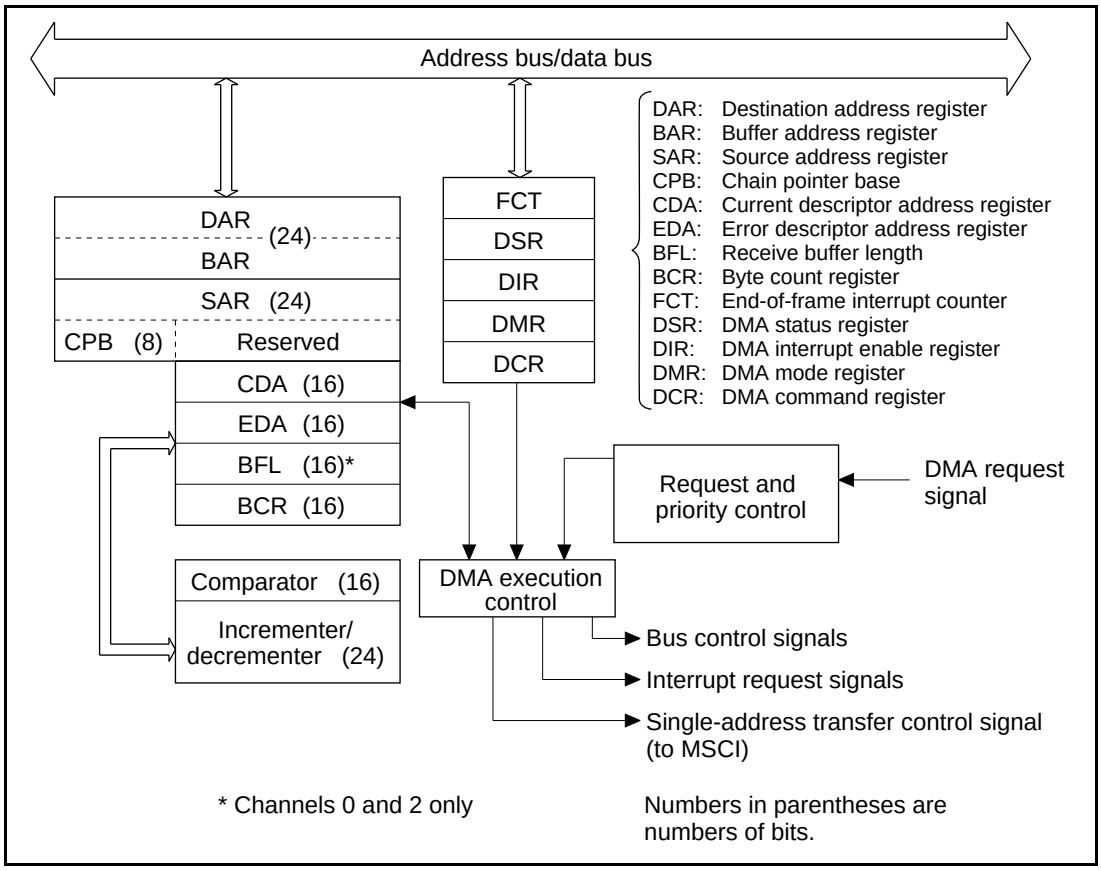


Figure 1.3 DMAC Block Diagram (one channel)

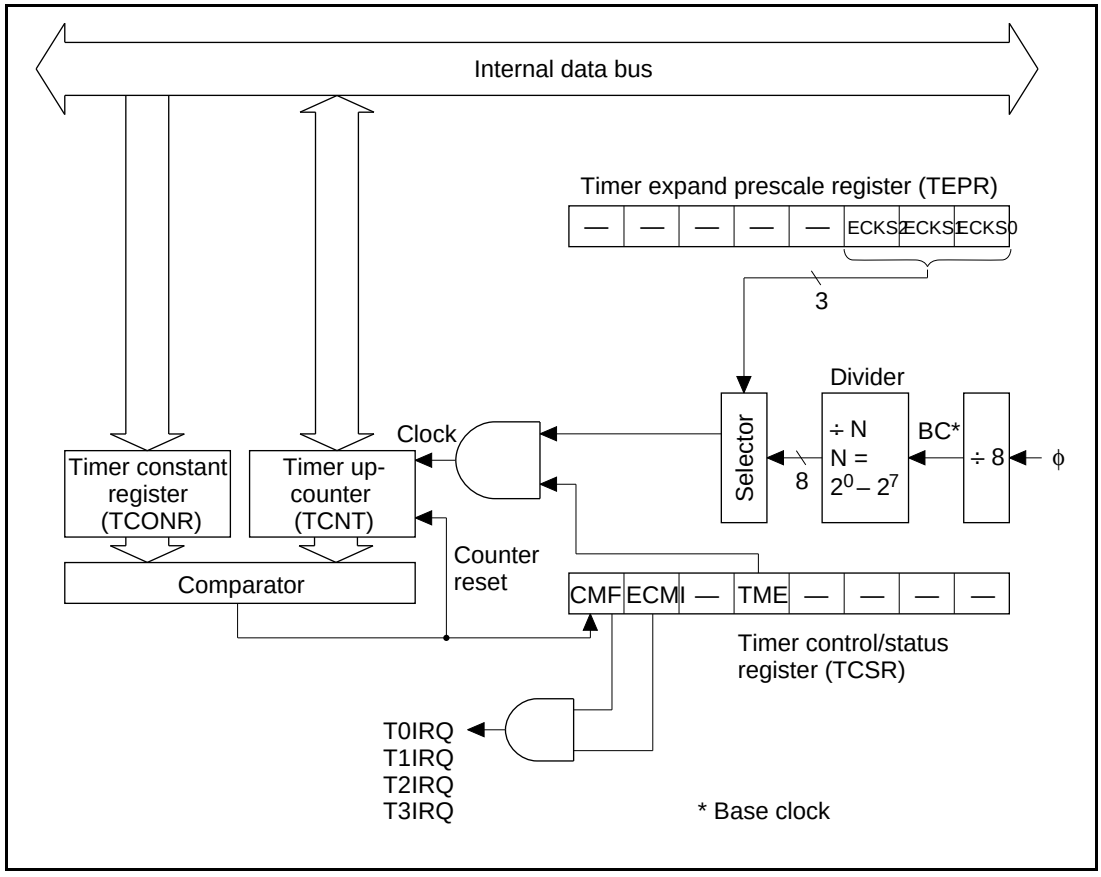


Figure 1.4 Timer Block Diagram

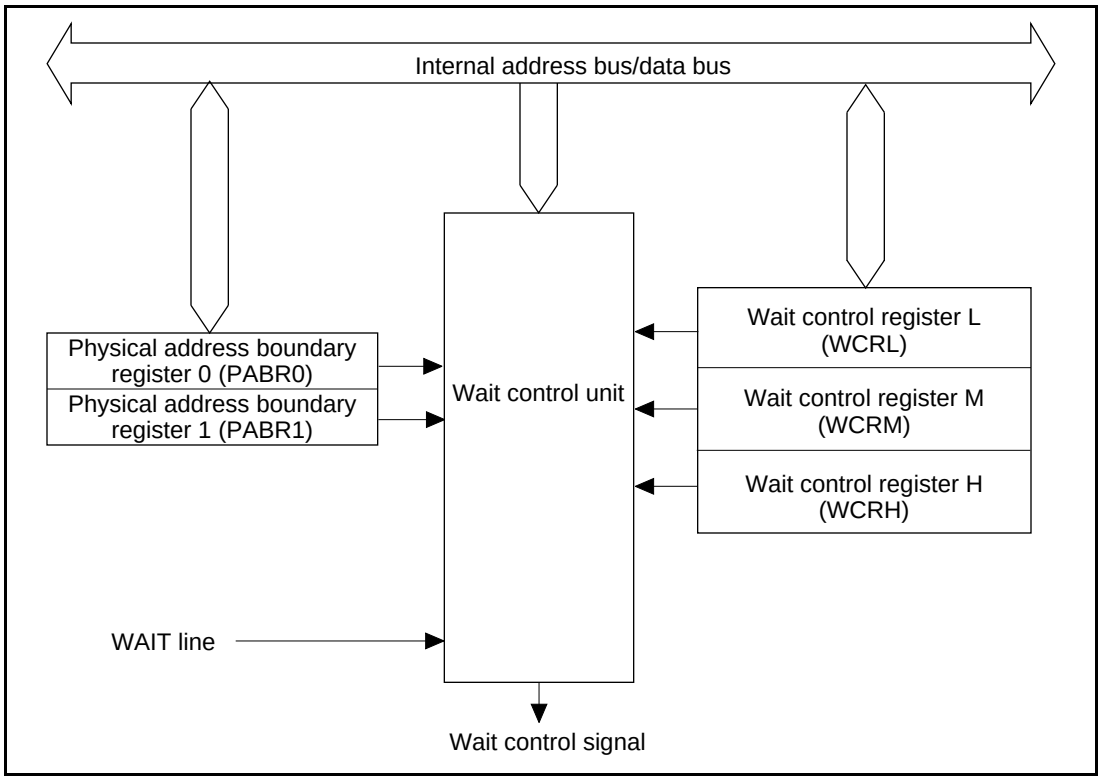


Figure 1.5 Wait Controller Block Diagram

1.5 Protocol Support

1.5.1 Asynchronous Mode

Item	Description
Character length	5 to 8 bits
Parity	Odd or even parity or no parity
Stop bits	1, 1.5, or 2
Transmit/receive clock	1x, 16x, 32x, or 64x mode
Error detection	Parity errors, overrun errors, framing errors
Break signal	Can generate break signal of arbitrary length
Break detection	Detects beginning and end of break
Multiprocessor support	By MP bit

1.5.2 Byte-Synchronous Mode

Item	Description
Character length	8 bits
Error control	Generates and checks CRC codes (CRC-16, CRC-CCITT)
Synchronous characters	1 or 2 characters
External synchronization	Supported
Synchronization	Can transmit, detect and delete SYN character
Underrun	Idle, or CRC + idle output
Idle	Mark or SYN character output
Error detection	CRC error, overrun error, underrun error

1.5.3 Bit-Synchronous Mode

Item	Description
Character length	8 bits
Error control	Generates and checks CRC codes (CRC-16, CRC-CCITT)
Bit pattern	Detects and generates flag, abort, and idle
Frame subdivision	Detects subdivision flag (single flag) between frames
Address field	Four address field checks (can recognize group addresses and global addresses)
End of frame	When EOM is received
Data input/output	Zero insert/delete
Residual bits	Detects residual bit frames
Short frame	Detects short (invalid) frames
Underrun	Abort + idle, or FCS + flag + idle
Idle	Mark or flag

1.6 Built-In Registers

1.6.1 Low-Power Mode Control Registers

Register Name	Symbol	CPU Modes 0 & 1	CPU Modes 2 & 3	Initial Value at Hardware Reset								Read/ Write
				MSB							LSB	
Low power register	LPR	00H	01H	0	0	0	0	0	0	0	0	R/W

1.6.2 Interrupt Control Registers

Register Name	Symbol	Channel 0	Channel 1	Initial Value at Hardware Reset								Read/ Write
				MSB							LSB	
Interrupt vector register	IVR	1AH	1BH	0	0	0	0	0	0	0	0	R/W
Interrupt modified vector register	IMVR	1CH	1DH	0	0	0	0	0	0	0	0	R/W
Interrupt control register	ITCR	18H	19H	0	0	0	0	0	0	0	0	R/W
Interrupt status register 0	ISR0	10H	11H	0	0	0	0	0	0	0	0	R
Interrupt status register 1	ISR1	11H	10H	0	0	0	0	0	0	0	0	R
Interrupt status register 2	ISR2	12H	13H	0	0	0	0	0	0	0	0	R
Interrupt enable register 0	IER0	14H	15H	0	0	0	0	0	0	0	0	R/W
Interrupt enable register 1	IER1	15H	14H	0	0	0	0	0	0	0	0	R/W
Interrupt enable register 2	IER2	16H	17H	0	0	0	0	0	0	0	0	R/W

1.6.3 MSCI Registers (1)

		Address				Initial Value at Reset*[1]								Read/ Write*[2]
		CPU Modes 0 & 1		CPU Modes 2 & 3										
Register Name	**Symbol**	**Channel 0**	**Channel 1**	**Channel 0**	**Channel 1**	**MSB**							**LSB**	
Mode register 0	MD0	2EH	4EH	2FH	4FH	0	0	0	0	0	0	0	0	R/W
Mode register 1	MD1	2FH	4FH	2EM	4EH	0	0	0	0	0	0	0	0	R/W
Mode register 2	MD2	30H	50H	31H	51H	0	0	0	0	0	0	0	0	R/W
Control register	CTL	31H	51H	30H	50H	0	0	0	0	0	0	0	1	R/W
RX clock source register	RXS	36H	56H	37H	57H	0	0	0	0	0	0	0	0	R/W
TX clock source register	TXS	37H	57H	36H	56H	0	0	0	0	0	0	0	0	R/W
Time constant register	TMC	35H	55H	34H	54H	0	0	0	0	0	0	0	1	R/W
Command register	CMD	2CH	4CH	2DH	4DH	—								W
Status register 0	ST0	22H	42H	23H	43H	0	0	0	0	0	0	0	0	R
Status register 1	ST1	23H	43H	22H	42H	0	0	0	0	0	0	0	0	R/W
Status register 2	ST2	24H	44H	25H	45H	0	0	0	0	0	0	0	0	R/W
Status register 3	ST3	25H	45H	24H	44H	0	0	0	0	×	×*[3]	0	0	R
Frame status register	FST	26H	46H	27H	47H	0	0	0	0	0	0	0	0	R/W
Interrupt enable register 0	IE0	28H	48H	29H	49H	0	0	0	0	0	0	0	0	R/W
Interrupt enable register 1	IE1	29H	49H	28H	48H	0	0	0	0	0	0	0	0	R/W
Interrupt enable register 2	IE2	2AH	4AH	2BH	4BH	0	0	0	0	0	0	0	0	R/W
Frame interrupt enable register	FIE	2BH	4BH	2AH	4AH	0	0	0	0	0	0	0	0	R/W
Sync/address register 0	SA0	32H	52H	33H	53H	1	1	1	1	1	1	1	1	R/W
Sync/address register 1	SA1	33H	53H	32H	52H	1	1	1	1	1	1	1	1	R/W
Idle pattern register	IDL	34H	54H	35H	55H	1	1	1	1	1	1	1	1	R/W

Notes: 1. These initial values apply after a hardware reset or execution of a channel reset command. Some registers are also initialized to these values by the RX reset command or TX reset command. See section 5.2, Registers, for details.

2. The functions of some bits vary depending on the operating mode (asynchronous, byte synchronous, or bit synchronous). See the register descriptions starting in section 5.2.1.

3. When bits 3 and 2 of status register 3 (ST3) are read, the logic level of the $\overline{\text{CTS}}$ and $\overline{\text{DCD}}$ lines are read.

1.6.3 MSCI Registers (2)

Register Name	Symbol	Address: CPU Modes 0 & 1 Channel 0	Address: CPU Modes 0 & 1 Channel 1	Address: CPU Modes 2 & 3 Channel 0	Address: CPU Modes 2 & 3 Channel 1	Initial Value at Reset*[1] MSB							LSB	Read/Write
TX/RX buffer register	TRBL	20H	40H	21H	41H	×	×	×	×	×	×	×	×	R/W*[3]
	TRBH	21H	41H	20H	40H	×	×	×	×	×	×	×	×	R/W*[3]
RX ready control register	RRC	3AH	5AH	3BH	5BH	0	0	0	0	0	0	0	0	R/W
TX ready control register 0	TRC0	38H	58H	39H	59H	0	0	0	0	0	0	0	0	R/W
TX ready control register 1	TRC1	39H	59H	38H	58H	0	0	0	1	1	1	1	1	R/W
Current status register 0	CST0	3CH	5CH	3DH	5DH	0	0	0	0	0	0	0	0	R/W
Current status register 1	CST1	3DH	5DH	3CH	5CH	0	0	0	0	0	0	0	0	R/W

Notes: 1. These initial values apply after a hardware reset or execution of a channel reset command. Some registers are also initialized to these values by the RX reset command or TX reset command. See section 5.2, Registers, for details.

2. The functions of some bits vary depending on the operating mode (asynchronous, byte synchronous, or bit synchronous). See the register descriptions starting in section 5.2.1.

3. The TX/RX buffer register (TRBL, TRBH) acts as the receive buffer register for the received character when read, and as the transmit buffer register for the transmitted character when written.

1.6.4 DMAC Registers Common to Channels 0 to 3

Register Name	Symbol	Address: CPU Modes 0 & 1	Address: CPU Modes 2 & 3	Initial Value at Hardware Reset MSB							LSB	Read/Write
DMA priority control register	PCR	08H	09H	0	0	0	0	0	0	0	0	R/W
DMA master enable register	DMER	09H	08H	1	0	0	0	0	0	0	0	R/W

Note: Use byte access to read and write PCR and DMER. These registers cannot be accessed by word access.

1.6.5 DMA Registers Provided Separately for Channels 0 to 3

Register Name	Symbol	Address: CPU Modes 0 & 1 Channel 0	Address: CPU Modes 0 & 1 Channel 1	Address: CPU Modes 0 & 1 Channel 2	Address: CPU Modes 0 & 1 Channel 3	Address: CPU Modes 2 & 3 Channel 0	Address: CPU Modes 2 & 3 Channel 1	Address: CPU Modes 2 & 3 Channel 2	Address: CPU Modes 2 & 3 Channel 3	Initial Value at Hardware Reset MSB	Initial Value at Hardware Reset LSB	Read/ Write
Destination address register L (buffer address register L)*1	DARL (BARL)	80H	A0H	C0H	E0H	81H	A1H	C1H	E1H	× × × × × × × ×		R/W
Destination address register H (buffer address register H)*1	DARH (BARH)	81H	A1H	C1H	E1H	80H	A0H	C0H	E0H	× × × × × × × ×		R/W
Destination address register B (buffer address register B)*1	DARB (BARB)	82H	A2H	C2H	E2H	83H	A3H	C3H	E3H	× × × × × × × ×		R/W
Source address register L*2	SARL	—	A4H	—	E4H	—	A5H	—	E5H	× × × × × × × ×		R/W
Source address register H*2	SARH	—	A5H	—	E5H	—	A4H	—	E4H	× × × × × × × ×		R/W
Source address register B (chain pointer base)*1	SARB (CPB)	86H	A6H	C6H	E6H	87H	A7H	C7H	E7H	× × × × × × × ×		R/W
Current descriptor address register L	CDAL	88H	A8H	C8H	E8H	89H	A9H	C9H	E9H	× × × × × × × ×		R/W

(×: undefined)

Notes: 1. Parentheses indicate registers with different functions in single-block transfer mode and chained-block transfer mode. The name in parentheses applies in chained-block transfer mode. See the register descriptions for details.

2. These registers are not used in chained-block transfer mode. Avoid writing to these registers in chained-block transfer mode.

1.6.5 DMA Registers Provided Separately for Channels 0 to 3 (cont)

Register Name	Symbol	Address: CPU Modes 0 & 1 Channel 0	Channel 1	Channel 2	Channel 3	Address: CPU Modes 2 & 3 Channel 0	Channel 1	Channel 2	Channel 3	Initial Value at Hardware Reset MSB							LSB	Read/ Write
Current descriptor address register H	CDAH	89H	A9H	C9H	E9H	88H	A8H	C8H	E8H	×	×	×	×	×	×	×	×	R/W
Error descriptor address register L	EDAL	8AH	AAH	CAH	EAH	8BH	ABH	CBH	EBH	×	×	×	×	×	×	×	×	R/W
Error descriptor address register H	EDAH	8BH	ABH	CBH	EBH	8AH	AAH	CAH	EAH	×	×	×	×	×	×	×	×	R/W
Receive buffer length L*2	BFLL	8CH	—	CCH	—	8DH	—	CDH	—	×	×	×	×	×	×	×	×	R/W
Receive buffer length H*2	BFLH	8DH	—	CDH	—	8CH	—	CCH	—	×	×	×	×	×	×	×	×	R/W
Byte count register L	BCRL	8EH	AEH	CEH	EEH	8FH	AFH	CFH	EFH	×	×	×	×	×	×	×	×	R/W
Byte count register H	BCRH	8FH	AFH	CFH	EFH	8EH	AEH	CEH	EEH	×	×	×	×	×	×	×	×	R/W
DMA status register*1	DSR	90H	B0H	D0H	F0H	91H	B1H	D1H	F1H	0	0	0	0	0	0	0	1	R/W
DMA mode register	DMR	91H	B1H	D1H	F1H	90H	B0H	D0H	F0H	0	0	0	0	0	0	0	0	R/W
End-of-frame interrupt counter	FCT	93H	B3H	D3H	F3H	92H	B2H	D2H	F2H	0	0	0	0	0	0	0	0	R
DMA interrupt enable register	DIR	94H	B4H	D4H	F4H	95H	B5H	D5H	F5H	0	0	0	0	0	0	0	0	R/W
DMA command register	DCR	95H	B5H	D5H	F5H	94H	B4H	D4H	F4H	—								W

(×: undefined)

Notes: 1. Some bits in the DMA status register are cleared by writing 1 to their bit positions, and one is a write-only bit. See section 6.2.7, DMA Status Register, for details.

2. These registers are used in receiving, so they are not provided for channels 1 and 3.

1.6.6 Timer Registers

Register Name	Symbol	Address: CPU Modes 0 & 1 Channel 0	Ch 1	Ch 2	Ch 3	Address: CPU Modes 2 & 3 Channel 0	Ch 1	Ch 2	Ch 3	Initial Value at Hardware Reset MSB							LSB	Read/ Write
Timer up-counter	TCNTL	60H	68H	70H	78H	61H	69H	71H	79H	0	0	0	0	0	0	0	0	R/W
	TCNTH	61H	69H	71H	79H	60H	68H	70H	78H	0	0	0	0	0	0	0	0	R/W
Timer constant register	TCONRL	62H	6AH	72H	7AH	63H	6BH	73H	7BH	1	1	1	1	1	1	1	1*	W
	TCONRH	63H	6BH	73H	7BH	62H	6AH	72H	7AH	1	1	1	1	1	1	1	1*	W
Timer control/status register	TCSR	64H	6CH	74H	7CH	65H	6DH	75H	7DH	0	0	0	0	0	0	0	0	R/W
Timer expand prescale register	TEPR	65H	6DH	75H	7DH	64H	6CH	74H	7CH	0	0	0	0	0	0	0	0	R/W

Note: The timer constant register is a write-only register that always reads as 0000H.

1.6.7 Wait Controller Registers

Register Name	Symbol	CPU Modes 0 & 1	CPU Modes 2 & 3	Initial Value at Hardware Reset MSB							LSB	Read/ Write
Physical address boundary register 0	PABR0	02H	03H	0	0	0	0	0	0	0	0	R/W
Physical address boundary register 1	PABR1	03H	02H	0	0	0	0	0	0	0	0	R/W
Wait control register L	WCRL	04H	05H	0	0	0	0	0	1	1	1	R/W
Wait control register M	WCRM	05H	04H	0	0	0	0	0	1	1	1	R/W
Wait control register H	WCRH	06H	07H	0	0	0	0	0	1	1	1	R/W

1.7 General Description of Functions

1.7.1 Operating Modes of Serial Section

The normal SCA operating mode is full duplex (figure 1.6). The SCA can transmit and receive simultaneously, using two separate lines.

In auto echo mode (figure 1.6), the SCA automatically retransmits all received data. Data received on the RXD line are retransmitted on the TXD line, and received data are simultaneously input to the receiver in the SCA.

In local loop back mode (figure 1.6), transmit data supplied to the SCA are automatically transferred to the receive data buffer in the SCA. Data received on the RXD line are retransmitted on the TXD line.

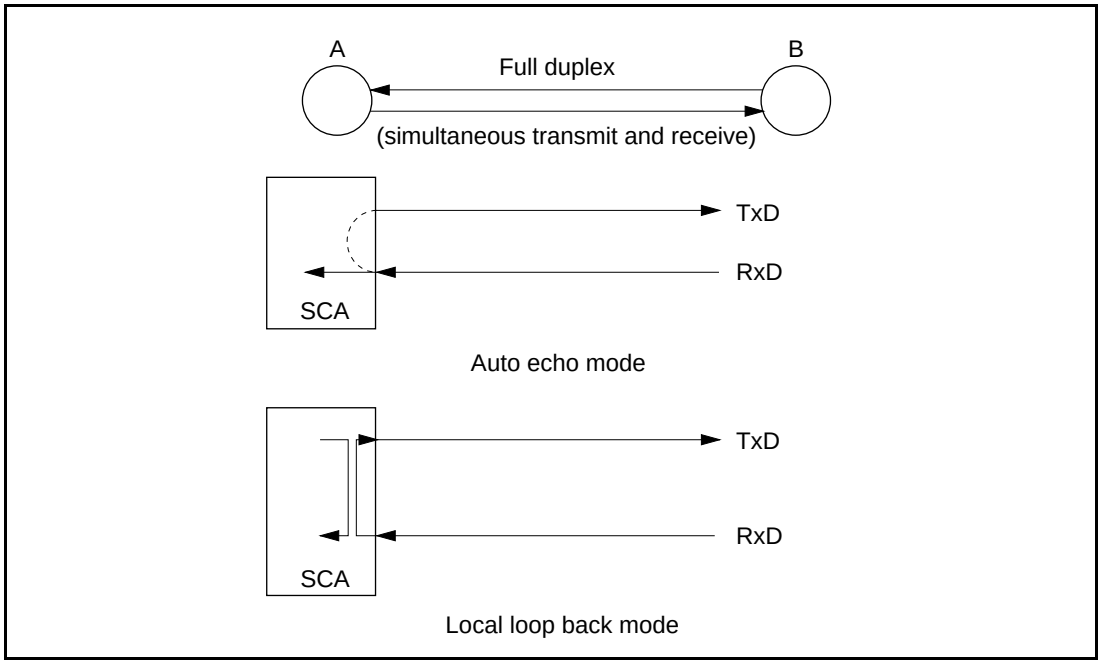


Figure 1.6 Operating Modes

1.7.2 Transmission Formats

The SCA supports asynchronous, byte-synchronous (bisync, X.21, DDCMP, etc.) and bit-synchronous (frame relay, HDLC, SDLC, X.25 link level/LAPB, LAPD, etc.) protocols. Each communication mode has its own format for transmitting data (figure 1.7).

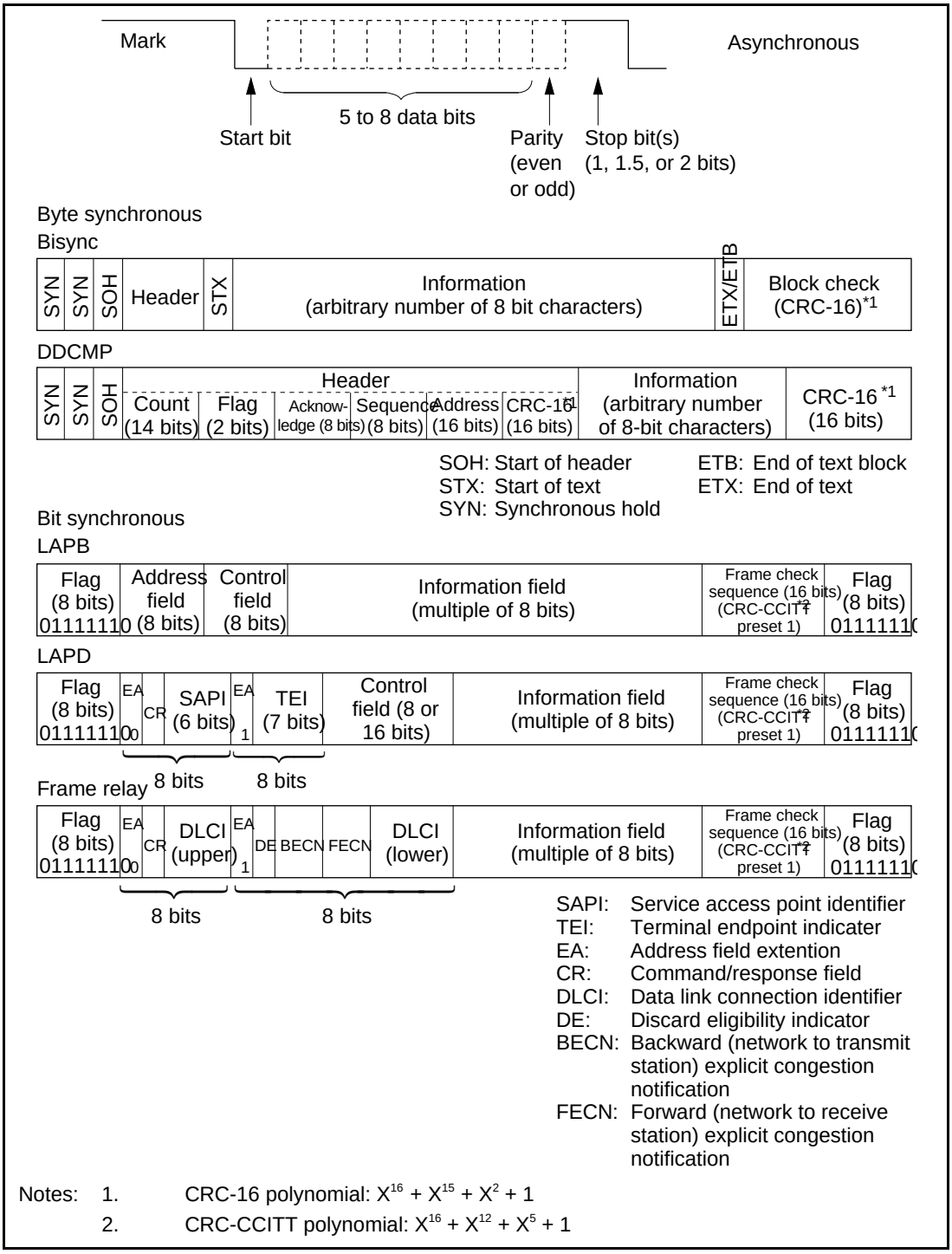


Figure 1.7 Examples of Transmission Formats

1.7.3 Transmission Error Detection

The SCA flags the following transmission errors in its status registers (ST1 and ST2) to notify the host MPU:

1. Parity error (asynchronous)

   This error occurs when the designated parity condition is not satisfied. It indicates that an incorrect bit (possibly the parity bit) was received.
2. Framing error (asynchronous)

   This error occurs if the RXD input is low (space) in the position of the first stop bit.
3. CRC error (byte synchronous or bit synchronous)

   This error occurs if the correct CRC value is not obtained, indicating that a bit error occurred on the transmission line.
4. Overrun error (asynchronous, byte synchronous, or bit synchronous)

   This error occurs if receive data are sent to the receive FIFO when the receive FIFO is full. After an overrun error, new receive data are overwritten on the last byte in the receive FIFO, destroying the preceding receive data but protecting other data already received.
5. Underrun error (byte synchronous or bit synchronous)

   This error occurs if the transmit FIFO is empty after transmission of the data in the transmit shift register.

1.7.4 Transmission Codes

The SCA supports five transmission codes: NRZ, NRZI, FM0, FM1, and Manchester (figure 1.8).

See figures 5.39, 5.40, and 5.41 for the timing relationships between the TX and RX clocks and each code.

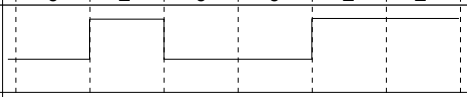
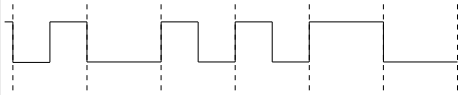
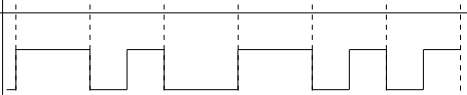
Type	No.	Code	Data and waveform (0 1 0 0 1 1)	Definition of 1 and 0
NRZ Encoding	1	NRZ (Nonreturn to zero)		1: High 0: Low
	2	NRZI (Nonreturn to zero inversion)		1: No transition 0: Transition
FM Encoding	1	FM0 (Frequency modulation space)		1: Transition at beginning of bit cell 0: Transitions at beginning and center of bit cell
	2	FM1 (Frequency modulation mark)		1: Transitions at beginning and center of bit cell 0: Transition at beginning of bit cell
	3	MANCHESTER (Manchester)		1: High-to-low transition at center of bit cell 0: Low-to-high transition at center of bit cell

Figure 1.8 Transmission Codes

1.7.5 Transmit/Receive Clock Selection

MSCI channels 0 and 1 in the SCA are full-duplex transceivers that support asynchronous, byte-synchronous, and bit-synchronous transmission formats. The transmit clock (figure 1.9) can be selected independently on each channel. The selectable clock sources include the built-in baud rate generator, external clock input, and the receive clock (figure 1.10).

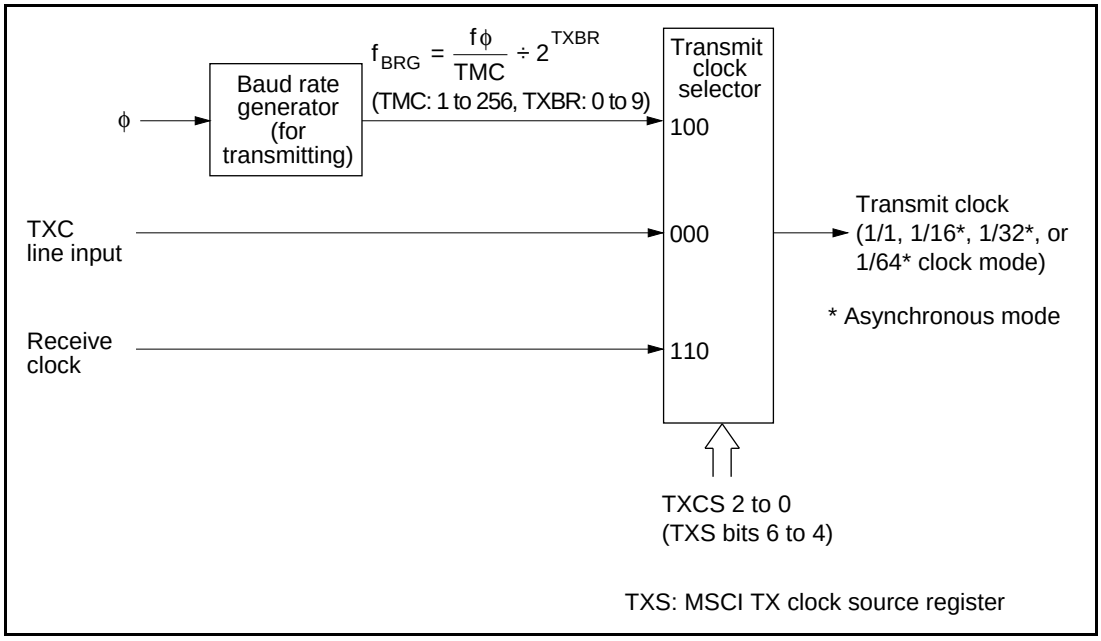


Figure 1.9 Transmit Clock Source

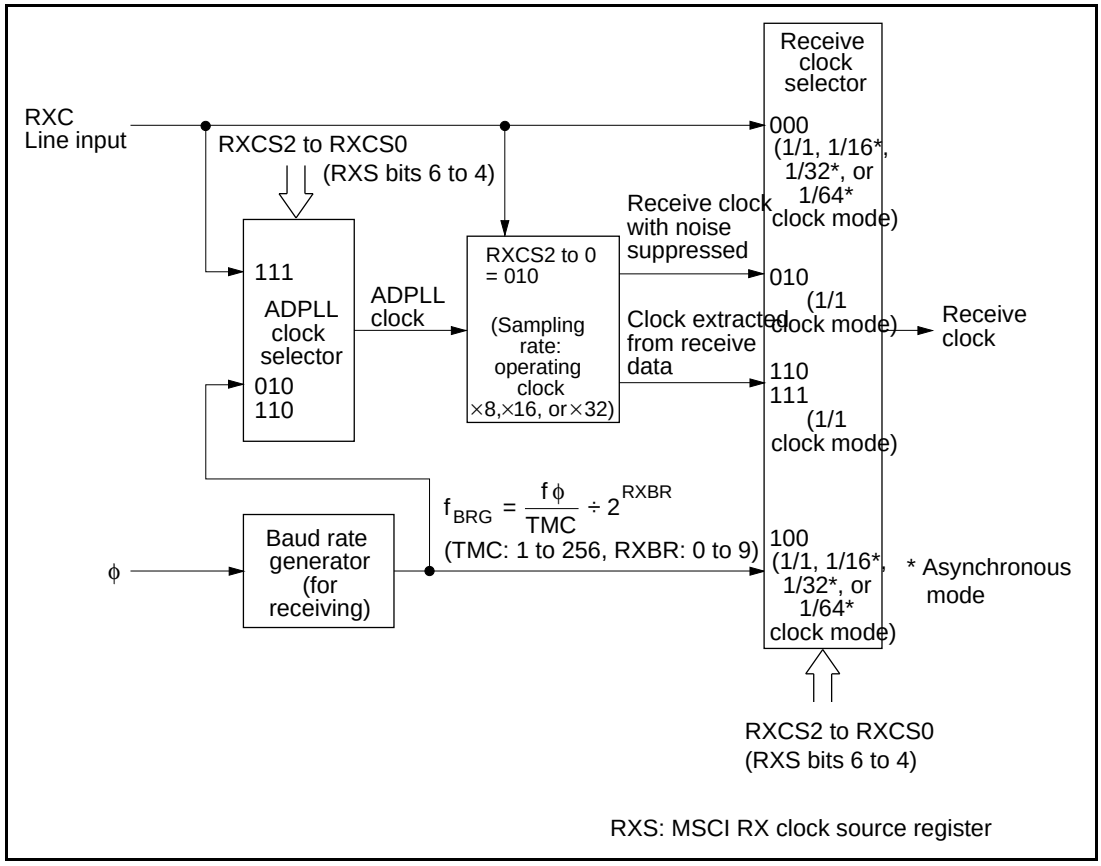


Figure 1.10 Receive Clock Source

1.7.6 Maximum Bit Rates

Table 1.2 lists the maximum bit rates supported by the MSCI in the SCA (when system clock (fϕ) = 10 MHz).

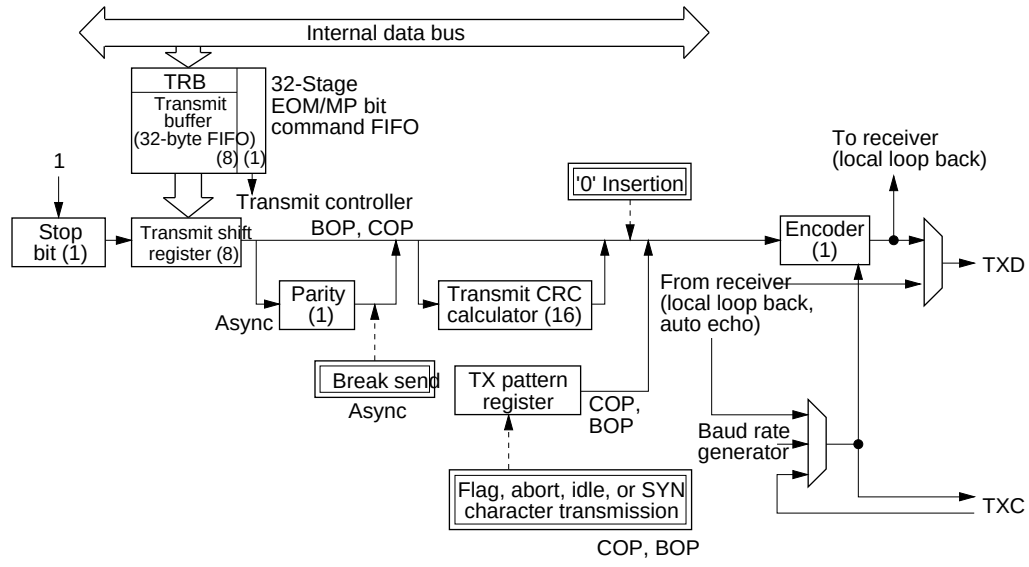
Table 1.2 Maximum Bit Rates

			Maximum Transfer Rate (bps)								
					Clock Extraction						
					Sampling Clock: External*1			Sampling Clock: BRG*2, 3		
Frequency (fϕ)	**Protocol Mode**	**Clock Mode*4**	**External Clock**	**BRG**	**× 8**	**× 16**	**× 32**	**× 8**	**× 16**	**× 32**
10 MHz*5	Asynchronous	1/64	62.5k*6	78.1k*7	—	—	—	—	—	—
		1/32	125k*6	156.3k*7	—	—	—	—	—	—
		1/16	250k*6	312.5k*7	—	—	—	—	—	—
		1/1	4.0M*6	3.3M*8	—	—	—	—	—	—
	Byte synchronous	1/1	7.1M*9	5M*7	2.2M	1.1M	0.55M	1.25M	0.62M	0.31M
	Bit synchronous HDLC mode	1/1	7.1M*9	5M*7	2.2M	1.1M	0.55M	1.25M	0.62M	0.31M
16.7 MHz*10	Asynchronous	1/64	104k*6	130k*7	—	—	—	—	—	—
		1/32	208k*6	260k*7	—	—	—	—	—	—
		1/16	416k*6	521k*7	—	—	—	—	—	—
		1/1	6.67M*6	5.56M*8	—	—	—	—	—	—
	Byte synchronous	1/1	12M*9	8.3M*7	2.2M	1.1M	0.55M	2.08M	1.04M	0.52M
	Bit synchronous HDLC mode	1/1	12M*9	8.3M*7	2.2M	1.1M	0.55M	2.08M	1.04M	0.52M

Notes:
1. 17.6 Mbps ÷ (sampling clock multiplier)
2. fϕ ÷ (sampling clock multiplier)
3. Same maximum transfer rate when receive clock noise is suppressed
4. Depends on setting of MSCI mode register 1 (MD1)
5. SCA (HD64570CP, HD64570F)
6. fϕ ÷ 2.5 × (clock mode)
7. fϕ ÷ 2 × (clock mode)
8. fϕ ÷ 3
9. fϕ ÷ 1.4 × (clock mode)
10. High-speed SCA (HD64570CP16, HD64570F16)

1.7.7 Transmitter

The transmitter (figure 1.11) loads parallel data supplied from the data bus into a transmit FIFO consisting of 32 eight-bit registers. Next, according to the selected transmission format, it moves the data into a transmit shift register which converts them to serial data. Data are transmitted LSB first.



Numbers in parentheses are bit lengths.

TRB: MSCI TX/RX buffer register
Async: Asynchronous mode
COP: Byte-synchronous mode
BOP: Bit-synchronous mode
‡ : Transmit data flow

Figure 1.11 Block Diagram of the MSCI Transmitter

1.7.8 Receiver

The receiver (figure 1.13) converts serial receive data into parallel data according to the selected communication format. The LSB of the data is received first. The data are shifted through a succession of receive shift registers, the last of which is the eight-bit receive shift register 4 (figure 1.12).

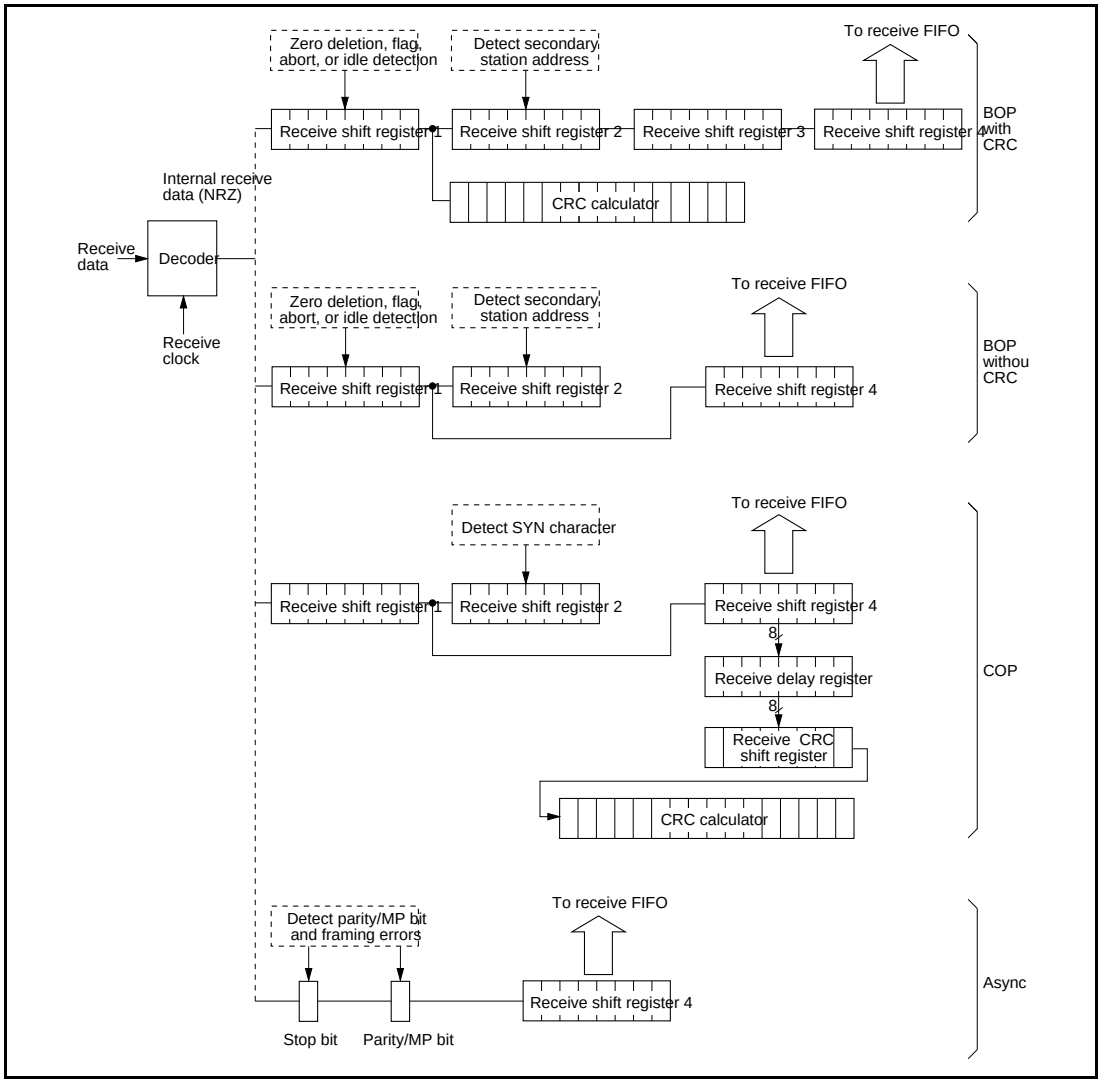
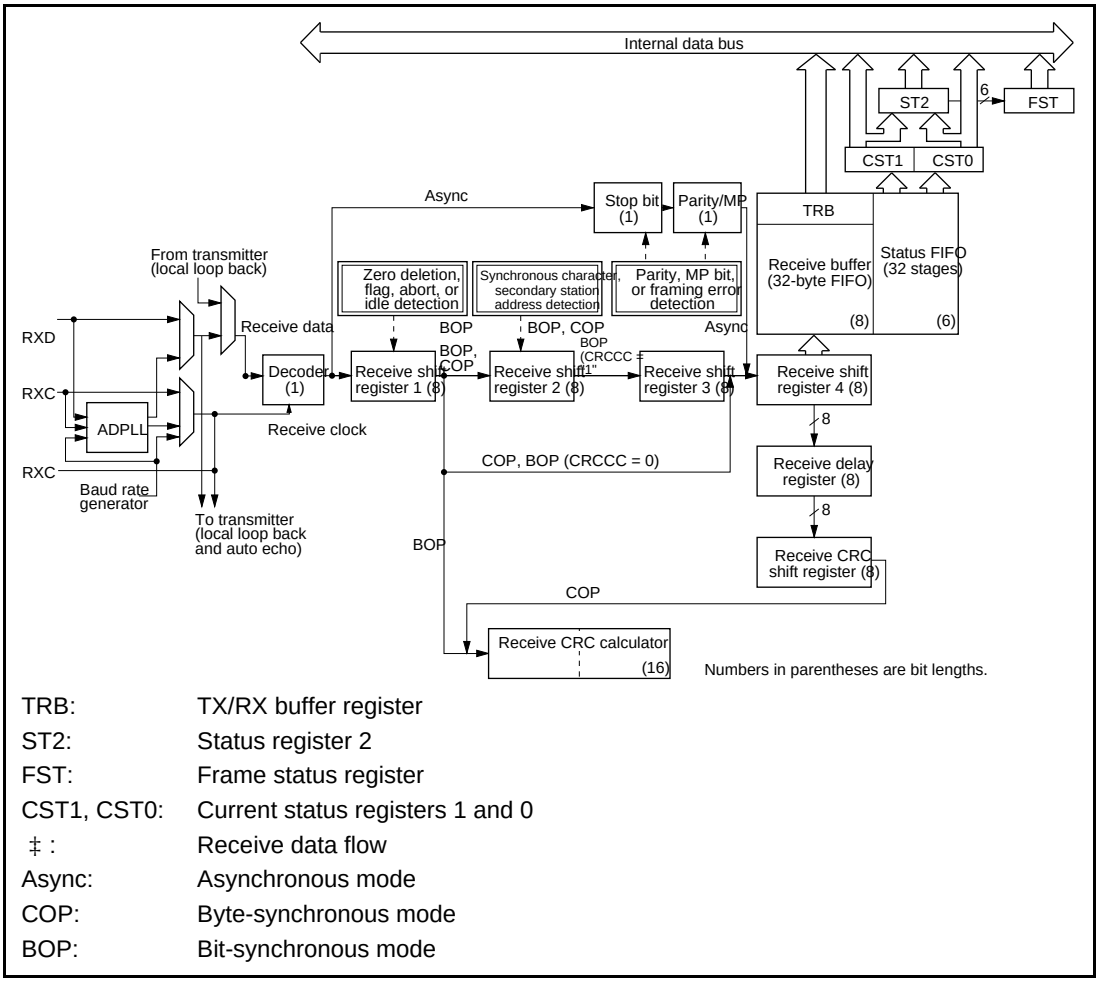


Figure 1.12 Receive Data Path (in each protocol mode)



TRB:	TX/RX buffer register
ST2:	Status register 2
FST:	Frame status register
CST1, CST0:	Current status registers 1 and 0
‡ :	Receive data flow
Async:	Asynchronous mode
COP:	Byte-synchronous mode
BOP:	Bit-synchronous mode

Figure 1.13 Block Diagram of the MSCI Receiver

1.7.9 DMAC

The DMAC module in the SCA has four independent channels (figure 1.14). The DMAC is used exclusively for single-address transfer between memory and the MSCI. Data can be transferred a word at a time or a byte at a time. A group of data transferred consecutively is referred to as a block. The DMAC can transfer single blocks individually, or can transfer a chained sequence of blocks (chained-block transfer). See figure 1.15 to 1.20.

Features:

- Four independent DMA channels
- Programmable channel priority
- Used exclusively for transfer between memory and MSCI
- Single-block transfer or chained-block transfer (supported by buffer management function)
- Two interrupt sources
- Maximum transfer rate: 11.1 Mbytes/s (operating on 16.7 MHz clock)
- Address space: 16 Mbytes

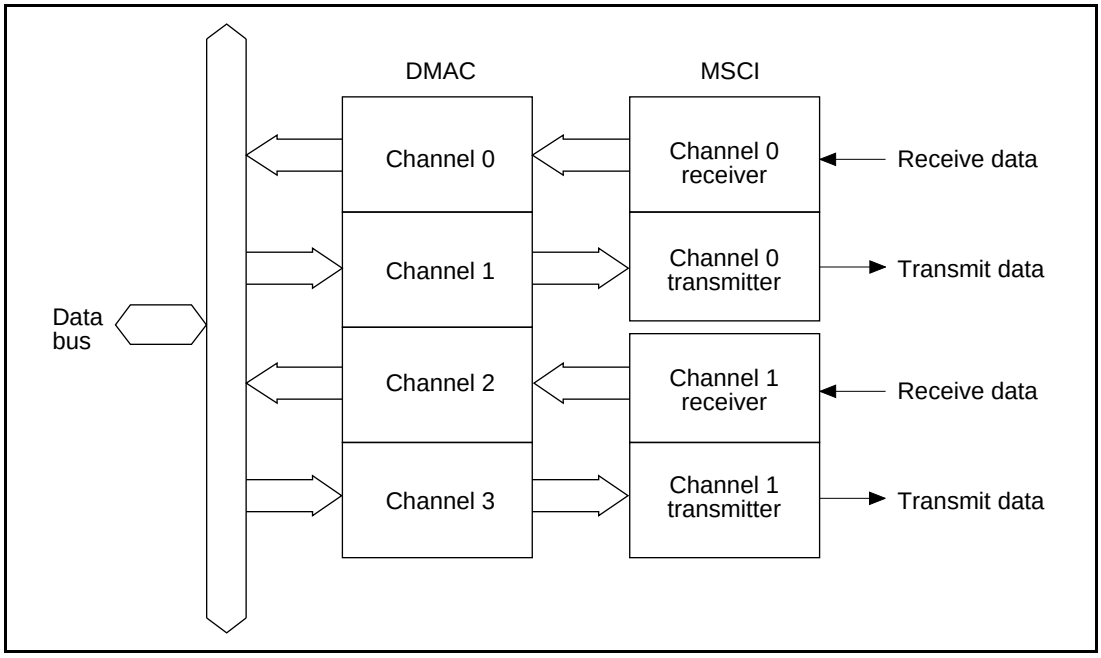


Figure 1.14 Interconnections between DMA Channels and MSCI Channels (concept)

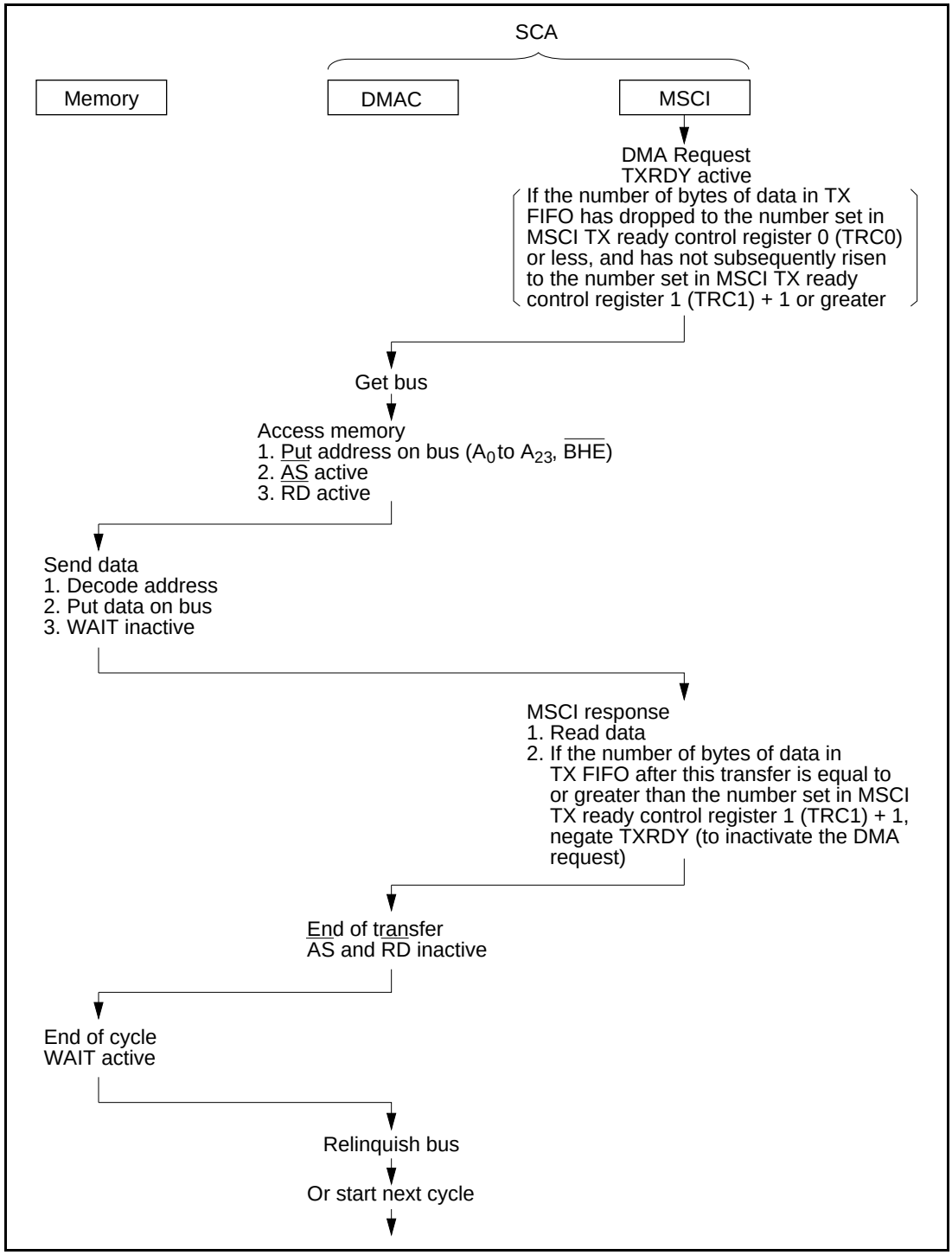


Figure 1.15 Transmit DMA Operation (CPU mode 0)

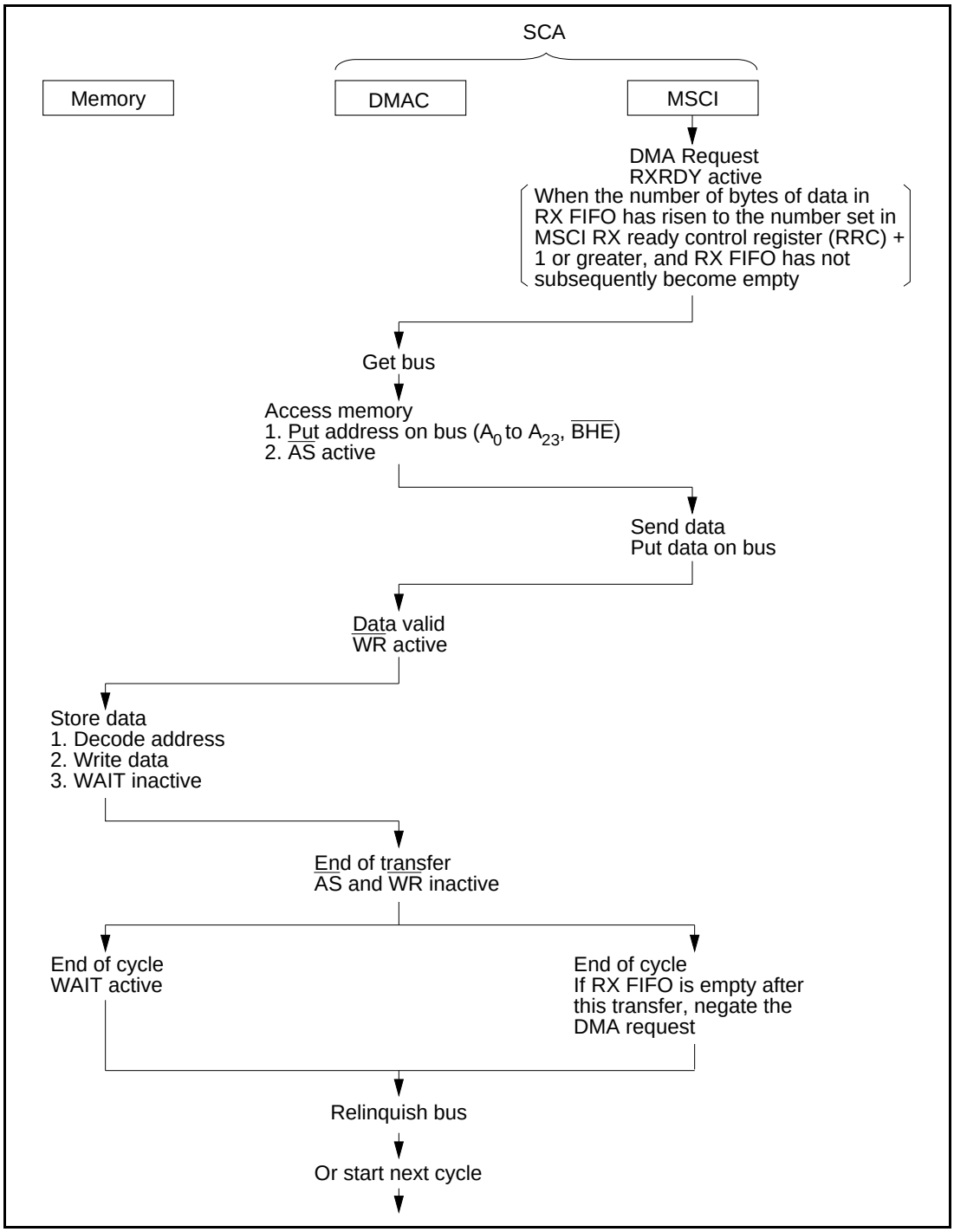


Figure 1.16 Receive DMA Operation (CPU mode 0)

HITACHI

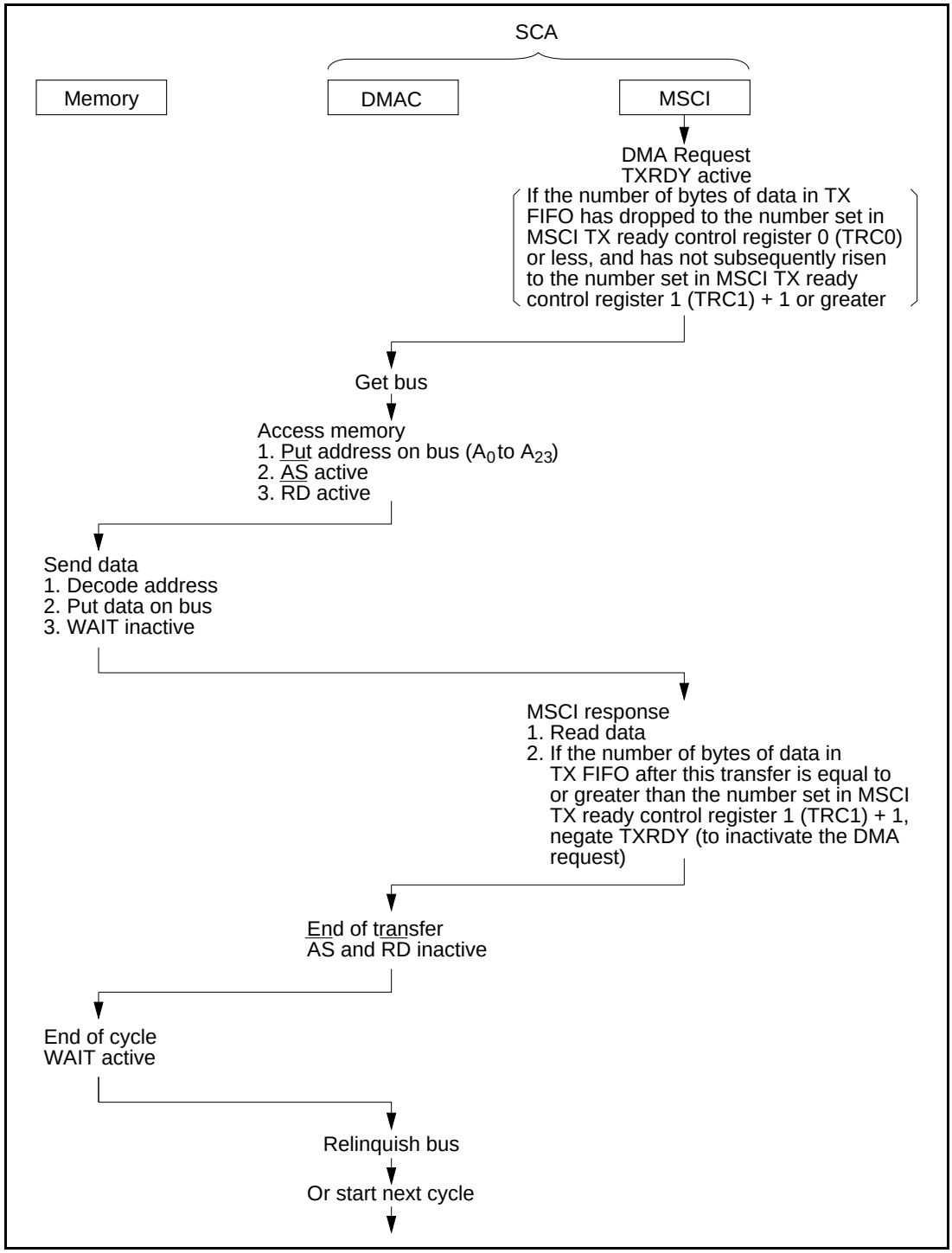


Figure 1.17 Transmit DMA Operation (CPU mode 1)

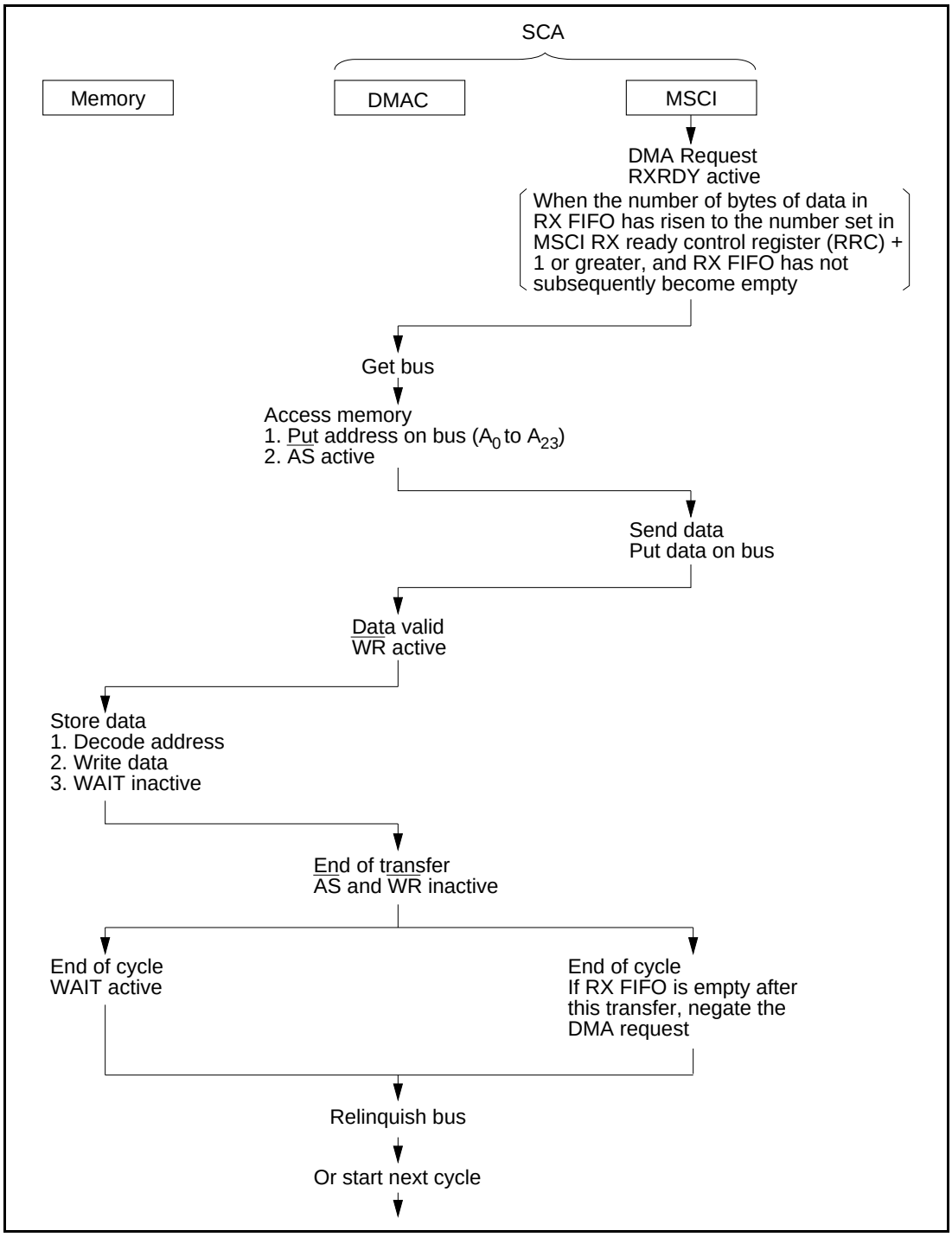


Figure 1.18 Receive DMA Operation (CPU mode 1)

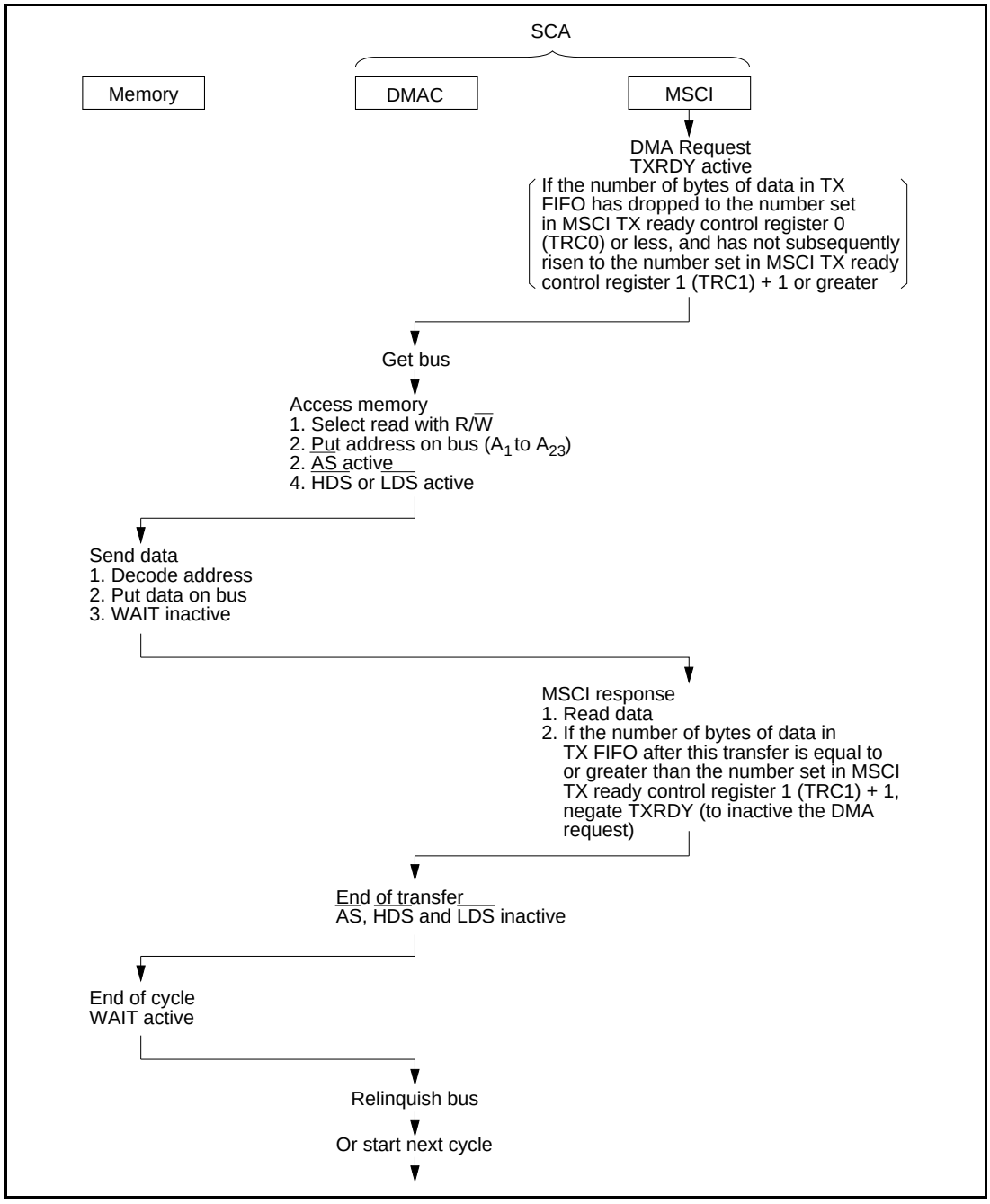


Figure 1.19 Transmit DMA Operation (CPU modes 2 and 3)

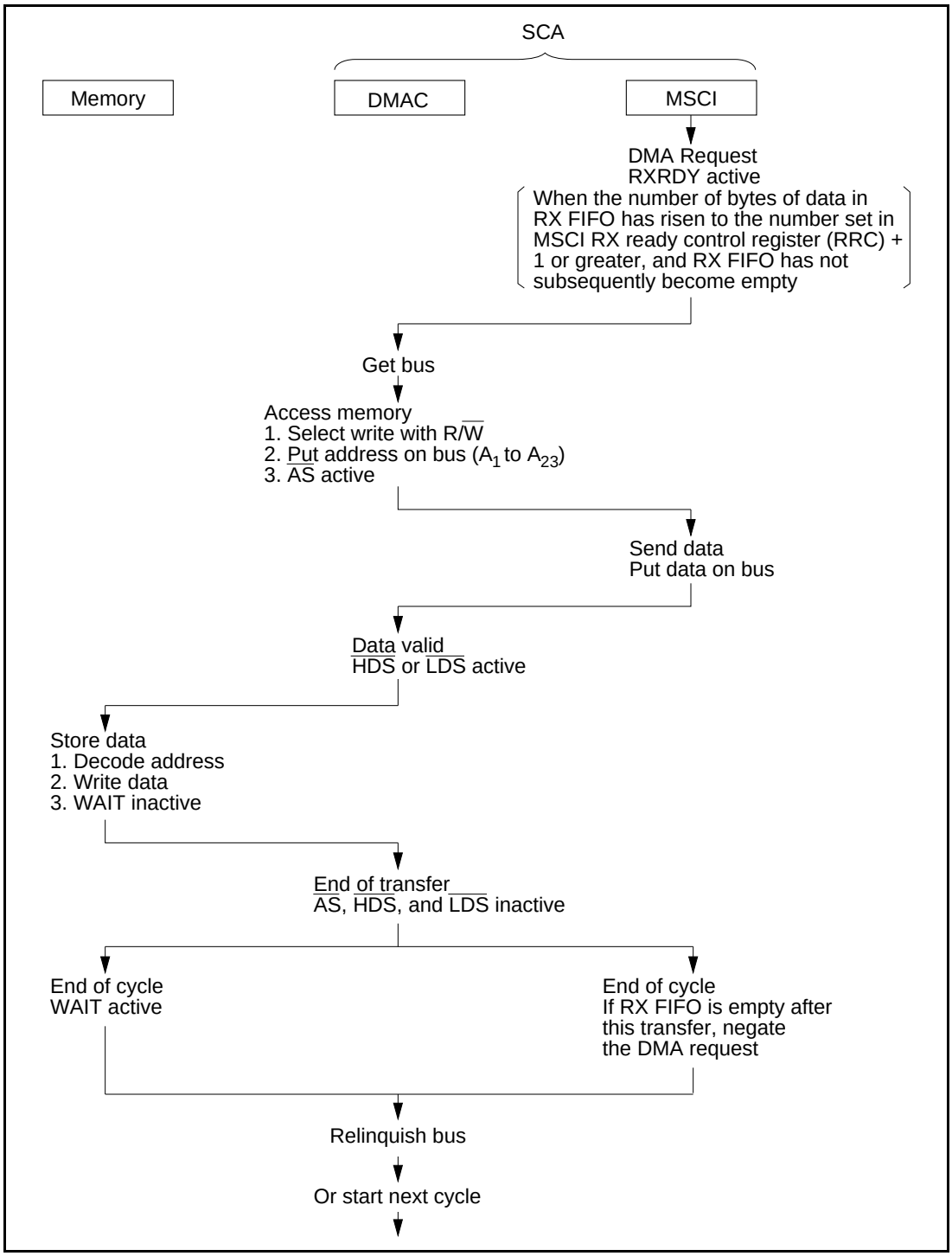


Figure 1.20 Receive DMA Operation (CPU modes 2 and 3)

1.7.10 DMA Buffer Chaining

In bit-synchronous mode, each DMAC channel in the SCA can perform chained-block transfer, in which one or more data blocks are transferred in a continuous sequence (figure 1.21).

To set up a chained-block transfer:

1. Create one or more descriptors in memory. A descriptor is a string of data giving the starting address of a data buffer (data block), the data length, the starting address of the next descriptor, and other information.
2. Write the starting address of the first descriptor in a DMAC register.
3. Set necessary values in other DMAC registers.
4. Enable the corresponding DMA channel.

On the DMA request from the MSCI, the DMAC will automatically fetch the first descriptor and begin chained-block transfer.

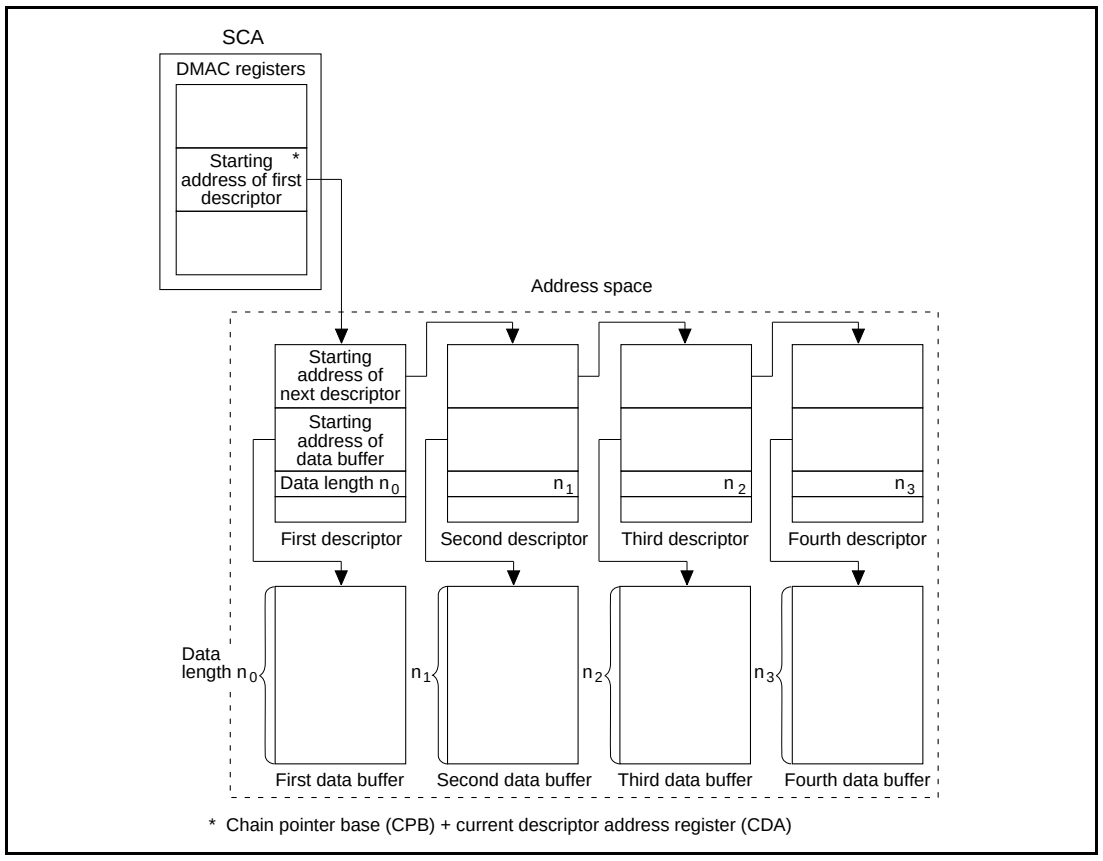


Figure 1.21 DMA Buffer Chaining

1.7.11 Descriptor Structure

Figure 1.22 shows the structure of a descriptor. Descriptors are allocated in memory in different ways, depending on the CPU mode.

Address	Bit 7 — Bit 0
2n + 9	Reserved (Note)
2n + 8	Status (ST) 8 bits
2n + 7	Data length H (DLH) 8 bits
2n + 6	Data length L (DLL) 8 bits
2n + 5	Reserved (Note)
2n + 4	Buffer pointer B (BPB) 8 bits
2n + 3	Buffer pointer H (BPH) 8 bits
2n + 2	Buffer pointer L (BPL) 8 bits
2n + 1	Chain pointer H (CPH) 8 bits
2n	Chain pointer L (CPL) 8 bits

1. CPU Mode 1

Address	Bit 15 — Bit 8	Bit 7 — Bit 0	Address
2n + 9	Reserved (Note)	Status (ST)	2n + 8
2n + 7	Data length H (DLH)	Data length L (DLL)	2n + 6
2n + 5	Reserved (Note)	Buffer pointer B (BPB)	2n + 4
2n + 3	Buffer pointer H (BPH)	Buffer pointer L (BPL)	2n + 2
2n + 1	Chain pointer H (CPH)	Chain pointer L (CPL)	2n

2. CPU Mode 0

Address	Bit 15 — Bit 8	Bit 7 — Bit 0	Address
2n + 8	Reserved (Note)	Status (ST)	2n + 9
2n + 6	Data length H (DLH)	Data length L (DLL)	2n + 7
2n + 4	Buffer pointer H (BPH)	Buffer pointer L (BPL)	2n + 5
2n + 2	Reserved (Note)	Buffer pointer B (BPB)	2n + 3
2n	Chain pointer H (CPH)	Chain pointer L (CPL)	2n + 1

3. CPU Modes 2, 3

Note: Reserved fields in descriptors are not written by the DMAC, but retain their previous values. The MPU may write in these fields, but this does not affect the DMAC.
Descriptors must start at an even address (2n). Correct operation is not assured if a descriptor starts at an odd address.

Figure 1.22 Descriptor Structure

1.7.12 Bus Arbiter

The SCA has a built-in bus arbiter for arbitrating the bus between the on-chip DMAC and an external bus master device. This bus arbiter provides an easy way to design a multi-channel system using two or more SCA chip (figure 1.23). The circuit shown here merely represents the concept of the multi-channel system. In practice, the user system should be designed while referring to the bus arbitration sequence (figure 3.6).

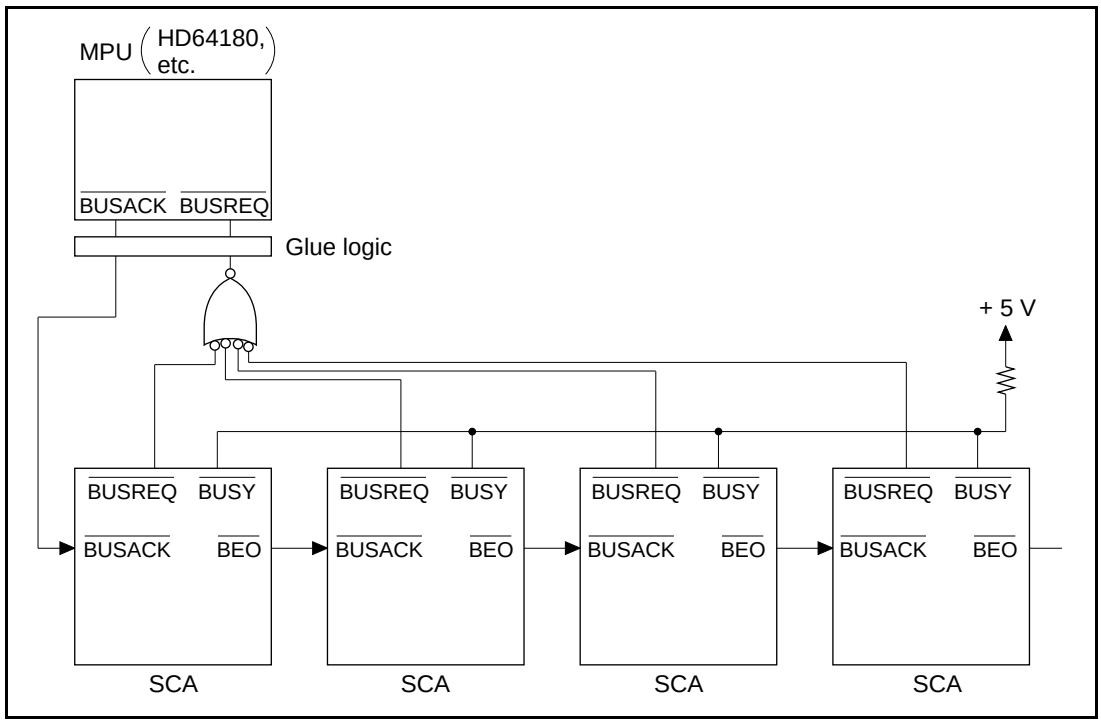


Figure 1.23 Daisy-Chained Multi-Channel System (example)

1.7.13 Interrupt Control

SCA interrupts are controlled by three interrupt status registers (ISR0, ISR1, ISR2) containing flags for 20 interrupt sources, three interrupt enable registers (IER0, IER1, IER2) which can mask the interrupt source flags individually, and one interrupt control register (ITCR) (figure 1.24).

The interrupt sources are located in corresponding functional modules (MSCI, DMAC, timers). When an interrupt source that is not masked becomes active, the SCA activates $\overline{\text{INT}}$ to request an MPU interrupt. When the MPU activates $\overline{\text{INTA}}$ in response, the SCA begins an acknowledge cycle and outputs an interrupt vector according to register settings.

The interrupt control register (ITCR) selects the interrupt priority order, the type of acknowledge cycle, and the type of vector output. ITCR bits IAK0 and IAK1 select the non-acknowledge, single acknowledge, or double acknowledge cycle. Nonacknowledge means that the SCA does not output an interrupt vector when $\overline{\text{INTA}}$ is activated. Single acknowledge means that the SCA outputs a vector the first time $\overline{\text{INTA}}$ is activated. Double acknowledge means that the SCA outputs a vector the second time $\overline{\text{INTA}}$ is activated.

ITCR bit VOS selects whether to output the contents of the interrupt vector register (IVR) or interrupt modified vector register (IMVR) as the vector. Any value can be set in IVR for unmodified output as the vector in the acknowledge cycle. The highest two bits of IMVR can be set to any value, but the lower six bits hold a hardware-generated code representing the interrupt source. If multiple interrupt sources are active simultaneously, IMVR holds the code of the source with the highest priority. ITCR bit IPC can switch the relative priority of the MSCI and DMAC.

See figures 1.25 to 1.27 for interrupt logic flow for MSCI, DMAC, and timer modules.

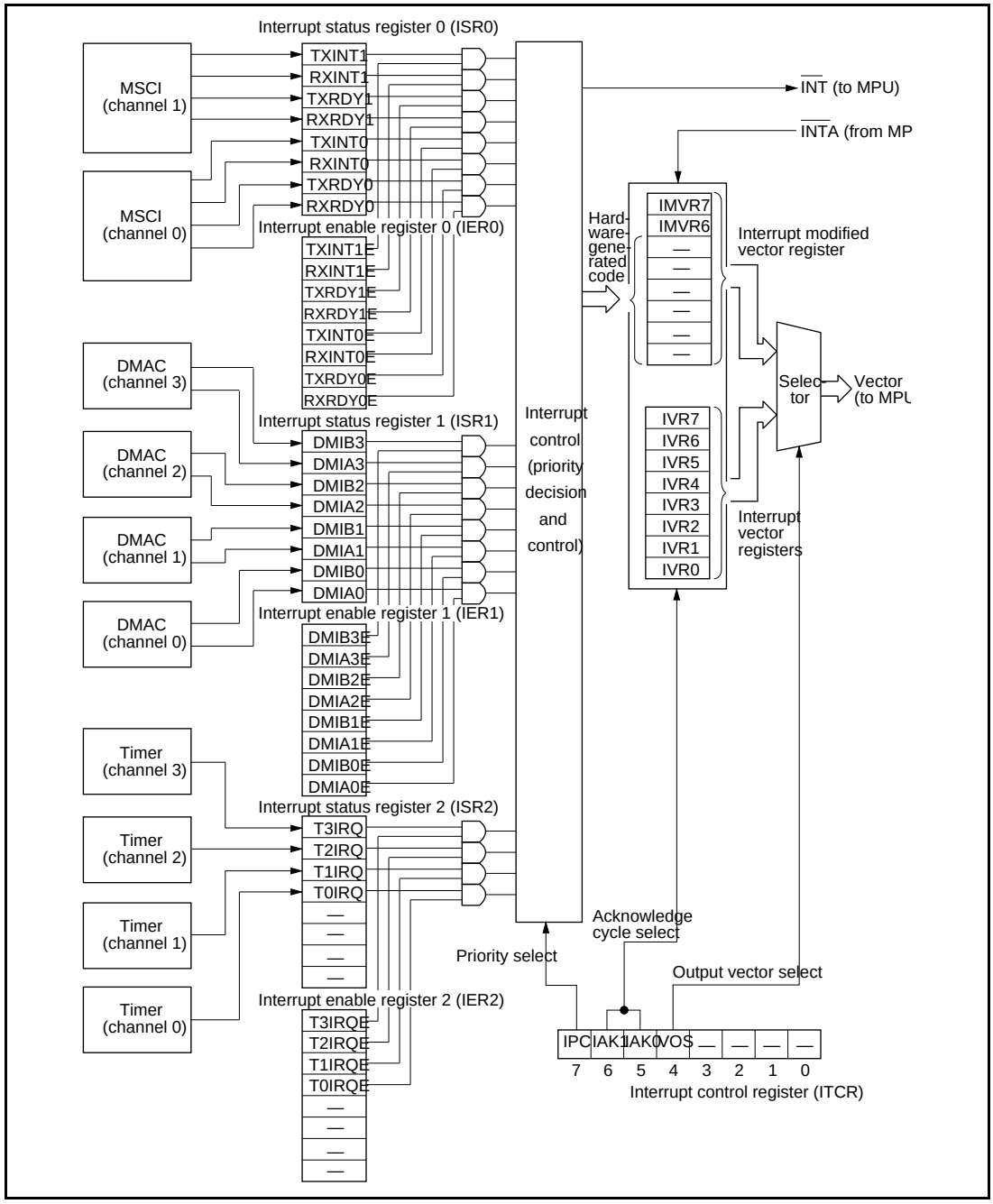


Figure 1.24 Interrupt Control

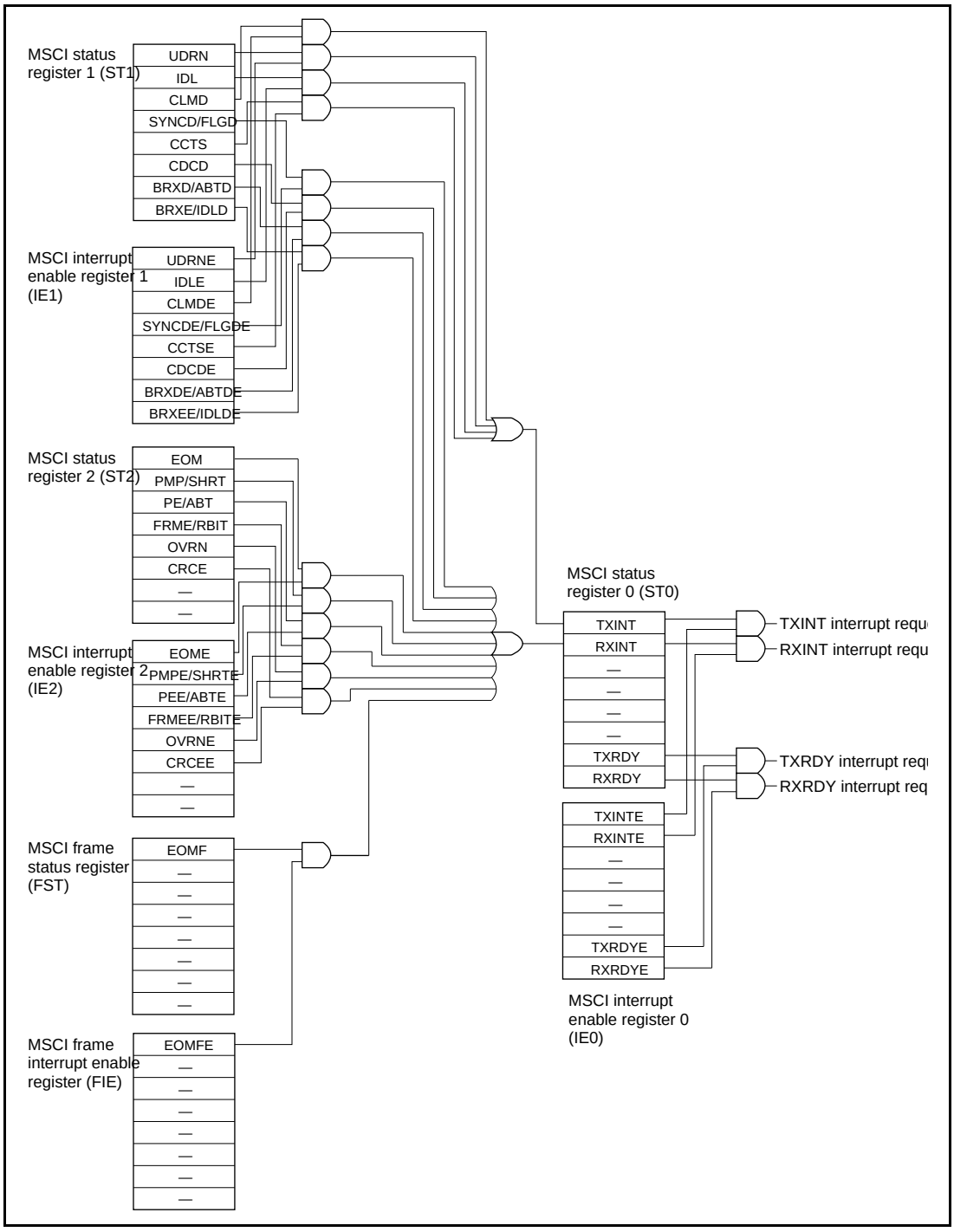


Figure 1.25 Logic Flow for Interrupt Requests, Status, and Enable Bits in MSCI Module

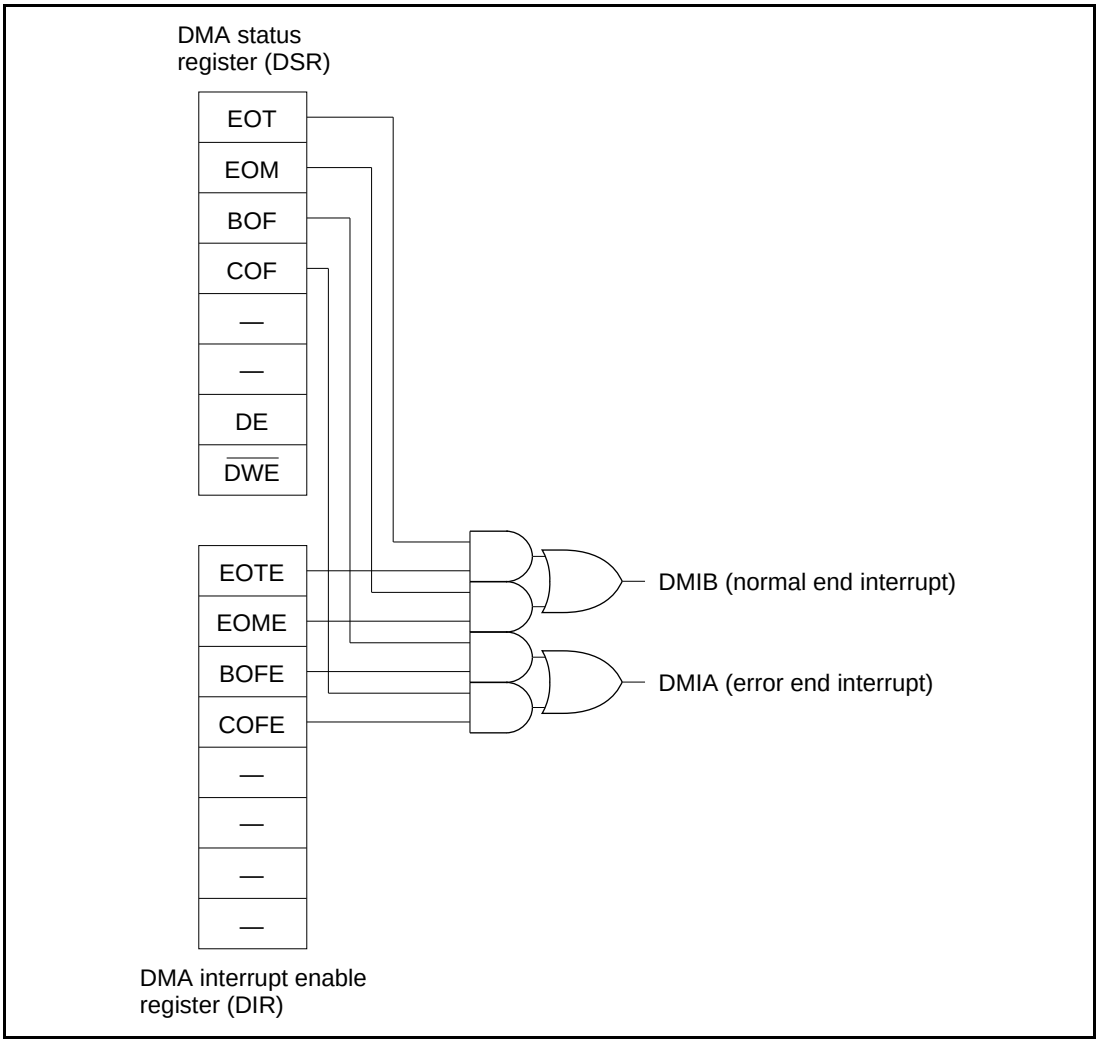


Figure 1.26 Logic Flow for Interrupt Requests, Status, and Enable Bits in DMAC Module

Figure 1.27 Logic Flow for Interrupt Requests, Status, and Enable Bits in Timer Module

1.7.14 Timers

The SCA has a built-in four-channel, 16-bit timer module. All channels have identical functions and specifications. They can be used as interval timers or watchdog timers, or for time-out detection or other purposes. The timer features are listed below.

- Each timer uses a 16-bit reloadable up-counter.
- The timer increments at a rate of BC/20 to BC/27, where BC is a base clock obtained by dividing the internal system clock (ϕ) by eight.
- A counter generates interrupt when it reaches a specified value.

1.7.15 Wait Controller

The SCA has a built-in wait controller. The wait controller can insert wait states to lengthen the DMA bus cycle.

The address space is divided into three areas (figure 1.28). The number of wait states inserted when each area is accessed can be set independently in the range from 0 to 7. This enables the SCA to support memory chips having different access times without requiring external wait control logic.

Wait states can also be inserted by the WAIT input. In this case there is no limit on the number of wait states inserted. In cases of conflict between the number of wait states set in the wait control register and the number requested via the WAIT line, the larger number of wait states is inserted.

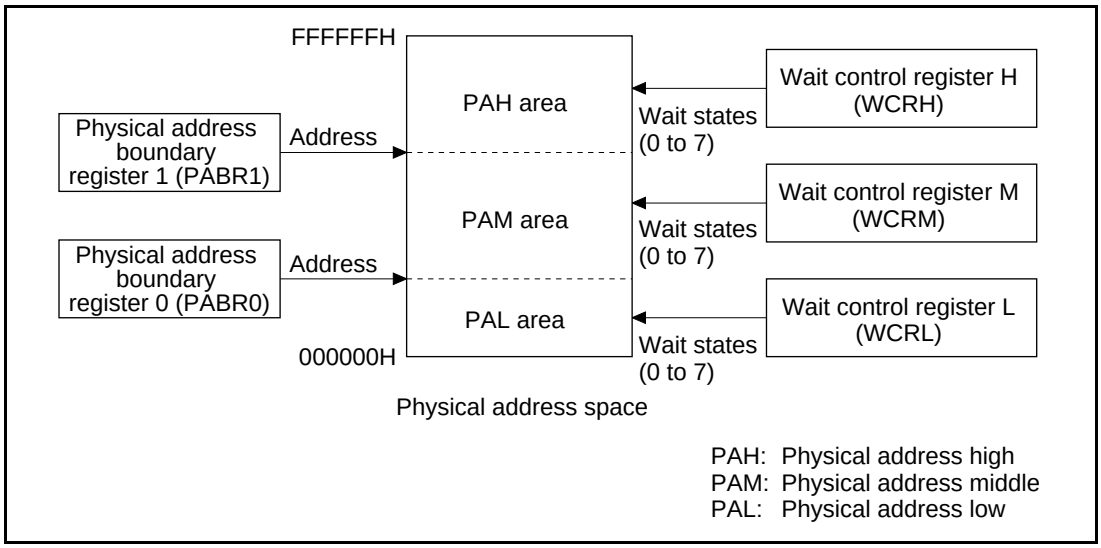


Figure 1.28 Subdivision of Address Space by Wait Controller and Insertion of Wait States

Section 2 Pin Arrangements and Functions

2.1 Pin Arrangements

Figures 2.1 and 2.2 show the pin arrangements of the SCA chip in QFJ (PLCC (CP-84)) and QFP (FP-88) packages, respectively.

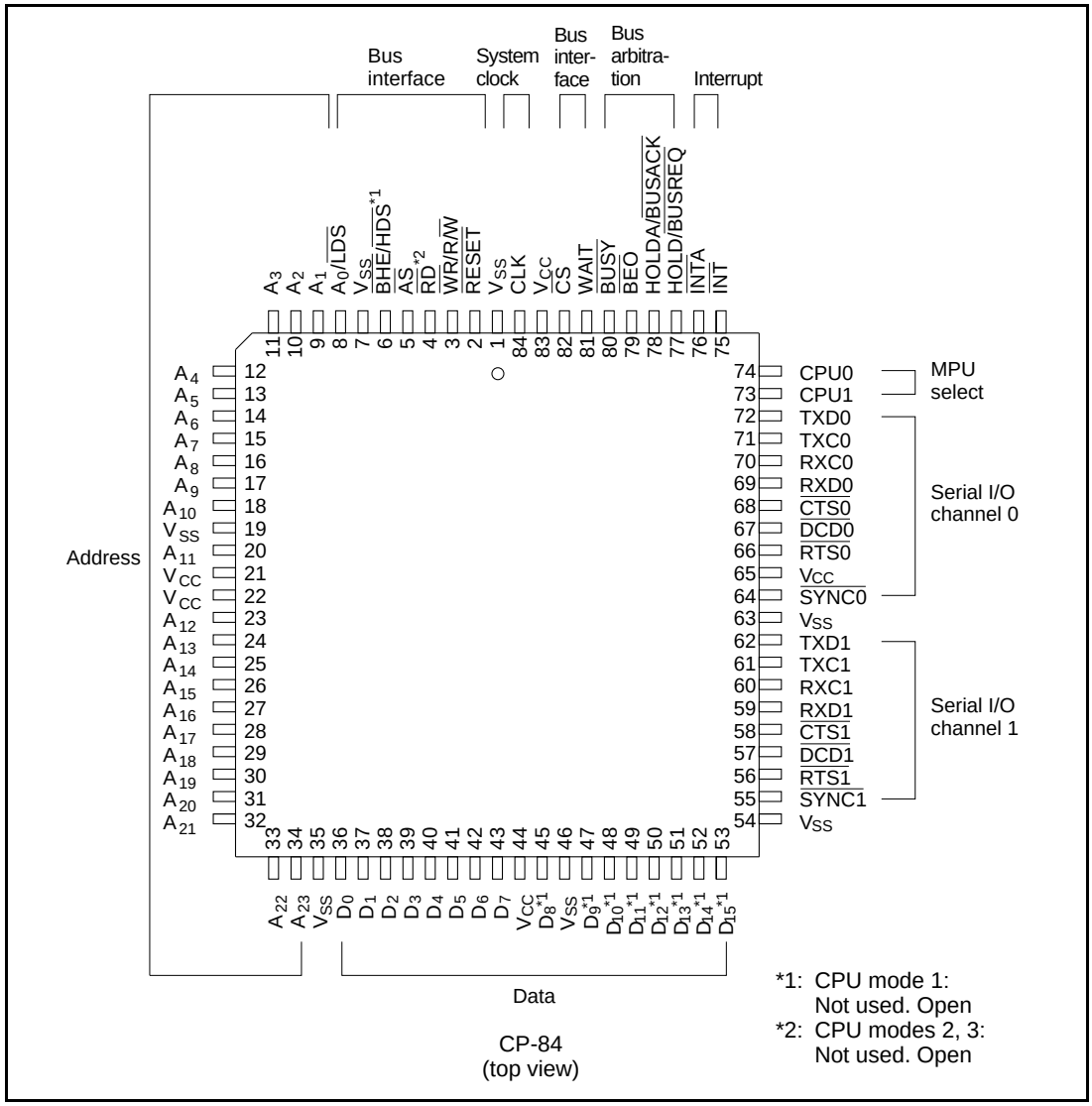


Figure 2.1 Pin Arrangement of CP-84

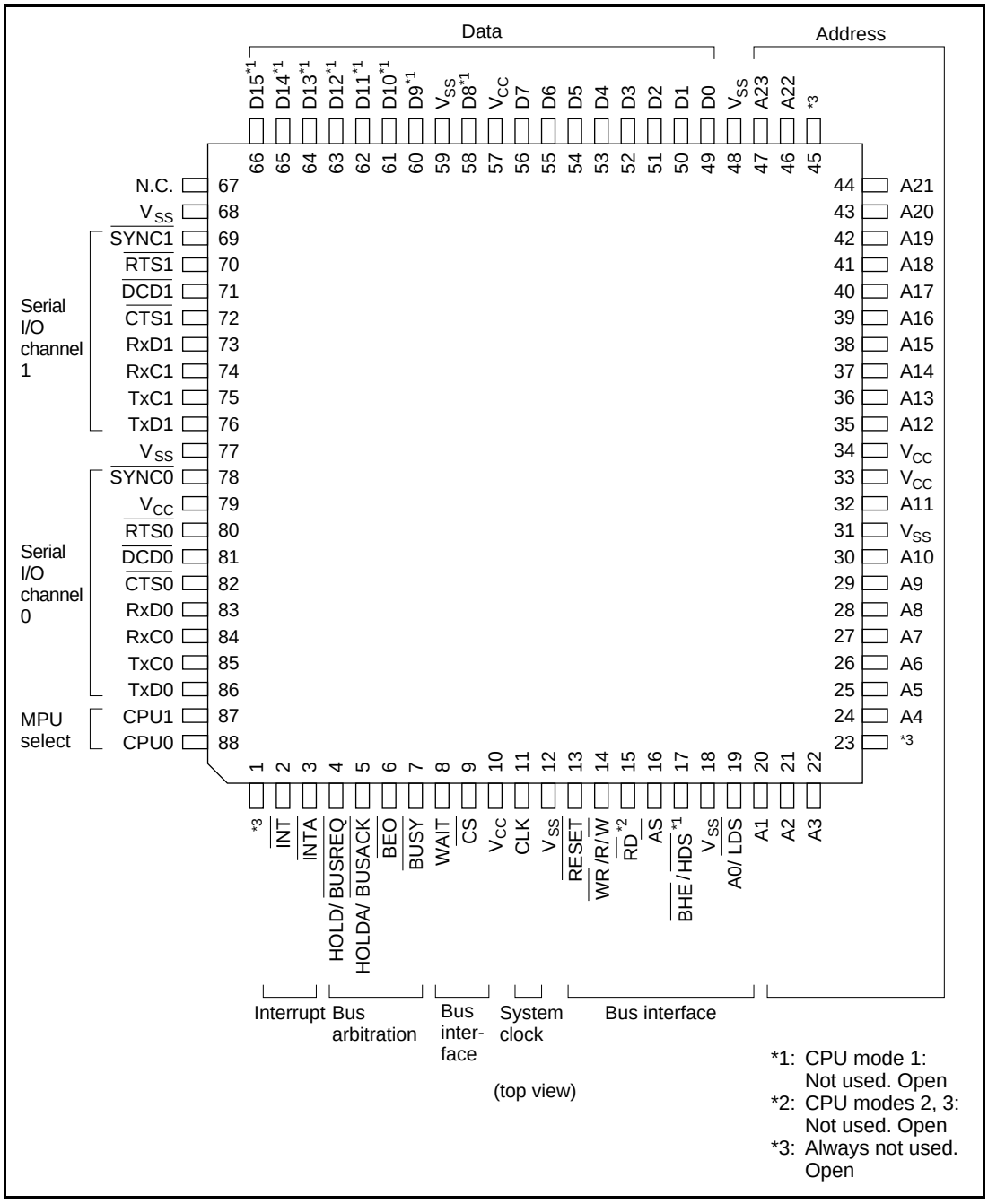


Figure 2.2 Pin Arrangement on FP-88

2.2 Pin Functions

The function of each signal line is described below. Note that permanent or temporary input lines must never be left unconnected.

Table 2.1 Power Supply

Symbol	Pin Number CP-84	Pin Number FP-88	Input/ Output	Description
V_{CC}	21, 22, 44, 65, 83	10, 33, 34 57, 79	Input	+5 V power supply: All VCC pins must be connected to the system power supply (+5 V).
V_{SS}	1, 7, 19, 35, 46, 54, 63	12, 18, 31, 48, 59, 68, 77	Input	Ground: All VSS pins must be connected to the system ground (0 V).

Note: To minimize potential differences in the chip, use the shortest possible lead length to the V_{CC} pins and V_{SS} pins.

Table 2.2 Clock

Symbol	Pin Number CP-84	Pin Number FP-88	Input/ Output	Description
CLK	84	11	Input	System clock input: The input is reshaped on chip and used as the ϕ clock.

Table 2.3 Reset Line

Symbol	Pin Number CP-84	Pin Number FP-88	Input/ Output	Description
$\overline{\text{RESET}}$	2	13	Input	Reset: When this line is driven active low for 6 or more clock cycles, the SCA enters reset mode. All functional modules are reset.

Table 2.4 Address Lines

Symbol	Pin Number CP-84	Pin Number FP-88	Input/ Output	Description	
A_8–A_{23}	16–18, 20, 23–34	28–30, 32, 35–44, 46, 47	Output (Three-state)	Address bus:	
				Master mode	Output lines for the 16 high-order bits of a 24-bit address when DMAC accesses a 16-Mbyte address space.
				Slave mode	High impedance.
				Reset mode	High impedance.
				System stop mode	High impedance.
A_1–A_7	9–15	20–22, 24–27	Input/ output	Address bus:	
				Master mode	Output lines for the 7 low-order bits of a 24-bit address when DMAC accesses a 16-Mbyte address space.
				Slave mode	Internal register address input.
				Reset mode	Input.
				System stop mode	Input.

Table 2.5 Data Lines

Symbol	Pin Number CP-84	Pin Number FP-88	Input/ Output	Description	
D_0–D_7	36–43	49–56	Input/ output (Three-state	Data bus:	
				CPU modes 0–3	
				Normal mode	Input/output lines for the 8 low- order bits of the 16-bit bidirectional data bus.
				Reset mode	High impedance.
				System stop mode	High impedance.
D_8*1–D_{15}*1	45, 47–53	58, 60–66	Input/ output (Three-state)	Data bus:	
				CPU modes 0, 2, 3	
				Normal mode	Input/output lines for the 8 high-order bits of the 16-bit bidirectional data bus.
				Reset mode	High impedance.
				System stop mode	High impedance.
				CPU mode 1	
				All modes	*1: Always high. Leave unconnected, or pull up to V_{CC}.

*1: Not connected

Table 2.6 Bus Interface Lines

Symbol	Pin Number CP-84	Pin Number FP-88	Input/ Output	Description	
$\overline{RD}$*2	4	15	Input/ output	CPU modes 0, 1	
				Read: Indicates that the SCA is executing a read cycle.	
				Master mode	Output line. When this line is driven active low, the data bus lines are used as inputs.
				Slave mode	Input line. When this line is driven active low, the data bus lines are used as outputs.
				Reset mode	Input.
				System stop mode	Input.
				CPU modes 2, 3	
				*2	Always high. Leave unconnected, or pull up to V_{CC}.
$\overline{WR}$/R$\overline{W}$	3	14	Input/ output	CPU modes 0, 1	
				Write: Indicates that the SCA is executing a write cycle.	
				Master mode	Output line. When this line is driven active low, the data bus lines are used as outputs.
				Slave mode	Input line. When this line is driven active low, the data bus lines are used as inputs.
				Reset mode	Input.
				System stop mode	Input.
				CPU modes 2, 3	
				Read/write: Indicates whether the SCA is executing a read or write cycle, to control the data direction.	

*2: Not connected

Table 2.6 Bus Interface Lines (cont)

Symbol	Pin Number CP-84	Pin Number FP-88	Input/ Output	Description	
$\overline{WR}$/R/$\overline{W}$	3	14	Input/ output	Master mode	Output line. When this line is driven high, data moves in the input direction. When this line is driven low, data moves in the output direction.
				Slave mode	Input line. When this line is driven high, data moves in the output direction. When this line is driven low, data moves in the input direction.
				Reset mode	Input.
				System stop mode	Input.
A_0 /$\overline{LDS}$	8	19	Input/ output	CPU modes 0, 1	
				Address: Least significant bit of the address bus.	
				Master mode	Output.
				Slave mode	Input.
				Reset mode	Input.
				System stop mode	Input.
				CPU modes 2, 3	
				Lower data strobe: Strobe timing for the low-order data bits.	
				Master mode	Output.
				Slave mode	Input.
				Reset mode	Input.
				System stop mode	Input.

Table 2.6 Bus Interface Lines (cont)

Symbol	Pin Number CP-84	Pin Number FP-88	Input/ Output	Description	
$\overline{BHE}$/$\overline{HDS}$*1	6	17	Input/ output	CPU mode 0	
				Bus high enable: High-order byte access signal.	
				Master mode	Output.
				Slave mode	Input.
				Reset mode	Input.
				System stop mode	Input.
				CPU mode 1	
				*1	Always high. Leave unconnected, or pull up to V_{CC}.
				CPU modes 2, 3 Higher data strobe: Strobe timing for the high-order data bits.	
				Master mode	Output.
				Slave mode	Input.
				Reset mode	Input.
				System stop mode	Input.
$\overline{CS}$	82	9	Input/ output	Chip select: Selects the SCA.	
				CPU modes 0–3	
				Master mode	Input, but ignored by the SCA.
				Slave mode	Indicates access by a host MPU. An internal register read/write cycle starts when this line is driven active low.
				Reset mode	Input, but ignored by the SCA.
				System stop mode	Input, but ignored by the SCA.

*1: Not connected

Table 2.6 Bus Interface Lines (cont)

Symbol	Pin Number CP-84	Pin Number FP-88	Input/ Output	Description	
WAIT	81	8	Input/ output	Wait: Used to extend read and write cycles.	
				CPU mode 0	
				Master mode	If this line is high at the rising edge of a T_2 state, a T_W state is inserted. If the line is still high at the rising edge of the next T_W state, another T_W state is inserted. If this line is low at the rising edge of a T_2 or TW state, the next state is a T_3 state.
				Slave mode	Output line. This line is driven high to request the host MPU to extend the bus cycle.
				Reset mode	High output.
				System stop mode	High output.
				CPU modes 1, 2, 3	
				Master mode	If this line is high at the falling edge of a T_2 state, a T_W state is inserted. If the line is still high at the falling edge of the next T_W state, another T_W state is inserted. If this line is low at the falling edge of a T_2 or T_W state, the next state is a T_3 state.
				Slave mode	Output line. This line is driven high to request the host MPU to extend the bus cycle.
				Reset mode	High output.
				System stop mode	High output.

Table 2.6 Bus Interface Lines (cont)

Symbol	Pin Number CP-84	Pin Number FP-88	Input/ Output	Description	
$\overline{AS}$	5	16	Input/ output	Address strobe: Indicates whether the address bus is active.	
				CPU modes 0,1	
				Master mode	Output line. The address bus (A_0–A_{23}) is valid when this line is driven active low.
				Slave mode	Input, but ignored by the SCA.
				Reset mode	Input.
				System stop mode	Input.
				CPU modes 2, 3	
				Master mode	Output line. The address bus (A_1–A_{23}) is valid when this line is driven active low.
				Slave mode	Input line. The SCA regards the address bits (A_1–A_7) as valid | when this line is driven active low.
				Reset mode	Input.
				System stop mode	Input.

Table 2.7 System Control Lines

Symbol	Pin Number CP-84	Pin Number FP-88	Input/ Output	Description
HOLD/ $\overline{\text{BUSREQ}}$	77	4	Output	CPU mode 0
				Hold: Used to request the bus. By driving HOLD active high, the SCA asks the host MPU to grant control of the bus.
				CPU modes 1, 2, 3
				Bus request: Used to request the bus. By driving $\overline{\text{BUSREQ}}$ active low, the SCA asks the host MPU to grant control of the bus.
HOLDA/ $\overline{\text{BUSACK}}$	78	5	Input	CPU mode 0
				Hold acknowledge: Used to indicate that the host MPU has received a HOLD signal and released the bus. When HOLDA is driven active high, the SCA assumes that control of the bus has been granted.
				If this line goes low (inactive) during a DMA transfer, the SCA releases control of the bus at the next bus cycle at which such release is permitted.
				CPU modes 1, 2, 3
				Bus acknowledge: Used to indicate that the host MPU has received a $\overline{\text{BUSREQ}}$ signal and released the bus. When $\overline{\text{BUSACK}}$ is driven active low, the SCA assumes that control of the bus has been granted.
				If this line goes high (inactive) during a DMA transfer, the SCA releases control of the bus at the next bus cycle at which such release is permitted.
$\overline{\text{BEO}}$	79	6	Output	Bus enable output: Used for daisy-chained bus arbitration. When the HOLD or $\overline{\text{BUSACK}}$ line is active, unless an internal DMA transfer request is present in the SCA, $\overline{\text{BEO}}$ is driven active low to pass the acknowledgment on to a lower-order device.

Table 2.7 System Control Lines (cont)

Symbol	Pin Number CP-84	Pin Number FP-88	Input/ Output	Description	
$\overline{\text{BUSY}}$	80	7	Input/ output (Open drain)	Bus busy: indicates that the bus is in use.	
				Slave mode	Input line. An external device drives $\overline{\text{BUSY}}$ active low to indicate that it is using the bus. When the SCA requests the bus after the HOLDA or $\overline{\text{BUSACK}}$ input becomes active, if $\overline{\text{BUSY}}$ is low (active), the SCA waits until $\overline{\text{BUSY}}$ goes high (inactive) before taking control of the bus.
				Master mode	The SCA drives this line active low as long as it holds the bus. When it finishes using the bus, the SCA drives this line high (inactive) and releases the bus after which this line becomes an input line.
				Note: Be sure to pull this line up to V_{CC}.	
CPU0, CPU1	74, 73	88, 87	Input	MPU select: Select the bus interface mode.	

CPU1	CPU0	CPU Mode
0	0	Mode 0 (8086-system 16-bit MPU mode)
1	0	Mode 1 (64180 mode)
0	1	Mode 2 (68000-system 16-bit MPU mode I)
1	1	Mode 3 (68000-system 16-bit MPU mode II)

Note: Do not change the mode when the SCA is turned on.

Table 2.8 Interrupt Lines

Symbol	Pin Number CP-84	Pin Number FP-88	Input/ Output	Description
$\overline{INT}$	75	2	Input/ output (Open drain)	Interrupt request: used to request an interrupt. The SCA drives $\overline{INT}$ active low when it has an interrupt request. Note: Be sure to pull this line up to V_{CC}.
$\overline{INTA}$	76	3	Input	Interrupt acknowledge: Used to acknowledge an interrupt. The SCA recognizes an interrupt acknowledge cycle when $\overline{INTA}$ goes active low.

Note: If no signal is input on this line in non-acknowledge mode, pull this line up to V_{CC}.

Table 2.9 Serial I/O (MSCI) Lines

There are two sets of serial I/O lines, for channels 0 and 1 respectively. The functions of both channels are the same (table 2.9).

Symbol	Pin Number CP-84	Pin Number FP-88	Input/ Output	Description
TXD0, TXD1	72, 62	86, 76	Output	Transmit data for MSCI: outputs transmit data from the MSCI.TXD1
RXD0, RXD1	69, 59	83, 73	Input	Receive data for MSCI: inputs receive data to the MSCI.RXD1
TXC0, TXC1	71, 61	85, 75	Input/ output	Transmit clock for MSCI: inputs or outputs the MSCI transmit clock. It has three programmable modes: Input: • External transmit clock Output: • Transmit clock from the onchip baud rate generator • Receive clock (used as the transmit clock)
RXC0, RXC1	70, 60	84, 74	Input/ output	Receive clock for MSCI: inputs or outputs the MSCI receive clock. This line can also be used to input the ADPLL operating clock. It has four programmable modes: Input: • External receive clock • ADPLL operating clock Output: • Receive clock extracted by the ADPLL (when the on-chip baud rate generator is used as the ADPLL operating clock) • Receive clock from the on-chip baud rate generator

Table 2.9 Serial I/O (MSCI) Lines (cont)

Symbol	Pin Number CP-84	Pin Number FP-88	Input/ Output	Description
$\overline{RTS0}$, $\overline{RTS1}$	66, 56	80, 70	Output	Request to send for MSCI: Indicates that the SCA has data to output to a communications device such as a modem. The output level on this line can be automatically controlled by MSCI operation (auto-enable function). This line can also be used as a general-purpose output port.
$\overline{DCD0}$, $\overline{DCD1}$	67, 57	81, 71	Input	Data carrier detect for MSCI: Indicates that a communications device such as a modem is receiving valid data from the communication line. MSCI receive operations can be automatically controlled by this input (auto-enable function). This line can also be used as a general-purpose input port.
$\overline{CTS0}$, $\overline{CTS1}$	68, 58	82, 72	Input	Clear to send for MSCI: Indicates that a communications device such as a modem is ready to send data to the communication line. MSCI transmit operations can be automatically controlled by this input (auto-enable function). This line can also be used as a general-purpose input port.
$\overline{SYNC0}$, $\overline{SYNC1}$	64, 55	78, 69	Input/ output	Synchronization for MSCI: Input line in external byte output synchronous mode. Synchronization is established at the falling edge of $\overline{SYNC0}$ or $\overline{SYNC1}$. This line is an output line in mono-sync and bi-sync byte synchronous modes and bit synchronous HDLC mode. It indicates the inverse of the SYNCD/FLGD bit in MSCI status register 1 (ST1). In mono-sync or bi-sync byte synchronous mode, a low pulse is output immediately after a SYN pattern is detected. In bit synchronous HDLC mode, a low pulse is output immediately after a flag pattern is detected. In asynchronous mode, this line is an input line, but the input value does not affect operations.

Note: For details concerning MSCI status register 1 (ST1), see section 5.2.1, MSCI Status Register 1.

Section 3 System Controller

3.1 Overview

Features of the SCA's system controller:

- Three chip operating modes
  - —Reset mode
  - —Normal operating mode
  - —System stop mode
- An on-chip bus arbiter arbitrates bus contention between the external bus master and on-chip DMA controller
- Four 8- and 16-bit MPU bus interfaces can be switched under external control

3.2 Chip Operating Modes

3.2.1 SCA Operating Modes

The SCA supports three chip operating modes:

- Reset mode
- Normal operating mode
- System stop mode (low-power mode)

In reset mode, normal chip operations halt and registers are initialized. The chip enters reset mode if a $\overline{\text{RESET}}$ signal is input during normal operating mode or system stop mode. Reset mode is released when the $\overline{\text{RESET}}$ signal returns to the high level. The chip then enters normal operating mode.

In normal operating mode, all on-chip functional modules operate at their normal performance levels. From normal operating mode, transitions to either of the other operating modes are possible.

System stop mode is a low-power mode in which all internal operations cease except for the clock generator and reset circuits, to reduce power dissipation. From system stop mode, the SCA can return to normal operating mode via reset mode, as shown in figure 3.1.

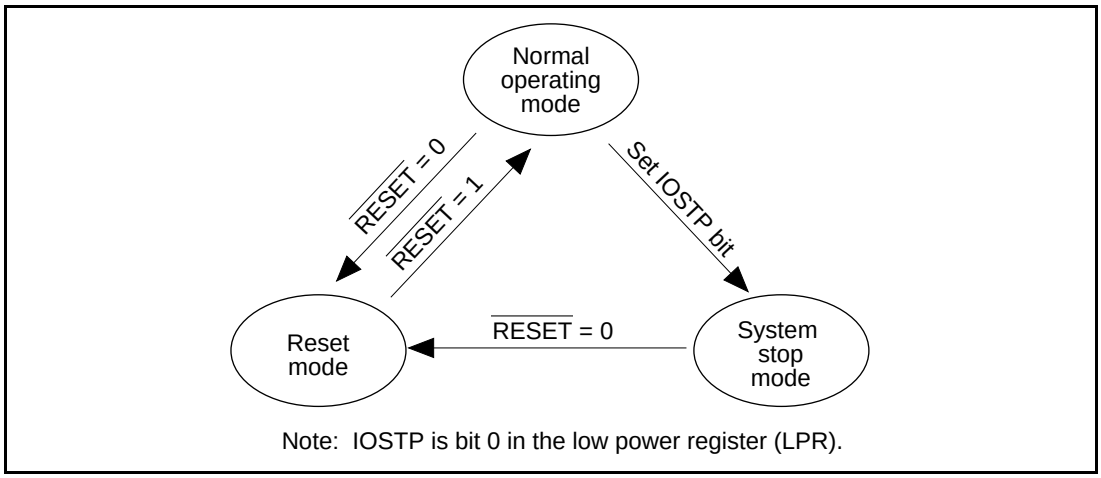


Figure 3.1 Chip Operating Mode Transitions

Table 3.1 indicates the operational status of the main functional modules in each of the operating modes.

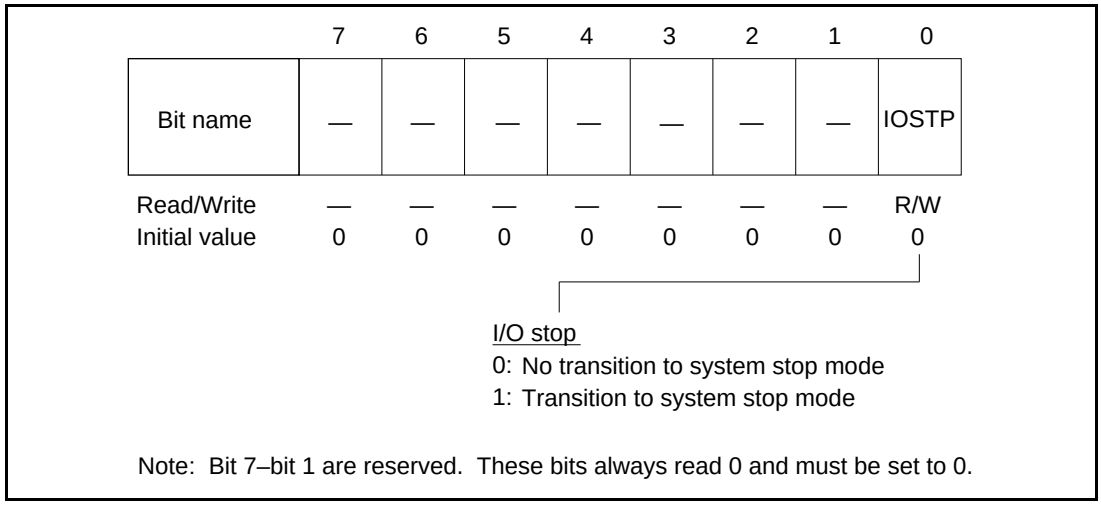
Table 3.1 Operational Status of On-Chip Functional Modules in Operating Modes

	Functional Module		
Operating Mode	**On-Chip DMAC**	**MSCI**	**Timers**
Reset mode	—	—	—
Normal operating mode	°	°	°
System stop mode	—	—	—

Note: °: operation enabled
—: operation disabled

3.2.2 Low-Power Register (LPR)

The low-power register controls transition to system stop mode.



Bits 7–1: Reserved. These bits always read 0 and must be set to 0.

Bit 0 (IOSTP: I/O Stop): Controls transition to low-power mode (system stop mode).

IOSTP = 0: The SCA remains in its current operating mode and does not enter system stop mode.

IOSTP = 1: The SCA enters system stop mode. Registers cannot be written or read in this mode, so once it is set to 1 to enter system stop mode, the IOSTP bit cannot be cleared to 0 except by a reset. Figure 3.3 shows transition timing in system stop mode.

3.2.3 Reset Mode

Holding the $\overline{\text{RESET}}$ line low for six or more clock cycles resets all SCA functional modules and puts the SCA into reset mode. In this mode, the SCA operates as follows:

- The MSCI, DMAC, and timers halt, their internal states are reset, and registers are initialized.
- The A8–A23 and D0–D15 lines go to high impedance and all output lines are initialized to predefined values.
- WAIT becomes an output line and goes high. Other input/output lines go to the input or high-impedance state.

The $\overline{\text{RESET}}$ line is sampled at the falling edge of every CLK clock (rising edge in CPU mode 0). If the $\overline{\text{RESET}}$ line is low on the falling CLK edge (rising edge in CPU mode 0) for six successive cycles, the SCA enters reset mode after a half clock cycle delay.

Make sure that $\overline{\text{RESET}}$ remains low long enough to be sampled on at least six consecutive falling edges of CLK (rising edges in CPU mode 0). Correct resetting is not guaranteed if $\overline{\text{RESET}}$ is low in fewer than six consecutive cycles.

The SCA leaves reset mode when the $\overline{\text{RESET}}$ line goes high. If the $\overline{\text{RESET}}$ line remains high for five successive falling CLK edges (rising edges in CPU mode 0), the SCA leaves reset mode after a half clock cycle delay and enters normal operating mode.

Figure 3.2 shows the timing for entering and leaving reset mode.

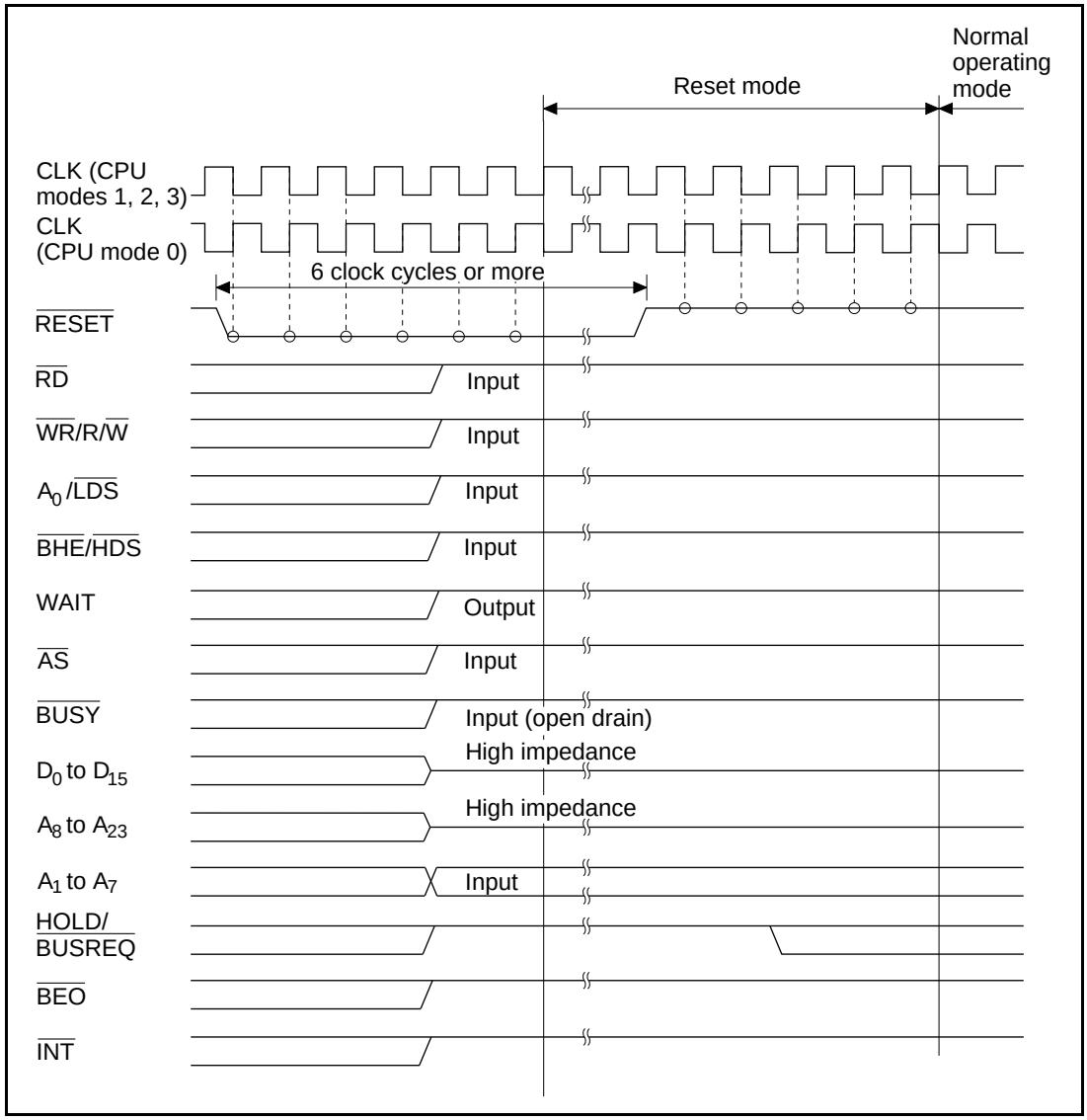


Figure 3.2 Reset Mode Timing

3.2.4 Normal Operating Mode

In normal operating mode, all functional modules in the SCA are active and communication is enabled. The SCA operates as follows in this mode:

- The MSCI, DMAC, and timers perform their regular functions with regular performance.
- Interrupt requests (INT) can be generated.
- The SCA can become the bus master through the action of its on-chip bus arbiter.

From the normal operating mode it is possible to enter any other chip operating mode, as follows:

- If the RESET signal is active for six clock cycles or more, the SCA enters reset mode.
- If the IOSTP bit is set to 1, the SCA enters system stop mode. IOSTP is bit 0 of the low power register (LPR). See section 3.2.5, System Stop Mode, for details.

3.2.5 System Stop Mode

Setting the IOSTP bit in the low-power register (LPR) to 1 puts the SCA into system stop mode. System stop mode is a low-power mode in which clock signals are not supplied to the on-chip functional modules.

Operation in System Stop Mode: In system stop mode, the SCA operates as follows:

- The MSCI, DMAC, and timers halt.
- The bus interface shuts down. Registers cannot be written or read.
- The on-chip clock generator circuit continues to operate, but no clock signals are supplied to the main functional modules.
- Pins are placed in the states listed in table 3.2.

Power dissipation in system stop mode can be further decreased by stopping external clock input. The external clock input line (CLK) should be held high (low in CPU mode 0). The SCA is not guaranteed to operate as described above if clock input stops in the low state (high in CPU mode 0).

Entering and Leaving System Stop Mode: Enter system stop mode from the normal operating mode by setting the IOSTP bit in the low-power register (LPR) to 1. The mode transition takes place during the write cycle in which IOSTP is set to 1. Figures 3.3 (a) to (d) show the timing.

The SCA leaves system stop mode by entering reset mode.

System stop mode is released when the $\overline{\text{RESET}}$ signal becomes active for six clock cycles or more, placing the SCA in reset mode.

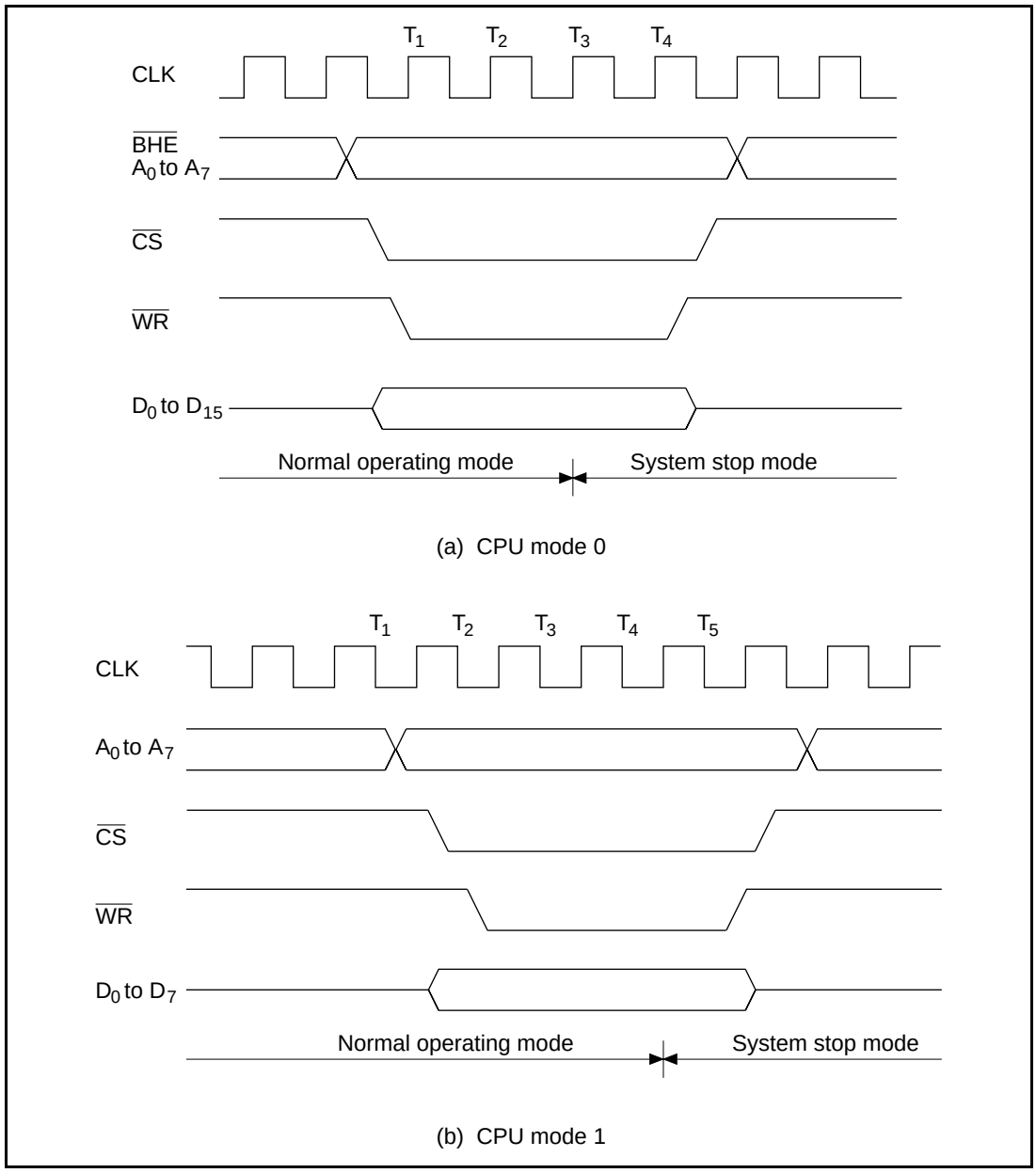


Figure 3.3 Timing of Transition to System Stop Mode

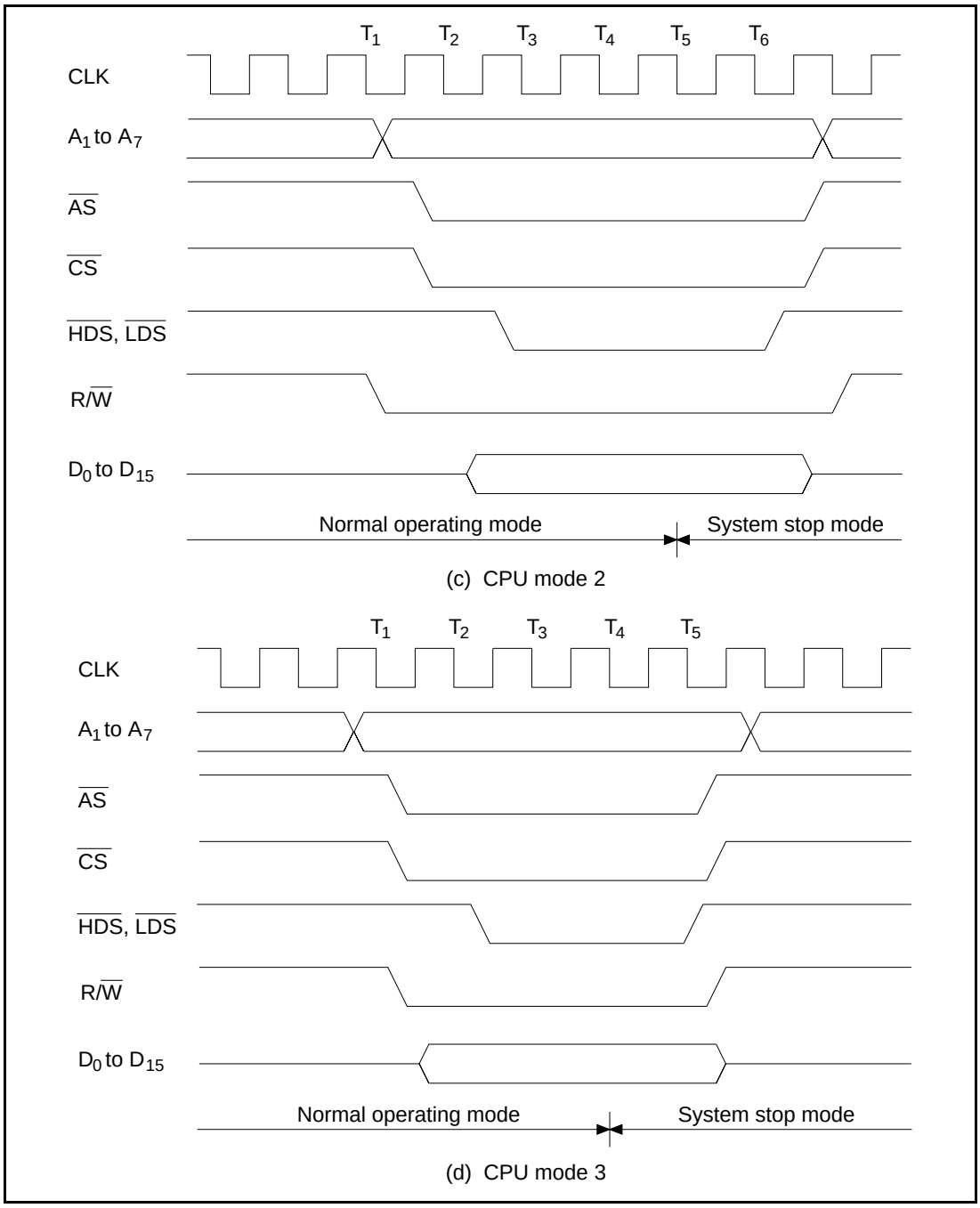


Figure 3.3 Timing of Transition to System Stop Mode (cont)

Table 3.2 Signal Line States in System Stop Mode

	Signal Line States		
Signal Line	**CPU Mode 0**	**CPU Mode 1**	**CPU Modes 2, 3**
A_1–A_7	Input	Input	Input
A_8–A_{23}	High impedance	High impedance	High impedance
$\overline{BUSY}$	Input	Input	Input
$\overline{BEO}$	High output	High output	High output
$\overline{RD}$/NC	Input	Input	NC
$\overline{WR}$/R/$\overline{W}$	Input	Input	Input
A_0/$\overline{LDS}$	Input	Input	Input
$\overline{BHE}$/$\overline{HDS}$	Input	NC	Input
WAIT	High output	High output	High output
$\overline{AS}$	Input	Input	Input
HOLD/$\overline{BUSREQ}$	Low output	High output	High output
$\overline{INT}$	High (open drain)	High (open drain)	High (open drain)
D_0–D_7	High impedance	High impedance	High impedance
D_8–D_{15}	High impedance	NC	High impedance

NC: Not connected

3.3 Bus Arbiter

3.3.1 Overview

The SCA is equipped with a bus arbiter which arbitrates bus contention between the on-chip DMAC and an external bus master device. The on-chip DMAC is connected internally to the bus arbiter. When the on-chip DMAC requests the bus, the bus arbiter drives $\overline{BUSREQ}$ active low (drives HOLD active high in CPU mode 0) to ask the host MPU for control of the bus.

When the host MPU makes $\overline{BUSACK}$ active low (or makes HOLDA active high in CPU mode 0), the bus arbiter monitors the $\overline{BUSY}$ line. If $\overline{BUSY}$ is high, the bus arbiter takes control of the bus and drives $\overline{BUSY}$ low to notify external devices that the SCA is using the bus. The on-chip DMAC then starts DMA transfer.

If $\overline{BUSACK}$ goes active low (or HOLDA goes active high in CPU mode 0) when there is no bus request from the on-chip DMAC, the bus arbiter drives $\overline{BEO}$ active low to pass the $\overline{BUSACK}$ signal (HOLDA in CPU mode 0) on to other bus master devices. $\overline{BEO}$ can be used for daisy chaining.

Figure 3.4 shows the interconnections of the bus arbiter and bus masters.

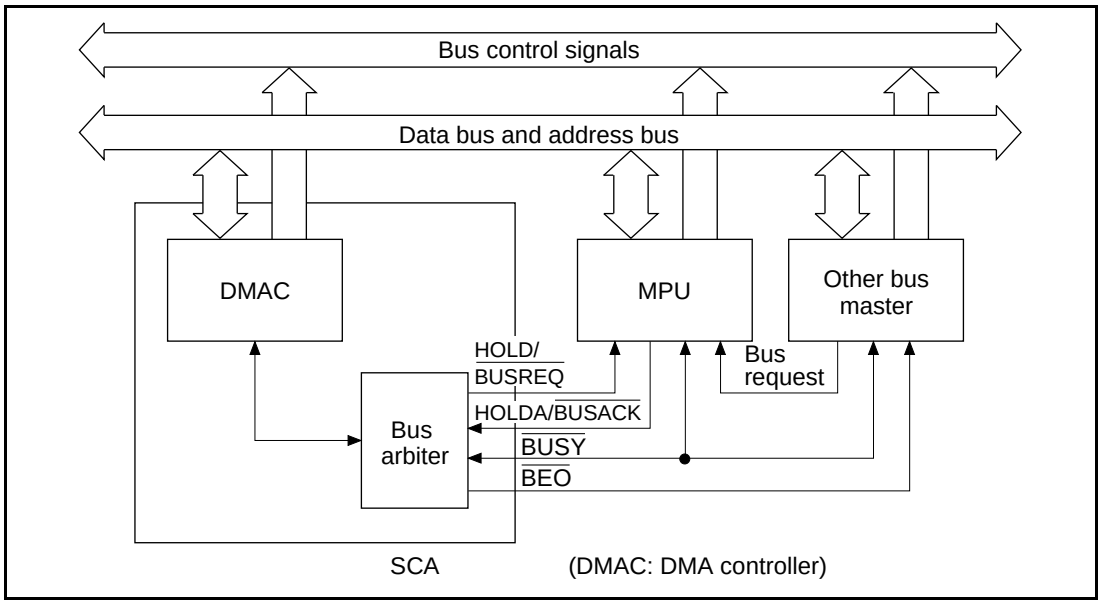


Figure 3.4 Bus Arbiter and Bus Masters

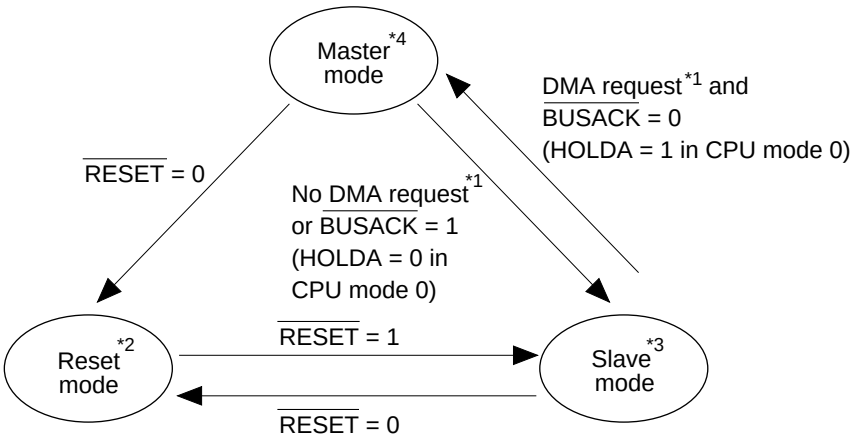
3.3.2 Timing for Passing Bus Control

If $\overline{\text{BUSACK}}$ becomes inactive (high) (HOLDA becomes low in CPU mode 0) during a DMA transfer, the bus arbiter releases control of the bus at an opportunity furnished by the on-chip DMAC controller.

The on-chip DMAC controller allows control of the bus to pass to another bus master at the end of each machine cycle, immediately after a T_3 or T_i state. See section 6, DMAC, for details. When $\overline{\text{BUSACK}}$ (HOLDA in CPU mode 0) becomes inactive, the on-chip DMAC suspends the transfer at the end of a machine cycle and makes $\overline{\text{BUSY}}$ inactive (high), passing control of the bus to another bus master. If $\overline{\text{BUSACK}}$ (HOLDA in CPU mode 0) later becomes active low (high in CPU mode 0), the DMAC waits for $\overline{\text{BUSY}}$ to become inactive, then takes control of the bus and resumes the transfer.

3.3.3 Bus Control Passing

Figure 3.5 shows how bus control is passed.



Notes:
1. See section 6, DMAC for information about DMA requests.
2. If the RESET signal is driven active low for six cycles or more, the SCA unconditionally enters reset mode. When RESET goes high, the SCA enters normal operating mode and control of the bus passes to the MPU (or an external bus master), leaving the SCA in slave mode.
3. Slave mode is the mode in which the MPU or external bus master has control of the bus. The SCA executes MPU read/write cycles and interrupt acknowledge cycles in this mode.
4. In master mode, the SCA has control of the bus and can execute DMA transfers.

Figure 3.5 Bus Control Passing

Figure 3.6 shows examples of bus arbitration sequences. The following describes the sequence in CPU modes 1 to 3 when the system is configured as shown in figure 3.7. The differences between CPU mode 0 and other modes are that in CPU mode 0, HOLD and HOLDA are used instead of $\overline{\text{BUSREQ}}$ and $\overline{\text{BUSACK}}$, respectively, and that the HOLD, HOLDA, and CLK signals have the opposite phase to their counterparts in the other modes. For details on DMA cycles, see Section 6.4, Operating Modes.

Figure 3.6 (a) shows the bus arbitration sequence in which the master MPU has control of the bus, that is, the $\overline{\text{BUSY}}$ input is high. In this sequence:

(a) The SCA requests the master MPU to release the bus by driving $\overline{\text{BUSREQ}}$ active (low), and the master MPU, in response, grants control of the bus to the SCA by driving $\overline{\text{BUSACK}}$ active (low).

(b) The SCA passes control of the bus to another bus master with $\overline{\text{BUSACK}}$ kept active because another bus master requests for control of the bus.

In this case, when a DMA transfer is requested in the SCA, the SCA drives $\overline{\text{BUSREQ}}$ active at the next falling edge of CLK to request control of the bus from the master MPU. The SCA then samples the $\overline{\text{BUSACK}}$ input from the master MPU at each rising edge of CLK. Here, the SCA also samples the $\overline{\text{BUSY}}$ input to check whether or not any other bus master is using the bus ($\overline{\text{BUSY}}$ is

an input in SCA slave mode and is an output in master mode.) When the SCA detects $\overline{\text{BUSACK}}$ low and if no other bus master is using the bus, that is, $\overline{\text{BUSY}}$ is high, the SCA acquires control of the bus. Having acquired control of the bus, the SCA drives $\overline{\text{BUSY}}$ (output) low at the next rising edge of CLK to indicate that it has received control of the bus. The SCA begins a DMA cycle at the next rising edge of CLK.

When a DMA transfer request has been serviced, the SCA releases control of the bus. Specifically, the SCA drives $\overline{\text{BUSREQ}}$ inactive, which terminates the DMA cycle at the next falling edge of CLK. At the same falling edge of CLK, the SCA drives $\overline{\text{BUSY}}$ inactive to indicate that the SCA has released control of the bus. (Since the $\overline{\text{BUSY}}$ line is an open-drain output, it must be pulled up to V_{CC}.)

Figure 3.6 (b) shows the bus arbitration sequence in which a DMA transfer is requested in the SCA while the master MPU does not have control of the bus.

In this case, since $\overline{\text{BUSACK}}$ is low, the SCA only samples the $\overline{\text{BUSY}}$ input at each rising edge of CLK. When the SCA detects $\overline{\text{BUSY}}$ high, that is, no other bus master is using the bus, the SCA immediately drives BUSY (output) low to indicate that the SCA has received control of the bus. The SCA then drives $\overline{\text{BUSREQ}}$ active at the next falling edge of CLK and drives $\overline{\text{BEO}}$ inactive (high) to stop $\overline{\text{BUSACK}}$ from being transferred to the lower bus masters. The SCA begins a DMA cycle at the next rising edge of CLK.

The bus release sequence is similar to that in figure 3.6 (a).

When the master MPU later drives $\overline{\text{BUSACK}}$ high, $\overline{\text{BEO}}$ goes high at the next rising edge of CLK.

Figure 3.6 (c) shows the bus arbitration sequence in which the SCA requests control of the bus while another bus master is using it, and the SCA acquires control of the bus after the bus master releases control of the bus with the BUSACK input from the master MPU kept low.

In this sequence, after driving $\overline{\text{BUSREQ}}$ active, the SCA samples the $\overline{\text{BUSACK}}$ input from the master MPU and the $\overline{\text{BUSY}}$ input from the other bus master at each rising edge of CLK. When the SCA detects $\overline{\text{BUSACK}}$ low and $\overline{\text{BUSY}}$ high, it acquires control of the bus. Having acquired control of the bus, the SCA drives $\overline{\text{BUSY}}$ (output) low at the next rising edge of CLK and begins a DMA cycle at the next rising edge of CLK.

If the master MPU drives $\overline{\text{BUSACK}}$ high while the SCA is still requesting control of the bus, the SCA temporarily releases control of the bus after executing the DMA cycle that began before the rising edge of CLK at which the SCA sampled $\overline{\text{BUSACK}}$ high. Here, the SCA drives $\overline{\text{BUSY}}$ (input) high, which indicates the end of the DMA cycles. The specific end timing of DMA cycles varies depending on the $\overline{\text{BUSACK}}$ input timing and the number of inserted wait states.

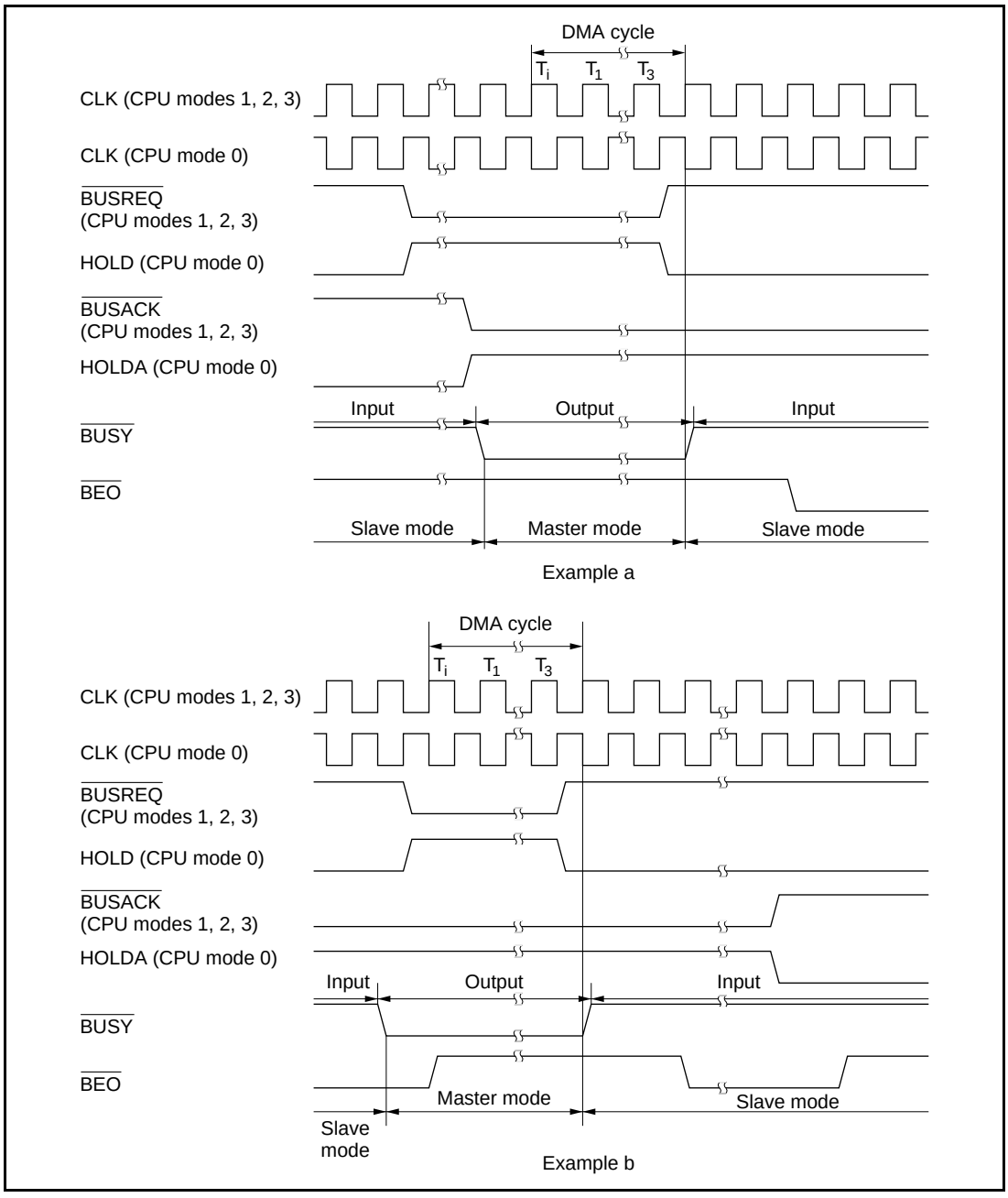


Figure 3.6 Bus Arbitration Sequence examples

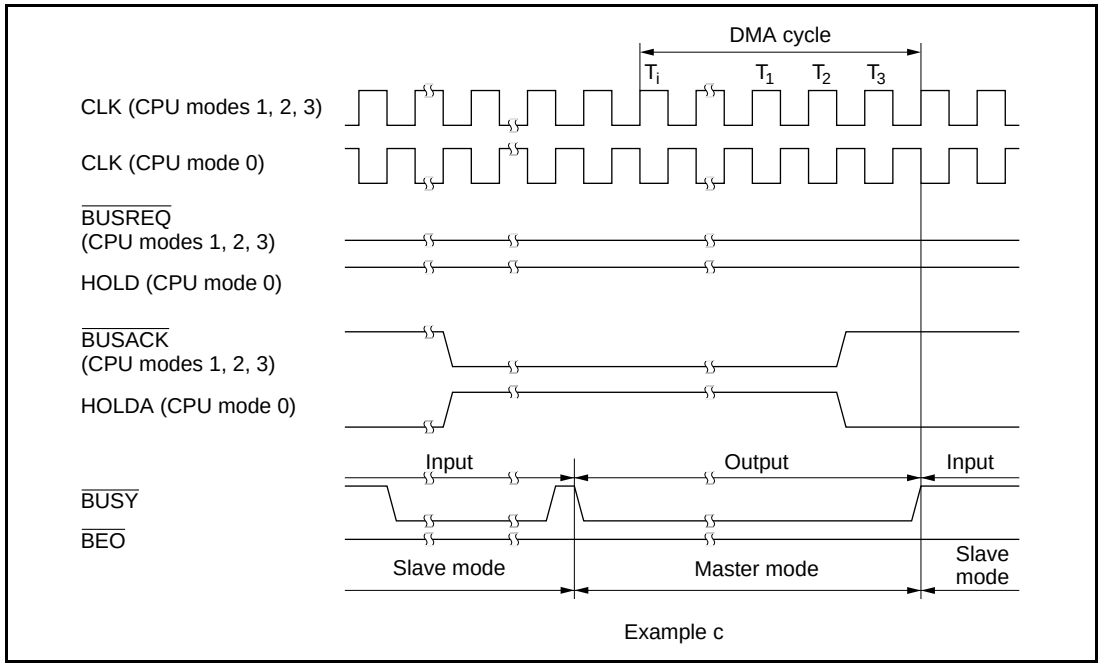


Figure 3.6 Bus Arbitration Sequence Examples (cont)

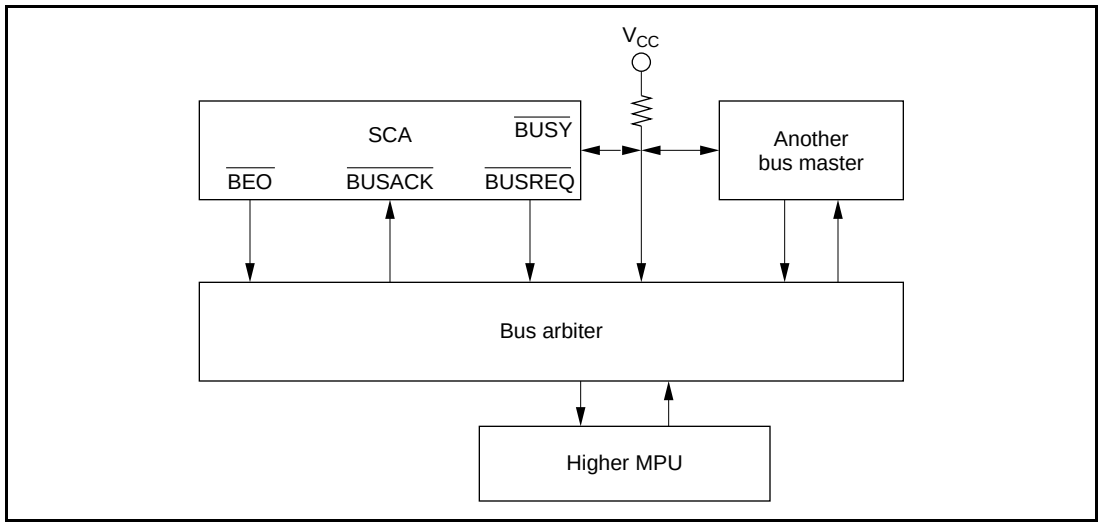


Figure 3.7 Bus Arbitration System Block Diagram

3.4 Bus Interface

3.4.1 Overview

The SCA has four 8- and 16-bit bus interfaces that can be switched under external control. The bus interface is selected according to the CPU mode as shown in table 3.3.

Table 3.3 CPU Mode and Bus Interface

		Address Relationships of Data Bus		Length of Bus Cycle (number of states)		
CPU Modes	**Bus Width**	**High Byte (D15 to D8)**	**Low Byte (D7 to D0)**	**Slave Mode*2, *5**	**Master Mode**	**MPU Type**
Mode 0	16 bits	Odd address	Even address	4 (5)*3/4 (5)*3	3*4	8086-system 16-bit MPU
Mode 1	8 bits	—	All addresses	4/5	3*4	64180-type MPU
Mode 2*1	16 bits	Even address	Odd address	5/6	3*4	68000-system 16-bit MPU I
Mode 3*1	16 bits	Even address	Odd address	5/5	3*4	68000-system 16-bit MPU II

Notes:
1. CPU modes 2 and 3 differ only in the bus timing. See section 10, Electrical Characteristics for details.
2. Number of states in read cycle/number of states in write cycle
3. Number of states for consecutive bus cycles in slave mode
4. When no wait states are inserted.
5. Shortest number of states. The number of states may increase if the MPU's strobe disable timing is delayed.

The SCA has three 16-bit bus interfaces (CPU modes 0, 2, 3). The high-byte/low-byte address relationship on the data bus in CPU mode 0 is opposite to the relationship in CPU modes 2 and 3. This gives the SCA a byte-swap capability. The data bus lines map onto even and odd memory banks as shown in figure 3.8.

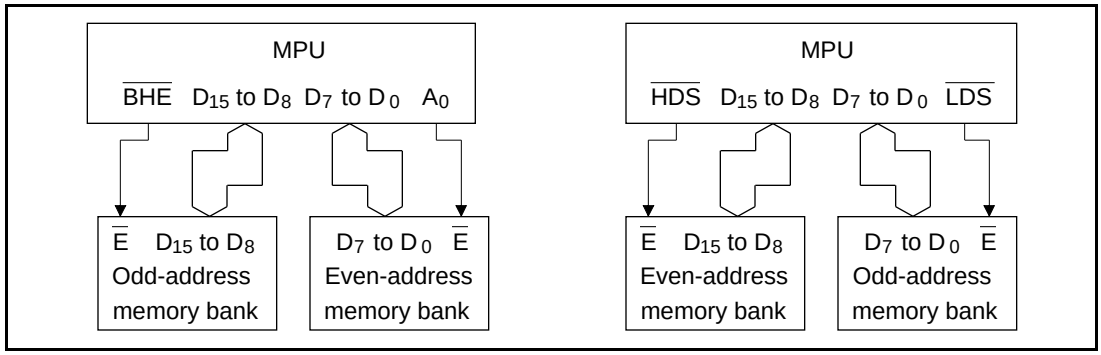


Figure 3.8 Data Bus Mapping onto Memory Banks in CPU Modes 0, 2, and 3

3.4.2 Slave Mode Bus Cycle

In slave mode, data moves from the SCA to MPU in a read cycle, and from MPU to the SCA in a write cycle. The address and bus interface signals are input signals, except for WAIT, which is an output signal.

CPU Mode 0: The SCA latches $\overline{BHE}$ and the address on lines A0 to A7 when $\overline{CS}$ is driven active low. $\overline{CS}$ must remain low throughout the bus cycle. After the bus cycle ends, $\overline{CS}$ may be either high or low. $\overline{CS}$ may also be low before the beginning of the bus cycle. Figure 3.9 shows the slave mode bus timing sequence in CPU mode 0.

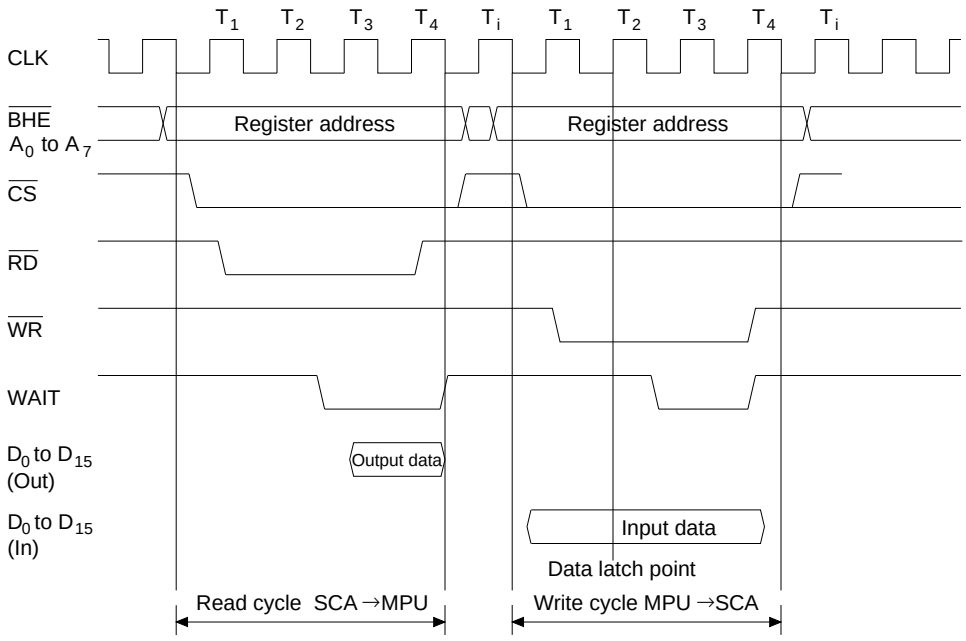
- Read cycle

  If $\overline{RD}$ is low (active) at the falling clock edge between the T_1 and T_2 states, the SCA outputs the contents of the register specified by the address on the data bus on the rising clock edge in the T_3 state. $\overline{RD}$ must remain low until the beginning of the T_4 state. When $\overline{RD}$ goes high (inactive), the cycle ends: the SCA then drives the WAIT output active high and lets the data bus float. The read cycle can be extended by delaying the high transition of $\overline{RD}$.

- Write cycle

  If $\overline{WR}$ is low (active) at the falling clock edge between the T_1 and T_2 states, the SCA latches the data on the data bus on the rising clock edge in the T_3 state, and stores the data in the register specified by the address. $\overline{WR}$ must remain low until the rising clock edge in the T_4 state. When $\overline{WR}$ goes high (inactive), the cycle ends: the SCA then drives the WAIT output active high.

  When successive slave mode bus cycles or interrupt acknowledge cycles occur in CPU mode 0, at least one T_i state (idle state) must be inserted between cycles. No T_i state is necessary when the next cycle is not a slave mode bus cycle or an interrupt acknowledge cycle.



Note: 1. Ti states are required between successive slave mode bus cycles or interrupt acknowledge cycles.

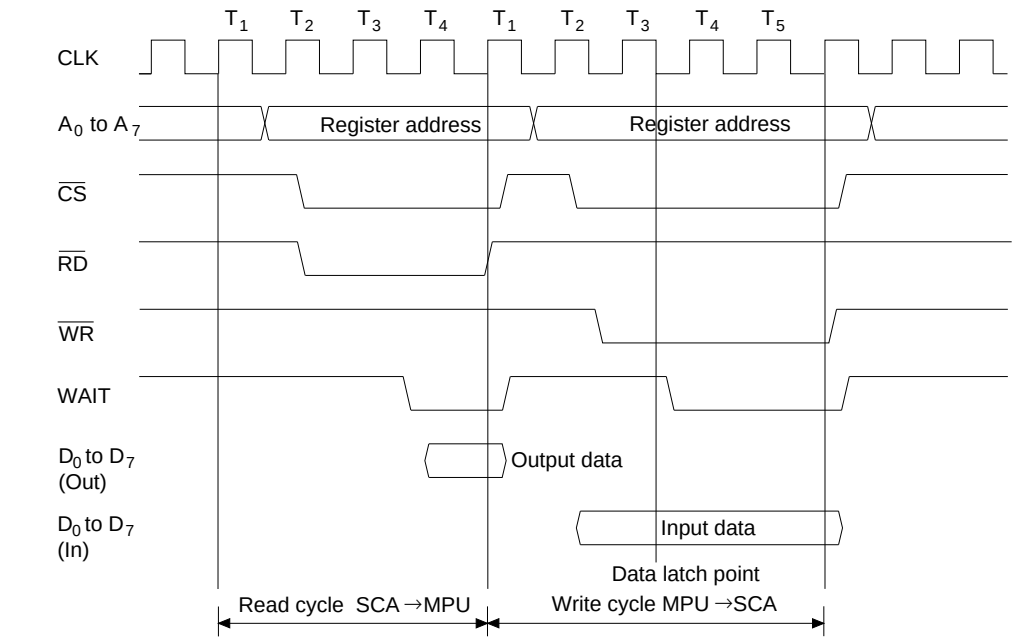
2. State numbers do not match MPU state numbers.

Figure 3.9 Slave Mode Bus Timing Sequence in CPU Mode 0

CPU Mode 1: The SCA latches the address on lines A_0 to A_7 when $\overline{CS}$ is driven active low. $\overline{CS}$ must remain low throughout the bus cycle. After the bus cycle ends, $\overline{CS}$ must go high (inactive). Figure 3.10 shows the slave mode bus timing sequence in CPU mode 1.

- Read cycle
  If $\overline{RD}$ is low (active) at the falling clock edge in the T_2 state, the SCA outputs the contents of the register specified by the address on the data bus on the rising clock edge between the T_3 and T_4 states. $\overline{RD}$ must remain low until the falling clock edge in the T_4 state. When $\overline{RD}$ goes high (inactive), the cycle ends: the SCA then drives the WAIT output active high and lets the data bus float. The read cycle can be extended by delaying the high transition of $\overline{RD}$.

- Write cycle
  If $\overline{WR}$ is low (active) at the rising clock edge between the T_2 and T_3 states, the SCA latches the data on the data bus on the falling clock edge in the T_3 state, and stores the data in the register specified by the address. $\overline{WR}$ must remain low until the falling clock edge in the T_5 state. When $\overline{WR}$ is driven high (inactive), the cycle ends: the SCA drives the WAIT output active high.



Note: State numbers do not match MPU state numbers.

Figure 3.10 Slave Mode Bus Timing Sequence in CPU Mode 1

CPU Mode 2: The SCA latches the address on lines A_1 to A_7 when $\overline{CS}$ and $\overline{AS}$ are both driven active low. $\overline{CS}$ and $\overline{AS}$ must remain low throughout the bus cycle. After the bus cycle ends, they must go high (inactive). Figure 3.11 shows the slave mode bus timing sequence in CPU mode 2.

- Read cycle

  When R/W is high, if $\overline{HDS}$ or $\overline{LDS}$ is low (active) at the rising clock edge between the T_2 and T_3 states, the SCA outputs the contents of the register specified by the address on the data bus on the falling clock edge in the T_4 state. $\overline{HDS}$ or $\overline{LDS}$ must remain low until the beginning of the T_5 state. When $\overline{HDS}$ or $\overline{LDS}$ goes high (inactive), the cycle ends: the SCA then drives the WAIT output active high and lets the data bus float. The read cycle can be extended by delaying the high transition of $\overline{HDS}$ or $\overline{LDS}$.

- Write cycle

  When R/$\overline{W}$ is low, if $\overline{HDS}$ or $\overline{LDS}$ is low (active) at the falling clock edge in the T_3 state, the SCA latches the data on the data bus on the falling clock edge in the T_4 state, and stores the data in the register specified by the address. $\overline{HDS}$ or $\overline{LDS}$ must remain low until the falling clock edge in the T_6 state. When $\overline{HDS}$ or $\overline{LDS}$ goes high (inactive), the cycle ends: the SCA then drives the WAIT output active high.

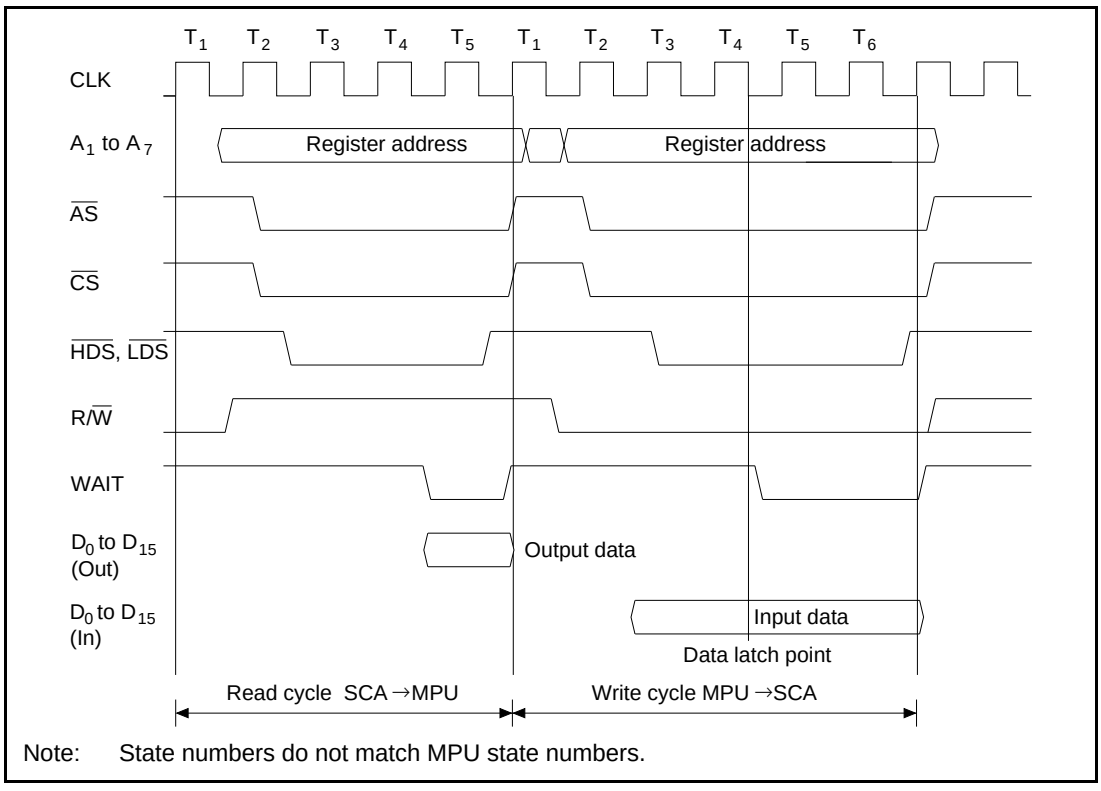


Figure 3.11 Slave Mode Bus Timing Sequence in CPU Mode 2

CPU Mode 3: The SCA latches the address on lines A_1 to A_7 when $\overline{CS}$ and $\overline{AS}$ are both driven active low. $\overline{CS}$ and $\overline{AS}$ must remain low throughout the bus cycle. After the bus cycle ends, they must go high (inactive). Figure 3.12 shows the slave mode bus timing sequence in CPU mode 3.

- Read cycle

  When R/W is high, if $\overline{HDS}$ or $\overline{LDS}$ is low (active) at the rising clock edge between the T_1 and T_2 states, the SCA outputs the contents of the register specified by the address on the data bus on the falling clock edge in the T_3 state. $\overline{HDS}$ or $\overline{LDS}$ must remain low until the beginning of the T_5 state. When $\overline{HDS}$ or $\overline{LDS}$ goes high (inactive), the cycle ends: the SCA then drives the WAIT output active high and lets the data bus float. The read cycle can be extended by delaying the high transition of $\overline{HDS}$ or $\overline{LDS}$.

- Write cycle

  When R/$\overline{W}$ is low, if $\overline{HDS}$ or $\overline{LDS}$ is low (active) at the rising clock edge between the T_2 and T_3 states, the SCA latches the data on the data bus on the falling clock edge in the T_3 state, and stores the data in the register specified by the address. $\overline{HDS}$ or $\overline{LDS}$ must remain low until the falling clock edge in the T_5 state. When $\overline{HDS}$ or $\overline{LDS}$ goes high (inactive), the cycle ends: the SCA drives the WAIT output active high.

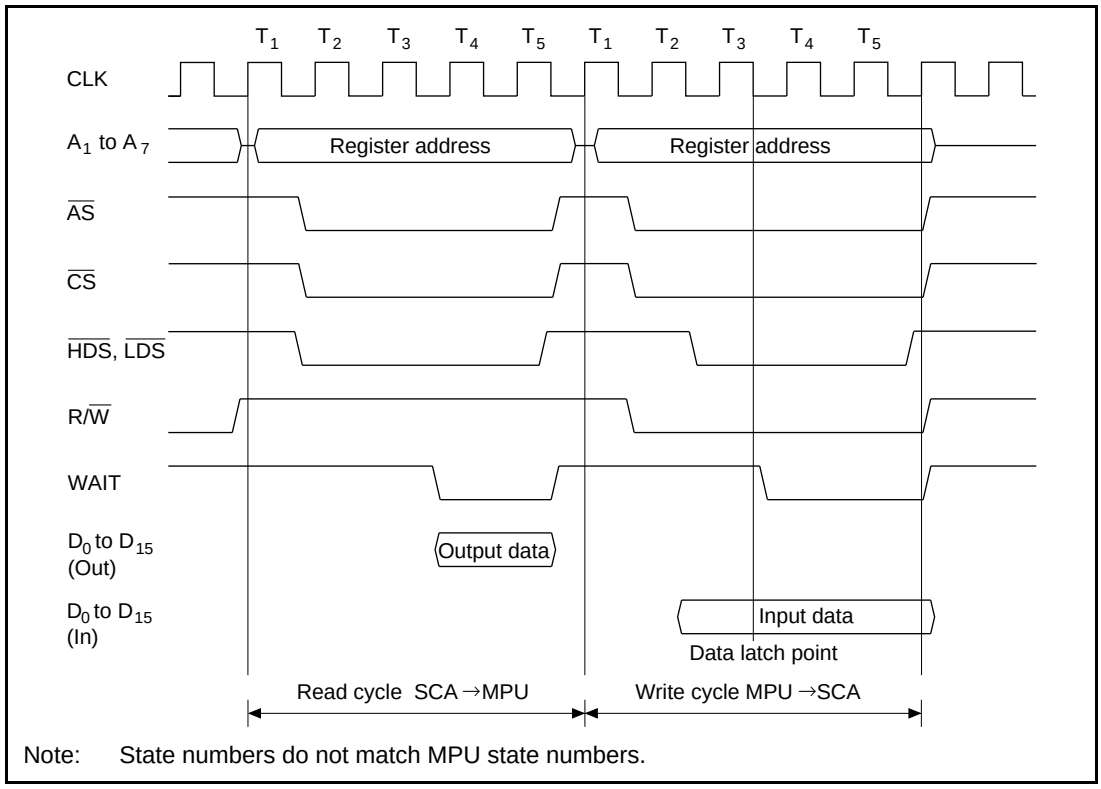


Figure 3.12 Slave Mode Bus Timing Sequence in CPU Mode 3

3.4.3 Master Mode Bus Cycle

In master mode (DMA mode), data moves from memory to the SCA in a read cycle, and from the SCA to memory in a write cycle. The address and bus interface signals are output signals, except for WAIT which is an input signal.

Word Transfer from Odd Address: In CPU modes 0, 2, and 3, DMA transfer of a word starting at an odd address is performed as two byte transfers. The DMAC first transfers one byte from the odd address, then transfers the remaining byte from the succeeding even address. Figure 3.13 shows how a word is transferred from an odd address.

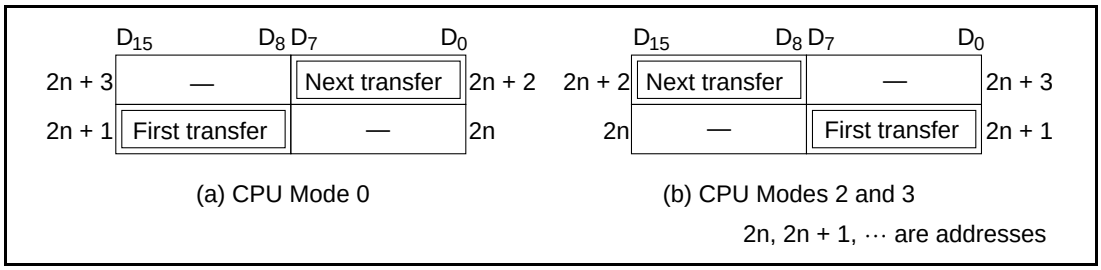


Figure 3.13 Word Transfer from Odd Address

Word Transfer from Even Address: In CPU modes 0, 2, and 3, a word is transferred from an even address by direct memory access in a single word transfer operation.

Transfer of Three or More Bytes from Odd Address: In CPU modes 0, 2, and 3, to transfer three or more bytes starting at an odd address by direct memory access, the DMAC first transfers one byte from the odd address, then transfers successive words starting from the next even address. If one byte remains to be transferred at the end, it is transferred from an even address. Figure 3.14 shows an example of data transfer starting from an odd address.

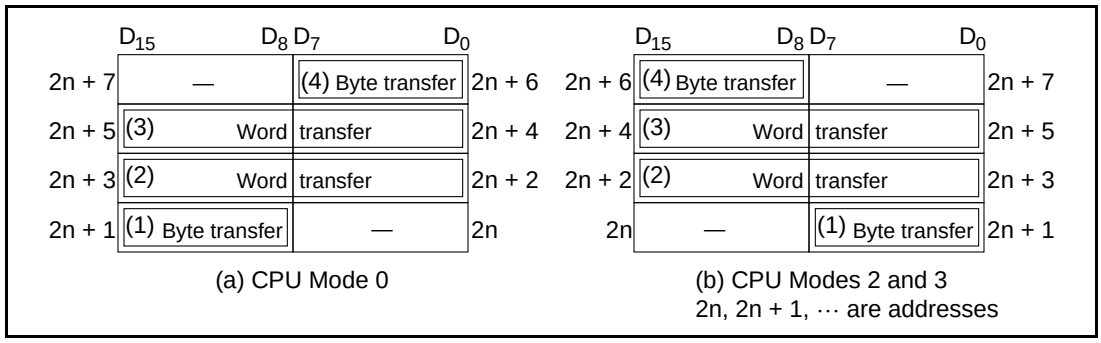


Figure 3.14 Data Transfer from Odd Address (example)

Figures 3.15 to 3.17 show the master mode bus transfer timing sequence in each CPU mode.

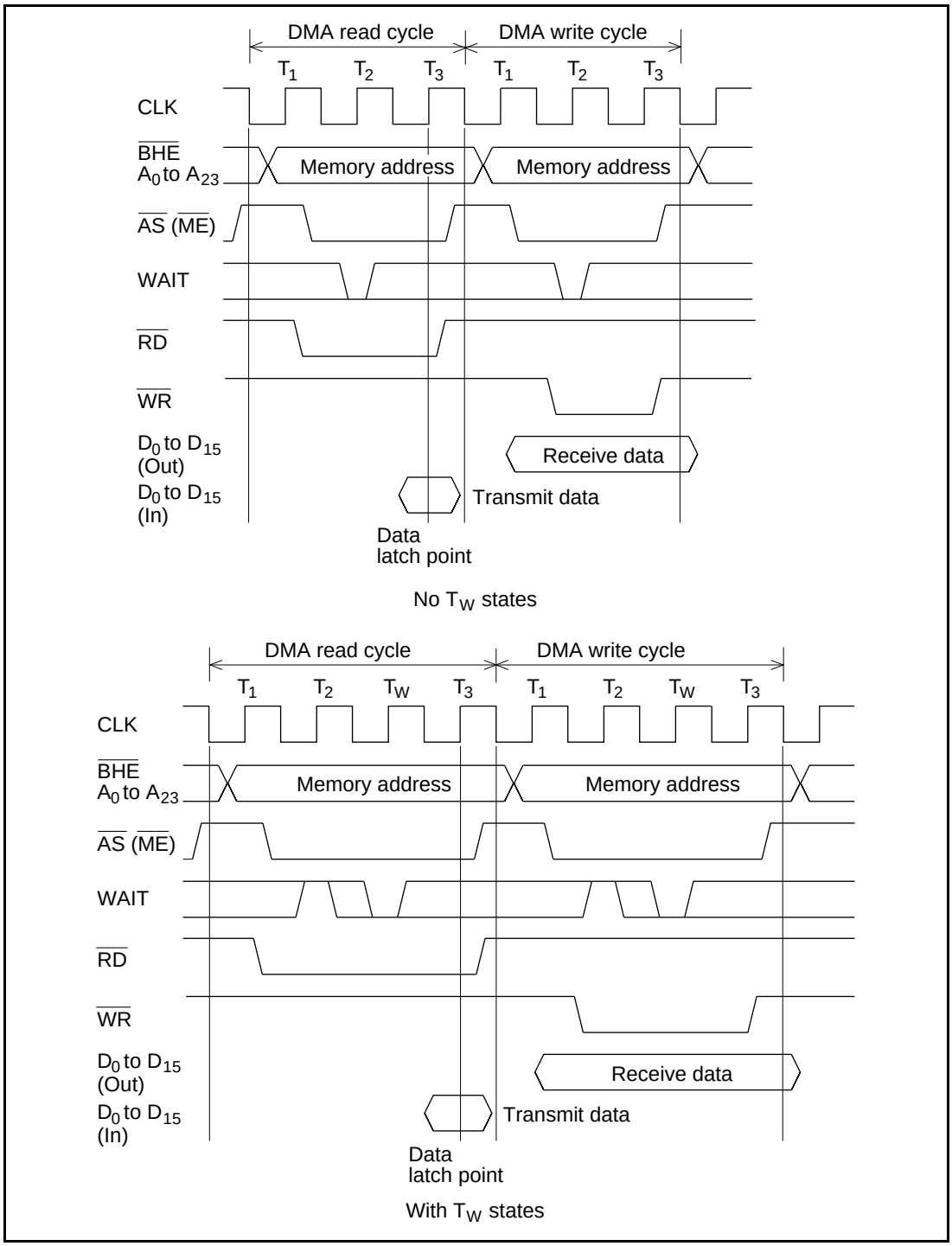


Figure 3.15 Master Mode Bus Timing Sequence in CPU Mode 0

HITACHI

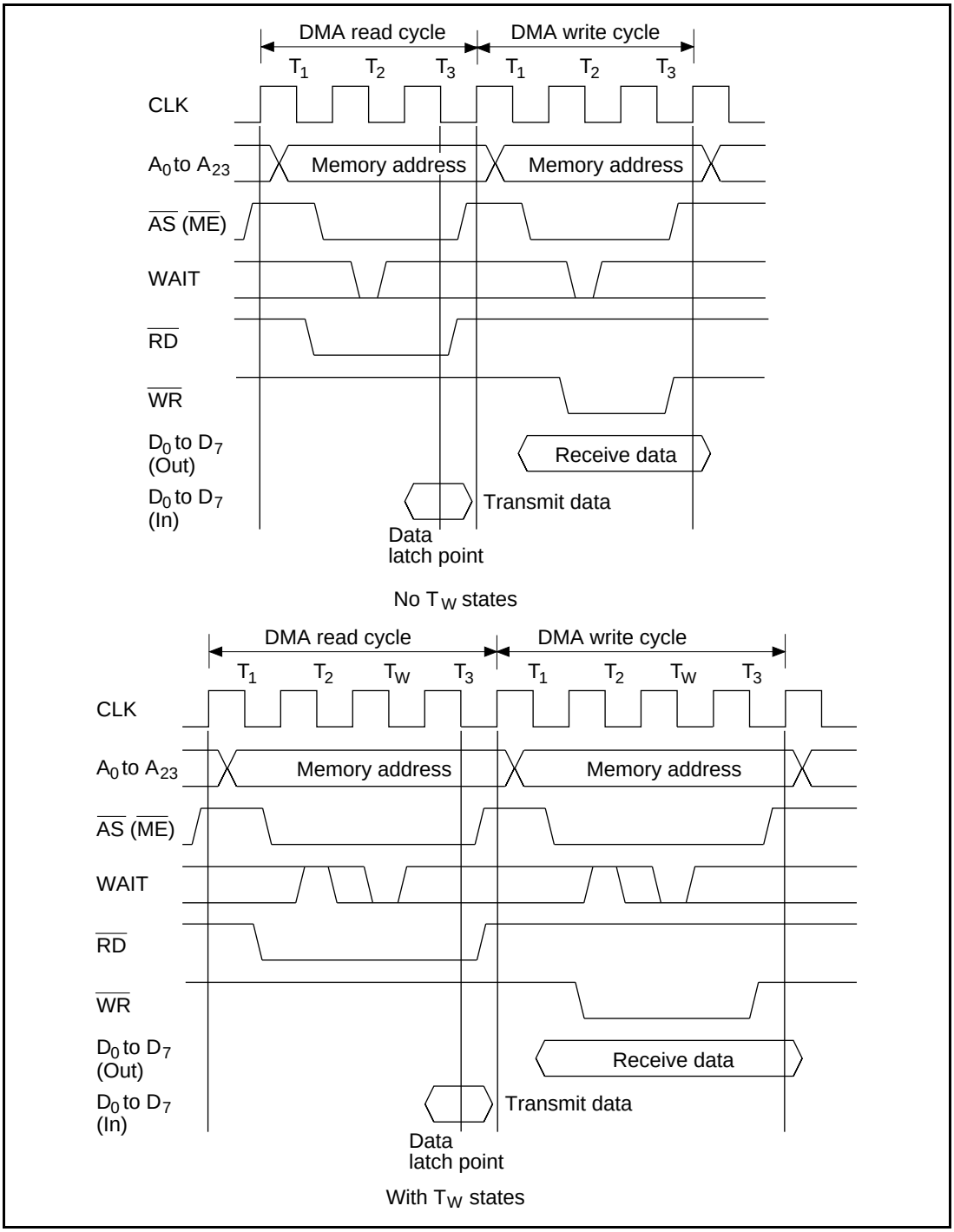


Figure 3.16 Master Mode Bus Timing Sequence in CPU Mode 1

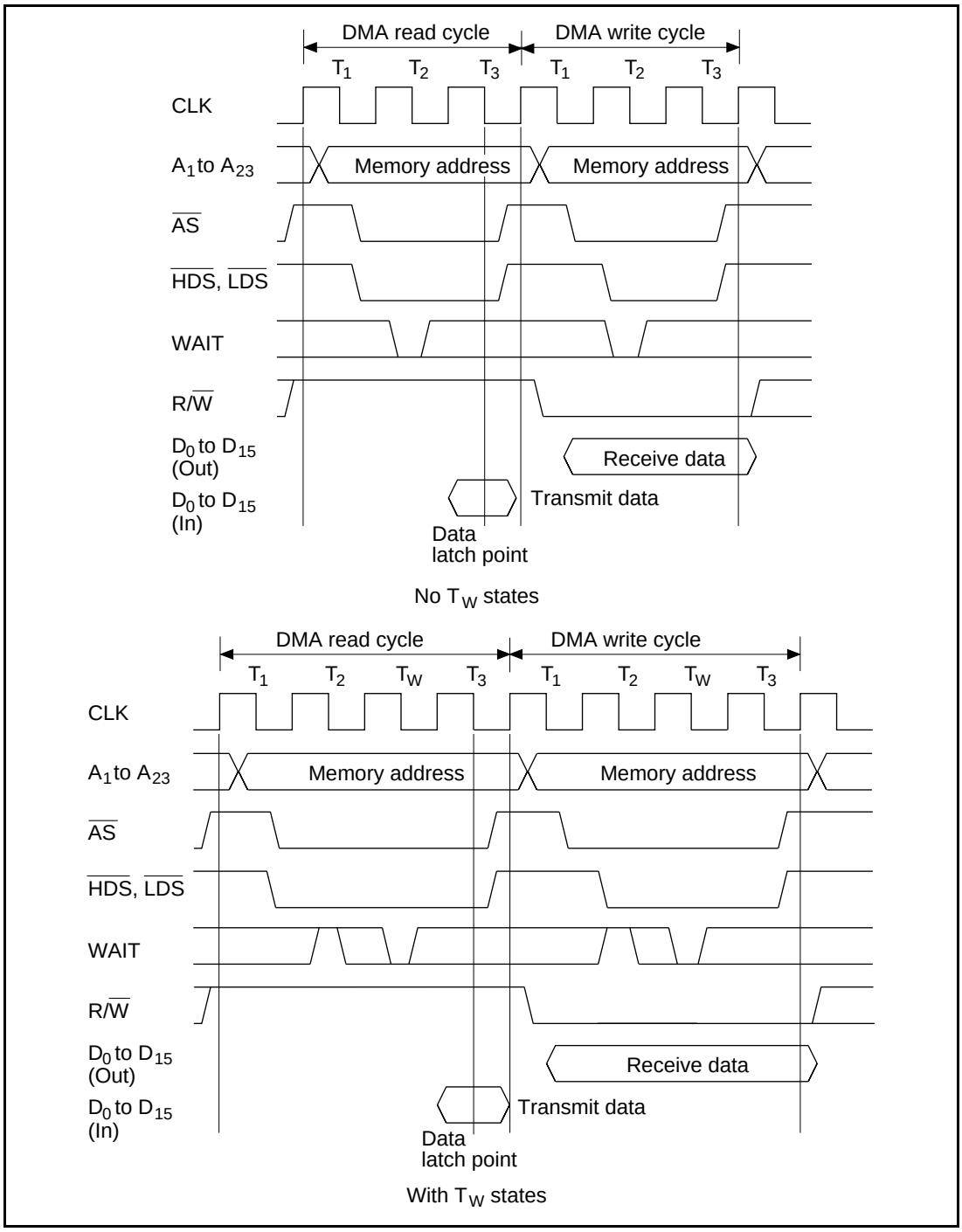


Figure 3.17 Master Mode Bus Timing Sequence in CPU Modes 2 and 3

Section 4 Interrupt Controller

4.1 Overview

The SCA has a single $\overline{INT}$ signal line for sending interrupt requests to a host MPU. The $\overline{INT}$ signal is generated by 20 interrupt sources on the SCA chip. Figure 4.1 shows the location of these interrupt sources in the SCA's functional modules.

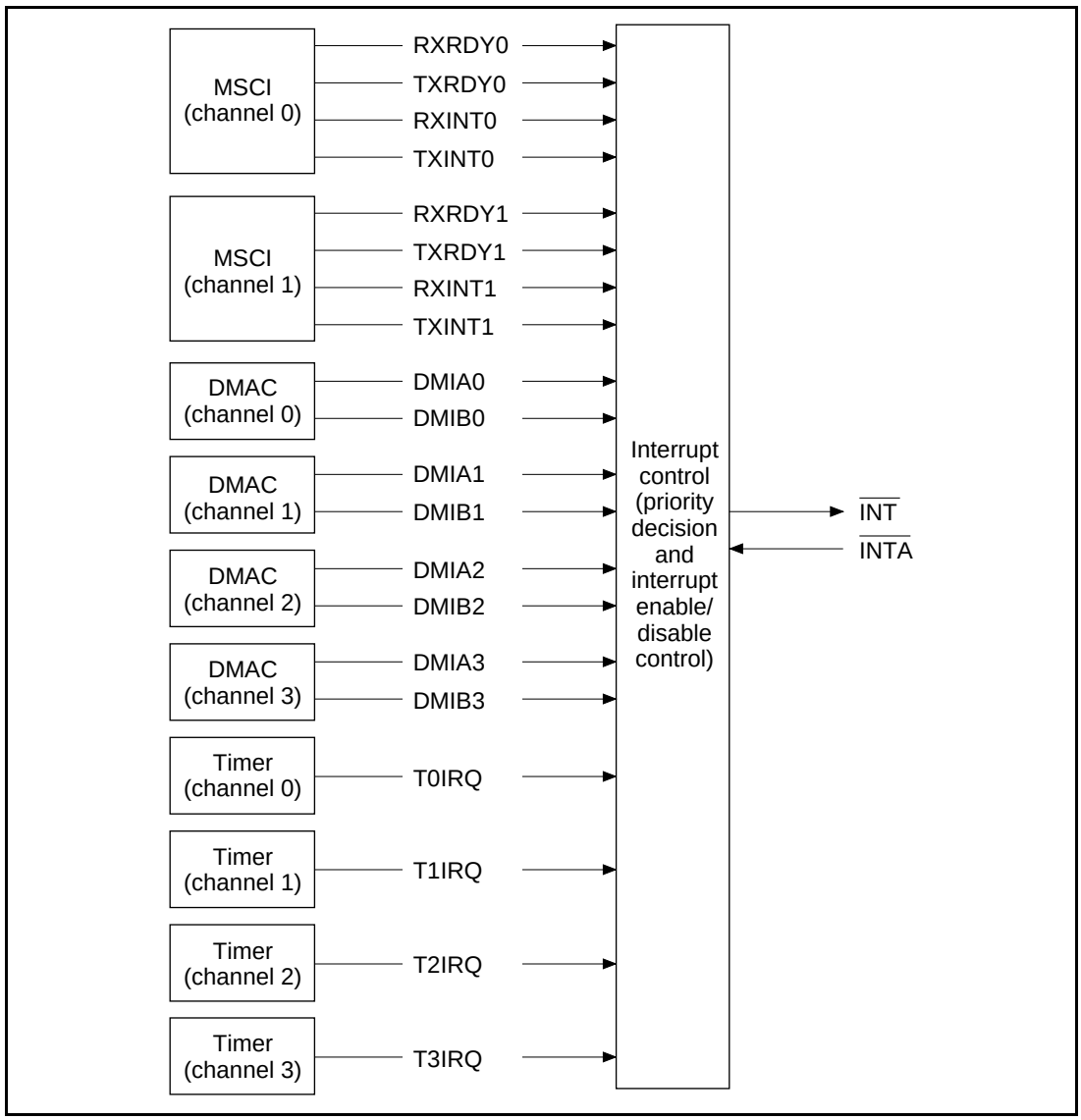


Figure 4.1 Interrupt Sources

Table 4.1 lists the interrupts in priority order and names of their sources.

Table 4.1 Interrupt Priority and Interrupt Sources

Module	Interrupt Name	Priority	Interrupt Source	
MSCI0	RXRDY0	High	Receive buffer ready	(channel 0)
MSCI0	TXRDY0		Transmit buffer ready	(channel 0)
MSCI0	RXINT0		Receive status	(channel 0)
MSCI0	TXINT0		Transmit status	(channel 0)
MSCI1	RXRDY1		Receive buffer ready	(channel 1)
MSCI1	TXRDY1		Transmit buffer ready	(channel 1)
MSCI1	RXINT1		Receive status	(channel 1)
MSCI1	TXINT1		Transmit status	(channel 1)
DMAC0	DMIA0		Error interrupt	(channel 0)
DMAC0	DMIB0		Normal end interrupt	(channel 0)
DMAC1	DMIA1		Error interrupt	(channel 1)
DMAC1	DMIB1		Normal end interrupt	(channel 1)
DMAC2	DMIA2		Error interrupt	(channel 2)
DMAC2	DMIB2		Normal end interrupt	(channel 2)
DMAC3	DMIA3		Error interrupt	(channel 3)
DMAC3	DMIB3		Normal end interrupt	(channel 3)
Timer 0	T0IRQ		Count match	(channel 0)
Timer 1	T1IRQ		Count match	(channel 1)
Timer 2	T2IRQ		Count match	(channel 2)
Timer 3	T3IRQ	Low	Count match	(channel 3)

Note: The MSCI and DMAC priorities can be interchanged by setting a bit in the interrupt control register (ITCR). For details, see section 4.2.3, Interrupt Control Register, and section 4.2.2, Interrupt Modified Vector Register.

Interrupt requests are generated by status bits set in interrupt status registers 0, 1, and 2 (ISR0, ISR1, ISR2). If a requested interrupt is enabled by the corresponding bit in interrupt enable register 0, 1, or 2 (IER0, IER1, IER2), the request is sent to the MPU.

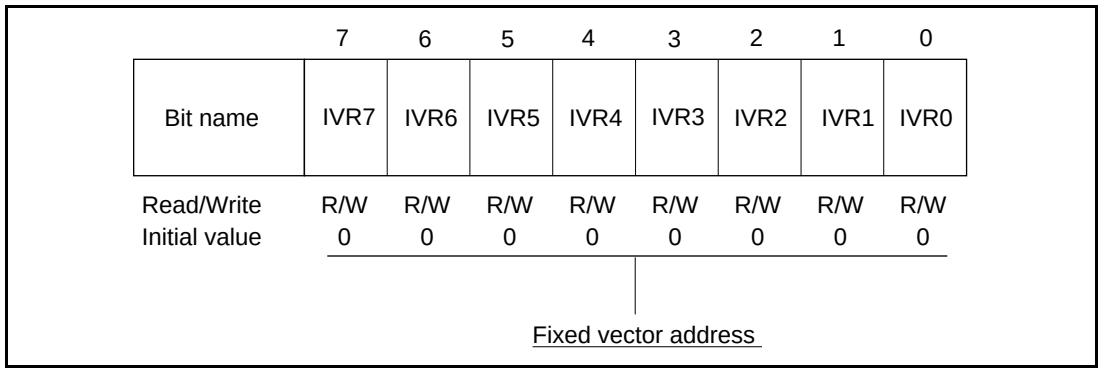
See sections 4.2.4 to 4.2.9 for details of interrupt status registers 0 to 2 and interrupt enable registers 0 to 2.

4.2 Registers

The SCA has nine registers for interrupt control. These registers can be accessed by read and write instructions from the MPU.

4.2.1 Interrupt Vector Register (IVR)

The interrupt vector register stores the vector address output to the MPU in an interrupt acknowledge cycle.

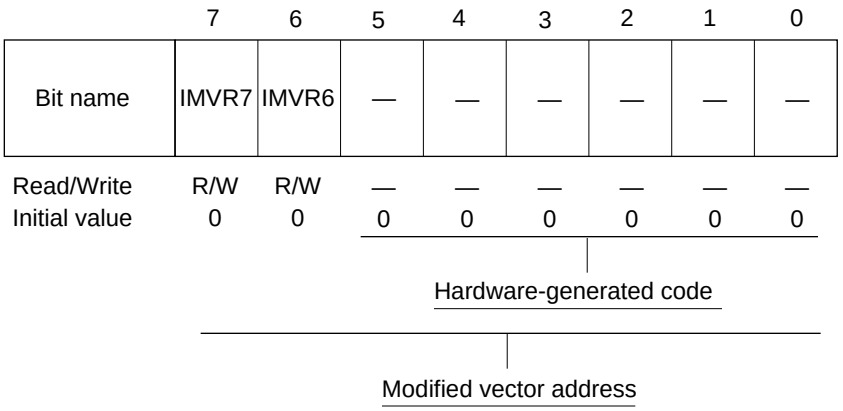


Any desired vector address can be set. Vector output will be detailed in section 4.3, Vector Output.

4.2.2 Interrupt Modified Vector Register (IMVR)

The interrupt modified vector register stores a modifiable vector address output to the MPU in an interrupt acknowledge cycle.

The register consists of eight bits. The six low-order bits hold a hardware-generated code identifying the interrupt source. If multiple interrupt sources are active simultaneously, IMVR holds the code of the source with the highest priority. Any desired value can be set in bits 7 and 6 (IMVR7, IMVR6).

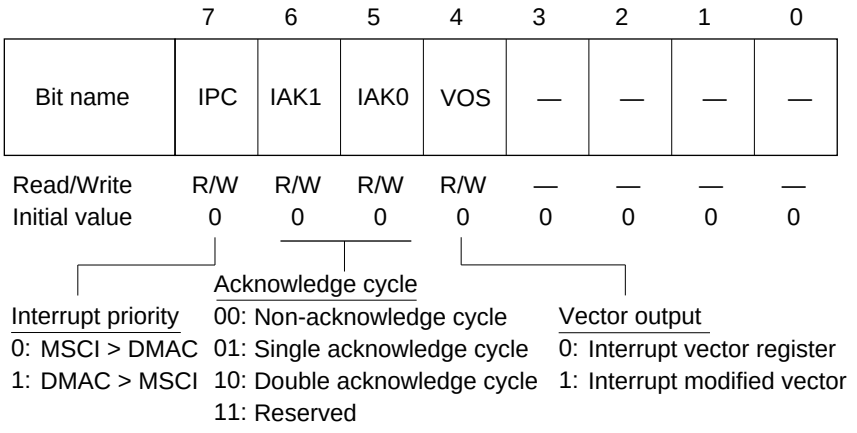


Note: The codes generated for each interrupt source are listed in table 4.2.

Bits 7 and 6 are cleared to 0 by a reset. Bits 5 to 0 always read 0. When writing to IMVR, write 0 in bits 5 to 0.

4.2.3 Interrupt Control Register (ITCR)

The interrupt control register controls the priority order of interrupt sources, and selects the type of acknowledge cycle and vector output.



Note: Bit 3–bit 0 are reserved. These bits always read 0 and must be set to 0.

Bit 7 (IPC: Interrupt Priority Control): Controls the priority order of interrupt sources. This bit is cleared to 0 at a reset.

IPC = 0: MSCI interrupt sources have higher priority than DMAC interrupt sources.

IPC = 1: DMAC interrupt sources have higher priority than MSCI interrupt sources.

Bit 6, Bit 5 (IAK1, IAK0: Interrupt Acknowledge Cycle): Select the type of acknowledge cycle. These bits are cleared to 0 at a reset.

IAK1, IAK0 = 0, 0: Non-acknowledge cycle; the data bus is left in the high-impedance state even when the $\overline{\text{INTA}}$ line is driven active low. Although the $\overline{\text{INTA}}$ line input is ignored if this type of acknowledge cycle is selected, this line should be pulled up to V_{CC}.

IAK1, IAK0 = 0, 1: Single acknowledge cycle; the value in IVR or IMVR is output on data bus lines D_7 to D_0 at the first active (low) input on the $\overline{\text{INTA}}$ line. Output for D_{15} to D_8 is undefined.

IAK1, IAK0 =1, 0: Double acknowledge cycle; the data bus is left in the high-impedance state at the first active (low) input on the $\overline{\text{INTA}}$ line. The value in IVR or IMVR is output on data bus lines D_7 to D_0 at the second active (low) input on the $\overline{\text{INTA}}$ line. Output for D_{15} to D_8 is undefined.

IAK1, IAK0 =1, 1: Reserved.

Bit 4 (VOS: Vector Output Select): Selects which vector to output in a single or double acknowledge cycle. This bit is cleared to 0 at a reset.

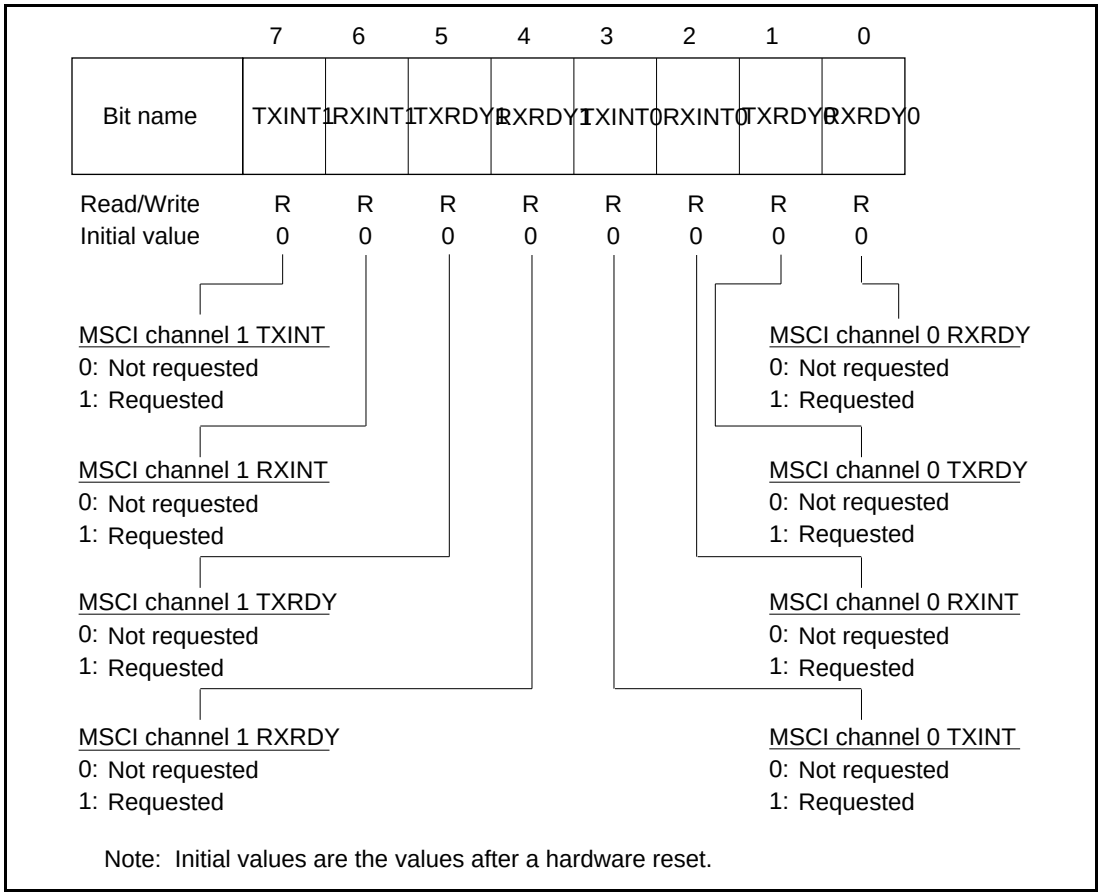
VOS = 0: The interrupt vector register is selected. The unmodified IVR contents are output in a single or double acknowledge cycle.

VOS = 1: The interrupt modified vector register is selected. The IMVR contents are output in a single or double acknowledge cycle.

Bits 3—0: Reserved. These bits always read 0 and must be set to 0.

4.2.4 Interrupt Status Register 0 (ISR0)

The read-only interrupt status register 0 indicates the status of interrupt request sources. All bits are cleared to 0 at a reset.



Bit 7 (TXINT1: MSCI Channel 1 TXINT):

TXINT1 = 0: MSCI channel 1 is not requesting a TxINT interrupt.

TXINT1 = 1: MSCI channel 1 is requesting a TxINT interrupt.

Bit 6 (RXINT1: MSCI Channel 1 RXINT):

RXINT1 = 0: MSCI channel 1 is not requesting a RXINT interrupt.

RXINT1 = 1: MSCI channel 1 is requesting a RXINT interrupt.

Bit 5 (TxRDY1: MSCI Channel 1 TXRDY):

TXRDY1 = 0: MSCI channel 1 is not requesting a TXRDY interrupt.

TXRDY1 = 1: MSCI channel 1 is requesting a TXRDY interrupt.

Bit 4 (RXRDY1: MSCI Channel 1 RXRDY):

RXRDY1 = 0: MSCI channel 1 is not requesting a RXRDY interrupt.

RXRDY1 = 1: MSCI channel 1 is requesting a RXRDY interrupt.

Bit 3 (TXINT0: MSCI Channel 0 TXINT):

TXINT0 = 0: MSCI channel 0 is not requesting a TXINT interrupt.

TXINT0 = 1: MSCI channel 0 is requesting a TXINT interrupt.

Bit 2 (RXINT0: MSCI Channel 0 RxINT):

RXINT0 = 0: MSCI channel 0 is not requesting a RXINT interrupt.

RXINT0 = 1: MSCI channel 0 is requesting a RXINT interrupt.

Bit 1 (TxRDY0: MSCI Channel 0 TXRDY):

TXRDY0 = 0: MSCI channel 0 is not requesting a TXRDY interrupt.

TXRDY0 = 1: MSCI channel 0 is requesting a TXRDY interrupt.

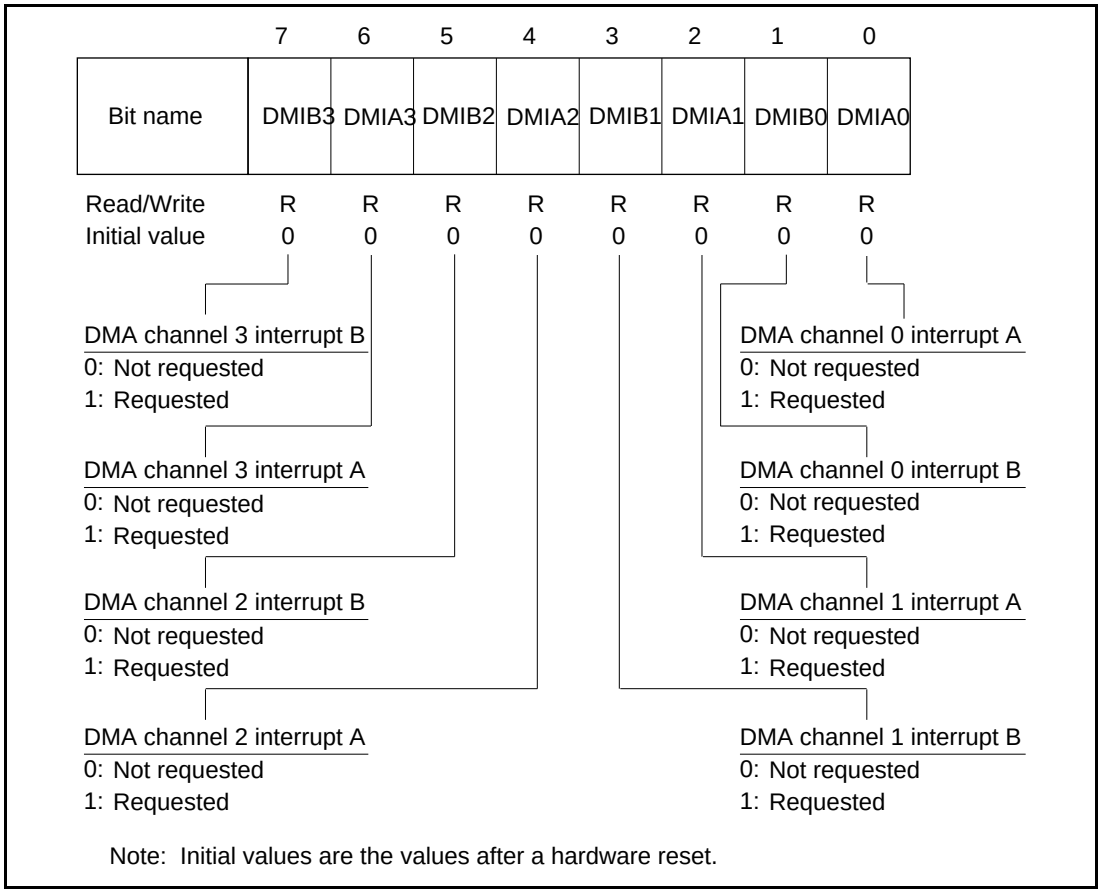
Bit 0 (RXRDY0: MSCI Channel 0 RXRDY):

RXRDY0 = 0: MSCI channel 0 is not requesting a RXRDY interrupt.

RXRDY0 = 1: MSCI channel 0 is requesting a RXRDY interrupt.

4.2.5 Interrupt Status Register 1 (ISR1)

The read-only interrupt status register 1 indicates the status of interrupt request sources. All bits are cleared to 0 at a reset.



Bit 7 (DMIB3: DMA Channel 3 Interrupt B):

DMIB3 = 0: DMAC channel 3 is not requesting a DMIB interrupt.

DMIB3 = 1: DMAC channel 3 is requesting a DMIB interrupt.

Bit 6 (DMIA3: DMA Channel 3 Interrupt A):

DMIA3 = 0: DMAC channel 3 is not requesting a DMIA interrupt.

DMIA3 = 1: DMAC channel 3 is requesting a DMIA interrupt.

Bit 5 (DMIB2: DMA Channel 2 Interrupt B):

DMIB2 = 0: DMAC channel 2 is not requesting a DMIB interrupt.

DMIB2 = 1: DMAC channel 2 is requesting a DMIB interrupt.

Bit 4 (DMIA2: DMA Channel 2 Interrupt A):

DMIA2 = 0: DMAC channel 2 is not requesting a DMIA interrupt.

DMIA2 = 1: DMAC channel 2 is requesting a DMIA interrupt.

Bit 3 (DMIB1: DMA Channel 1 Interrupt B):

DMIB1 = 0: DMAC channel 1 is not requesting a DMIB interrupt.

DMIB1 = 1: DMAC channel 1 is requesting a DMIB interrupt.

Bit 2 (DMIA1: DMA Channel 1 Interrupt A):

DMIA1 = 0: DMAC channel 1 is not requesting a DMIA interrupt.

DMIA1 = 1: DMAC channel 1 is requesting a DMIA interrupt.

Bit 1 (DMIB0: DMA Channel 0 Interrupt B):

DMIB0 = 0: DMAC channel 0 is not requesting a DMIB interrupt.

DMIB0 = 1: DMAC channel 0 is requesting a DMIB interrupt.

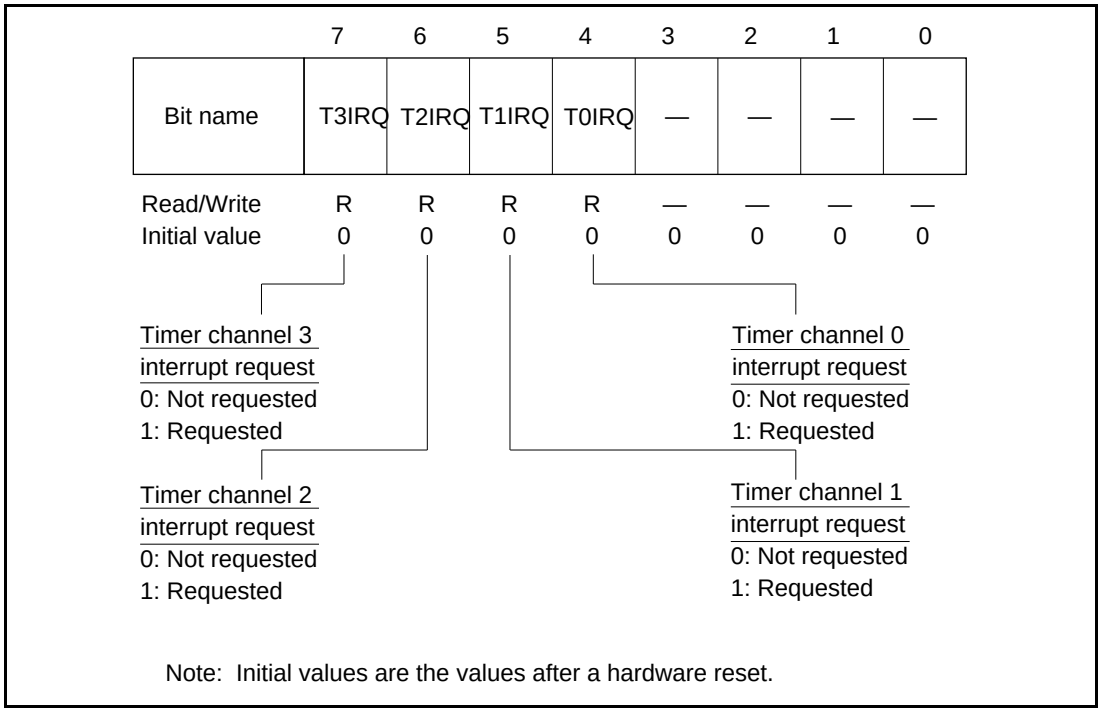
Bit 0 (DMIA0: DMA Channel 0 Interrupt A):

DMIA0 = 0: DMAC channel 0 is not requesting a DMIA interrupt.

DMIA0 = 1: DMAC channel 0 is requesting a DMIA interrupt.

4.2.6 Interrupt Status Register 2 (ISR2)

The read-only interrupt status register 2 indicates the status of interrupt request sources. Bits 7 to 4 are cleared to 0 at a reset. Bits 3 to 0 are reserved bits that always read 0.



Note: Initial values are the values after a hardware reset.

Bit 7 (T3IRQ: Timer Channel 3 Interrupt Request):

T3IRQ = 0: Timer channel 3 is not requesting a T3IRQ interrupt.

T3IRQ = 1: Timer channel 3 is requesting a T3IRQ interrupt.

Bit 6 (T2IRQ: Timer Channel 2 Interrupt Request):

T2IRQ = 0: Timer channel 2 is not requesting a T2IRQ interrupt.

T2IRQ = 1: Timer channel 2 is requesting a T2IRQ interrupt.

Bit 5 (T1IRQ: Timer Channel 1 Interrupt Request):

T1IRQ = 0: Timer channel 1 is not requesting a T1IRQ interrupt.

T1IRQ = 1: Timer channel 1 is requesting a T1IRQ interrupt.

Bit 4 (T0IRQ: Timer Channel 0 Interrupt Request):

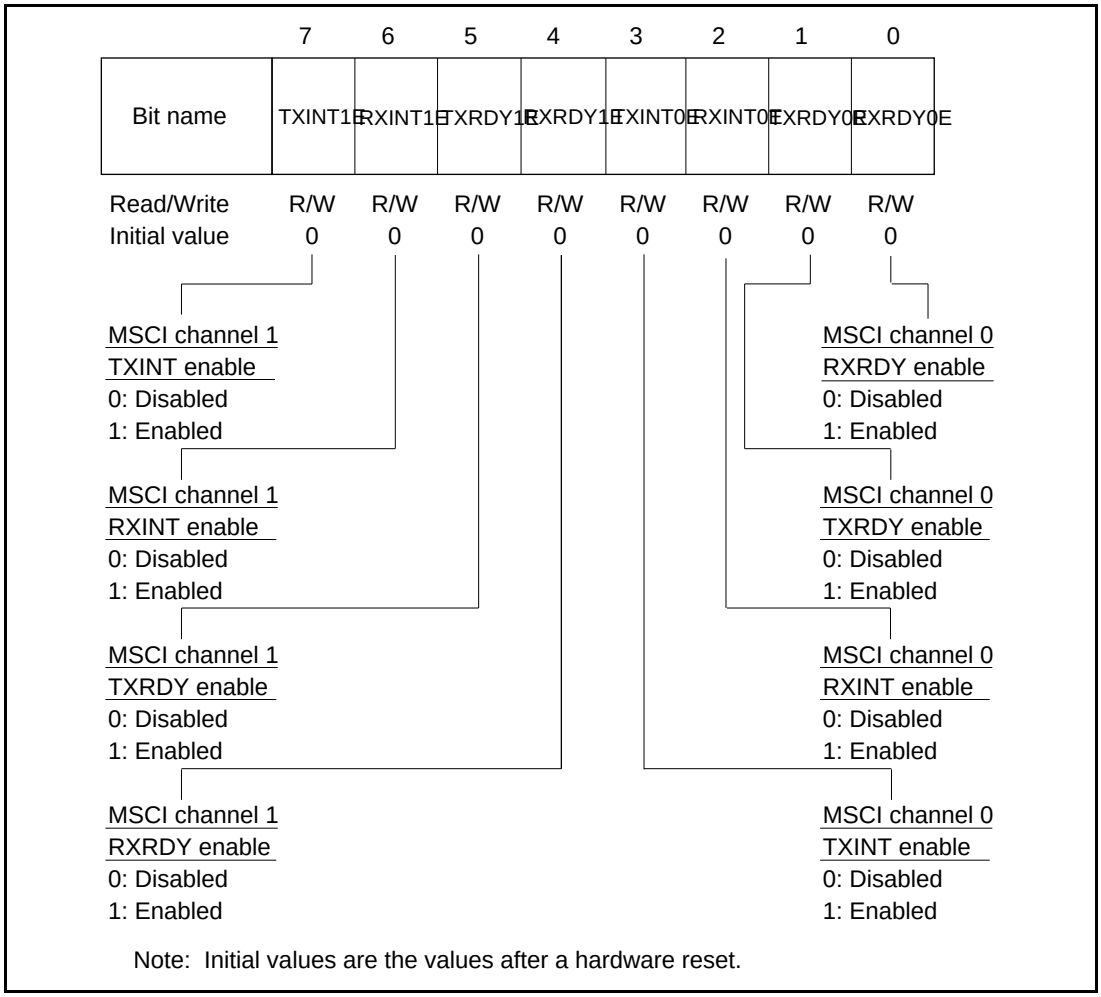
T0IRQ = 0: Timer channel 0 is not requesting a T0IRQ interrupt.

T0IRQ = 1: Timer channel 0 is requesting a T0IRQ interrupt.

Bit 3–Bit 0: Reserved. These bits always read 0 .

4.2.7 Interrupt Enable Register 0 (IER0)

The interrupt enable register 0 enables or disables interrupt requests indicated in interrupt status register 0 (ISR0). All IER0 bits are cleared to 0 at a reset.



Bit 7 (TXINT1E: MSCI Channel 1 TXINT Enable):

TXINT1E = 0: The MSCI channel 1 TXINT interrupt is disabled.

TXINT1E = 1: The MSCI channel 1 TXINT interrupt is enabled.

Bit 6 (RXINT1E: MSCI Channel 1 RXINT Enable):

RXINT1E = 0: The MSCI channel 1 RXINT interrupt is disabled.

RXINT1E = 1: The MSCI channel 1 RXINT interrupt is enabled.

Bit 5 (TXRDY1E: MSCI Channel 1 TXRDY Enable):

TXRDY1E = 0: The MSCI channel 1 TXRDY interrupt is disabled.

TXRDY1E = 1: The MSCI channel 1 TXRDY interrupt is enabled.

Bit 4 (RXRDY1E: MSCI Channel 1 RXRDY Enable):

RXRDY1E = 0: The MSCI channel 1 RXRDY interrupt is disabled.

RXRDY1E = 1: The MSCI channel 1 RXRDY interrupt is enabled.

Bit 3 (TXINT0E: MSCI Channel 0 TXINT Enable):

TXINT0E = 0: The MSCI channel 0 TXINT interrupt is disabled.

TXINT0E = 1: The MSCI channel 0 TXINT interrupt is enabled.

Bit 2 (RXINT0E: MSCI Channel 0 RXINT Enable):

RXINT0E = 0: The MSCI channel 0 RXINT interrupt is disabled.

RXINT0E = 1: The MSCI channel 0 RXINT interrupt is enabled.

Bit 1 (TXRDY0E: MSCI Channel 0 TXRDY Enable):

TXRDY0E = 0: The MSCI channel 0 TXRDY interrupt is disabled.

TXRDY0E = 1: The MSCI channel 0 TXRDY interrupt is enabled.

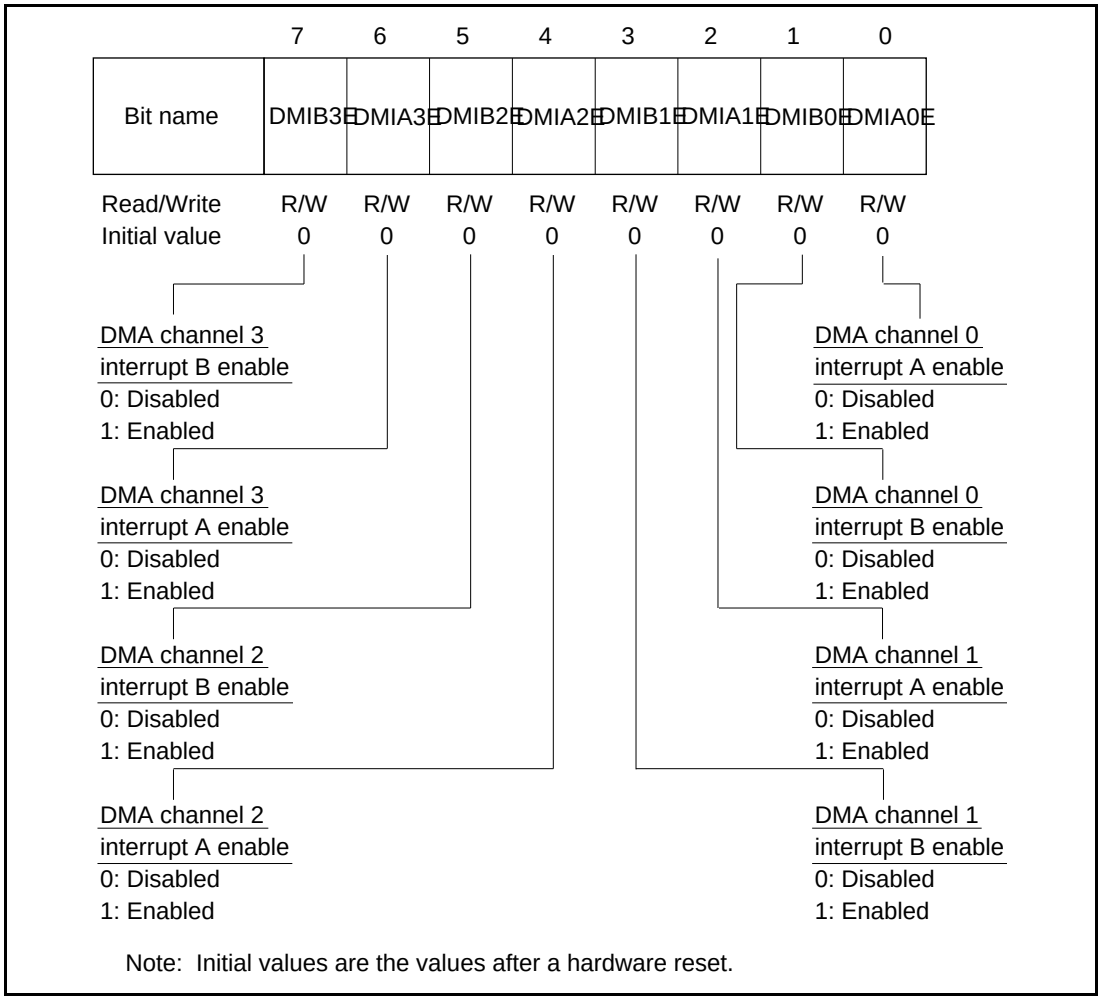
Bit 0 (RXRDY0E: MSCI Channel 0 RXRDY Enable):

RXRDY0E = 0: The MSCI channel 0 RXRDY interrupt is disabled.

RXRDY0E = 1: The MSCI channel 0 RXRDY interrupt is enabled.

4.2.8 Interrupt Enable Register 1 (IER1)

The interrupt enable register 1 enables or disables interrupt requests indicated in interrupt status register 1 (ISR1). All IER1 bits are cleared to 0 at a reset.



Bit 7 (DMIB3E: DMA Channel 3 Interrupt B Enable):

DMIB3E = 0: The DMAC channel 3 DMIB interrupt is disabled.

DMIB3E = 1: The DMAC channel 3 DMIB interrupt is enabled.

Bit 6 (DMIA3E: DMA Channel 3 Interrupt A Enable):

DMIA3E = 0: The DMAC channel 3 DMIA interrupt is disabled.

DMIA3E = 1: The DMAC channel 3 DMIA interrupt is enabled.

Bit 5 (DMIB2E: DMA Channel 2 Interrupt B Enable):

DMIB2E = 0: The DMAC channel 2 DMIB interrupt is disabled.

DMIB2E = 1: The DMAC channel 2 DMIB interrupt is enabled.

Bit 4 (DMIA2E: DMA Channel 2 Interrupt A Enable):

DMIA2E = 0: The DMAC channel 2 DMIA interrupt is disabled.

DMIA2E = 1: The DMAC channel 2 DMIA interrupt is enabled.

Bit 3 (DMIB1E: DMA Channel 1 Interrupt B Enable):

DMIB1E = 0: The DMAC channel 1 DMIB interrupt is disabled.

DMIB1E = 1: The DMAC channel 1 DMIB interrupt is enabled.

Bit 2 (DMIA1E: DMA Channel 1 Interrupt A Enable):

DMIA1E = 0: The DMAC channel 1 DMIA interrupt is disabled.

DMIA1E = 1: The DMAC channel 1 DMIA interrupt is enabled.

Bit 1 (DMIB0E: DMA Channel 0 Interrupt B Enable):

DMIB0E = 0: The DMAC channel 0 DMIB interrupt is disabled.

DMIB0E = 1: The DMAC channel 0 DMIB interrupt is enabled.

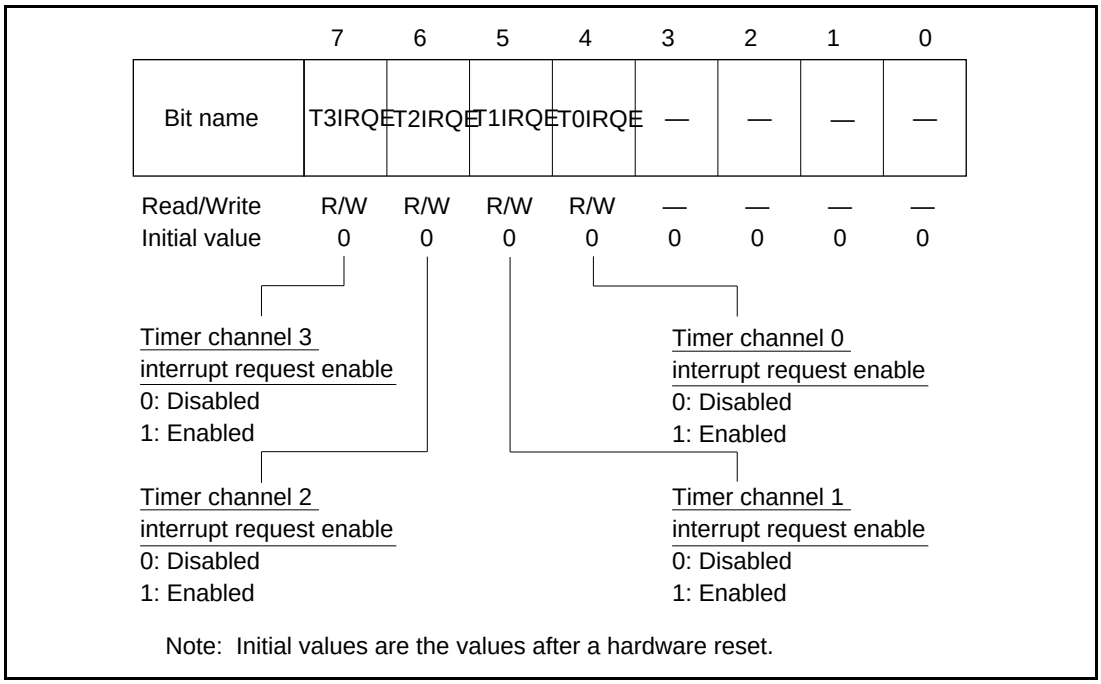
Bit 0 (DMIA0E: DMA Channel 0 Interrupt A Enable):

DMIA0E = 0: The DMAC channel 0 DMIA interrupt is disabled.

DMIA0E = 1: The DMAC channel 0 DMIA interrupt is enabled.

4.2.9 Interrupt Enable Register 2 (IER2)

The interrupt enable register 2 enables or disables interrupt requests indicated in interrupt status register 2 (ISR2). IER2 bits 7 to 4 are cleared to 0 at a reset. Bits 3 to 0 are reserved bits that always read 0. When writing to IER2, write 0 in bits 3 to 0.



Note: Initial values are the values after a hardware reset.

Bit 7 (T3IRQE: Timer Channel 3 Interrupt Request Enable):

T3IRQE = 0: The timer channel 3 T3IRQ interrupt is disabled.

T3IRQE = 1: The timer channel 3 T3IRQ interrupt is enabled.

Bit 6 (T2IRQE: Timer Channel 2 Interrupt Request Enable):

T2IRQE = 0: The timer channel 2 T2IRQ interrupt is disabled.

T2IRQE = 1: The timer channel 2 T2IRQ interrupt is enabled.

Bit 5 (T1IRQE: Timer Channel 1 Interrupt Request Enable):

T1IRQE = 0: The timer channel 1 T1IRQ interrupt is disabled.

T1IRQE = 1: The timer channel 1 T1IRQ interrupt is enabled.

Bit 4 (T0IRQE: Timer Channel 0 Interrupt Request Enable):

T0IRQE = 0: The timer channel 0 T0IRQ interrupt is disabled.

T0IRQE = 1: The timer channel 0 T0IRQ interrupt is enabled.

Bit 3–**Bit 0:** Reserved. These bits always read 0 and must be set to 0.

4.3 Vector Output

Two types of vectors can be selected for output from the SCA. The vector is output on data bus lines D_7 to D_0. (The output on lines D_{15} to D_8 is undetermined.)

1. Fixed vector: An arbitrary 8-bit value can be set in the interrupt vector register (IVR) for output as a fixed interrupt vector.
2. Modified vector: An arbitrary 2-bit value can be set in bits 7 and 6 of the interrupt modified vector register (IMVR). The other six bits are modified according to the interrupt source. The IMVR value is output as the interrupt vector.

These two types of vector output are selected by setting a bit in the interrupt control register (ITCR). See section 4.2.3, Interrupt Control Register.

4.4 Acknowledge Cycle

Three types of acknowledge cycles can be selected for the SCA.

1. Non-acknowledge cycle: The data bus remains in the high-impedance state even when $\overline{\text{INTA}}$ is driven active (low).
2. Single acknowledge cycle: The IVR or IMVR contents are output on the data bus at the first active (low) $\overline{\text{INTA}}$ input (figure 4.2).
3. Double acknowledge cycle: The first active (low) $\overline{\text{INTA}}$ input is ignored; the data bus is left in the high-impedance state. The IVR or IMVR contents are output on the data bus at the second active (low) $\overline{\text{INTA}}$ input (figure 4.3).

If $\overline{\text{INTA}}$ goes low when no interrupt is requested (when $\overline{\text{INT}}$ is inactive), no vector is output.

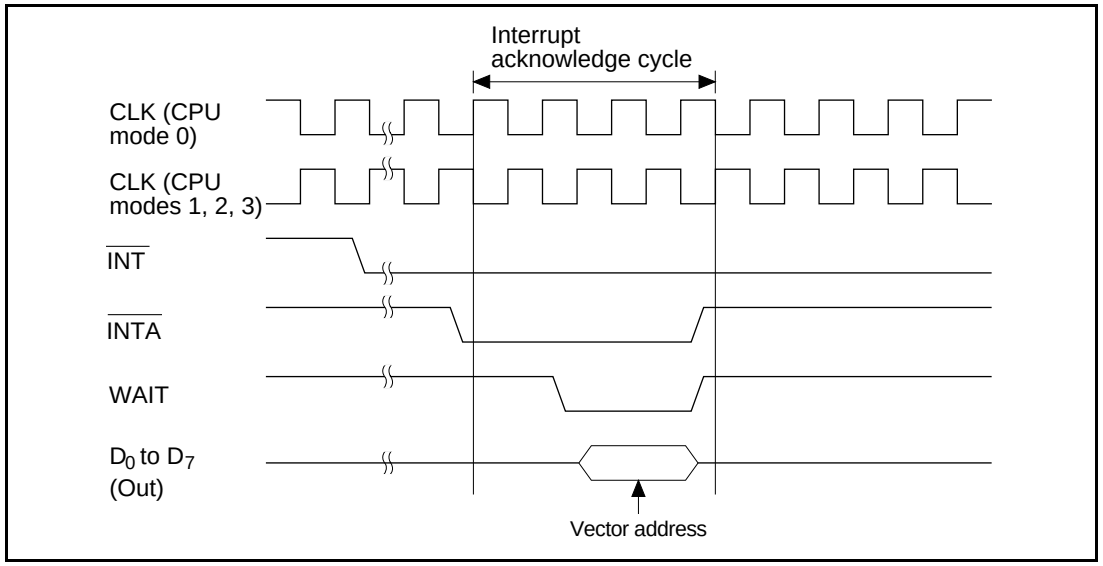


Figure 4.2 Timing Sequence of Single Acknowledge Cycle

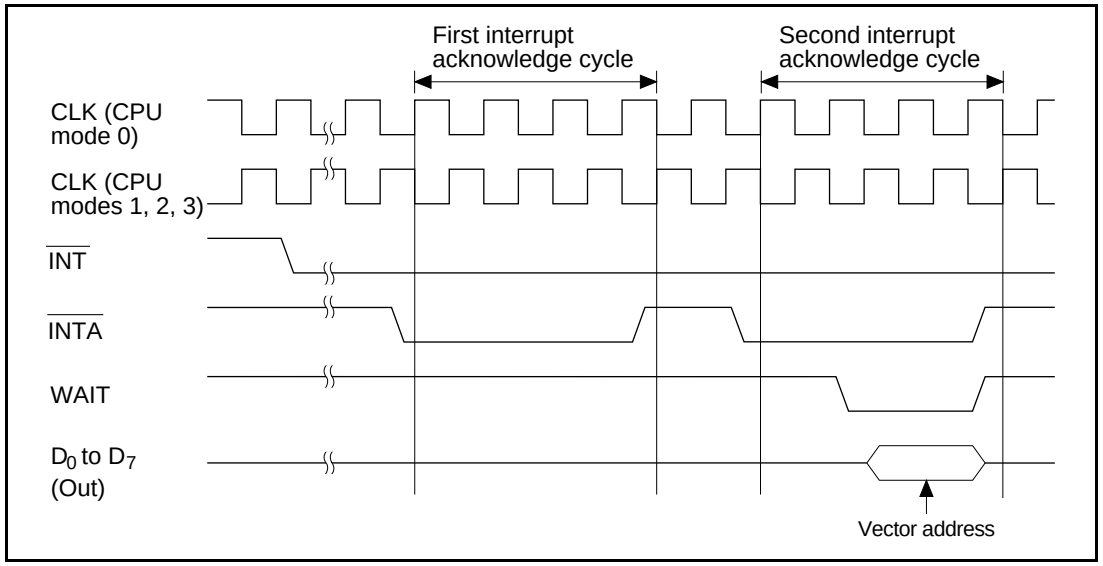


Figure 4.3 Timing Sequence of Double Acknowledge Cycle

4.5 Interrupt Sources and Vector Addresses

The interrupt modified vector register (IMVR) is an eight-bit register in which the six low bits (bits 5 to 0) hold a hardware-generated code identifying an interrupt source, as listed in table 4.2. The two high bits (bits 7 and 6) can be set to arbitrary values by the MPU.

Table 4.2 Interrupt Sources and Vector Addresses

			Priority*[1]		Vector Address							
			VOS*[2] = 0	VOS*[2] = 1	Programmable		Hardware-Generated Code					
No.	Interrupt Source				b_7	b_6	b_5	b_4	b_3	b_2	b_1	b_0
1	MSCI channel 0	RXRDY	1	9	×	×	0	0	0	1	0	0
2	MSCI channel 0	TXRDY	2	10	×	×	0	0	0	1	1	0
3	MSCI channel 0	RXINT	3	11	×	×	0	0	1	0	0	0
4	MSCI channel 0	TXINT	4	12	×	×	0	0	1	0	1	0
5	MSCI channel 1	RXRDY	5	13	×	×	1	0	0	1	0	0
6	MSCI channel 1	TXRDY	6	14	×	×	1	0	0	1	1	0
7	MSCI channel 1	RXINT	7	15	×	×	1	0	1	0	0	0
8	MSCI channel 1	TXINT	8	16	×	×	1	0	1	0	1	0
9	DMAC channel 0	DMIA0	9	1	×	×	0	1	0	1	0	0
10	DMAC channel 0	DMIB0	10	2	×	×	0	1	0	1	1	0
11	DMAC channel 1	DMIA1	11	3	×	×	0	1	1	0	0	0
12	DMAC channel 1	DMIB1	12	4	×	×	0	1	1	0	1	0
13	DMAC channel 2	DMIA2	13	5	×	×	1	1	0	1	0	0
14	DMAC channel 2	DMIB2	14	6	×	×	1	1	0	1	1	0
15	DMAC channel 3	DMIA3	15	7	×	×	1	1	1	0	0	0
16	DMAC channel 3	DMIB3	16	8	×	×	1	1	1	0	1	0
17	Timer channel 0	T0IRQ	17	17	×	×	0	1	1	1	0	0
18	Timer channel 1	T1IRQ	18	18	×	×	0	1	1	1	1	0
19	Timer channel 2	T2IRQ	19	19	×	×	1	1	1	1	0	0
20	Timer channel 3	T3IRQ	20	20	×	×	1	1	1	1	1	0

(×: Arbitrary value)

Notes: 1. Smaller priority values indicate higher priority. Larger priority values indicate lower priority.

2. The VOS bit in the interrupt control register (ITCR).

Section 5 Multiprotocol Serial Communication Interface (MSCI)

5.1 Overview

The multiprotocol serial communication interface (MSCI) supports three different communication modes: asynchronous, byte synchronous, and bit synchronous.

The two full-duplex channels of the MSCIs, built in the SCA, have identical functions but operate independently.

5.1.1 Functions

The MSCI features the following functions:

- Program-selectable operating modes: Asynchronous, byte synchronous, and bit synchronous
- Transmission codes NRZ, NRZI, Manchester, FM0, and FM1 supported (only NRZ code supported in asynchronous mode)
- Full-duplex communications, auto echo, and local loop-back functions available
- 32-stage transmit and receive buffers provided
- Modem control signals RTS, CTS, and DCD automatically controlled using the auto-enable function
  - RTS (request to send): General-purpose output/transmit request
  - CTS (clear to send): General-purpose input/transmit enable/transition-triggered interrupt
  - DCD (data carrier detect): General-purpose input/receive carrier detection/transition-triggered interrupt
- Programmable on-chip baud rate generator for transmission and reception
- Selectable clock sources: External clock input, on-chip baud rate generator output, and internal ADPLL (advanced digital PLL) output
- Noise suppression function for receive clock and receive data
- Data transmission rate of 50 bps to 7.1 Mbps for a 10-MHz system clock, and 50 bps to 12 Mbps for a 16.7-MHz system clock
- Four interrupt signals: RXRDY, TXRDY, RXINT, and TXINT

Functions of the MSCI in asynchronous, byte synchronous, and bit synchronous operating modes can be summarized as follows:

- Asynchronous Mode
  - Programmable character length (5-8 bits/character) for transmission and reception
  - Programmable stop bit length (1, 1.5, or 2 bits)

— Programmable parity (odd, even, or no parity)
— Detection of parity, overrun, and framing errors
— Break transmission and reception
— Multiprocessor (MP) bit transmission and reception
— Programmable bit rate (input clock frequency × 1/1, 1/16, 1/32, or 1/64)

- Byte Synchronous Mode
  — 8-bit character length
  — Mono-sync, bi-sync, and external synchronous modes supported
  — CRC code generation and check. Initial value (all 0s or 1s) selectable for each of CRC-16 and CRC-CCITT generator polynomials
  — Automatic SYN character transmission, detection, and deletion
  — CRC code transmission or no-transmission in underrun state program-selectable
  — SYN character or mark transmission in idle state program-selectable
  — Detection of CRC, overrun, and underrun errors
- Bit Synchronous Mode
  — 8-bit character length
  — HDLC mode supported
  — Information (I) field configured in bytes
  — Automatic zero insertion in transmit data and deletion from receive data
  — Flag or mark transmission program-selectable in idle state
  — 8- or 16-bit address (A) field selectable; Four address field check modes program-selectable
  — End-of-message detection
  — CRC code generation and detection

5.1.2 Configuration and Operation

The MSCI block diagram is shown in figure 1.2.

The MSCI has 27 internal registers that the user can access. These registers specify the operating mode and control transmission and reception operations.

Receiver: The following describes the operations of the MSCI receiver, referring to figure 1.13.

The MSCI receiver has one 32-stage FIFO receive buffer, five 8-bit shift registers, and one delay register. The receiver also has a 6-bit status buffer (FIFO). This buffer retains status information, such as parity or framing errors, related to the received data. Note that status register 2 (ST2) and current status registers 0 and 1 (CST0 and CST1) are located at the top of the status buffer (FIFO) and interface with the internal data bus. For details, see sections 5.2.11, MSCI Status Register 2

(ST2), 5.2.25, MSCI Current Status Register 0 (CST0), and 5.2.26, MSCI Current Status Register 1 (CST1).

Input data is received via the RXD line and enters the MSCI internal circuitry after passing through a decoder. The data path inside the MSCI differs according to the operating mode (asynchronous, byte synchronous, or bit synchronous).

In asynchronous mode, the MSCI checks input data for the parity/MP bit and for framing errors before passing it to receive shift register 4. When one character of data is received, the data is sent to the receive buffer. The MPU or DMAC can read the data from the receive buffer (TX/RX buffer register (TRB)) via the internal data bus. Note that the TX/RX buffer register (TRB) is located at the top of the receive buffer and interfaces with the internal data bus. For details, see section 5.2.21, MSCI TX/RX Buffer Register (TRB).

In byte synchronous mode, input data enters receive shift register 1 before branching toward receive shift register 2 and receive shift register 4.

The data received by receive shift register 2 is used to detect SYN character(s). The data received by receive shift register 4 is transmitted to the receive buffer, and is also transmitted to the RX CRC calculator, for CRC calculation, via the RX delay register and RX CRC shift register.

Output from the CRC calculator goes to status register 2 (ST2). The MPU or DMAC can read the received data and its status.

In bit synchronous mode, input data enters receive shift register 1, which deletes 0s, and detects flags, abort status, and idle status. The data then branches toward receive shift register 2 and the RX CRC calculator. Output from the CRC calculator is sent to ST2, as in byte synchronous mode, and is also sent to the frame status register (FST) at the completion of frame reception. In other words, FST always holds the status of the most recently received frame.

The data sent to receive shift register 2 is sent via receive shift registers 3 and 4 to the receive buffer if the data's secondary station address detected coincides with the present station (SCA) address. The MPU or DMAC can read the received data and its status via the internal data bus. When CRC calculation is disabled (the CRCCC bit of mode register 0 (MD0) is 0), the received data is sent directly from receive shift register 1 to receive shift register 4. The secondary station address is also detected in this case.

Transmitter: The following describes the operations of the MSCI transmitter, referring to figure 1.11.

The MSCI transmitter has one 32-stage FIFO transmit buffer, one transmit shift register, and one TX pattern register. The transmitter also has one CRC calculator, as the receiver does.

The MPU or DMA writes output data via the internal data bus to the transmit buffer (TX/RX buffer register (TRB)). Information necessary to assemble frames in each communications mode

is appended to the output data in the TX shift register. The data is then output to the TXD line via the encoder.

See sections 5.2.1, MSCI Mode Register 0 (MD0), 5.2.2, MSCI Mode Register 1 (MD1), 5.2.4, MSCI Control Register (CTL), 5.2.18, MSCI Synchronous/Address Register 0 (SA0), and 5.2.19, MSCI Synchronous/Address Register 1 (SA1), for details on the specification of parity, stop bit length, and break transmission in asynchronous mode. These sections also contain information on the specification of SYN characters, aborts, flags, and CRC calculation in byte and bit synchronous modes.

Each stage of the transmit buffer has 1-bit EOM/MP bit command FIFO. Refer to section 5.3, Operations. For MP bit transmission in asynchronous mode and EOM transmission in bit synchronous mode, see section 5.2.8, MSCI Command Register (CMD).

5.2 Registers

The MSCI has 27 registers which select the operating mode (asynchronous, byte synchronous, or bit synchronous), and control the transmitter, receiver, ADPLL, and baud rate generator. These registers are accessed with MPU instructions.

For changing the operating mode, these registers must be pre-initialized with a channel reset command by the command register (CMD).

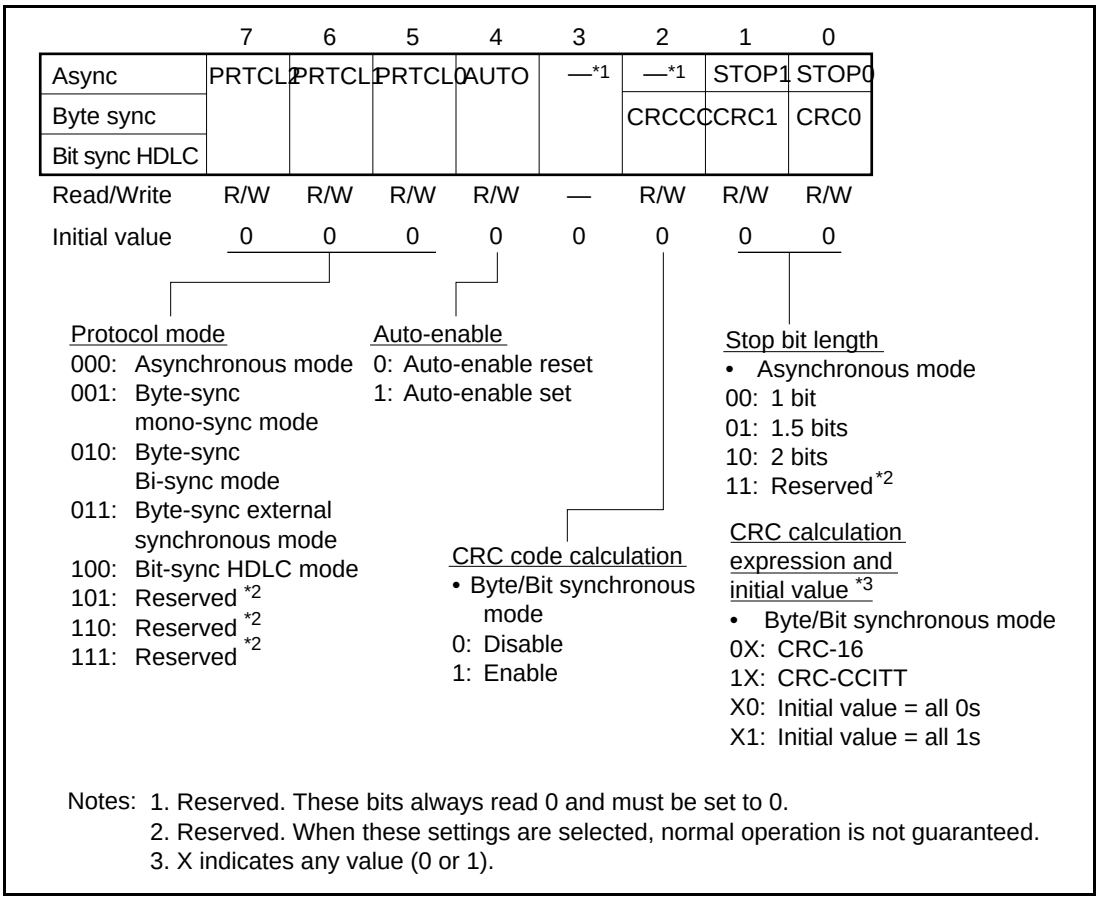
5.2.1 MSCI Mode Register 0 (MD0)

Mode register 0 (MD0) specifies the operating mode (asynchronous, byte synchronous, or bit synchronous), CRC calculation expression, and stop bit length for asynchronous mode, and sets the auto-enable function.

This register is reset under either of the following conditions:

- Hardware reset
- Channel reset command

The receive reset command must be issued immediately after rewriting the contents of MD0, otherwise the contents of the status register may change especially when the contents of MD0 are rewritten after being initially set up after power-on or initialization.



Bits 7–5 (PRTCL2–PRTCL0: Protocol Mode): Specify the transmission protocol (transmission control procedure). These bits must be initialized with a channel reset command before the bit setting is changed. Otherwise, normal operation is not guaranteed.

PRTCL2, PRTCL1, PRTCL0 = 0, 0, 0: Specifies asynchronous mode

PRTCL2, PRTCL1, PRTCL0 = 0, 0, 1: Specifies byte synchronous mono-sync mode

PRTCL2, PRTCL1, PRTCL0 = 0, 1, 0: Specifies byte synchronous bi-sync mode

PRTCL2, PRTCL1, PRTCL0 = 0, 1, 1: Specifies byte synchronous external synchronous mode

PRTCL2, PRTCL1, PRTCL0 = 1, 0, 0: Specifies bit synchronous HDLC mode

PRTCL2, PRTCL1, PRTCL0 = 1, 0, 1: Reserved

PRTCL2, PRTCL1, PRTCL0 = 1, 1, 0: Reserved

PRTCL2, PRTCL1, PRTCL0 = 1, 1, 1: Reserved

Bit 4 (AUTO: Auto-Enable): Controls the modem control signals ($\overline{\text{CTS}}$, $\overline{\text{DCD}}$, and $\overline{\text{RTS}}$).

- Asynchronous/Byte synchronous/Bit synchronous mode

  AUTO = 0: Specifies CTS and DCD as general-purpose inputs, and RTS as a general-purpose output.

  CTS, DCD, and RTS have no effect on MSCI transmission or reception

  AUTO = 1: Sets the auto-enable function. CTS, DCD, and RTS serve as modem control signals for an RS-232C interface or the like. (Note that the auto-enable function of CTS and DCD is available in any operating mode (asynchronous, byte synchronous, or bit synchronous mode), while the function of RTS is available only in asynchronous mode.

  For example, the CTS input controls transmission operations. In asynchronous mode, when the CTS input goes high, the transmitter sends the data from the transmit shift register, then enters idle state with TXD held high. In idle state, no data is transferred from the transmit buffer to the transmit shift register. In byte or bit synchronous mode, when the CTS input goes high, the transmitter sends the data from the transmit shift register, then stops transferring data to this register from the transmit buffer. This generates an underrun error (the UDRN bit of ST1 register is set), and the transmitter enters the idle state according to the sequence specified by the UDRNC bit of the MSCI control register (CTL).

  The DCD input controls reception operations. When DCD is high, reception is disabled. If DCD goes high during character assembly, the data being assembled is lost. However, the data in the receive buffer remains intact. (Character assembly implies sampling of received data and assembly of a character in the receive shift register.)

  The RTS output is low during transmission in asynchronous mode. Otherwise (when TX is disabled or in idle state), the RTS line outputs the value of the RTS bit of the control register (CTL). In byte or bit synchronous mode, the RTS output is independent of transmission operation and the RTS line outputs the value of the RTS bit of the control register (CTL).

  The timing for modem control signal RTS is shown in figure 5.1 (a) to figure 5.1 (h). The RTS output goes high one clock cycle after the TXD line has been set to "mark" after data transmission (figure 5.1 (a)). The RTS output during data write to the transmit buffer (TRB) by the MPU is shown in figure 5.1 (b) to figure 5.1 (e). The RTS output during data transmission to the transmit buffer (TRB) in DMA cycles is shown in figure 5.1 (f) to figure 5.1 (h).

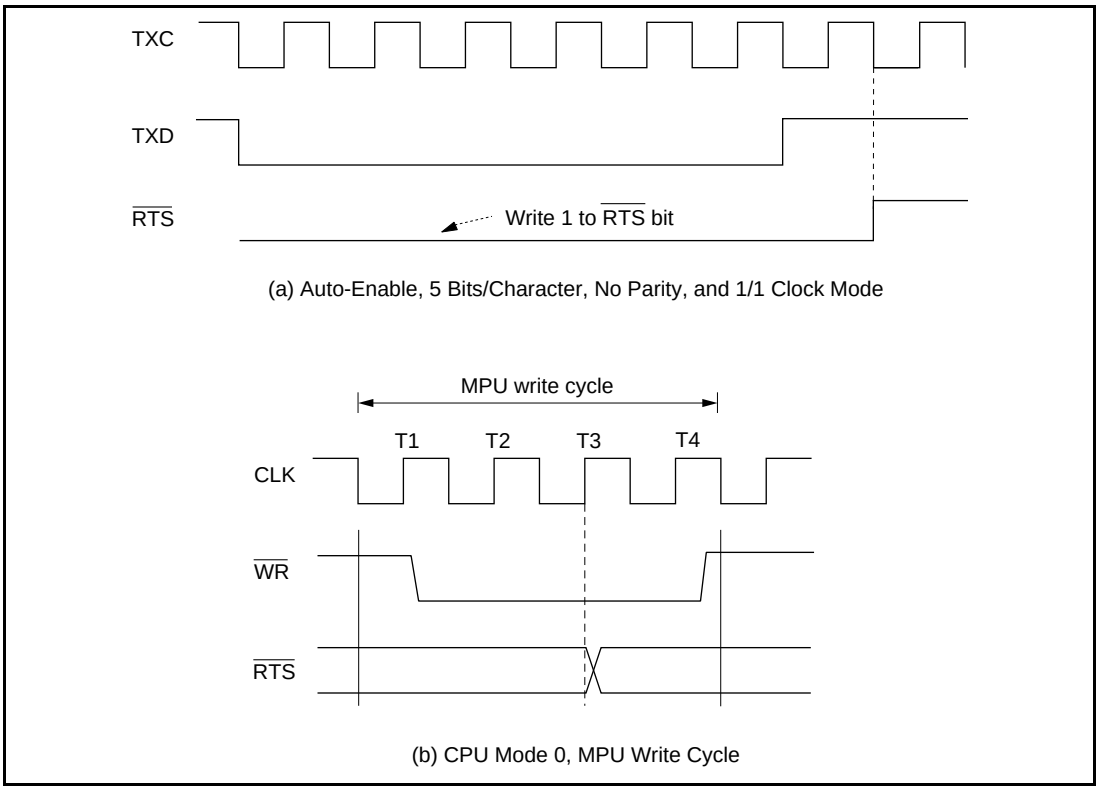


Figure 5.1 Modem Control Signal Timing

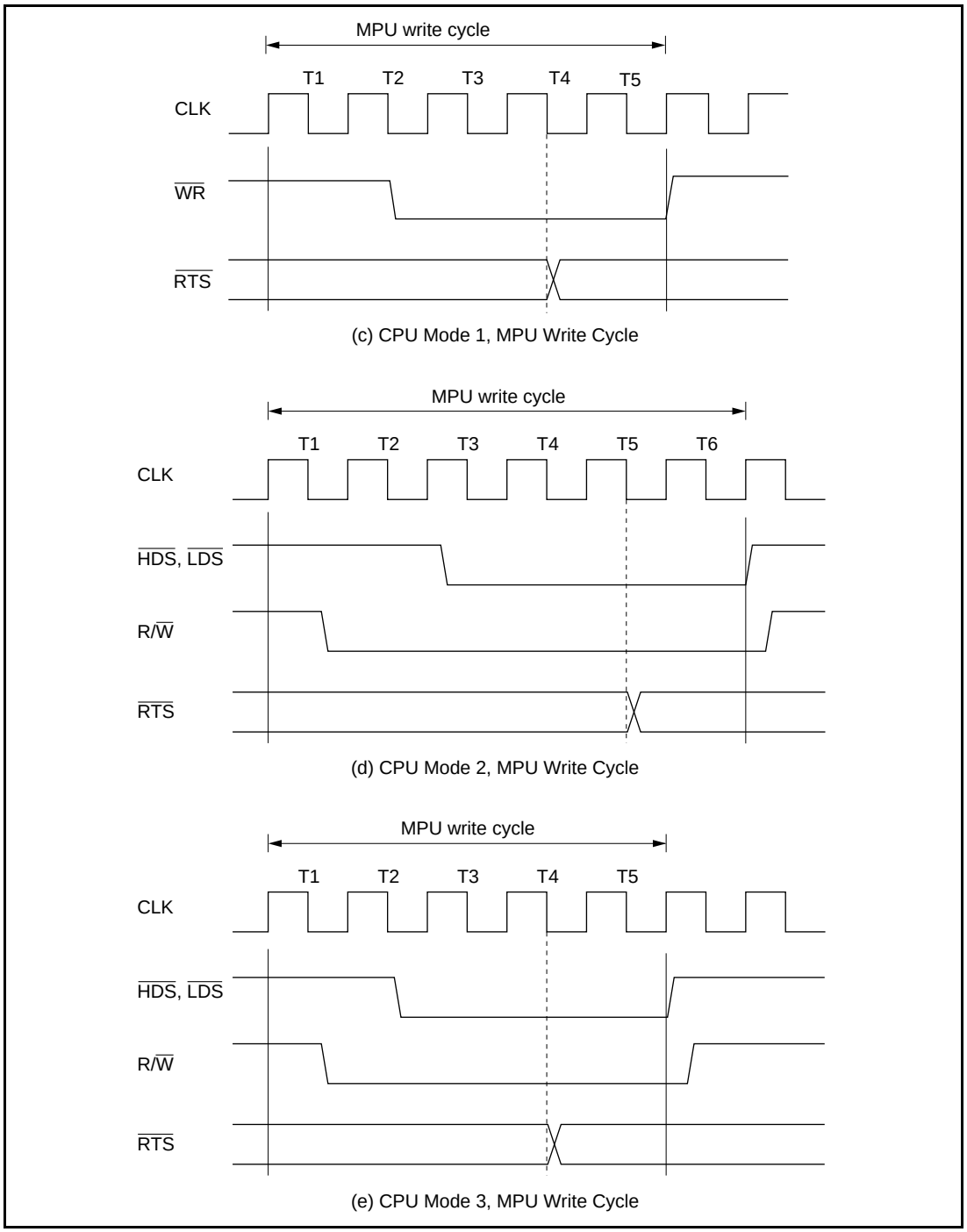
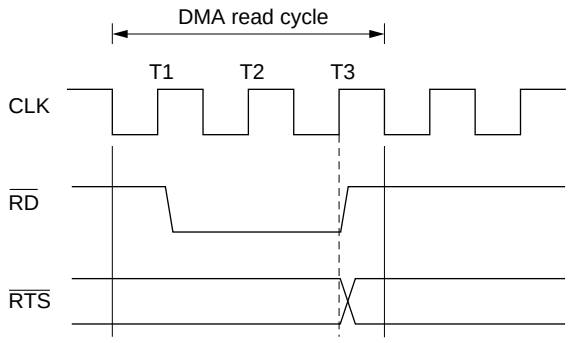
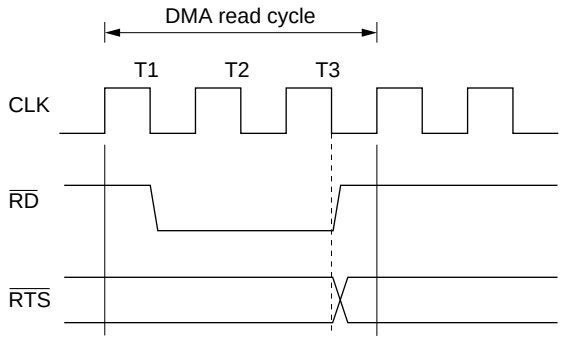


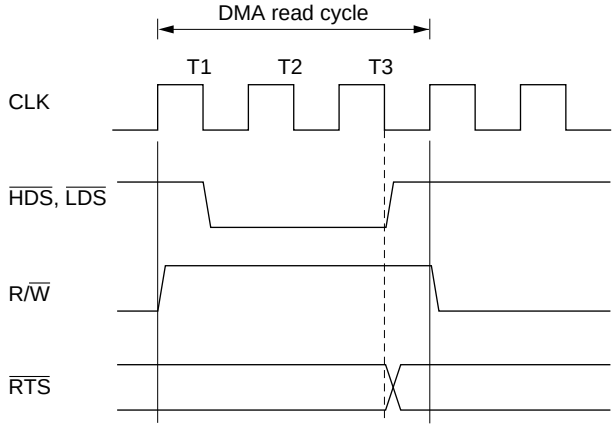
Figure 5.1 Modem Control Signal Timing (cont)



(f) CPU Mode 0, DMA Read Cycle



(g) CPU Mode 1, DMA Read Cycle



(h) CPU Mode 2 and 3, DMA Read Cycle

Figure 5.1 Modem Control Signal Timing (cont)

Bit 3: Reserved. This bit always reads 0 and must be set to 0.

Bit 2 (CRCCC: CRC Code Calculation): Specifies CRC code generation/detection in byte synchronous or bit synchronous mode.

- Asynchronous mode
  Reserved. This bit always reads 0 and must be set to 0.
- Byte synchronous/Bit synchronous mode
  CRCCC = 0: Disables CRC code generation/detection.
  CRCCC = 1: Enables CRC calculation for transmission and reception in byte or bit synchronous mode. Results of the CRC calculation for transmission are sent as CRC codes, while results of the CRC calculation for reception are reflected by the CRCE bit of status register 2 (ST2).
  Deletes FCS (CRC) without transferring it to the receive buffer in bit synchronous mode.

The polarity of the CRC code on the transmit and receive data lines depends on the protocol mode as follows:

Byte synchronous: The calculated CRC value is transmitted on the TXDM line without complementation, regardless of the setting of bits CRC1 and CRC0.
In received CRC calculations, the CRC value on the RXCM line is regarded as non-complemented.

Bit synchronous: The one's complement of the calculated CRC value is transmitted on the TXDM line, regardless of the setting of bits CRC1 and CRC0. In received CRC calculations, the CRC value on the RXDM line is regarded as a one's complement.

Bits 1–0 (STOP1–STOP0/CRC1–CRC0: Stop Bit Length/CRC Calculation Expression and Initial Value): Specify the stop bit length in asynchronous mode and the CRC calculation expression in bit or byte synchronous mode.

- Asynchronous mode
  STOP1, STOP0 = 0, 0: Stop bit length = 1
  STOP1, STOP0 = 0, 1: Stop bit length = 1.5
  STOP1, STOP0 = 1, 0: Stop bit length = 2
  STOP1, STOP0 = 1, 1: Reserved
- Byte synchronous/Bit synchronous mode
  CRC1 = 0: Specifies CRC-16 (X16 + X15 + X2 + 1) for CRC calculators in the transmitter and receiver
  CRC1 = 1: Specifies CRC-CCITT (X16 + X12 + X5 + 1) for CRC calculators in the transmitter and receiver
  CRC0 = 0: Specifies all 0s as the CRC calculator initial value

CRC0 = 1: Specifies all 1s as the CRC calculator initial value

The following CRC bit patterns are sent starting with the most significant bit.

Protocol	CRC-16 Preset 0	CRC-CCITT Preset 0	CRC-16 Preset 1	CRC-CCITT Preset 1*[1]
BOP (bit synchronous mode)	Complemented*[2]	Complemented*[2]	Complemented*[2]	Complemented*[2]
COP (byte synchronous mode)	Not complemented	Not complemented	Not complemented	Not complemented

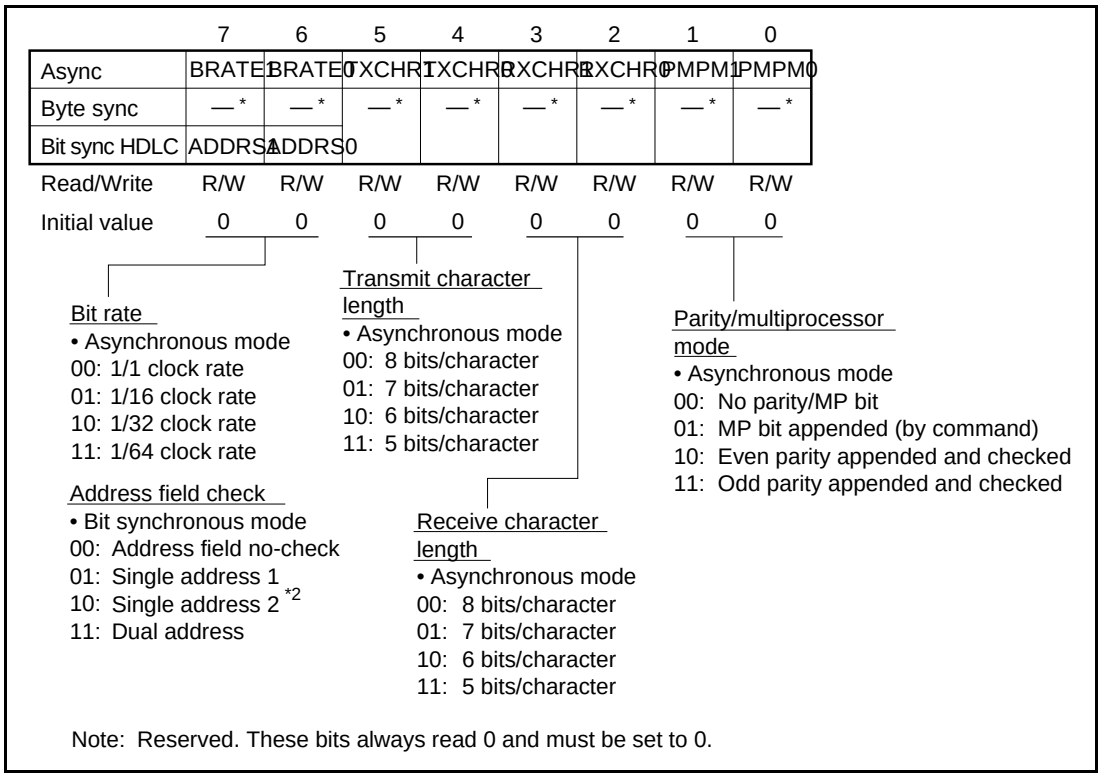
Notes: 1. CRC-CCITT preset 1 is recommended for the HDLC modes in such recommendations LAPB and X.25.
2. One's complement

5.2.2 MSCI Mode Register 1 (MD1)

Mode register 1 (MD1) specifies the transmit/receive character length, the parity/MP bit, and the relationship between the transmit/receive data and the transmit/receive clock, all in asynchronous mode. This register also specifies the checking method for the address field in bit synchronous mode. This register does not function in byte synchronous mode. For the parity/MP bit, see section 5.3, Operations.

This register is reset under either of the following conditions:

- Hardware reset
- Channel reset command



Note: Reserved. These bits always read 0 and must be set to 0.

Bits 7–6 (BRATE1–BRATE0/ADDRS1–ADDRS0: Bit Rate/Address Field Check): Specify the relationship between the bit rate and the transmit/receive clock in asynchronous mode, and the checking method for the address field in bit synchronous mode. These bits are used for both transmission and reception.

- Asynchronous mode
  BRATE1, BRATE0 = 0, 0: Bit rate = 1/1 clock rate
  BRATE1, BRATE0 = 0, 1: Bit rate = 1/16 clock rate
  BRATE1, BRATE0 = 1, 0: Bit rate = 1/32 clock rate
  BRATE1, BRATE0 = 1, 1: Bit rate = 1/64 clock rate
  For details on asynchronous mode, see section 5.3.1, Asynchronous Mode.
- Byte synchronous mode
  Reserved. These bits always read 0 and must be set to 0.
- Bit synchronous mode
  ADDRS1, ADDRS0 = 0, 0: Disables the address field check
  ADDRS1, ADDRS0 = 0, 1: Sets single address 1
  ADDRS1, ADDRS0 = 1, 0: Sets single address 2
  ADDRS1, ADDRS0 = 1, 1: Sets dual address
  For details, see Address Field Check, in section 5.3.4, Bit Synchronous Loop Mode.

Bits 5–4 (TXCHR1–TXCHR0: Transmit Character Length): Specify the character length of the transmit data in asynchronous mode. Rewriting these bits during operation updates the character length for subsequent transmit characters.

- Asynchronous mode
  TXCHR1, TXCHR0 = 0, 0: 8 bits/character
  TXCHR1, TXCHR0 = 0, 1: 7 bits/character
  TXCHR1, TXCHR0 = 1, 0: 6 bits/character
  TXCHR1, TXCHR0 = 1, 1: 5 bits/character
- Byte synchronous/Bit synchronous mode
  Reserved. These bits always read 0 and must be set to 0.

Bits 3–2 (RXCHR1–RXCHR0: Receive Character Length): Specify the character length of the receive data in asynchronous mode. Rewriting these bits during operation updates the character length for subsequent receive characters.

- Asynchronous mode
  RXCHR1, RXCHR0 = 0, 0: 8 bits/character
  RXCHR1, RXCHR0 = 0, 1: 7 bits/character
  RXCHR1, RXCHR0 = 1, 0: 6 bits/character

RXCHR1, RXCHR0 = 1, 1: 5 bits/character

- Byte synchronous/Bit synchronous mode

  Reserved. These bits always read 0 and must be set to 0.

Bits 1–0 (PMPM1–PMPM0: Parity/Multiprocessor Mode): Specify the multiprocessor (MP) mode, and whether or not to use the parity check in asynchronous mode. Rewriting these bits during operation activates the contents of the new setting for subsequent transmit/receive characters.

- Asynchronous mode

  PMPM1, PMPM0 = 0, 0: Appends no parity/MP bit; performs no parity check

  PMPM1, PMPM0 = 0, 1: Appends an MP bit according to the commands

  PMPM1, PMPM0 = 1, 0: Appends even parity for parity check

  PMPM1, PMPM0 = 1, 1: Appends odd parity for parity check

  For the parity check and multiprocessor mode (MP) in asynchronous mode, see Parity /MP Bit, in section 5.3.1, Asynchronous Mode. For commands, see section 5.2.8, MSCI Command Register.

- Byte synchronous/Bit synchronous mode

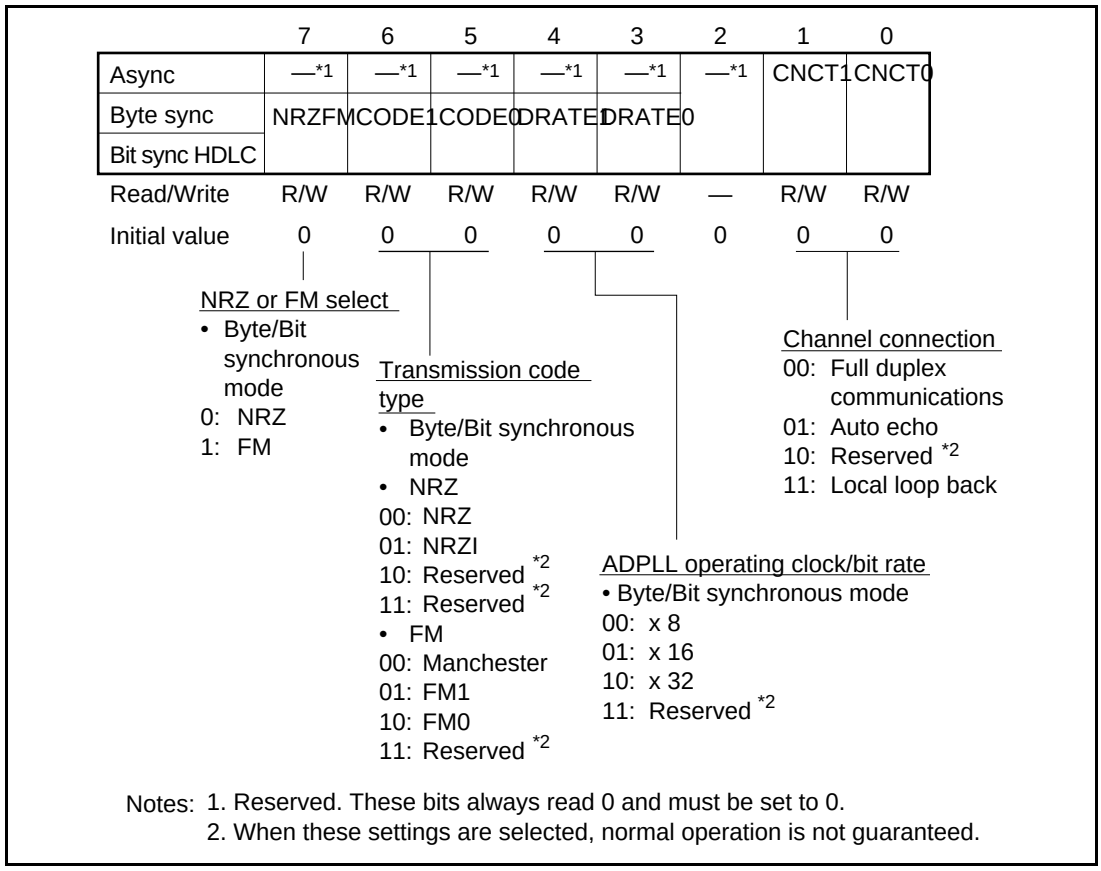
  Reserved. These bits always read 0 and must be set to 0.

5.2.3 MSCI Mode Register 2 (MD2)

Mode register 2 (MD2) specifies the transmission/reception data code type, the ratio of the advanced digital phase locked loop (ADPLL) operating clock to the bit rate, and the connection between the transmit/receive data and the TXD/RXD lines. For the ADPLL, see section 5.5, ADPLL.

This register is reset under either of the following conditions:

- Hardware reset
- Channel reset command



Notes: 1. Reserved. These bits always read 0 and must be set to 0.
2. When these settings are selected, normal operation is not guaranteed.

Bit 7 (NRZFM: NRZ/FM Select): Used in conjunction with the CODE1–CODE0 bits (below) to specify the transmission code type (NRZ or FM). This bit specifies MSCI decoding and encoding types. Only the NRZ type is available in asynchronous mode.

- Asynchronous mode
  Reserved. This bit always reads 0 and must be set to 0.
- Byte synchronous/Bit synchronous mode
  NRZFM = 0: Specifies NRZ transmission code type
  NRZFM = 1: Specifies FM transmission code type

Bits 6–5 (CODE1–CODE0: Transmission Code Type): Used in conjunction with the NRZFM bit (above) to specify the transmission/reception signal decoding and encoding type. Only the NRZ type is available in asynchronous mode.

- Asynchronous mode
  Reserved. These bits always read 0 and must be set to 0.
- Byte synchronous/Bit synchronous mode
  NRZFM = 0: NRZ transmission code type
  — CODE1, CODE0 = 0, 0: Specifies the NRZ transmission code type
  — CODE1, CODE0 = 0, 1: Specifies the NRZI transmission code type
  — CODE1, CODE0 = 1, 0: Reserved
  — CODE1, CODE0 = 1, 1: Reserved
  NRZFM = 1: FM transmission code type
  — CODE1, CODE0 = 0, 0: Specifies the Manchester transmission code type
  — CODE1, CODE0 = 0, 1: Specifies the FM1 transmission code type
  — CODE1, CODE0 = 1, 0: Specifies the FM0 transmission code type
  — CODE1, CODE0 = 1, 1: Reserved

Bits 4–3 (DRATE1–DRATE0: ADPLL Operating Clock/Bit Rate): Specify the ratio of the ADPLL operating clock frequency to the bit rate in byte or bit synchronous mode.

- Asynchronous mode
  Reserved. These bits always read 0 and must be set to 0.
- Byte synchronous/Bit synchronous mode
  DRATE1, DRATE0 = 0, 0: ADPLL operating clock frequency = bit rate × 8
  DRATE1, DRATE0 = 0, 1: ADPLL operating clock frequency = bit rate × 16
  DRATE1, DRATE0 = 1, 0: ADPLL operating clock frequency = bit rate × 32
  DRATE1, DRATE0 = 1, 1: Reserved

Bits 1–0 (CNCT1–CNCT0: Channel Connection): The functions of these bits are described below.

CNCT1, CNCT0 = 0, 0: Specifies the full duplex communications mode (normal operation)

CNCT1, CNCT0 = 0, 1: Specifies the auto-echo mode. In this mode, input data via the RXD line is directly output to the TXD line. (If specified, the data is first processed by the ADPLL to suppress noise and extract the clock signal, and the resulting data is output on the TXD line.) This mode enables data reception, but disables data transmission. If the MSCI TX clock source register (TXS) designates TXC as an output line, the receive clock selected by the MSCI RX clock source register (RXS) is output on the TXC line. If TXC is designated as an input line, the TXC input does not affect operations.

CNCT1, CNCT0 = 1, 0: Reserved.

CNCT1, CNCT0 = 1, 1: Specifies the local loop-back mode. In this mode, the transmit shift register output is internally connected to the receive shift register input for the receive shift register to directly receive the transmit data without passing through the ADPLL. Here, the receive clock selected by the MSCI RX clock source register (RXS) is used as both the transmit clock and receive clock.

Independently of the above operation, data input on the RXD line is output again on the TXD line. (If specified, the data is first processed by the ADPLL to suppress noise and extract the clock signal, and the resulting data is output on the TXD line.)

In addition, if the TXS register designates TXC as an output line, the receive clock selected by the RXS register is output on the TXC line. If TXC is designated as an input line, the TXC input does not affect operations.

Note that if the ADPLL output is selected as the receive clock, the clock signal extracted from the RXD input is supplied to both the transmitter and receiver.

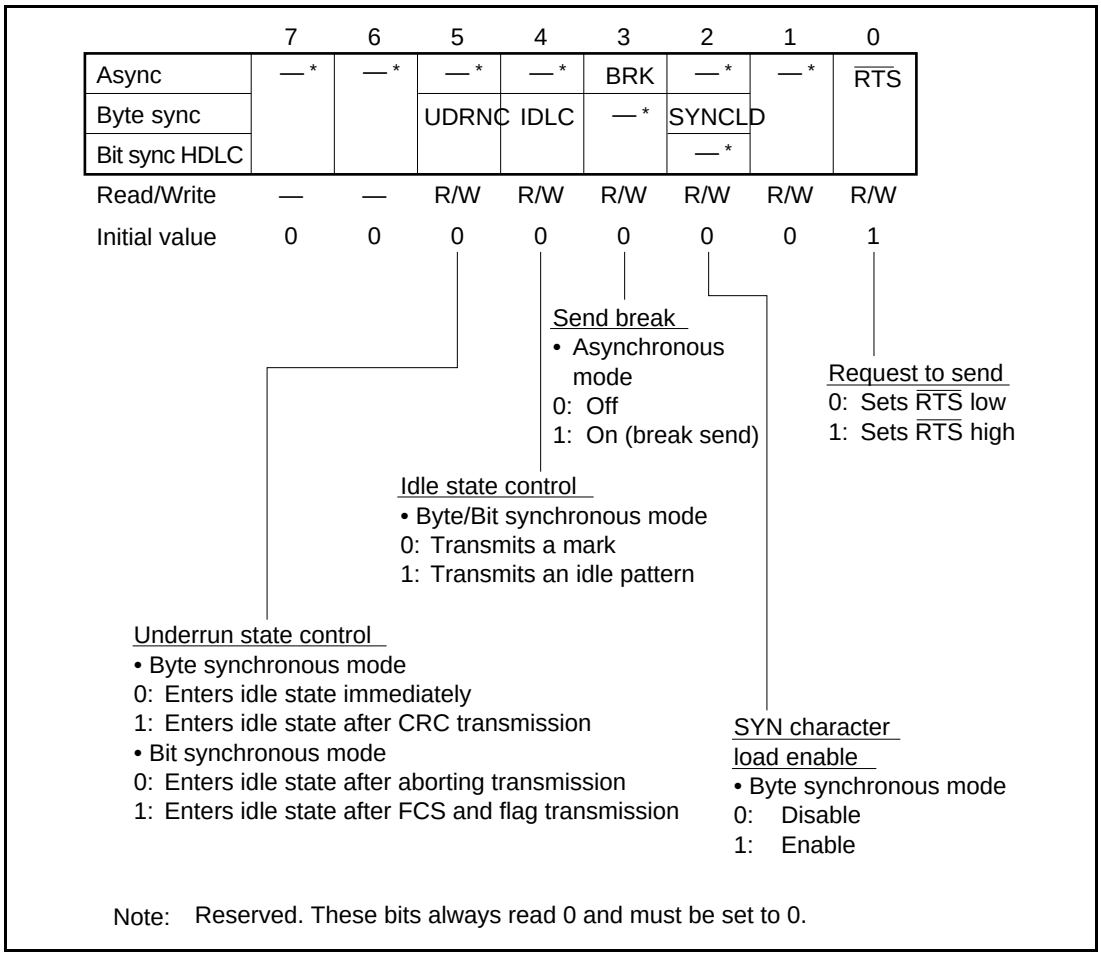
5.2.4 MSCI Control Register (CTL)

The control register (CTL) specifies the transmit operation in underrun state, an output pattern for idle state in byte or bit synchronous mode, break send in asynchronous mode, SYN character transfer from the data field to the receive buffer, and the $\overline{\text{RTS}}$ line output level.

This register is reset under either of the following conditions:

- Hardware reset
- Channel reset command

The BRK bit (bit 3) is also cleared by a TX reset command.



Note: Reserved. These bits always read 0 and must be set to 0.

Bits 7–6: Reserved. These bits always read 0 and must be set to 0.

Bit 5 (UDRNC: Underrun State Control): Specifies the transmit operation in underrun state in byte or bit synchronous mode.

- Asynchronous mode
  Reserved. This bit always reads 0 and must be set to 0.
- Byte synchronous mode
  UDRNC = 0: Sets the transmitter to idle state in underrun state
  UDRNC = 1: Sets the transmitter to idle state after CRC code transmission in underrun state
- Bit synchronous mode
  UDRNC = 0: Sets the transmitter to idle state after aborting transmission in underrun state
  UDRNC = 1: Sets the transmitter to idle state after FCS (CRC code) and flag transmission in underrun state

If the UDRNC bit is set to 0 so as to set the transmitter to the idle state after transmitting an abort frame, a zero may not be inserted immediately before the abort frame. The receiver should thus discard the data immediately preceding the abort frame.

Bit 4 (IDLC: Idle State Control): Specifies the TXD line output in idle state in byte or bit synchronous mode.

- Asynchronous mode
  Reserved. This bit always reads 0 and must be set to 0.
- Byte synchronous/Bit synchronous mode
  IDLC = 0: Sets the TXD line high (mark) in idle state
  IDLC = 1: Repeatedly transmits 8-bit idle patterns in the idle pattern register (IDL) in idle state

Bit 3 (BRK: Send Break): Specifies whether or not to transmit a break in asynchronous mode.

- Asynchronous mode
  BRK = 0: Transmits no break (normal operation).
  BRK = 1: Transmits a break by setting the TXD line low (space) at the next falling edge of the transmit clock. The TXD line must be low for two or more character cycles to transmit a break.
  The BRK bit is cleared by a TX reset command.
  For details on breaks, see Break Transmission and Detection, in section 5.3.1, Asynchronous Mode.
- Byte synchronous/Bit synchronous mode
  Reserved. This bit always reads 0 and must be set to 0.

Bit 2 (SYNCLD: SYN Character Load Enable): Specifies whether or not to transfer the SYN character specified by synchronous/address register 0 (SA0) in the data field to the receive buffer in byte synchronous mode. See section 5.3.2, Byte Synchronous Mode.

- Asynchronous mode
  Reserved. This bit always reads 0 and must be set to 0.
- Byte synchronous mode
  SYNCLD = 0: Deletes SYN character in the data field instead of transferring it to the receive buffer
  SYNCLD = 1: Transfers the SYN character in the data field to the receive buffer
- Bit synchronous mode
  Reserved. This bit always reads 0 and must be set to 0.

Bit 1: Reserved. This bit must always be set to 0. If set to 1, the SCA operation is not guaranteed.

Bit 0 (RTS: Request to Send): Specifies the $\overline{\text{RTS}}$ line output level.

- Asynchronous/Byte synchronous/Bit synchronous mode
  $\overline{\text{RTS}}$ = 0: Sets the $\overline{\text{RTS}}$ line level low
  $\overline{\text{RTS}}$ = 1: Sets the $\overline{\text{RTS}}$ line level high
  In asynchronous mode, when the auto-enable mode is selected, (the AUTO bit of mode register 0 (MD0) is 1) the $\overline{\text{RTS}}$ line is driven low by transmit operation, regardless of the $\overline{\text{RTS}}$ bit setting.

5.2.5 MSCI RX Clock Source Register (RXS)

The RX clock source register (RXS) specifies the receive clock source and the baud rate of the baud rate generator (BRG) in the receiver. For the baud rate generator, see section 5.6, Baud Rate Generator.

This register is reset under either of the following conditions:

- Hardware reset
- Channel reset command

	7	6	5	4	3	2	1	0
Async Byte sync Bit sync HDLC	—*1	RXCS2	RXCS1	RXCS0	RXBR3	RXBR2	RXBR1	RXBR0
Read/Write	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

Receive clock source
000: RXC line input
010: RXC line input (noise suppression)
100: Internal baud rate generator (BRG) output
110: ADPLL output
(BRG output for ADPLL operating clock)
111: ADPLL output
(RXC line input for ADPLL operating clock)
Others: Reserved *2

Receiver baud rate
• Clock division ratio
0000: 1/1
0001: 1/2
0010: 1/4
0011: 1/8
0100: 1/16
0101: 1/32
0110: 1/64
0111: 1/128
1000: 1/256
1001: 1/512
Others: Reserved *2

Notes: 1. Reserved. This bit always reads 0 and must be set to 0.
2. Reserved. When these settings are selected, normal operation is not guaranteed.

Bit 7: Reserved. This bit always reads 0 and must be set to 0.

Bits 6–4 (RXCS2–RXCS0: Receive Clock Source): Specify the receive clock source.

- Asynchronous/Byte synchronous/Bit synchronous mode

  RXCS2–RXCS0 = 0, 0, 0: Specifies the RXC line input as the receive clock. The noise suppressor does not function for the receive clock and receive data.

  RXCS2–RXCS0 = 0, 1, 0: Specifies the RXC line input as the receive clock. The noise suppressor of the ADPLL functions for the receive clock and receive data.

  RXCS2–RXCS0 = 1, 0, 0: Specifies the internal BRG output as the receive clock, which is output from the RXC line.

  RXCS2–RXCS0 = 1, 1, 0: Specifies the clock extracted by the ADPLL as the receive clock, which is output from the RXC line. The BRG output serves as the ADPLL operating clock, and the noise suppressor functions for the receive data.

  RXCS2–RXCS0 = 1, 1, 1: Specifies the clock extracted by the ADPLL as the receive clock. The RXC line input serves as the ADPLL operating clock, and the noise suppressor functions for the receive data.

  Other settings: Reserved.

Bits 3–0 (RXBR3–RXBR0: Receiver Baud Rate): Used in conjunction with the time constant register (TMC) to specify the baud rate of the baud rate generator in the receiver. For details, see section 5.6, Baud Rate Generator.

- Asynchronous/Byte synchronous/Bit synchronous mode

  RXBR3–RXBR0 = 0, 0, 0, 0: Division ratio = 1/1

  RXBR3–RXBR0 = 0, 0, 0, 1: Division ratio = 1/2

  RXBR3–RXBR0 = 0, 0, 1, 0: Division ratio = 1/4

  RXBR3–RXBR0 = 0, 0, 1, 1: Division ratio = 1/8

  RXBR3–RXBR0 = 0, 1, 0, 0: Division ratio = 1/16

  RXBR3–RXBR0 = 0, 1, 0, 1: Division ratio = 1/32

  RXBR3–RXBR0 = 0, 1, 1, 0: Division ratio = 1/64

  RXBR3–RXBR0 = 0, 1, 1, 1: Division ratio = 1/128

  RXBR3–RXBR0 = 1, 0, 0, 0: Division ratio = 1/256

  RXBR3–RXBR0 = 1, 0, 0, 1: Division ratio = 1/512

  RXBR3–RXBR0 = 1, 0, 1, 0: Reserved

  → →

  RXBR3–RXBR0 = 1, 1, 1, 1: Reserved

5.2.6 MSCI TX Clock Source Register (TXS)

The TX clock source register (TXS) specifies the transmit clock source and the baud rate of the baud rate generator (BRG) in the transmitter. For details on the baud rate generator, see section 5.6, Baud Rate Generator.

This register is reset under either of the following conditions:

- Hardware reset
- Channel reset command

	7	6	5	4	3	2	1	0
Async Byte sync Bit sync HDLC	—*1	TXCS2	TXCS1	TXCS0	TXBR3	TXBR2	TXBR1	TXBR0
Read/Write	—	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

Transmit clock source
000: TXC line input
100: Internal baud rate generator (BRG) output
110: Receive clock
Others: Reserved*2

Transmitter baud rate
• Clock division ratio
0000: 1/1
0001: 1/2
0010: 1/4
0011: 1/8
0100: 1/16
0101: 1/32
0110: 1/64
0111: 1/128
1000: 1/256
1001: 1/512
Others: Reserved*2

Notes: 1. Reserved. This bit always reads 0 and must be set to 0.
2. Reserved. When these settings are selected, normal operation is not guaranteed.

Bit 7: Reserved. This bit always reads 0 and must be set to 0.

Bits 6–4 (TXCS2–TXCS0: Transmit Clock Source): Specify the transmit clock source.

- Asynchronous/Byte synchronous/Bit synchronous mode

  TXCS2–TXCS0 = 0, 0, 0: Specifies the TXC line input as the transmit clock.

  TXCS2–TXCS0 = 1, 0, 0: Specifies the internal BRG output as the transmit clock, which is output from the TXC line.

  TXCS2–TXCS0 = 1, 1, 0: Specifies the receive clock as the transmit clock. This setting is used for using the clock extracted by the ADPLL as the transmit clock.

  Other settings: Reserved.

Bits 3–0 (TXBR3–TXBR0: Transmitter Baud Rate): Used in conjunction with the time constant register (TMC) to specify the baud rate of the baud rate generator in the transmitter. For details, see section 5.6, Baud Rate Generator.

- Asynchronous /Byte synchronous/Bit synchronous mode

  TXBR3–TXBR0 = 0, 0, 0, 0: Division ratio = 1/1

  TXBR3–TXBR0 = 0, 0, 0, 1: Division ratio = 1/2

  TXBR3–TXBR0 = 0, 0, 1, 0: Division ratio = 1/4

  TXBR3–TXBR0 = 0, 0, 1, 1: Division ratio = 1/8

  TXBR3–TXBR0 = 0, 1, 0, 0: Division ratio = 1/16

  TXBR3–TXBR0 = 0, 1, 0, 1: Division ratio = 1/32

  TXBR3–TXBR0 = 0, 1, 1, 0: Division ratio = 1/64

  TXBR3–TXBR0 = 0, 1, 1, 1: Division ratio = 1/128

  TXBR3–TXBR0 = 1, 0, 0, 0: Division ratio = 1/256

  TXBR3–TXBR0 = 1, 0, 0, 1: Division ratio = 1/512

  TXBR3–TXBR0 = 1, 0, 1, 0: Reserved

  → →

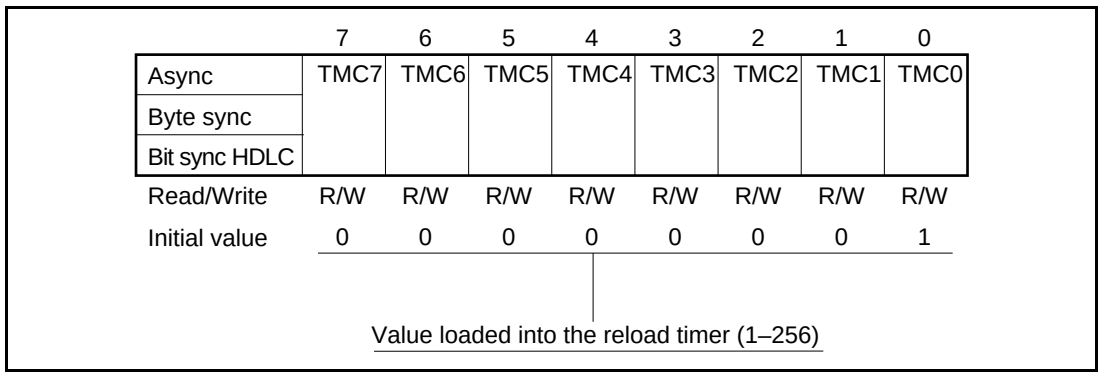
  TXBR3–TXBR0 = 1, 1, 1, 1: Reserved

5.2.7 MSCI Time Constant Register (TMC)

The time constant register (TMC) specifies the value (1–256) to be loaded to the reload timer in the internal baud rate generator (BRG). For details, see section 5.6, MSCI Baud Rate Generator.

This register is reset under either of the following conditions:

- Hardware reset
- Channel reset command



Bits 7–0 (TMC7–TMC0: Time Constant): The functions of these bits are described below.

- Asynchronous/Byte synchronous/Bit synchronous mode

  Bits 7–0 specify the value (1–256) to be loaded into the reload timer in the internal baud rate generator. (Zero is assumed to be 256.) These bits are used in conjunction with the TXBR3–TXBR0 bits of the TX clock source register (TXS) and the RXBR3–RXBR0 bits of the RX clock source register (RXS) to determine the BRG output frequency for transmission and reception.

The BRG output frequency can be calculated by the following expression:

$$f_{BRG} = \frac{f_{CLK}}{TMC} \div 2^{BR}$$

f_{BRG}: Transmit (receive) BRG output frequency
f_{CLK}: System clock frequency
TMC: Value (1–256) set in TMC
BR: Value (0–9) set in the TXBR3–TXBR0 bits of TXS, or the RXBR3–RXBR0 bits of RXS

Specific values are listed in table 5.21, Register Values and Bit Rates in Asynchronous Mode, and table 5.22, Register Values and Bit Rates in Byte Synchronous/Bit Synchronous Mode.

5.2.8 MSCI Command Register (CMD)

The command register (CMD) specifies the command for MSCI transmission/reception control. This is a write-only register and always reads 00H.

	7	6	5	4	3	2	1	0
Async Byte sync Bit sync HDLC	—*1	—*1	CMD5	CMD4	CMD3	CMD2	CMD1	CMD0
Read/Write	—	—	W	W	W	W	W	W
Initial value	—	—	—	—	—	—	—	—

Command

• Transmit commands
000001: TX reset
000010: TX enable
000011: TX disable
000100: TX CRC initialization
000101: TX CRC calculation exclusion
000110: End-of-message
000111: Abort transmission
001000: MP bit on
001001: TX buffer clear
Others: Reserved *2

• Receive commands
010001: RX reset
010010: RX enable
010011: RX disable
010100: RX CRC initialization
010101: Message reject
010110: Search MP bit
010111: RX CRC calculation exclusion
011000: Forcing RX CRC calculation

• Other commands
100001: Channel reset
110001: Enter search mode
000000: No operation

Notes: 1. Reserved. These bits always read 0 and must be set to 0.
2. When this setting is selected, normal operation is not guaranteed.

Bits 7–6: Reserved. These bits always read 0 and must be set to 0.

Bits 5–0 (CMD5–CMD0: Command): The functions of these bits are described below.

- Asynchronous/Byte synchronous/Bit synchronous mode
  Bits 5–0 specify three types of commands: transmit, receive, and other commands. These commands and their values are described in table 5.1 to table 5.3.

Table 5.1 Transmit Commands

Command Name (Set Value)	Function
TX reset (01H)	Immediately sets the transmitter to TX disable state (the transmit line goes to mark): clears the transmit buffer, transmit status in status registers 3–0 (ST3–ST0), and the BRK bit of the control register (CTL). No other register is affected.
TX enable (02H)	Sets the transmitter to idle state when the transmitter is in TX disable state. For auto-enable operation, see the description of the AUTO bit in section 5.2.1, MSCI Mode Register 0 (MD0).
TX disable (03H)	Immediately sets the TXRDY bit of MSCI status register 0 to stop the CPU or DMAC writing data into the transmit buffer. The transmitter enters the TX disable state after sending the transmit buffer data. In byte or bit synchronous mode, after sending the transmit buffer data, the transmitter first enters the underrun state, then enters the idle state according to the sequence specified by the UDRNC bit of the MSCI control register and the CRCCC bit of MSCI mode register 0. Finally, the transmitter enters the TX disable state after a one-bit idle state.
TX CRC initialization (04H)	Initializes the transmitter CRC calculator as specified by the CRC0 bit of MD0 when the first transmit character is transferred to the transmit shift register after this command is issued. This command is used in byte or bit synchronous mode.
TX CRC calculation exclusion (05H)	Excludes one specific character from the transmit CRC calculation. This command is valid only for the first character transferred to the transmit shift register after this command is issued. To exclude the first character, issue the command during transmission of the SYN character preceding the character to be excluded. (If SYN character transmission timing is not explicit, write a SYN character to the TX/RX buffer register (TRB) before the first character, and then issue the command during the SYN character transmission.) Command operation is not guaranteed in modes other than byte synchronous mode.

Table 5.1 Transmit Commands (cont)

Command Name (Set Value)	Function
End-of-message (06H)	Specifies the transmit character, which is first transferred to the transmit buffer after this command is issued, as the last character of the frame. When the transmit character transmitted is a word, this command specifies the transmit character written to D15–D8 as the last character of the frame in CPU mode 0, and also specifies the transmit character written to D7–D0 as such in CPU modes 2 and 3. Allows the transmitter to send the character marked with the end-of-message attribute, transmit the CRC code in byte synchronous mode, or sequentially transmit the FCS (CRC code) and flag in bit synchronous mode.
Abort transmission (07H)	Immediately transmits an 8-bit abort pattern 11111111 and clears the transmit buffer. This command is used in bit synchronous mode.
MP bit on (08H)	Sets the transmit data MP bit to 1, and then transmits a character. This command is valid only for the first character transferred to the transmit buffer after this command is issued. For details, see Multiprocessor Support, in section 5.3.1. This command operation is not guaranteed in modes other than asynchronous mode.
TX buffer clear (09H)	Clears the transmit buffer, deleting buffer contents. No other register is affected.

Table 5.2 Receive Commands

Command Name (Set Value)	Function
RX reset (11H)	Halts the receive shift register and sets the receiver to RX disable state; clears the receive buffer and receive status in status registers 3–0 (ST3–ST0). No other register is affected.

Table 5.2 Receive Commands (cont)

Command Name (Set Value)	Function
RX enable (12H)	Sets the receiver to start bit search state in asynchronous mode, SYN1 wait state in byte synchronous mode, and flag wait state in bit synchronous mode. When the receiver is in enable state, this command is invalid. For auto-enable operation , see the description of the AUTO bit in section 5.2.1, MSCI Mode Register 0 (MD0).
RX disable (13H)	Halts the receive shift register and sets the receiver to RX disable state while deleting the receive shift register contents, but without affecting the receive buffer contents.
RX CRC initialization (14H)	Initializes the receiver CRC calculator, as specified by the CRC0 bit of MD0, when the first receive character is transferred to the receive shift register after this command is issued. This command is used in byte or bit synchronous mode
Message reject (15H)	Allows the receiver to re-establish character synchronization in byte synchronous mode. Prevents transfer of the current data frame to the receive buffer in bit synchronous mode. Data transfer to the receive buffer resumes in the next frame. In bit synchronous mode, the message reject command must be issued only during character reception, or must be immediately followed by a receive buffer clear command. Otherwise, the frame currently being received may not be received correctly.
Search MP bit (16H)	Prevents receive characters with MP bit = 0 from being loaded into the receive buffer. This command remains valid until a character with MP bit = 1 is received. If necessary, re-issue this command after receiving a character with MP bit = 1. For details, see Multiprocessor Support, in section 5.3.1. This command is valid only in asynchronous mode.
RX CRC calculation exclusion (17H)	Excludes one specific character from the receiver CRC calculation. This command must be issued within 8 bit cycles after the character that will be excluded from the CRC calculation is input to the receive buffer. This command operation is not guaranteed in modes other than byte synchronous mode.

Table 5.2 Receive Commands (cont)

Command Name (Set Value)	Function
Forcing RX CRC calculation (18H)	Forcibly starts CRC calculation of the 8-bit data in the RX delay register. In byte synchronous mode, this command must be issued after the second byte of the CRC code enters the receive buffer. This allows CRC calculation to be completed even when the receive clock is halted after CRC code reception. CRC error status is activated 15 system clock cycles after this command is issued and remains valid until the next data enters the receive buffer.

Table 5.3 Other Commands

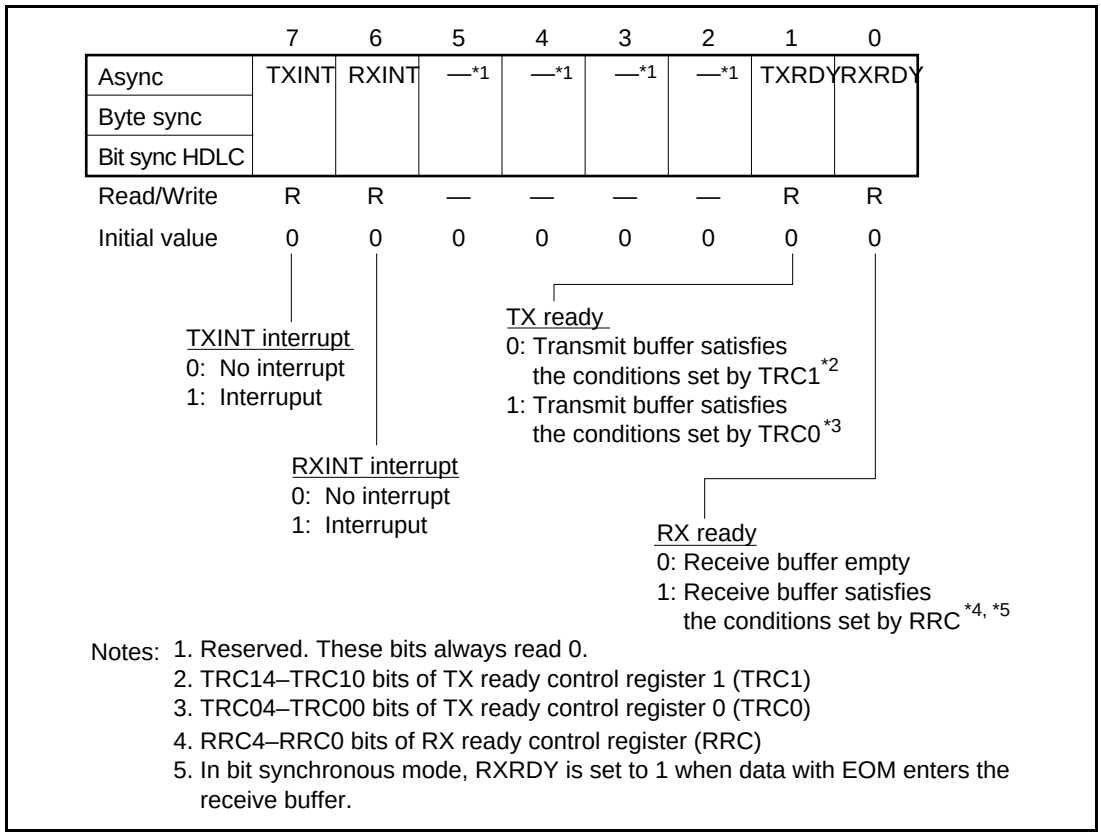
Command Name (Set Value)	Function
Channel reset (21H)	Initializes all MSCI registers, sets the transmitter and receiver to disable state, and clears the transmit/receive buffer.
Enter search mode (31H)	Sets the ADPLL to search mode. For FM code transmission, synchronization between the extracted receive clock and receive data can be established by a single transition point. This command also sets the SRCH bit of status register 3 (ST3) to 1. For details, see section 5.5, ADPLL.
No operation (00H)	Allows the transmitter and receiver to continue the current operation.

5.2.9 MSCI Status Register 0 (ST0)

Status register 0 (ST0) indicates the status of interrupts (TXINT and RXINT) and the transmit/receive buffer. When any bit of this register is set to 1, an MPU interrupt request is generated (if enabled).

This register is reset under either of the following conditions:

- Hardware reset
- Channel reset command
- System stop mode



Notes: 1. Reserved. These bits always read 0.
2. TRC14–TRC10 bits of TX ready control register 1 (TRC1)
3. TRC04–TRC00 bits of TX ready control register 0 (TRC0)
4. RRC4–RRC0 bits of RX ready control register (RRC)
5. In bit synchronous mode, RXRDY is set to 1 when data with EOM enters the receive buffer.

Bit 7 (TXINT: TXINT Interrupt): Indicates whether or not the TXINT interrupt has occurred. A TXINT interrupt request is issued to the MPU when this bit and the TXINTE bit of interrupt enable register 0 (IE0) are both 1.

- Asynchronous/Byte synchronous/Bit synchronous mode.

  TXINT = 0: Indicates that no TXINT interrupt has occurred.

  TXINT = 1: Indicates that a TXINT interrupt has occurred.

  This bit is set to 1 under the following conditions:

  TXINT = UDRN • UDRNE + IDL • IDLE + CCTS • CCTSE

  UDRN, IDL, CCTS: Bits 7, 6, and 3 of status register 1 (ST1)

  UDRNE, IDLE, CCTSE: Bits 7, 6, and 3 of interrupt enable register 1 (IE1)

  In other words, this bit is set to 1 under either of the following conditions:

  - The UDRNE bit is set to 1 and an underrun error has occurred.
  - The IDLE bit is set to 1 in idle state.
  - The CCTSE bit is set to 1 and the $\overline{\text{CTS}}$ line level has changed.

Bit 6 (RXINT: RXINT Interrupt): Indicates whether or not the RXINT interrupt has occurred. An RXINT interrupt request is issued to the MPU when this bit and the RXINTE bit of IE0 are both 1.

- Asynchronous/Byte synchronous/Bit synchronous mode

  RXINT = 0: Indicates that no RXINT interrupt has occurred.

  RXINT = 1: Indicates that an RXINT interrupt has occurred.

  This bit is set to 1 under the following conditions:

  RXINT = CLMD • CLMDE + (SYNCD/FLGD) • (SYNCDE/FLGDE) + CDCD • CDCDE + (BRKD/ABTD/GAPD) • (BRKDE/ABTDE/GAPDE) + (BRKE/IDLD) • (BRKEE/IDLDE) + EOM • EOME + (PMP/SHRT) • (PMPE/SHRTE) + (PE/ABT) • (PEE/ABTE) + (FRME/RBIT) • (FRMEE/RBITE) + OVRN • OVRNE + CRCE • CRCEE + EOMF • EOMFE

  CLMD, SYNCD/FLGD, CDCD, BRKD/ABTD/GAPD, BRKE/IDLD:

  Bits 5, 4, 2, 1, and 0 of status register 1 (ST1)

  EOM, PMP/SHRT, PE/ABT, FRME/RBIT, OVRN, CRCE:

  Bit 7–bit 2 of status register 2 (ST2)

  EOMF: Bit 7 of the frame status register (FST)

  CLMDE, SYNCDE/FLGDE, CDCDE, BRKDE/ABTDE/GAPDE, BRKEE/IDLDE:

  Bits 5, 4, 2, 1, and 0 of interrupt enable register (IE1)

  EOME, PMPE/SHRTE, PEE/ABTE, FRMEE/RBITE, OVRNE, CRCEE:

  Bit 7–bit 2 of interrupt enable register 2 (IE2)

  EOMFE: Bit 7 of the frame interrupt enable register (FIE)

In other words, RXINT is set to 1 under one of the following conditions:

- The CLMDE bit is set to 1 and no RXD transition has been detected in the ADPLL window twice successively in FM mode.
- The SYNCDE/FLGDE bit is set to 1 and a SYN character or a flag has been detected.
- The CDCDE bit is set to 1 and the $\overline{\text{DCD}}$ line level has changed.
- The BRKDE/ABTDE/GAPDE bit is set to 1 and a break start, abort, or a GA pattern has been detected.
- The BRKEE/IDLDE bit is set to 1 and a break end or an idle start has been detected.
- The EOME bit is set to 1 and the receive frame has ended.
- The PMPE/SHRTE bit is set to 1, and the parity/MP bit is set to 1 or a short frame has been detected.
- The PEE/ABTE bit is set to 1 and a parity error or an abort frame has been detected.
- The FRMEE/RBITE bit is set to 1 and a framing error or a residual bit frame has been detected.
- The OVRNE bit is set to 1 and an overrun error has been detected.
- The CRCEE bit is set to 1 and a CRC error has been detected.
- The EOMFE bit is set to 1, the receive frame has ended, and the last character has been read from the receive buffer (TRB).

Bits 5–2: Reserved. These bits always read 0.

Bit 1 (TXRDY: TX Ready): Indicates the transmit buffer status. When the transmitter is enabled and the data byte count in the transmit buffer is equal to or less than the value set by TX ready control register 0 (TRC0), this bit is set to 1. When the transmitter is enabled and the data byte count in the transmit buffer is equal to or greater than the value set by TX ready control register 1 (TRC1) + 1, this bit is cleared. When the transmitter is disabled, this bit is cleared, regardless of the data byte count in the transmit buffer. This means that the transmit buffer can be written only while this bit is 1.

A TXRDY interrupt request is issued to the MPU when this bit and the TXRDYE bit of IE0 are both set to 1. A DMA request is issued to the on-chip DMAC when this bit is set to 1. For details, see section 5.8.1, Serial Data Transfer by the MPU and DMAC.

- Asynchronous/Byte synchronous/Bit synchronous mode

  TXRDY = 0: In TX enable state, indicates that the data byte count in the transmit buffer is equal to or greater than TXF1 + 1 (the value set by the TRC14–TRC10 bits of TRC1 + 1) or indicates that the data byte count in the transmit buffer is NOT equal to or less than TXF0 (the value set by the TRC04–TRC00 bits of TRC0) after the data byte count has temporarily been equal to or greater than TXF1 + 1. This bit is cleared in TX disable state or when an underrun error has occurred.

TXRDY = 1: In TX enable state, indicates that the data byte count in the transmit buffer is equal to or less than TXF0, or indicates that the data byte count in the transmit buffer is NOT equal to or greater than TXF1 + 1 after the data byte count has temporarily been equal to or less than TXF0.

The TXRDY bit has a hysteresis as shown in figure 5.2.

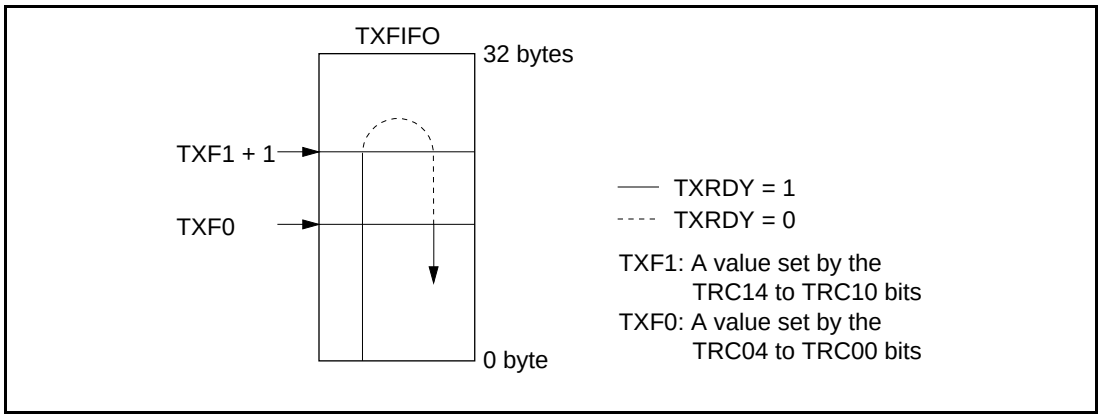


Figure 5.2 TXRDY Hysteresis

Writing a data word to the TX/RX buffer register (TRB) using the MPU with TXF1 set to 31 (1FH) eliminates the second byte of the data word. In this case, the TXRDY bit is asserted even when 31 bytes of data are in the TX FIFO.

This situation does not occur in DMA transfer mode.

Bit 0 (RXRDY: RX Ready): Indicates the receive buffer status. This bit is set to 1 when the data byte count in the receive buffer is equal to or greater than RXF + 1, (the value set by the RRC4–RRC0 bits of the RX control register (RRC) + 1) regardless of the RX enable or RX disable status. In bit synchronous mode, this bit is also set to 1 when data with EOM enters the receive buffer.

Once set to 1, this bit is not cleared until all the data has been read from the receive buffer.

An RXRDY interrupt request is issued to the MPU when this bit and the RXRDYE bit of IE0 are both set to 1. A DMA request is issued to the on-chip DMAC when this bit is set to 1. For details, see section 5.8.1, Serial Data Transfer by the MPU and DMAC.

- Asynchronous/Byte synchronous/Bit synchronous mode

  RXRDY = 0: Indicates that the receive buffer is empty, or that, after the receive buffer is empty, the data byte count in the receive buffer is NOT equal to or greater than RXF + 1. In bit synchronous mode, this bit also indicates that no EOM data is in the receive buffer.

RXRDY = 1: Indicates that the data byte count in the receive buffer is equal to or greater than RXF + 1, or that at least one byte of data still remains in the receive buffer after the data byte count in the receive buffer is temporarily equal to or greater than RXF + 1. In bit synchronous mode, this bit also indicates that one or more bytes of EOM data are in the receive buffer.

5.2.10 MSCI Status Register 1 (ST1)

Status register 1 (ST1) indicates status information such as break start/stop detection in asynchronous mode, underrun error, and SYN pattern detection in byte synchronous mode, underrun error, flag, abort, DPLL error, and idle start detection in bit synchronous mode, and also indicates transmitter idle status, and $\overline{CTS}$ and $\overline{DCD}$ input level changes in all modes.

The reset descriptions of this register's bits are as follows:

- Bits 7, 4, 3, 2, 1, and 0 are reset when 1 is written to the corresponding bit.
- Bits 7, 6, and 3 are reset by a TX reset command.
- Bit 6 is reset when data is written to the transmit buffer.
- Bits 4, 2, 1, and 0 are reset by an RX reset command.
- All bits are reset by a channel reset command or in system stop mode.

When any bit of this register is set to 1, an MPU interrupt request is generated (if enabled).

	7	6	5	4	3	2	1	0
Async	—*1	IDL	—*2	—*1	CCTS	CDCD	BRKD	BRKE
Byte sync	UDRN		CLMD	SYNCD			—*1	—*1
Bit sync HDLC				FLGD			ABTD	IDLD
Read/Write	R/W	R	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

Underrun error
- Byte/Bit synchronous mode

0: No underrun detected
1: Underrun detected

SYN pattern detection
- Byte synchronous mode

0: No pattern detected
1: Pattern detected

Flag detection
- Bit synchronous mode

0: No flag detected
1: Flag detected

Transmitter idle status
0: Not idle
1: Idle

Two-clock missing detection
- Byte/Bit synchronous mode

0: Two missing clock transitions not detected
1: Two missing clock transitions detected

$\overline{CTS}$ line level change
0: Not changed
1: Changed

$\overline{DCD}$ line level change
0: Not changed
1: Changed

Break end
- Asynchronous mode

0: Break sequence end not detected
1: Break sequence end detected

Idle start detection
- Bit synchronous mode

0: Idle sequence start not detected
1: Idle sequence start detected

Break detection
- Asynchronous mode

0: Break sequence starts not detected
1: Break sequence starts detected

Abort detection
- Bit synchronous mode

0: Abort sequence start not detected
1: Abort sequence start detected

Notes: 1. Reserved. These bits always read 0 and can be set to 0 or 1.
2. Reserved. When read, this bit is undefined and must be set to 0.

Bit 7 (UDRN: Underrun Error): Indicates whether or not an underrun error has occurred in byte or bit synchronous mode. (In asynchronous mode, underrun errors do not occur.) This bit is cleared when 1 is written to this bit position.

- Asynchronous mode
  Reserved. This bit always reads 0 and can be set to 0 or 1.
- Byte synchronous/Bit synchronous mode
  UDRN = 0: Indicates that no underrun error has occurred
  UDRN = 1: Indicates that an underrun error has occurred

Bit 6 (IDL: Transmitter Idle Status): Indicates whether or not the MSCI transmitter is in idle state. This bit is cleared when the transmit data is written to the transmit buffer.

- Asynchronous/Byte synchronous/Bit synchronous mode
  IDL = 0: Indicates that the transmitter is not in idle state
  IDL = 1: Indicates that the transmitter is in idle state

Bit 5 (CLMD: Two-Clock Missing Detection): Indicates the detection of two missing clock transitions in byte or bit synchronous mode when the FM codes and ADPLL are used. This occurs when no level transition on the RXD line is detected in the search window twice in a row (two clock cycles). If this bit is set to 1, the ADPLL automatically enters search mode.

- Asynchronous mode
  Reserved. The value of this bit is undefined when read, and must be set to 0.
- Byte synchronous/Bit synchronous mode
  Reserved. The value of this bit is undefined when read, and must be set to 0.
  CLMD = 0: Indicates that missing level transitions have not been detected
  CLMD = 1: Indicates that missing level transitions have been detected

Bit 4 (SYNCD/FLGD: SYN Pattern Detection/Flag Detection): Indicates whether or not synchronization has been established in byte or bit synchronous mode. This bit is cleared when 1 is written to this bit position.

- Asynchronous mode
  Reserved. This bit always reads 0 and can be set to 0 or 1.
- Byte synchronous mode
  SYNCD = 0: Indicates that synchronization has not been established
  SYNCD = 1: Indicates that synchronization has been established ($\overline{\text{SYN}}$ pattern detection in mono-sync or bi-sync mode, or by the $\overline{\text{SYNC}}$ line input in external synchronous mode)
- Bit synchronous mode
  FLGD = 0: Indicates that synchronization has not been established
  FLGD = 1: Indicates that synchronization has been established (flag pattern detection)

Bit 3 (CCTS: CTS Line Level Change): Indicates whether or not the $\overline{CTS}$ line level has changed. This bit is cleared when 1 is written to this bit position.

- Asynchronous/Byte synchronous/Bit synchronous mode
  CCTS = 0: Indicates that the $\overline{CTS}$ line level has not changed
  CCTS = 1: Indicates that the $\overline{CTS}$ line level has changed

Bit 2 (CDCD: DCD Line Level Change): Indicates whether or not the $\overline{DCD}$ line level has changed. This bit is cleared when 1 is written to this bit position.

- Asynchronous/Byte synchronous/Bit synchronous mode
  CDCD = 0: Indicates that the $\overline{DCD}$ line level has not changed
  CDCD = 1: Indicates that the $\overline{DCD}$ line level has changed

Bit 1 (BRKD/ABTD: Break Start Detection/Abort Detection): Indicates the detection of a break start (space state) in asynchronous mode or an abort in bit synchronous HDLC mode. This bit is cleared when 1 is written to this bit position.

- Asynchronous mode
  BRKD = 0: Indicates that no break start has been detected
  BRKD = 1: Indicates that a break start has been detected
- Byte synchronous mode
  Reserved. This bit always reads 0 and can be set to 0 or 1.
- Bit synchronous mode
  ABTD = 0: Indicates that no abort has been detected
  ABTD = 1: Indicates that an abort has been detected in bit synchronous HDLC mode

Bit 0 (BRKE/IDLD: Break End Detection/Idle Start Detection): Indicates the detection of a break end in asynchronous mode, or an idle start in bit synchronous mode. This bit is cleared when 1 is written to this bit position.

- Asynchronous mode
  BRKE = 0: Indicates that no break end has been detected
  BRKE = 1: Indicates that a break end has been detected
- Byte synchronous mode
  Reserved. This bit always reads 0 and can be set to 0 or 1.
- Bit synchronous mode
  IDLD = 0: Indicates that no idle start has been detected
  IDLD = 1: Indicates that an idle start has been detected

5.2.11 MSCI Status Register 2 (ST2)

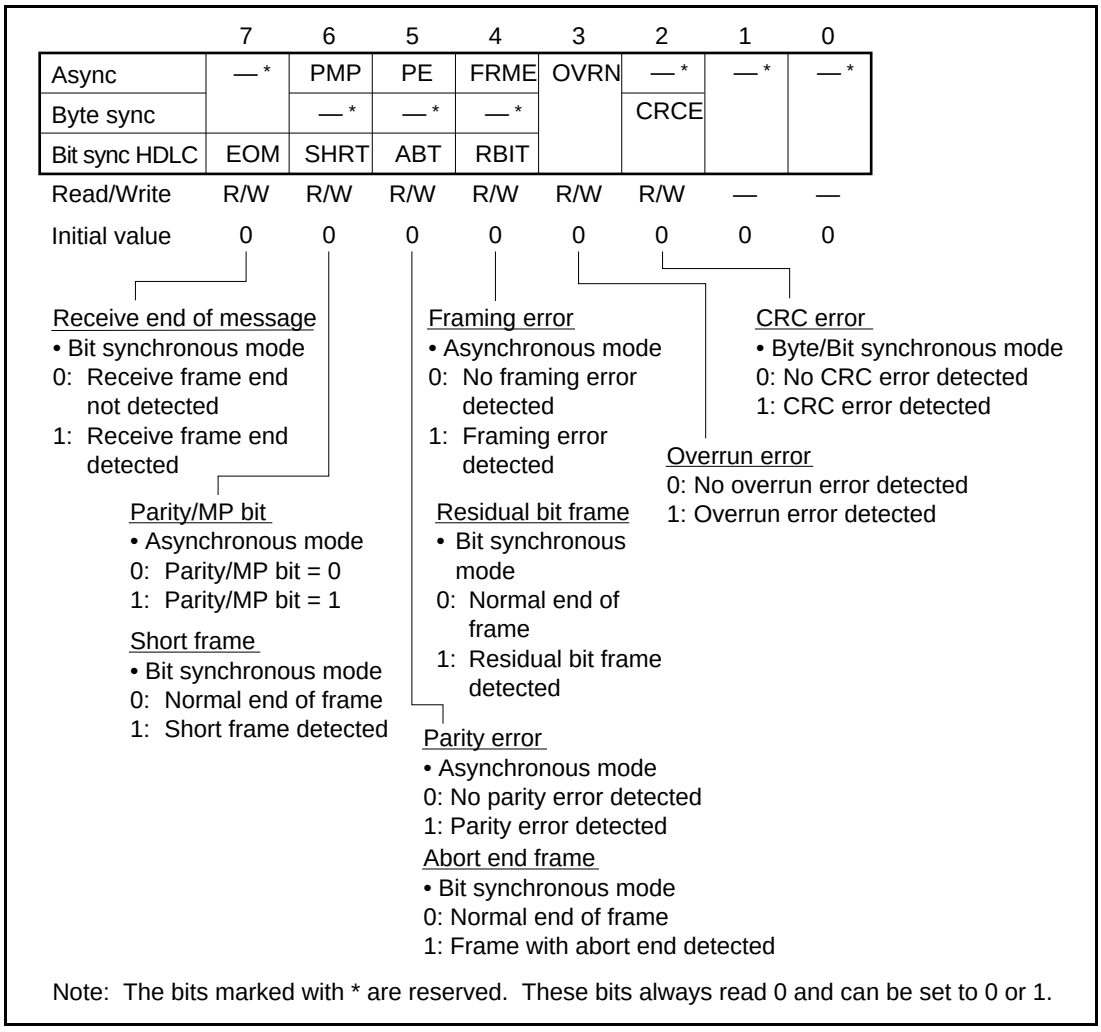
Status register 2 (ST2) indicates status information such as parity/MP bit value, parity error detection, and framing error detection in asynchronous mode, CRC error detection in byte synchronous mode, receive frame end, short frame, abort end frame, residual bit frame, and CRC error detection in bit synchronous mode, and also indicates overrun error detection in all modes.

This register is located at the top of the 32-stage status FIFO that corresponds to the receive buffer (figure 1.13). In CPU mode 1, this register is set by the top stage status of the status FIFO. In CPU modes 0, 2, and 3, this register is set by the OR of the second stage status and the top stage status of the status FIFO, except when the EOM bit of the top stage is 1. In this case, this register is set only by the top stage status. Once set to 1, no bit of this register is reset by a status FIFO change. For the CRCE bit clear conditions, see Bit 2 (CRCE: CRC Error) in this section. Note that the PMP bit is updated when the next receive character is ready to be read.

The reset descriptions of this register's bits are as follows:

- When 1 is written to a particular bit position, that bit is reset.
- All bits are reset by an RX or a channel reset command.
- All bits are reset in system stop mode.
- All bits are reset when data is transferred to the frame status register (FST) (See section 5.2.13, MSCI Frame Status Register (FST)).

When any bit of this register is set to 1, an MPU interrupt request is generated (if enabled).



Note: The bits marked with * are reserved. These bits always read 0 and can be set to 0 or 1.

Bit 7 (EOM: Receive End of Message): Indicates whether or not a receive frame has ended in bit synchronous mode. This bit is cleared when 1 is written to this bit position.

- Asynchronous/Byte synchronous mode

  Reserved. This bit always reads 0 and can be set to 0 or 1.
- Bit synchronous mode

  The EOM bit indicates whether or not a receive frame has ended. When the CRCCC bit of mode register 0 (MD0) is 1, the EOM bit is set to 1 by the last character in the I field of the receive frame. When the CRCCC bit of MD0 is 0, the EOM bit is set to 1 by the last character of FCS. Also, when the receive frame end status indicates either a short frame, residual bit frame, or abort, the EOM bit is set to 1.

EOM = 0: Indicates that the receive frame has not ended

EOM = 1: Indicates that the receive frame has ended

Bit 6 (PMP/SHRT: Parity/MP Bit/Short Frame): Indicates the parity/MP bit value in asynchronous mode, or short frame detection in bit synchronous mode. This bit is cleared when 1 is written to this bit position.

- Asynchronous mode

  The PMP bit indicates the status of the parity bit, MP bit, or receive character MSB according PMPM1–PMPM0: to the condition: indicates parity bit status when mode register 1's (MD1) PMPM1–PMPM0 bits are 10 or 11; indicates MP bit status when MD1's PMPM1–PMPM0 bits are 01; and indicates receive character MSB status when MD1's PMPM1–PMPM0 bits are 00.

  The PMP bit is updated when the next receive character is ready to be read.

  PMP = 0: Indicates that the parity bit, MP bit, or receive character MSB is 0

  PMP = 1: Indicates that the parity bit, MP bit, or receive character MSB is 1
- Byte synchronous mode

  Reserved. This bit always reads 0 and can be set to 0 or 1.
- Bit synchronous mode

  The SHRT bit indicates short frame detection. When the CRCCC bit of MD0 is 1, this bit is set to 1 by the last character in the I field if the receive frame is short and a part of the data is sent to the receive buffer. When the CRCCC bit is 0, this bit is set to 1 by the last character of FCS.

  However, even when the receive frame is short, this bit is not set to 1 if no data is sent to the receive buffer.

  When the SHRT bit is set to 1, the EOM bit is also set to 1.

  For details, see Short Frame Detection, in section 5.3.3, Bit Synchronous Mode.

  SHRT = 0: Indicates that no short frame has been detected

  SHRT = 1: Indicates that a short frame has been detected and that a part of the data has been sent to the receive buffer

Bit 5 (PE/ABT: Parity Error/Abort End Frame): Indicates detection of a parity error in asynchronous mode, or an abort end frame in bit synchronous mode.

This bit is cleared when 1 is written to this bit position.

- Asynchronous mode

  PE = 0: Indicates that no parity error has occurred

  PE = 1: Indicates that a parity error has occurred

  Once set to 1, this bit is not cleared until the receiver is reset or 1 is written to this bit position.
- Byte synchronous mode

Reserved. This bit always reads 0 and can be set to 0 or 1.

- Bit synchronous mode

  The ABT bit indicates whether or not an abort end frame has been detected.

  This bit is set to 1 by the character preceding the abort sequence when the receive frame ends with an abort. When this bit is set to 1, the EOM bit is also set to 1. See Abort End Frame Reception Operation below.

  ABT = 0: Indicates that no abort end frame has been detected

  ABT = 1: Indicates that an abort end frame has been detected

Bit 4 (FRME/RBIT: Framing Error/Residual Bit Frame): Indicates a framing error detection in asynchronous mode, and residual bit frame detection in bit synchronous mode. This bit is cleared when 1 is written to this bit position.

- Asynchronous mode

  FRME = 0: Indicates that no framing error has occurred

  FRME = 1: Indicates that a framing error has occurred

  Once set to 1, this bit is not cleared until the receiver is reset or 1 is written to this bit position.
- Byte synchronous mode

  Reserved. This bit always reads 0 and can be set to 0 or 1.
- Bit synchronous mode

  RBIT = 0: Indicates that no residual bit frame has been detected

  RBIT = 1: Indicates that a residual bit frame has been detected

  When the CRCCC bit of MD0 is 1, this bit is set to 1 by the residual bit of the last character in the receive frame I field. When the CRCCC bit is 0, the RBIT bit is set to 1 by the residual bit of the last character of FCS.

  When the RBIT bit is set to 1, the EOM bit is also set to 1. See Residual Bit Frame Reception Operation below.

Bit 3 (OVRN: Overrun Error): Indicates whether or not an overrun has occurred. This bit is cleared when 1 is written to this bit position. In asynchronous and byte synchronous modes, this bit is cleared when 1 is written to this bit position, or the receiver is reset. In bit synchronous mode, all bits of this register are also reset when the status data is loaded into the frame status register (FST).

- Asynchronous/Byte synchronous/Bit synchronous modes

  OVRN = 0: Indicates that no overrun error has occurred

  OVRN = 1: Indicates that an overrun error has occurred

Bit 2 (CRCE: CRC Error): Indicates whether or not a CRC error has occurred in byte or bit synchronous mode.

- Asynchronous mode

Reserved. This bit always reads 0 and can be set to 0 or 1.

- Byte synchronous/Bit synchronous mode

  The CRCE bit indicates whether or not a CRC error has occurred. When the CRCCC bit of MD0 is 1, this bit is set to 1 when a CRC error occurs. When the CRCCC bit is 0, this bit is not set to 1.

  This bit is cleared when 1 is written to this bit position or when the CRC calculation result is normal. This bit is the only bit of ST2 that changes status as the status FIFO changes status. For the timing of enabling this bit, see CRC errors, in Error Checking in sections 5.3.2, Byte Synchronous Mode and 5.3.3, Bit Synchronous Mode.

  CRCE = 0: Indicates that no CRC error has occurred

  CRCE = 1: Indicates that a CRC error has occurred

Bits 1–0: Reserved. These bits always read 0 and can be set to 0 or 1.

Residual bit frame reception operation: A residual bit frame reception operation is shown in figure 5.3. Residual bit frame data is transferred from the receive shift register to the receive buffer, and the residual bit frame status is set in the status FIFO.

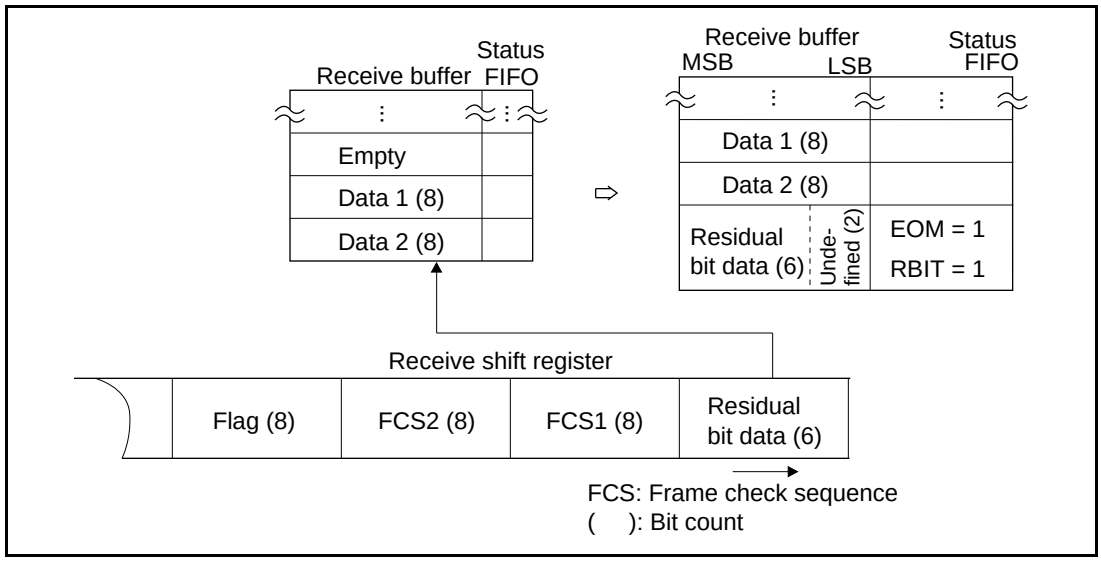


Figure 5.3 Residual Bit Frame Reception Operation (CRCCC = 1)

1. Residual bit data is transferred from the receive shift register to the receive buffer. At this time, the bits other than the residual data are undefined.

2. The EOM and RBIT bits of the status FIFO are set to 1.

3. If enabled, an interrupt request is generated when the residual bit data is ready to be read. However, in bit synchronous mode, the residual bit interrupt must be disabled because receive status is usually read from the frame status register (FST).

Abort end frame reception operation: An abort end frame reception operation is shown in figure 5.4. Abort end frame data is transferred from the receive shift register to the receive buffer, and the abort end frame status is set in the status FIFO.

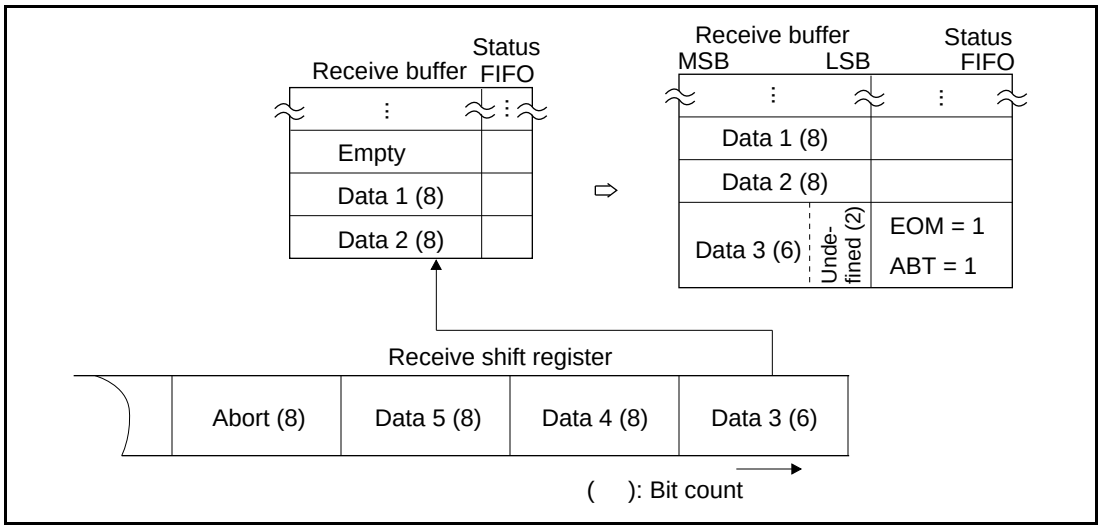


Figure 5.4 Abort End Frame Reception Operation (CRCCC = 1)

1. Part of the aborted data (data 3 in figure 5.4) is transferred from the receive shift register to the receive buffer. (Data 4 and 5 in figure 5.4 are not transferred to the receive buffer.) At this time, the bits other than this data are undefined. However, during abort frame reception in bit synchronous mode, if a zero is inserted in the character immediately before the abort frame, the preceding 17 bits of data will be discarded. As a result, the three bytes of data before the abort frame are not received correctly.

2. The EOM and ABT bits of the status FIFO are set to 1.

3. If enabled, an interrupt request is generated when the last data in the frame is ready to be read. However, in bit synchronous mode, the abort end frame interrupt must be disabled because receive status is usually read from the frame status register (FST).

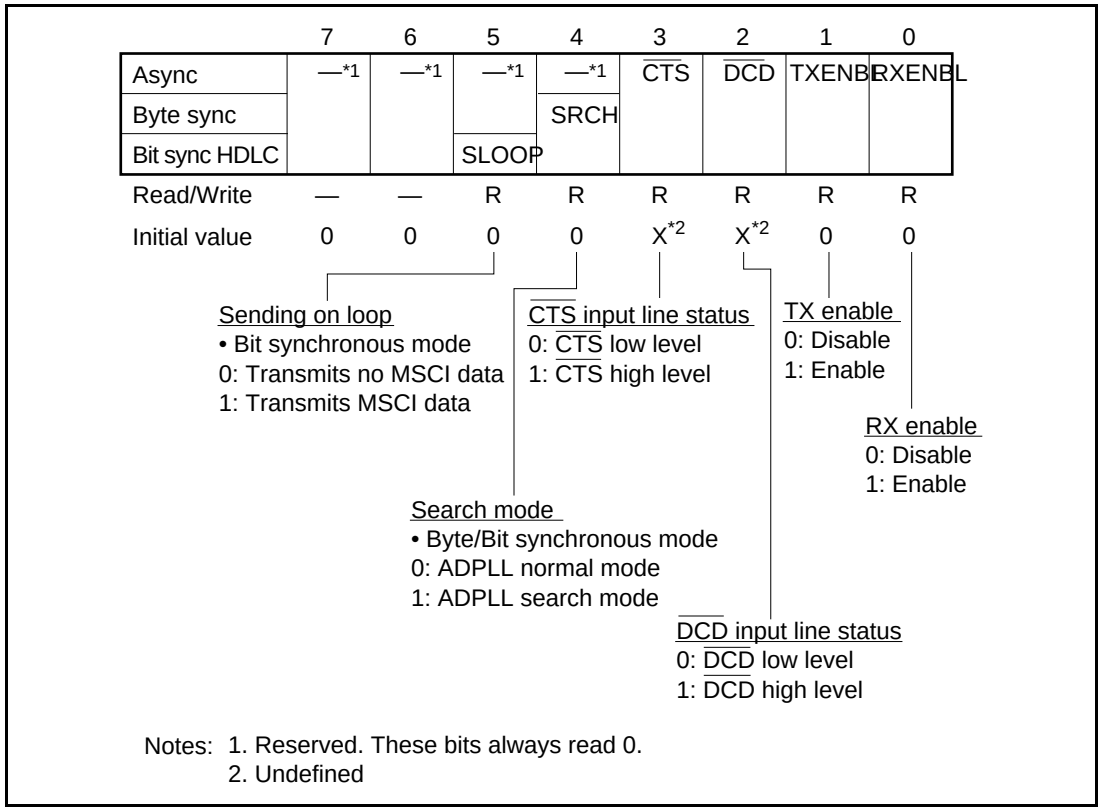
5.2.12 MSCI Status Register 3 (ST3)

Status register 3 (ST3) indicates data transmit status in bit synchronous mode, whether or not the ADPLL is in search mode in byte or bit synchronous mode, and also indicates the $\overline{CTS}$ and $\overline{DCD}$ line levels and the transmitter/receiver status (enable or disable). This is a read-only register.

The reset descriptions of this register's bits are as follows:

- Bits 5, 3, and 1 are reset by a TX reset command
- Bits 2 and 0 are reset by an RX reset command
- All bits are reset by a hardware reset or a channel reset command
- All bits are reset in system stop mode

No bit of this register generates an interrupt.



Notes: 1. Reserved. These bits always read 0.
2. Undefined

Bits 7–6: Reserved. These bits always read 0.

Bit 5 (SLOOP: Sending on Loop): Indicates MSCI data transmission status in bit synchronous mode. This bit is set to 1 when the MSCI is transmitting data, and is cleared otherwise. This is a read-only bit, and writing to it has no effect.

- Asynchronous/Byte synchronous mode
  Reserved. This bit always reads 0.
- Bit synchronous mode
  SLOOP = 0: Indicates that the MSCI is not transmitting data
  SLOOP = 1: Indicates that the MSCI is transmitting data

Bit 4 (SRCH: Search Mode): Indicates whether or not the ADPLL is in search mode. This bit is valid for transmission using FM codes in byte or bit synchronous mode. This is a read-only bit, and writing to it has no effect. This bit is set to 1 by an enter search command or automatic search mode initiated by two-clock missing detection.

- Asynchronous mode
  Reserved. This bit always reads 0.
- Byte synchronous/Bit synchronous mode
  SRCH = 0: Indicates that the ADPLL is not in search mode
  SRCH = 1: Indicates that the ADPLL is in search mode

Bit 3 (CTS: CTS Input Line Status): Indicates the $\overline{\text{CTS}}$ line level. This is a read-only bit, and writing to it has no effect.

- Asynchronous/Byte synchronous/Bit synchronous mode
  $\overline{\text{CTS}}$ = 0: Indicates that the $\overline{\text{CTS}}$ input line is low
  $\overline{\text{CTS}}$ = 1: Indicates that the $\overline{\text{CTS}}$ input line is high

Bit 2 (DCD: DCD Input Line Status): Indicates the $\overline{\text{DCD}}$ line level. This is a read-only bit, and writing to it has no effect.

- Asynchronous/Byte synchronous/Bit synchronous mode
  $\overline{\text{DCD}}$ = 0: Indicates that the $\overline{\text{DCD}}$ input line is low
  $\overline{\text{DCD}}$ = 1: Indicates that the $\overline{\text{DCD}}$ input line is high

Bit 1 (TXENBL: TX Enable): Indicates whether the transmitter is enabled or disabled. Transmitter enable/disable selection is performed by a command. This is a read-only bit, and writing to it has no effect.

- Asynchronous/Byte synchronous/Bit synchronous mode
  TXENBL = 0:Indicates that the transmitter is disabled

TXENBL = 1:Indicates that the transmitter is enabled

Bit 0 (RXENBL: RX Enable): Indicates whether the receiver is enabled or disabled. Receiver enable/disable selection is performed by a command. This is a read-only bit, and writing to it has no effect.

- Asynchronous/Byte synchronous/Bit synchronous mode

  RXENBL = 0: Indicates that the receiver is disabled

  RXENBL = 1: Indicates that the receiver is enabled

5.2.13 MSCI Frame Status Register (FST)

The frame status register (FST) (figure 5.16) holds the status of the last frame received in bit synchronous mode.

The reset descriptions of this register's bits are as follows:

- When 1 is written to a particular bit position, that bit is reset
- All bits are reset by an RX or channel reset command
- All bits are reset in system stop mode

When the EOMF bit is set to 1, an MPU interrupt request is generated (if enabled). The other bits do not generate interrupts.

	7	6	5	4	3	2	1	0
Async	— *	— *	— *	— *	— *	— *	— *	— *
Byte sync								
Bit sync HDLC	EOMF	SHRTF	ABTF	RBITF	OVRNF	CRCEF		
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	—	—
Initial value	0	0	0	0	0	0	0	0

Frame status at receive completion

Note: The bits marked with * are reserved. These bits always read 0 and can be set to 0 or 1.

When data with EOM bit = 1 (last character of the frame) is read from the receive buffer, the character status, which is reflected by bits 7–2 of status register 2 (ST2), is transferred and set in FST. This clears all bits of ST2 (figure 5.5). The definition of each bit of FST is the same as that of the corresponding bit of ST2. See section 5.2.11, MSCI Status Register 2 (ST2).

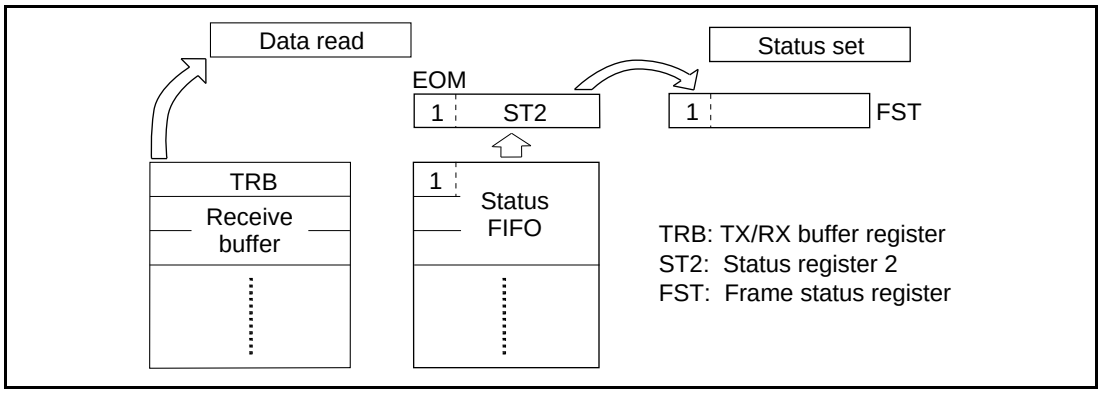


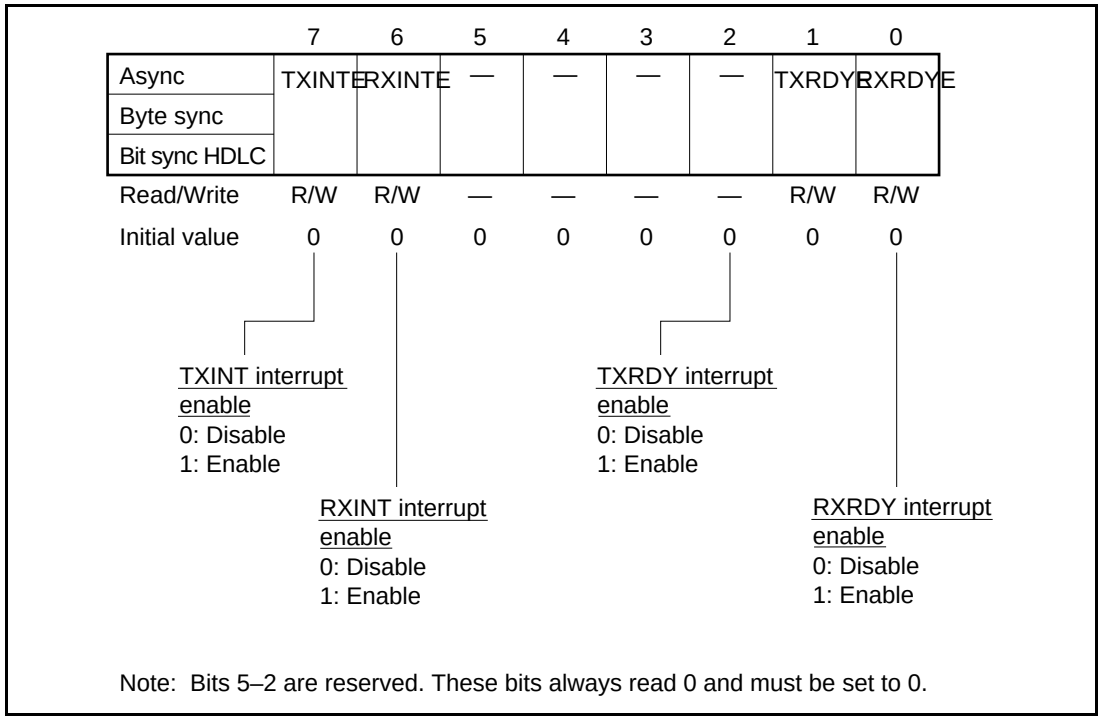
Figure 5.5 Frame Status Register (FST)

A frame end interrupt is generated when status data is set in FST. After the interrupt has been generated, the status of the received frame can be read from FST.

This method is used for MPU data transfer. In this case, residual bit frame interrupt, abort end frame interrupt, and CRC error interrupt must be disabled.

5.2.14 MSCI Interrupt Enable Register 0 (IE0)

Interrupt enable register 0 (IE0) enables or disables the TXINT, RXINT, TXRDY, and RXRDY interrupt requests. Interrupt requests are issued to the MPU when both the status register 0 (ST0) bits and the corresponding bits of this register are set to 1. For details on interrupts, see section 5.7, Interrupts.



Note: Bits 5–2 are reserved. These bits always read 0 and must be set to 0.

Bit 7 (TXINTE: TXINT Interrupt Enable): The function of this bit is described below.

- Asynchronous/Byte synchronous/Bit synchronous mode
  TXINTE = 0: Disables an interrupt request set by the TXINT bit of ST0
  TXINTE = 1: Enables an interrupt request set by the TXINT bit of ST0; a TXINT interrupt request is issued to the MPU when the TXINT bit of ST0 is set to 1

Bit 6 (RXINTE: RXINT Interrupt Enable): The function of this bit is described below.

- Asynchronous/Byte synchronous/Bit synchronous mode.
  RXINTE = 0: Disables an interrupt request set by the RXINT bit of ST0
  RXINTE = 1: Enables an interrupt request set by the RXINT bit of ST0; a RXINT interrupt request is issued to the MPU when the RXINT bit of ST0 is set to 1

Bits 5–2: Reserved. These bits always read 0 and must be set to 0.

Bit 1 (TXRDYE: TXRDY Interrupt Enable): The function of this bit is described below.

- Asynchronous/Byte synchronous/Bit synchronous mode

  TXRDYE = 0: Disables an interrupt request set by the TXRDY bit of ST0

  TXRDYE = 1: Enables an interrupt request set by the TXRDY bit of ST0; a TXRDY interrupt request is issued to the MPU when the TXRDY bit of ST0 is set to 1

Bit 0 (RXRDYE: RXRDY Interrupt Enable): The function of this bit is described below.

- Asynchronous/Byte synchronous/Bit synchronous mode

  RXRDYE = 0: Disables an interrupt request set by the RXRDY bit of ST0

  RXRDYE = 1: Enables an interrupt request set by the RXRDY bit of ST0; a RXRDY interrupt request is issued to the MPU when the RXRDY bit of ST0 is set to 1

The relationship between the interrupt enable bit and status bit is shown in figure 5.6.

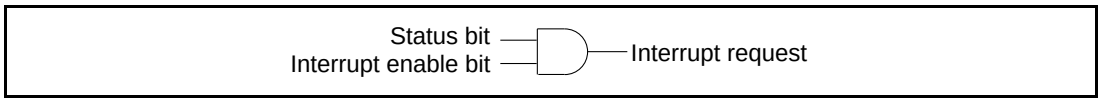
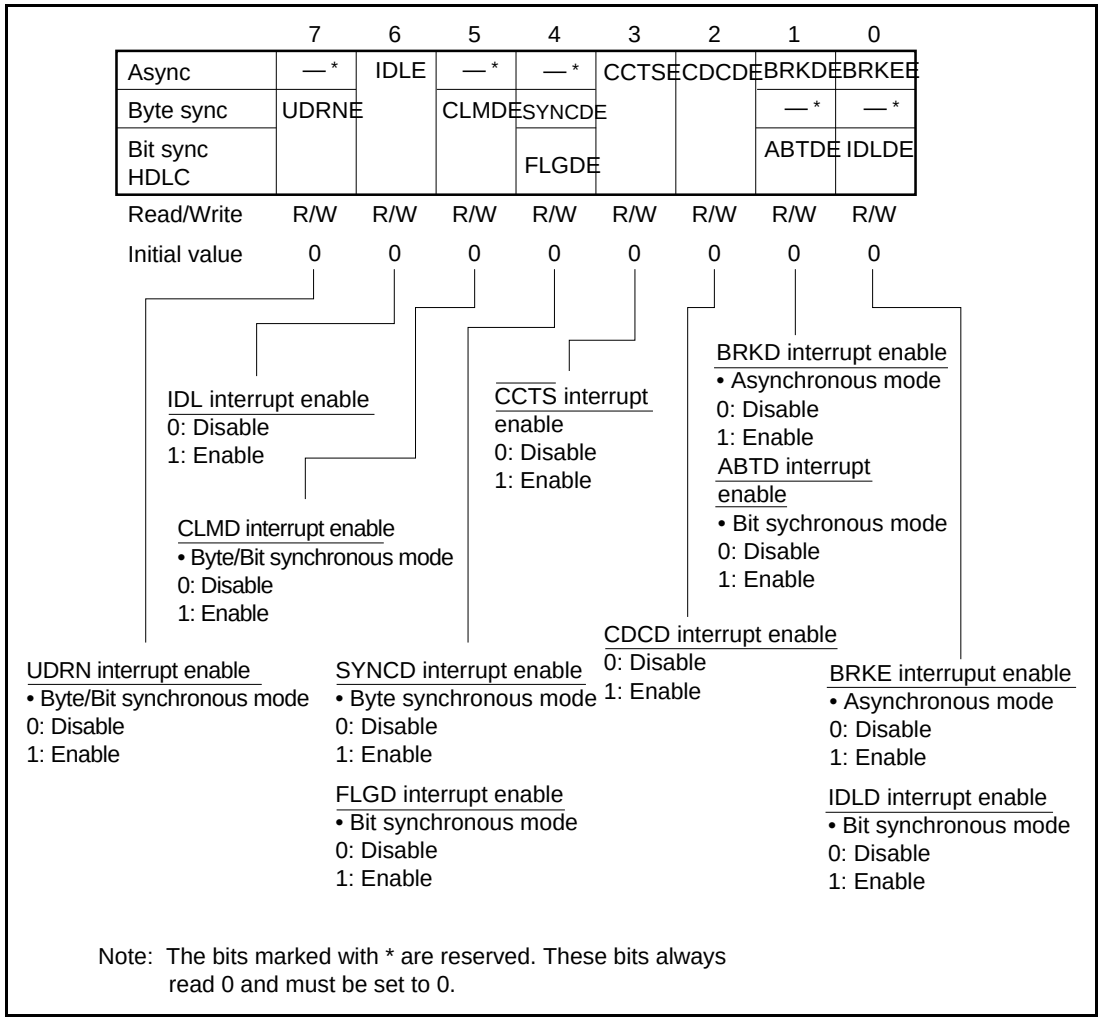


Figure 5.6 Interrupt Conditions

As shown in figure 5.6, an interrupt request is issued only when both the status bit and the interrupt enable bit are 1. The same relationship holds true between interrupt enable registers 0–2 (IE0–IE2) and status registers 0–2 (ST0–ST2), and between the frame interrupt enable register (FIE) and the frame status register (FST).

5.2.15 MSCI Interrupt Enable Register 1 (IE1)

Interrupt enable register 1 (IE1) enables or disables interrupt requests when the status bits of status register 1 (ST1) are set to 1. For details on interrupts, see section 5.7, Interrupts.



Note: The bits marked with * are reserved. These bits always read 0 and must be set to 0.

Bit 7 (UDRNE: UDRN Interrupt Enable): The function of this bit is described below.

- Asynchronous mode
  Reserved. This bit always reads 0 and must be set to 0.
- Byte synchronous/Bit synchronous mode
  UDRNE = 0: Disables an interrupt set by the UDRN bit of ST1
  UDRNE = 1: Enables an interrupt set by the UDRN bit of ST1; the TXINT bit of status register 0 (ST0) is set to 1 when the UDRN and UDRNE bits are both 1

Bit 6 (IDLE: IDL Interrupt Enable): The function of this bit is described below.

- Asynchronous/Byte synchronous/Bit synchronous mode
  IDLE = 0: Disables an interrupt set by the IDL bit of ST1
  IDLE = 1: Enables an interrupt set by the IDL bit of ST1; the TXINT bit of ST0 is set to 1 when the IDL and IDLE bits are both 1

Bit 5 (CLMDE: CLMD Interrupt Enable): The function of this bit is described below.

- Asynchronous mode
  Reserved. This bit always reads 0 and must be set to 0.
- Byte synchronous/Bit synchronous mode
  CLMDE = 0: Disables an interrupt set by the CLMD bit of ST1
  CLMDE = 1: Enables an interrupt set by the CLMD bit of ST1; the RXINT bit is set to 1 when the CLMD and CLMDE bits are both 1

Bit 4 (SYNCDE/FLGDE: SYNCD/FLGD Interrupt Enable): The function of this bit is described below.

- Asynchronous mode
  Reserved. This bit always reads 0 and must be set to 0.
- Byte synchronous/Bit synchronous mode
  SYNCDE/FLGDE = 0:Disables an interrupt set by the SYNCD/FLGD bit of ST1
  SYNCDE/FLGDE = 1:Enables an interrupt set by the SYNCD/FLGD bit of ST1; the RXINT bit of ST0 is set to 1 when the SYNCD/FLGD and SYNCDE/FLGDE bits are both 1

Bit 3 (CCTSE: CCTS Interrupt Enable): The function of this bit is described below.

- Asynchronous/Byte synchronous/Bit synchronous mode
  CCTSE = 0: Disables an interrupt set by the CCTS bit of ST1
  CCTSE = 1: Enables an interrupt set by the CCTS bit of ST1; the TXINT bit of ST0 is set to 1 when the CCTS and CCTSE bits are both 1

Bit 2 (CDCDE: CDCD Interrupt Enable): The function of this bit is described below.

- Asynchronous/Byte synchronous/Bit synchronous mode

  CDCDE = 0: Disables an interrupt set by the CDCD bit of ST1

  CDCDE = 1: Enables an interrupt set by the CDCD bit of ST1; the RXINT bit of ST0 is set to 1 when the CDCD and CDCDE bits are both 1

Bit 1 (BRKDE/ABTDE: BRKD/ABTD Interrupt Enable): The function of this bit is described below.

- Asynchronous/Bit synchronous mode

  BRKDE/ABTDE = 0: Disables an interrupt set by the BRKD/ABTD bit of ST1

  BRKDE/ABTDE = 1: Enables an interrupt set by the BRKD/ABTD bit of ST1; the RXINT bit of ST0 is set to 1 when the BRKD/ABTD and BRKDE/ABTDE bits are both 1
- Byte synchronous mode

  Reserved. This bit always reads 0 and must be set to 0.

Bit 0 (BRKEE/IDLDE: BRKE/IDLD Interrupt Enable): The function of this bit is described below.

- Asynchronous/Bit synchronous mode

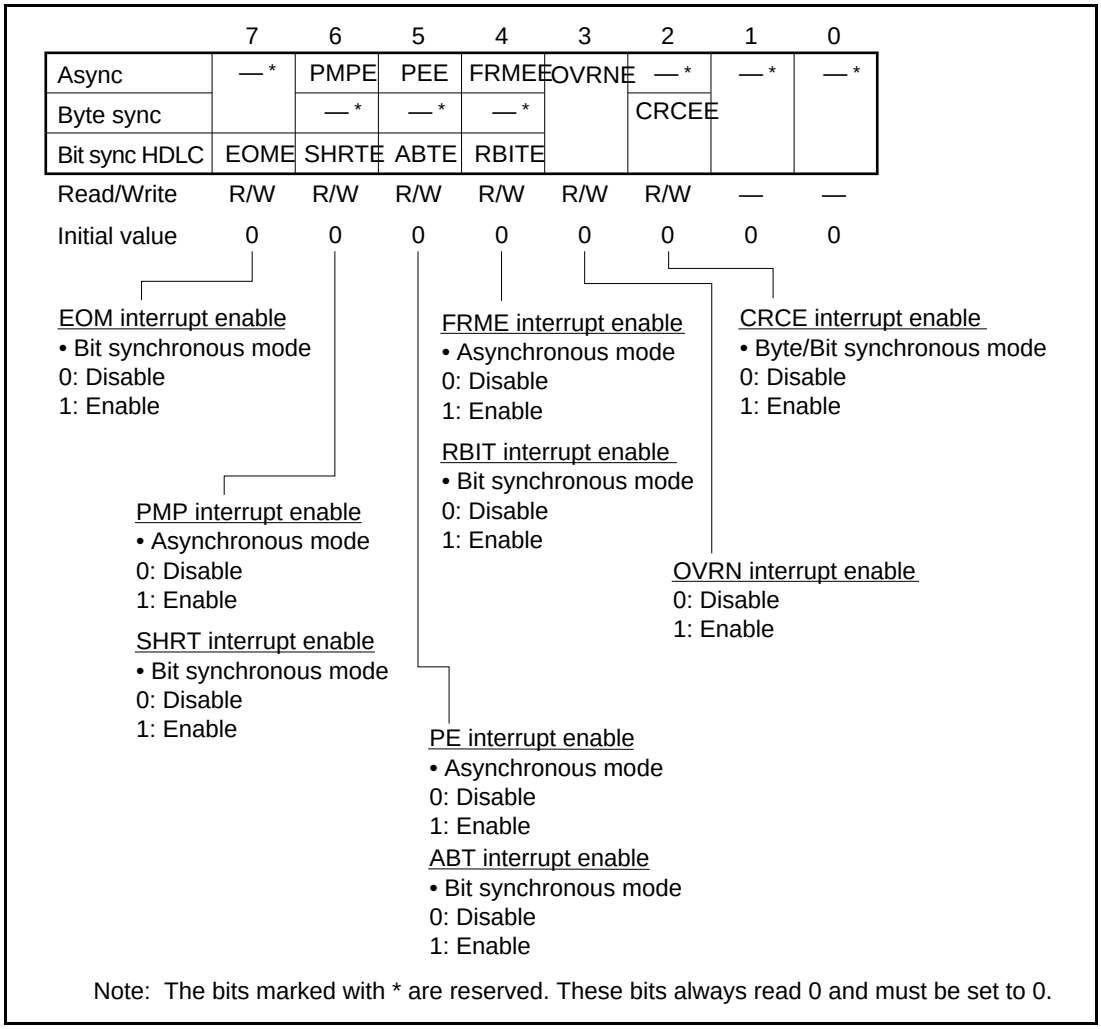
  BRKEE/IDLDE = 0: Disables an interrupt set by the BRKE/IDLD bit of ST1

  BRKEE/IDLDE = 1: Enables an interrupt set by the BRKE/IDLD bit of ST1; the RXINT bit of ST0 is set to 1 when the BRKE/IDLD and BRKEE/IDLDE bits are both 1
- Byte synchronous mode

  Reserved. This bit always reads 0 and must be set to 0.

5.2.16 MSCI Interrupt Enable Register 2 (IE2)

Interrupt enable register 2 (IE2) enables or disables interrupt requests when the status bits of status register 2 (ST2) are set to 1. For details on interrupts, see section 5.7, Interrupts.



Note: The bits marked with * are reserved. These bits always read 0 and must be set to 0.

Bit 7 (EOME: EOM Interrupt Enable): The function of this bit is described below.

- Asynchronous/Byte synchronous mode

  Reserved. This bit always reads 0 and must be set to 0.
- Bit synchronous mode

  EOME = 0: Disables an interrupt set by the EOM bit of ST2

  EOME = 1: Enables an interrupt set by the EOM bit of ST2; the RXINT bit of ST0 is set to 1 when the EOM and EOME bits are both 1

Bit 6 (PMPE/SHRTE: PMP/SHRT Interrupt Enable): The function of this bit is described below.

- Asynchronous/Bit synchronous mode

  PMPE/SHRTE = 0: Disables an interrupt set by the PMP/SHRT bit of ST2

  PMPE/SHRTE = 1: Enables an interrupt set by the PMP/SHRT bit of ST2; the RXINT bit of ST0 is set to 1 when the PMP/SHRT and PMPE/SHRTE bits are both 1
- Byte synchronous mode

  Reserved. This bit always reads 0 and must be set to 0.

Bit 5 (PEE/ABTE: PE/ABT Interrupt Enable): The function of this bit is described below.

- Asynchronous/Bit synchronous mode

  PEE/ABTE = 0: Disables an interrupt set by the PE/ABT bit of ST2

  PEE/ABTE = 1: Enables an interrupt set by the PE/ABT bit of ST2; the RXINT bit of ST0 is set to 1 when the PE/ABT and PEE/ABTE bits are both 1
- Byte synchronous mode

  Reserved. This bit always reads 0 and must be set to 0.

Bit 4 (FRMEE/RBITE: FRME/RBIT Interrupt Enable): The function of this bit is described below.

- Asynchronous/Bit synchronous mode

  FRMEE/RBITE = 0: Disables an interrupt set by the FRME/RBIT bit of ST2

  FRMEE/RBITE = 1: Enables an interrupt set by the FRME/RBIT bit of ST2; the RXINT bit of ST0 is set to 1 when FRME/RBIT and FRMEE/RBITE bits are both 1
- Byte synchronous mode

  Reserved. This bit always reads 0 and must be set to 0.

Bit 3 (OVRNE: OVRN Interrupt Enable): The function of this bit is described below.

- Asynchronous/Byte synchronous/Bit synchronous mode
  OVRNE = 0: Disables an interrupt set by the OVRN bit of ST2
  OVRNE = 1: Enables an interrupt set by the OVRN bit of ST2; the RXINT bit of ST0 is set to 1 when the OVRN and OVRNE bits are both 1

Bit 2 (CRCEE: CRCE interrupt enable): The function of this bit is described below.

- Asynchronous mode
  Reserved. This bit always reads 0 and must be set to 0.
- Byte synchronous/Bit synchronous mode
  CRCEE = 0: Disables an interrupt set by the CRCE bit of ST2
  CRCEE = 1: Enables an interrupt set by the CRCE bit of ST2; the RXINT bit of ST0 is set to 1 when the CRCE and CRCEE bits are both 1

Bits 1–0: Reserved. These bits always read 0 and must be set to 0.

5.2.17 MSCI Frame Interrupt Enable Register (FIE)

The frame interrupt enable register (FIE) enables or disables interrupt requests when the EOMF bit of the frame status register (FST) is set to 1.

	7	6	5	4	3	2	1	0
Async	— *	— *	— *	— *	— *	— *	— *	— *
Byte sync								
Bit sync HDLC	EOMFE							
Read/Write	R/W	—	—	—	—	—	—	—
Initial value	0	0	0	0	0	0	0	0

EOMF interrupt enable
• Bit synchronous mode
0: Disable
1: Enable

Note: The bits marked with * are reserved. These bits always read 0 and must be set to 0.

Bit 7 (EOMFE; EOMF Interrupt Enable): The function of this bit is described below.

- Asynchronous/Byte synchronous mode
  Reserved. This bit always reads 0 and must be set to 0.
- Bit synchronous mode
  EOMFE = 0: Disables an interrupt set by the EOMF bit of FST
  EOMFE = 1: Enables an interrupt set by the EOMF bit of FST; the RXINT bit of ST0 is set to 1 when the EOMF and EOMFE bits are both 1

Bits 6–0: Reserved. These bits always read 0 and must be set to 0.

5.2.18 MSCI Synchronous/Address Register 0 (SA0)

Synchronous/address register 0 (SA0) specifies the SYN character pattern for reception in byte synchronous mono-sync mode, the low-order eight bits of the SYN character pattern for transmission and reception in byte synchronous bi-sync mode, and the secondary station address in bit synchronous mode. This register is not used in asynchronous or byte synchronous external-sync mode.

	7	6	5	4	3	2	1	0
Async	—	—	—	—	—	—	—	—
Byte sync Bit sync HDLC	SA07	SA06	SA05	SA04	SA03	SA02	SA01	SA00
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial/value	1	1	1	1	1	1	1	1

SYN pattern for reception/address field check

• Byte synchronous mode

Mono-sync	SYN pattern for reception
Bi-sync	SYN pattern for transmission and reception (bits 7–0)
External-sync	Not used

• Bit synchronous mode

HDLC mode	No address field checked	Not used
	Single address 1	Bits 7–0 of the secondary station address
	Single address 2	Not used
	Dual address	Bits 7–0 of the secondary station address

Note: This register is not used in asynchronous mode.

Bits 7–0 (SA07–SA00: Synchronous/Address): The function of these bits is described below.

- Asynchronous mode

  Not used
- Byte synchronous (mono- or bi-sync) mode

  The SA07–SA00 bits specify bits 7–0 of the SYN character pattern for reception in mono-sync mode, and the low-order eight bits (bits 7–0) of the SYN character pattern for transmission and reception in bi-sync mode.
- Bit synchronous mode

  The SA07–SA00 bits set the values shown in table 5.4 according to the address field check mode selected in HDLC mode. The contents of this register are not used for transmission; the address must be written in the FIFO.

Table 5.4 SA07–SA00 Function in Bit Synchronous Mode

Mode	Address Field Check	Bits 7–0 of SA0
HDLC mode	No address field checked	Not used
	Single address 1	Bits 7–0 of the secondary station address
	Single address 2	Not used
	Dual address	Bits 7–0 of the secondary station address

When using a two-octet SYN character pattern or address, the first and second octets of data must be set in SA0 and SA1, respectively (figure 5.7).

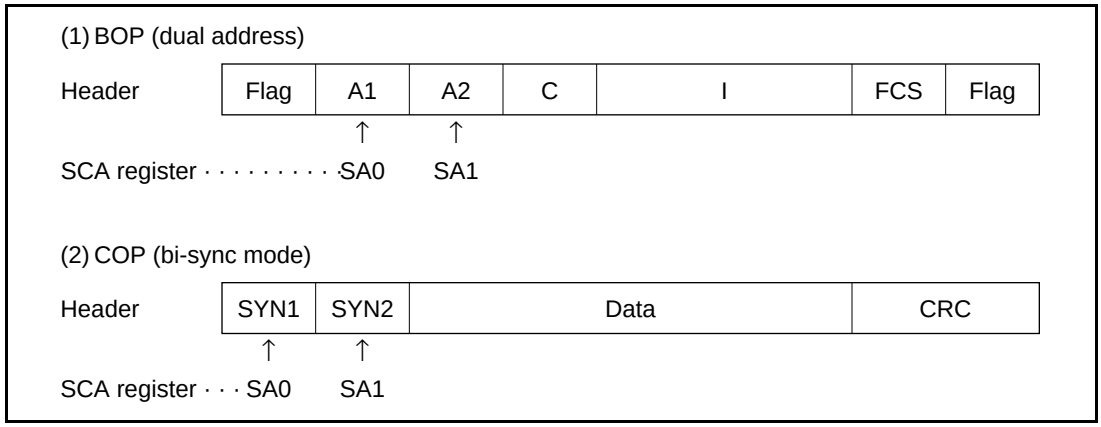


Figure 5.7 MSCI Synchronous/Address Registers (SA0 and SA1) and Two-Octet Data

5.2.19MSCI Synchronous/Address Register 1 (SA1)

Synchronous/address register 1 (SA1) specifies the SYN character pattern for transmission in byte synchronous mono-sync or byte synchronous external-sync mode, the SYN character pattern for transmission and reception in byte synchronous bi-sync mode, and the secondary station address in bit synchronous mode. This register is not used in asynchronous mode.

	7	6	5	4	3	2	1	0
Async	—	—	—	—	—	—	—	—
Byte sync Bit sync HDLC	SA17	SA16	SA15	SA14	SA13	SA12	SA11	SA10
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	1	1	1	1	1	1	1	1

SYN pattern for transmission/address field check

• Byte synchronous mode

Mono-sync	SYN pattern for transmission
Bi-sync	SYN pattern for transmission and reception (bits 15–8)
External-sync	SYN pattern for transmission

• Bit synchronous mode

HDLC mode	No address field checked	Not used
	Single address 1	Not used
	Single address 2	Bits 15–8 of the secondary station address
	Dual address	Bits 15–8 of the secondary station address

Note: This register is not used in asynchronous mode.

Bits 7–0 (SA17–SA10: Synchronous/Address): The function of these bits is described below.

- Asynchronous mode

  Not used
- Byte synchronous mode

  The SA17–SA10 bits specify bits 7–0 of the SYN character pattern for transmission in byte synchronous mono-sync or byte synchronous external-sync mode, and the high-order eight bits (bits 15–8) of the SYN character pattern in bi-sync mode.
- Bit synchronous mode

  The SA17–SA10 bits set the values shown in table 5.5 according to the address field check mode selected in HDLC mode. The contents of this register are not used for transmission; the address must be written in the FIFO.

Table 5.5 SA17–SA10 Function in Bit Synchronous Mode

Mode	Address Field Check	Bits 7–0 of SA1
HDLC mode	No address field checked	Not used
	Single address 1	Not used
	Single address 2	Bits 15–8 of the secondary station address
	Dual address	Bits 15–8 of the secondary station address

5.2.20 MSCI Idle Pattern Register (IDL)

The idle pattern register (IDL) specifies the idle pattern output by the transmitter when it is in idle state.

	7	6	5	4	3	2	1	0
Async	—	—	—	—	—	—	—	—
Byte sync Bit sync HDLC	IDL7	IDL6	IDL5	IDL4	IDL3	IDL2	IDL1	IDL0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	1	1	1	1	1	1	1	1

Idle pattern

Note: This register is not used in asynchronous mode.

Bits 7–0 (IDL7–IDL0: Idle Pattern): The function of these bits is described below.

- Asynchronous mode
  Not used
- Byte synchronous/Bit synchronous mode
  When the IDLC bit of the control register (CTL) is 1, the idle pattern set in this register is output from the TXD line during the idle state. When the IDLC bit is 0, the TXD line is fixed high.

5.2.21 MSCI TX/RX Buffer Register (TRB: TRBH, TRBL)

The TX/RX buffer register (TRB: TRBH, TRBL), located at the top of the 32-stage transmit/receive buffer (TX/RX buffer), interfaces with the internal data bus. Although the TX and RX buffers are physically different, this register is used for both reading receive data from the RX register and writing transmit data to the TX buffer.

TRBH

	7	6	5	4	3	2	1	0
Async / Byte sync / Bit sync HDLC	TRB15 (TRBH7)	TRB14 (TRBH6)	TRB13 (TRBH5)	TRB12 (TRBH4)	TRB11 (TRBH3)	TRB10 (TRBH2)	TRB9 (TRBH1)	TRB8 (TRBH0)
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	X	X	X	X	X	X	X	X

Value written to, or read from, the transmit/receive buffer

X: Undefined

TRBL

	7	6	5	4	3	2	1	0
Async / Byte sync / Bit sync HDLC	TRB7 (TRBL7)	TRB6 (TRBL6)	TRB5 (TRBL5)	TRB4 (TRBL4)	TRB3 (TRBL3)	TRB2 (TRBL2)	TRB1 (TRBL1)	TRB0 (TRBL0)
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	X	X	X	X	X	X	X	X

Value written to, or read from, the transmit/receive buffer

X: Undefined

TRBH Bits 7–0 (TRB15–TRB8/TRBH7–TRBH0: TX/RX Buffer High Byte (TRBH)): The function of these bits is described below.

- Asynchronous/Byte synchronous/Bit synchronous mode

  Reading TRBH bits 7–0 reads a receive character from the receive buffer. If data is read from these bits with the RXRDY bit of status register 0 (ST0) cleared to 0, the values are undefined and subsequent operation is not guaranteed.

  Writing TRBH bits 7–0 writes a transmit character to the transmit buffer. If data is written to these bits with the TXRDY bit of ST0 cleared to 0, the write data and/or the data in the transmit buffer may be lost.

TRBL Bits 7–0 (TRB7–TRB0/TRBL7–TRBL0: TX/RX Buffer Low Byte (TRBL)): The function of these bits is described below.

- Asynchronous/Byte synchronous/Bit synchronous mode

  Reading TRBL bits 7–0 reads a receive character from the receive buffer. If data is read from these bits, with the RXRDY bit of ST0 cleared to 0, the values are undefined and subsequent operation is not guaranteed.

  Writing TRBL bits 7–0 writes a transmit character to the transmit buffer. If data is written to these bits, with the TXRDY bit of ST0 cleared to 0, the write data and/or the data in the transmit buffer may be lost.

TRB read operation: Data is read from the receive buffer at TRB read in the procedure listed in table 5.6 to table 5.8.

Table 5.6 TRB Read Operation in CPU Mode 0

		Data Byte Count in Receive Buffer*[1]		
Read Mode	**Accessed Register*[2]**	**2 or More Bytes**	**1 Byte**	**None**
Word read	TRBH	Data 1*[3]	Undefined*[4]	Undefined*[4]
	TRBL	Data 0*[3]	Data 0*[3]	Undefined*[4]
Byte read	TRBH	Data 0*[3]	Data 0*[3]	Undefined*[4]
	TRBL	Data 0*[3]	Data 0*[3]	Undefined*[4]

Table 5.7 TRB Read Operation in CPU Mode 1

		Data Byte Count in Receive Buffer*[1]		
Read Mode	**Accessed Register*[2]**	**2 or More Bytes**	**1 Byte**	**None**
Byte read	TRBH	Data 0*[3]	Data 0*[3]	Undefined*[4]
	TRBL	Data 0*[3]	Data 0*[3]	Undefined*[4]

Table 5.8 TRB Read Operation in CPU Modes 2, 3

		Data Byte Count in Receive Buffer*[1]		
Read Mode	**Accessed Register*[2]**	**2 or More Bytes**	**1 Byte**	**None**
Word read	TRBH	Data 0*[3]	Data 0*[3]	Undefined*[4]
	TRBL	Data 1*[3]	Undefined*[4]	Undefined*[4]
Byte read	TRBH	Data 0*[3]	Data 0*[3]	Undefined*[4]
	TRBL	Data 0*[3]	Data 0*[3]	Undefined*[4]

Notes: 1. Data byte count in the receive buffer is reflected by the CDE1 and CDE0 bits of current status registers 1 and 0 (CTS1 and CTS0).
2. TRBH and TRBL are simultaneously accessed in word read mode, and either TRBH or TRBL is accessed in byte read mode.
3. Data 0 and 1 are arranged in the order shown in figures 5.8 (a) and (b). The serial unit sends data 0 and data 1 to the receive buffer in that order.
4. If undefined data is read, subsequent operation is not guaranteed. Undefined data is not read in a built-in DMA transfer.

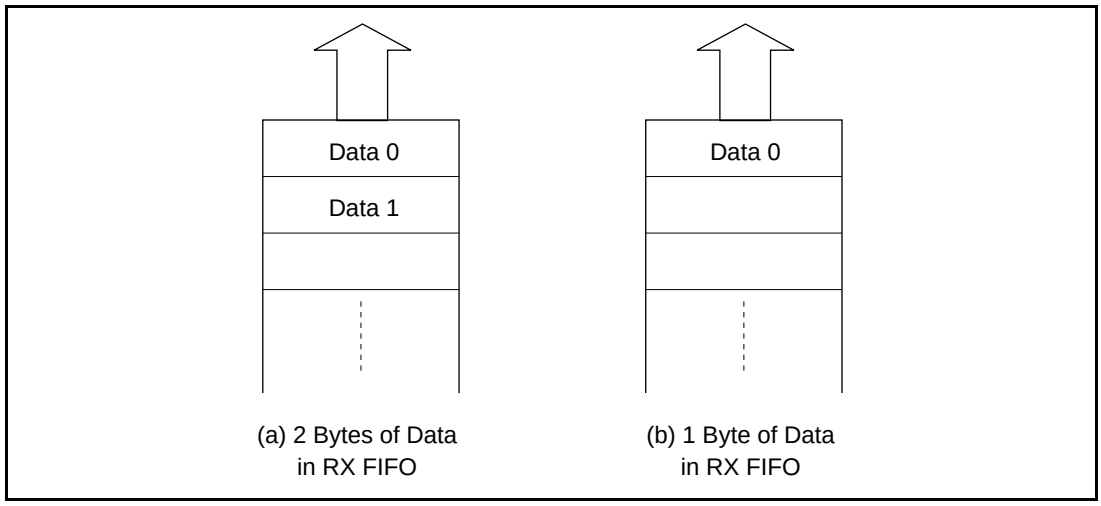


Figure 5.8 Data Arrangement in Receive Buffer

TRB write operation: Data is written to the transmit buffer at TRB write in the procedure as listed in table 5.9 to table 5.11.

Table 5.9 TRB Write Operation in CPU Mode 0

		Empty Data Byte Count in Transmit Buffer*1		
Read Mode	**Accessed Register*2**	**2 or More Bytes**	**1 Byte**	**None**
Word write	TRBH	B*3	—*4	—*4
	TRBL	A*3	—*4	—*4
Byte write	TRBH	A*3	A*3	—*4
	TRBL	A*3	A*3	—*4

Table 5.10 TRB Write Operation in CPU Mode 1

		Empty Data Byte Count in Transmit Buffer*1		
Read Mode	**Accessed Register*2**	**2 or More Bytes**	**1 Byte**	**None**
Byte write	TRBH	A*3	A*3	—*4
	TRBL	A*3	A*3	—*4

Table 5.11 TRB Write Operation in CPU Modes 2, 3

		Empty Data Byte Count in Transmit Buffer*1		
Read Mode	**Accessed Register*2**	**2 or More Bytes**	**1 Byte**	**None**
Word write	TRBH	A*3	—*4	—*4
	TRBL	B*3	—*4	—*4
Byte write	TRBH	A*3	A*3	—*4
	TRBL	A*3	A*3	—*4

Notes:
1. Because the empty data byte count in the transmit buffer is unknown to the user, extra care must be taken in writing data to the transmit buffer using the MPU. Set the TRC14–TRC10 bits of TX ready control register 1 (TRC1) to 1EH or less in CPU modes 0, 2, and 3 (1FH in CPU mode 1), and confirm that the TXRDY bit of status register 0 (ST0) is 1 before writing data to the transmit buffer. (Do not write data to the transmit buffer when the TXRDY bit is 0.) However, this procedure is not necessary in built-in DMA transfer.
2. TRBH and TRBL are simultaneously accessed in word write mode, and either TRBH or TRBL is accessed in byte write mode.
3. Empty bytes A and B are arranged in the order as shown in figures 5.9 (a) and (b). The transmit buffer sends A and B to the serial unit in that order.
4. Data is lost except in built-in DMA transfer. Data in the transmit buffer is not lost, but writing data to the full buffer does not guarantee subsequent operation.

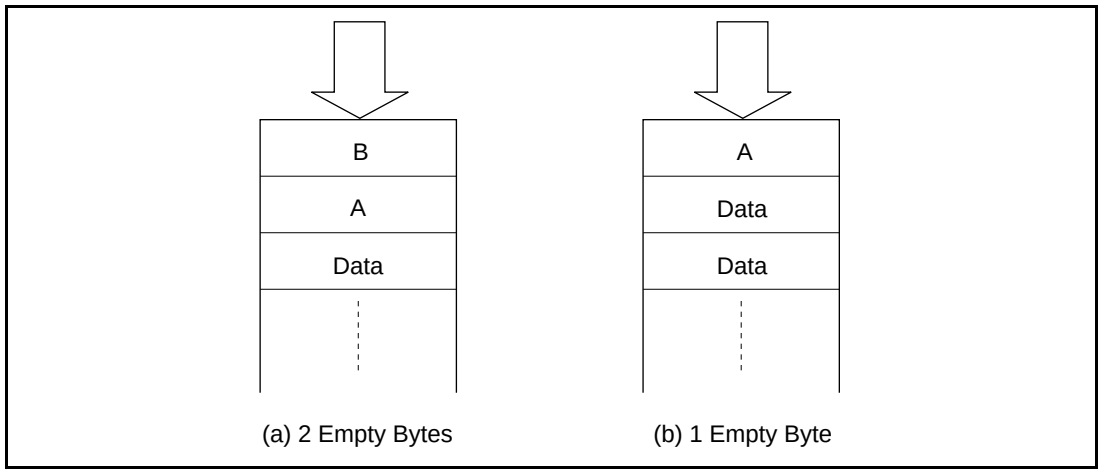


Figure 5.9 Empty Data Byte Arrangement in Transmit Buffer

5.2.22 MSCI RX Ready Control Register (RRC)

The RX ready control register (RRC) determines the MSCI RX ready (RXRDY) activation condition. The function of this register is the same in asynchronous, byte synchronous, and bit synchronous modes.

	7	6	5	4	3	2	1	0
Async Byte sync Bit sync HDLC	—	—	—	RRC4	RRC3	RRC2	RRC1	RRC0
Read/Write	—	—	—	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

RX ready control (RXF)

Note: Bits 7-5 are reserved. These bits always read 0 and must be set to 0.

Bits 7–5: Reserved. These bits always read 0 and must be set to 0.

Bits 4–0 (RRC4–RRC0: RX Ready Control): Determine the MSCI RX ready (RXRDY) activation condition. When the data byte count in the receive buffer is equal to or greater than RXF + 1, that is, the value set by these bits + 1, RX ready is activated. In other words, the RXRDY bit of status control register 0 (ST0) is set to 1. (The RXRDY bit is set to 0 when there is no data left in the receive buffer.) Any value can be set in the range from 00H–1FH.

5.2.23 MSCI TX Ready Control Register 0 (TRC0)

TX ready control register 0 (TRC0) determines the MSCI TX ready (TXRDY) activation condition. The function of this register is the same in asynchronous, byte synchronous, and bit synchronous modes.

	7	6	5	4	3	2	1	0
Async Byte sync Bit sync HDLC	—	—	—	TRC04	TRC03	TRC02	TRC01	TRC00
Read/Write	—	—	—	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

TX ready control 0 (TXF0)

Note: Bits 7–5 are reserved. These bits always read 0 and must be set to 0.

Bits 7–5: Reserved. These bits always read 0 and must be set to 0.

Bits 4–0 (TRC04–TRC00: TX Ready Control 0): Determine the MSCI TX ready (TXRDY) activation condition. When the data byte count in the transmit buffer is equal to or less than TXF0, that is, the value set by these bits, TX ready is activated. In other words, the TXRDY bit of status control register 0 (ST0) is set to 1. Any value can be set in the range from 00H–1FH.

5.2.24 MSCI TX Ready Control Register 1 (TRC1)

TX ready control register 1 (TRC1) determines the MSCI TX ready (TXRDY) inactivation condition. The function of this register is the same in asynchronous, byte synchronous, and bit synchronous modes.

	7	6	5	4	3	2	1	0
Async Byte sync Bit sync HDLC	—	—	—	TRC14	TRC13	TRC12	TRC11	TRC10
Read/Write	—	—	—	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	1	1	1	1	1

TX ready control 1 (TXF1)

Note: Bits 7–5 are reserved. These bits always read 0 and must be set to 0.

Bits 7–5: Reserved. These bits always read 0 and must be set to 0.

Bits 4–0 (TRC14–TRC10: TX Ready Control 1): Determine the MSCI TX ready (TXRDY) inactivation condition. When the data byte count in the transmit buffer is equal to or greater than TXF1 + 1, that is, the value set by these bits + 1, TX ready is inactivated. In other words, the TXRDY bit of status control register 0 (ST0) is set to 0. Any value can be set in the range from 00H–1FH. Note that TX ready is inactivated (the TXRDY bit of ST0 is set to 0) when the data byte count in the transmit buffer is equal to or greater than TXF1 + 1, under the condition that TXF1 is less than TXF0 (the value set by TRC04–TRC00 bits of TX ready control register 0 (TRC0)).

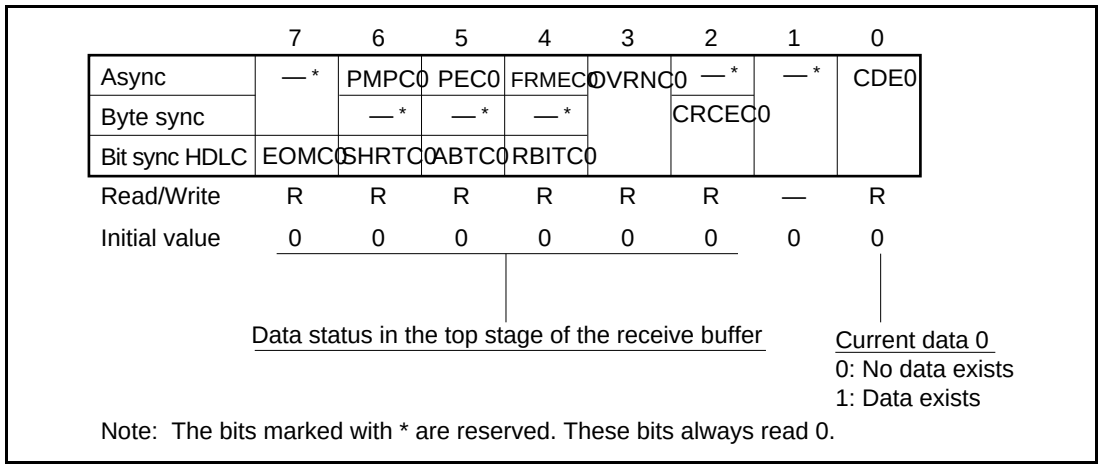
5.2.25 MSCI Current Status Register 0 (CST0)

Current status register 0 (CST0) monitors the top stage of the MSCI's 32-stage status FIFO. This register indicates whether or not data is in the top stage of the receive buffer, and if there is any data, indicates the status of the data.

This register is reset under either of the following conditions:

- RX reset command
- Channel reset command
- System stop mode

No bit of this register generates any interrupt.



Bits 7–2: Indicate the status of the data in the top stage of the receive buffer. These bits are arranged in the same way as bits 7–2 of the status register (ST2). When data is in the top stage of the receive buffer, the status of the top stage of the status FIFO is set to these bits. The status is activated when the TX/RX buffer register (TRB) is ready to be read. When data is read from TRB, the status of the data is cleared and replaced by the status of the following data. When there is no subsequent data, the status remains cleared.

Bit 1: Reserved. This bit always reads 0 and must be set to 0.

Bit 0 (CDE0: Current Data 0): Indicates that data is in the top stage of the receive buffer. This bit is set to 1 when TRB is ready to be read, and is cleared when data has been read and there is no subsequent data.

CDE0 = 0: Indicates that no data is in the top stage of the receive buffer

CDE0 = 1: Indicates that data is in the top stage of the receive buffer

This register monitors the status of only the top stage of the receive status FIFO in CPU modes 0, 2, and 3, which is different from status register 2 (ST2). ST2 monitors the OR of the status of the top and second stages of the receive status FIFO. This register is also characterized by its capability of monitoring the presence/absence of data in the top stage of the receive buffer. As a result, monitoring this register and current status register 1 (CST1) enables the user to determine which data status has generated an interrupt, and whether or not the following data can be read by word. For CST1, see section 5.2.26, MSCI Current Status Register 1 (CST1).

5.2.26 MSCI Current Status Register 1 (CST1)

Current status register 1 (CST1) monitors the second stage of the MSCI's 32-stage status FIFO. This register indicates whether or not data is in the second stage of the receive buffer, and if there is any data, indicates the status of the data.

This register is reset under either of the following conditions:

- RX reset command
- Channel reset command
- System stop mode

No bit of this register generates an interrupt.

	7	6	5	4	3	2	1	0
Async	—*	PMPC1	PEC1	FRMEC1	OVRNC1	—*	—*	CDE1
Byte sync		—*	—*	—*		CRCEC1		
Bit sync HDLC	EOMC1	SHRTC1	ABTC1	RBITC1				
Read/Write	R	R	R	R	R	R	–	R
Initial value	0	0	0	0	0	0	0	0

Data status in the second stage of the receive buffer

Current data 1
0: No data exists
1: Data exists

Note: The bits marked with * are reserved. These bits always read 0.

Bits 7–2: Indicate the status of the data in the second stage of the receive buffer. These bits are arranged in the same way as bits 7–2 of status register 2 (ST2). When data is in the second stage of the receive buffer, the status of the second stage of the status FIFO is set to these bits. The status is activated when the TX/RX buffer register (TRB) is ready to be read. When data is read from TRB, the status of the data is cleared and replaced by the status of the following data. When there is no subsequent data, the status remains cleared.

Bit 1: Reserved. This bit always reads 0 and must be set to 0.

Bit 0 (CDE1: Current Data 1): Indicates that data is in the second stage of the receive buffer. This bit is set to 1 when TRB is ready to be read, and is cleared when data has been read with no subsequent data.

CDE1 = 0: Indicates that no data is in the second stage of the receive buffer

CDE1 = 1: Indicates that data is in the second stage of the receive buffer

This register and current status register 0 (CST0) are used by the MPU for interrupt processing and receive buffer access. For details, see section 5.2.25, MSCI Current Status Register 0 (CST0).

5.3 Operation

5.3.1 Asynchronous Mode

In asynchronous mode, a start bit and stop bit(s) are appended to the character before transmission to synchronize character. In this mode, the transmission line is normally high (mark); when the line goes low, that is, when a start bit is detected, data transmission starts.

The start bit is followed by data, which begins with the least significant bit (LSB), that is, bit 0. The data may be optionally followed by a parity/MP bit. The data transmission ends with 1, 1.5, or 2 stop bits.

Figure 5.10 shows the character format for asynchronous mode. In this mode, data is transmitted and received in character units which may be 5 to 8 bits in length. When the character length is 5 to 7 bits, each received character is extended to 8 bits, by padding the high-order bits with 0s.

The PRTCL2–PRTCL0 bits of mode register 0 (MD0) specify asynchronous mode. The TXCHR1–TXCHR0 and RXCHR1–RXCHR0 bits of mode register 1 (MD1) specify the character length; the STOP1–STOP0 bits of MD0 specify the stop bit length (for details on how to set these bits, see section 5.2.1, MSCI Mode Register 0 (MD0), and section 5.2.2, MSCI Mode Register 1 (MD1)); and the PMPM1–PMPM0 bits of MD1 specify the parity/MP bit setting. In asynchronous mode, only the NRZ type code is available.

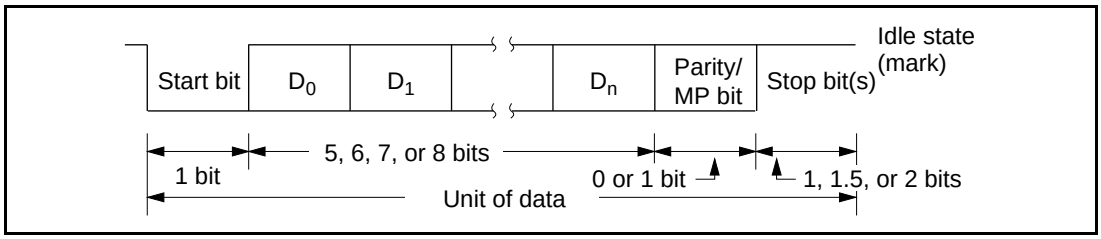


Figure 5.10 Character Format for Asynchronous Mode

The transmit and receive bit rates can be independently selected from the input frequency ratios 1/1, 1/16, 1/32, or 1/64 by using the BRATE1−BRATE0 bits of MD1 (figure 5.11). (The selected bit rate is used for both transmission and reception.) Since data is sampled at the rising edge of the clock pulse, bit-by-bit synchronization is necessary when the 1/1 clock mode is selected (figure 5.12).

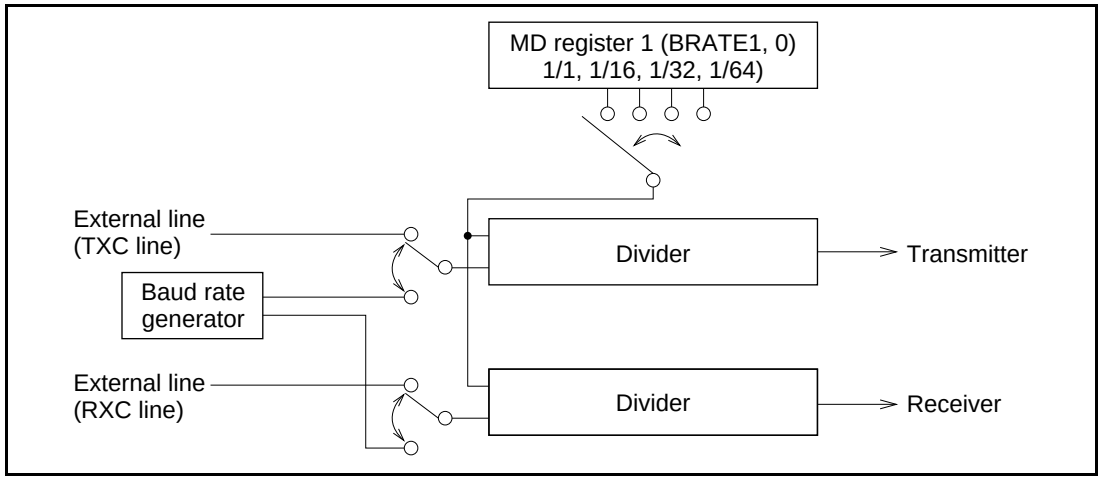


Figure 5.11 Bit Rate Selection

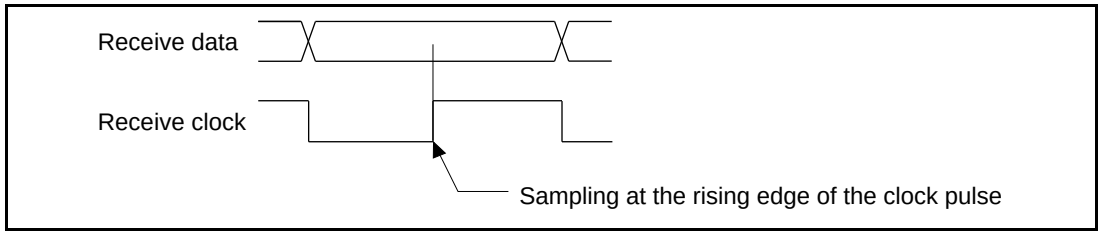


Figure 5.12 Data Sampling Timing (1/1 clock mode)

The external clock or internal baud rate generator output can be program-selected as the input/output clock. The ADPLL clock extraction function is not available in asynchronous mode. The clock can be specified with the RX clock source register (RXS) and TX clock source register (TXS). For details, see section 5.2.5, MSCI RX Clock Source Register (RXS), and section 5.2.6, MSCI TX Clock Source Register (TXS).

For details on the internal baud rate generator, see section 5.6, Baud Rate Generator.

Transmission Operation: Figure 5.13 is the state transition diagram for transmission in asynchronous mode.

- TX disable state

  The transmitter is placed in TX disable state by a hardware reset, a channel reset, a TX reset, or a TX disable command. In this state, the TXD line is high (mark), and the TXRDY bit of status register 0 (ST0) is cleared.

  Data must not be written in the transmit buffer during the transmit disable state. At other times, when writing data in the transmit buffer, the CPU should poll the TXRDY bit. Alternatively, an interrupt or DMA transfer can be used.
- Idle state

  The TX enable command sets the transmitter in idle state from TX disable state. In idle state, the TXD line remains high (mark) until transmit data is written to the transmit buffer. When the transmit data is written, the transmitter enters start bit transmit state.
- Start bit transmit state

  The TXD line is low (space) for one bit cycle. Then the transmitter enters character transmit state.
- Character transmit state

  The transmitter transmits a character from the transmit buffer, beginning with the LSB.
- Parity/MP bit transmit state

  The transmitter sends a parity or an MP bit as specified with the PMPM1–PMPM0 bits of MD1. For details, see Parity/MP Bit below.
- Stop bit transmit state

  The transmitter sends stop bit(s) as specified with the STOP1–STOP0 bits of MD0, and then returns to idle state.
- Break transmit state

  The TXD line goes low (space). The transmitter transmits a break when the BRK bit of the control register (CTL) is set to 1. The TXD line remains low until the BRK bit is cleared.
- One-bit cycle mark transmit state

  The TXD line goes high (mark) and remains so for one bit cycle after the break transmit state is canceled.

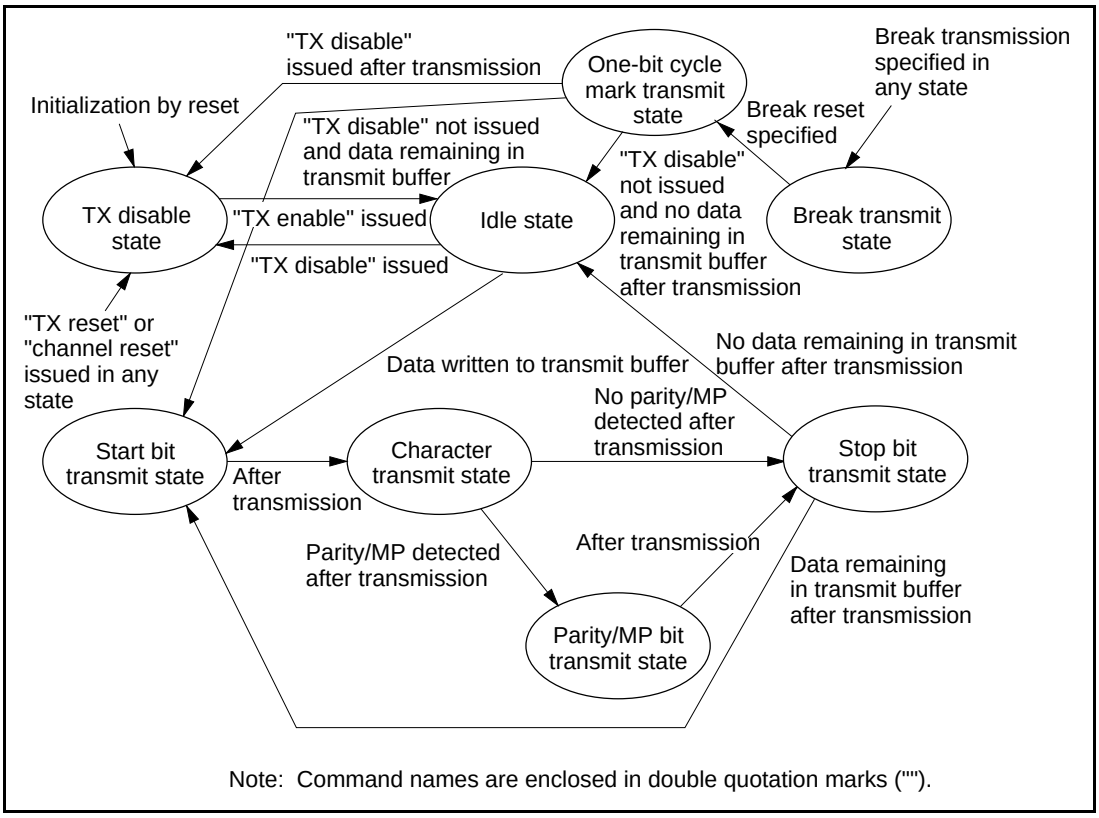


Figure 5.13 State Transition Diagram for Transmission in Asynchronous Mode

Transmission operation starts when a transmit character is written to the transmit buffer in idle state. The transmit line output changes at the falling edge of the transmit clock pulse as shown in figures 5.14 (a) and (b). This example uses an 8-bit character, one stop bit, with parity.

A stop bit length of 1, 1.5, or 2 can be specified in 1/16, 1/32, or 1/64 clock mode. In 1/1 clock mode, only a stop bit length of 1 or 2 is available. Even if 1.5 is specified in this mode, 1 or 2 stop bits will be used.

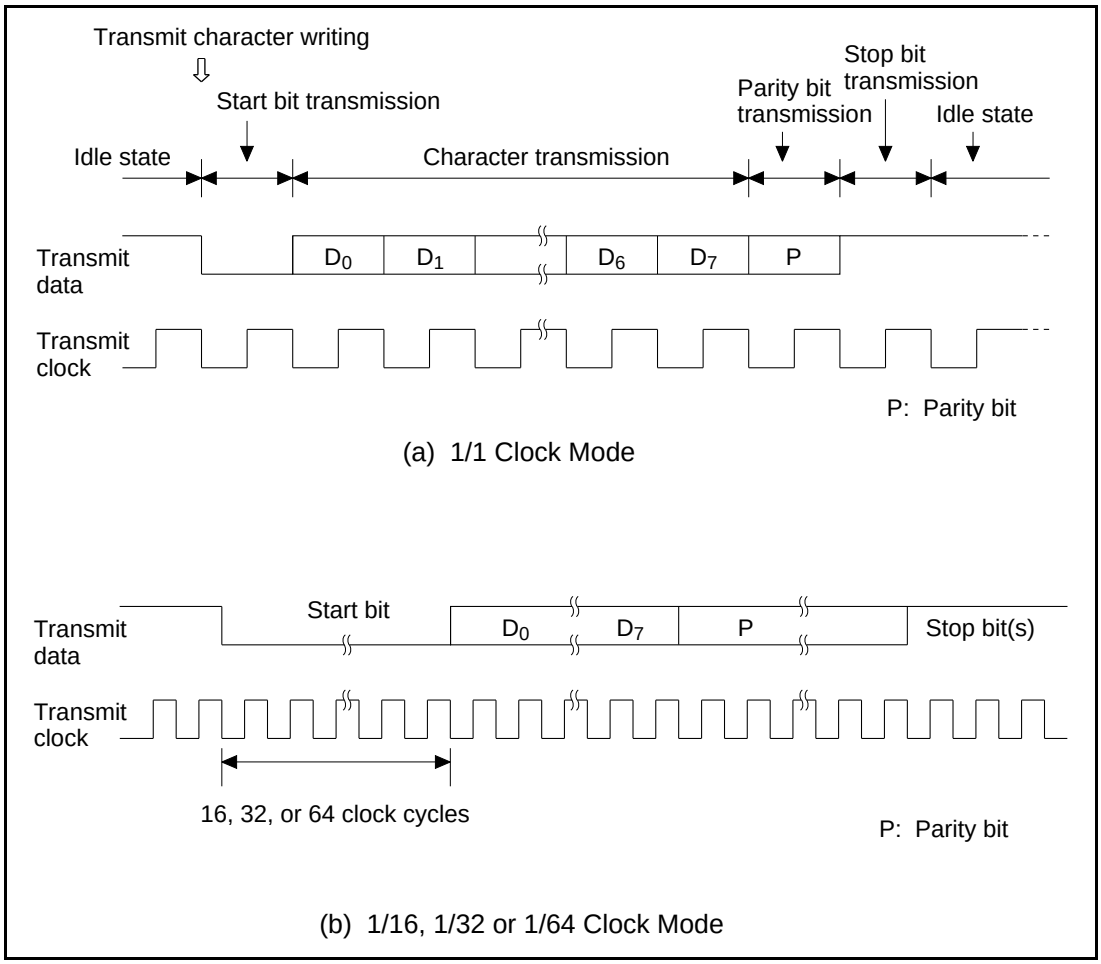


Figure 5.14 Transmission Operation

Reception Operation: Figure 5.15 is the state transition diagram for reception in asynchronous mode.

- RX disable state

  The receiver is placed in RX disable state by a hardware reset, a channel reset, an RX reset, or an RX disable command. In this state, the receiver ignores the input from the RXD line and does not perform a reception operation. The contents of the receive shift register are lost, but the value of the receive buffer is not affected.

- Start bit search state

  RX enable sets the receiver in start bit search state from RX disable state. In the start bit search state, the receiver samples the RXD line level at the rising edge of each receive clock pulse until a low level is detected.

- Start bit check state

  On detecting a low level while in start bit search state, the receiver enters start bit check state. In this state, the receiver waits for half a bit cycle and then samples the RXD line again. If the line is still low, the receiver enters character assembly state. If the line is not low, the receiver returns to start bit search state. If the line remains at space, the receiver enters character assembly state.

  In 1/1 clock mode, the RXD line is not retested, and the receiver enters character assembly state immediately after space detection.

- Character assembly state

  The receiver samples the received data at each bit cycle and assembles a character. The receiver ends character assembly when the first stop bit is detected.

- Half-bit cycle wait state

  If a framing error occurs after character assembly, the receiver waits for half a bit cycle in order to skip the stop bit associated with the framing error, and then enters start bit search state.

  For details on a framing error, see Error Checking in this section.

- Break end wait state

  On detecting a break after character assembly, the receiver enters break end wait state, where it samples the RXD line level at each clock cycle until the line goes high (mark).

  For details on a break, see Break Transmission and Detection.

- Break end check state

  On detecting a high level while in break end wait state, the receiver enters break end check state. In this state, the receiver waits for half a bit cycle and then samples the RXD line level again. If the line remains high, the receiver enters start bit search state. If the line does not remain high, the receiver returns to break end wait state.

  In 1/1 clock mode, the RXD line is not retested, and the receiver enters start bit search state immediately after mark detection.

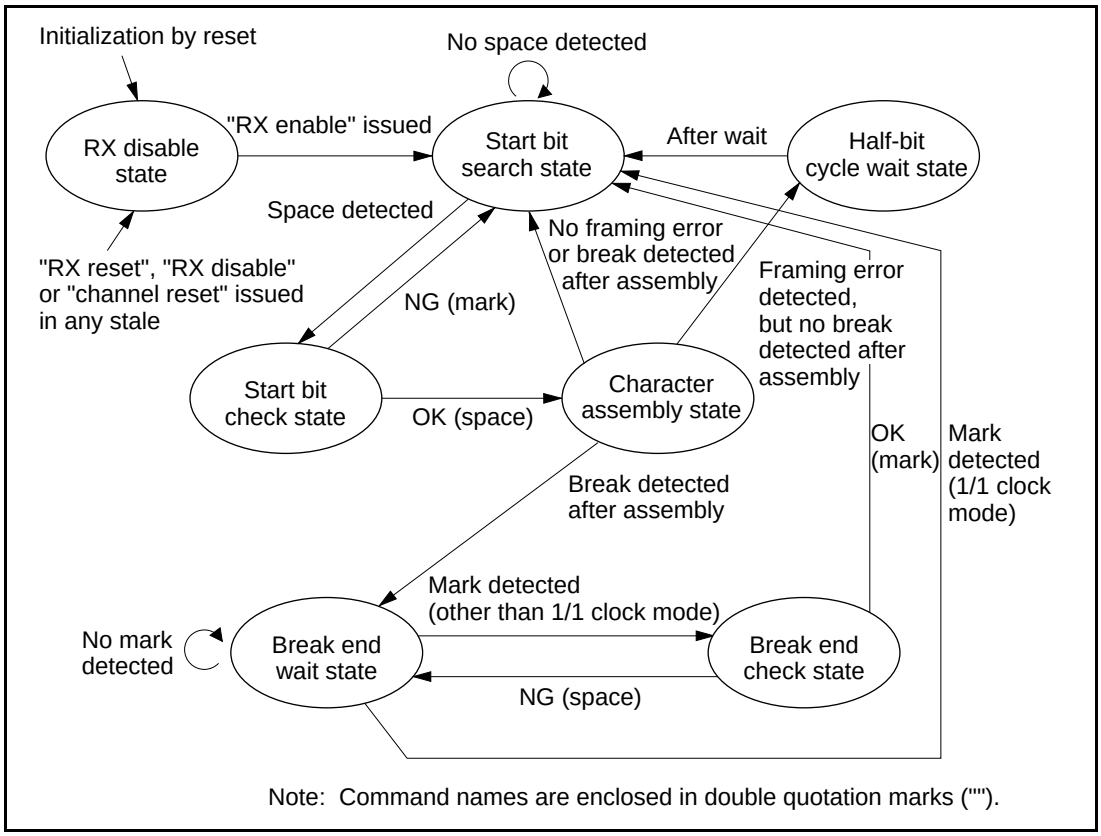


Figure 5.15 State Transition Diagram for Reception in Asynchronous Mode

The timing of sampling receive data is shown in figures 5.16 (a) and (b). This example uses an 8-bit character and one stop bit, with parity.

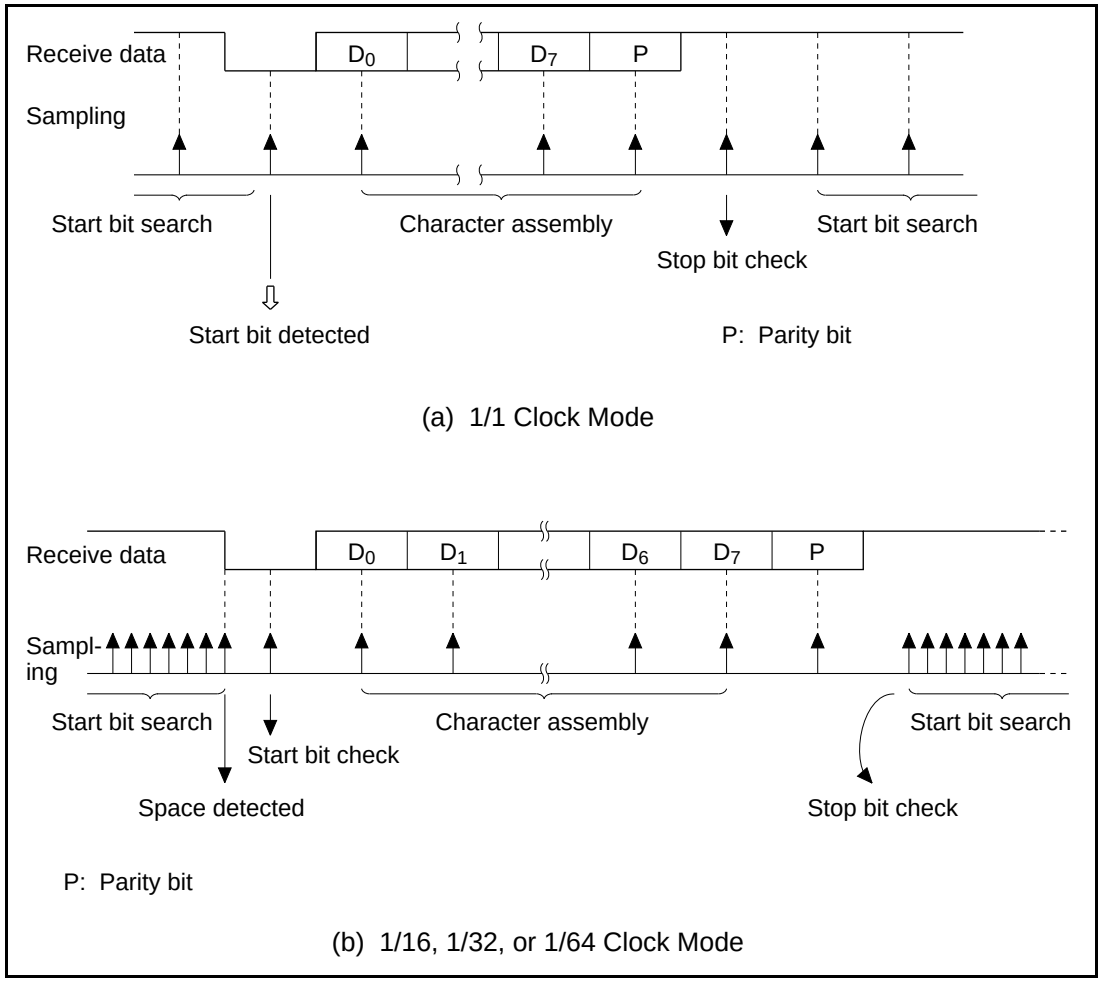


Figure 5.16 Receive Data Sampling Timing

Reception operation starts when an RX enable command is issued.

In 1/1 clock mode, the receiver searches for a start bit at the rising edge of each clock pulse. On detecting a space (low level), the receiver begins character assembly at the rising edge of the next clock pulse.

Character assembly involves assembling a character by loading bit data, which has been sampled at each clock cycle, into the receive shift register, as shown in figure 5.17.

More specifically, the receiver loads data into the receive shift register so that the data has the character length specified with the RXCHR1–RXCHR0 bits of MD1, and then samples the parity or MP bit (if it exists). In the next clock cycle, the receiver samples the stop bit to complete the character assembly process. Here, the receiver shift register value is loaded into the receive buffer.

The receiver resumes searching for a start bit one clock cycle after the completion of the character assembly process.

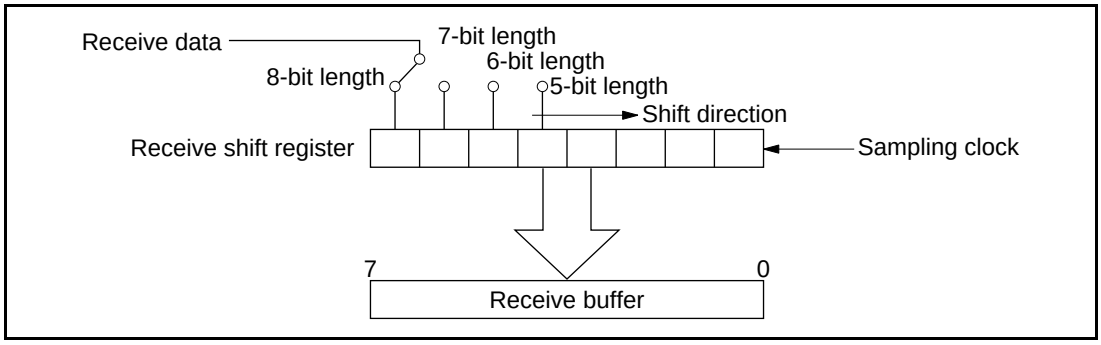


Figure 5.17 Character Assembly in the Receive Shift Register

Similarly in 1/16, 1/32, or 1/64 clock mode, the receiver searches for a start bit by sampling the line level at the rising edge of each clock pulse. On detecting a space (low level), the receiver waits for half a bit cycle, and samples the line level again to verify that the line remains low. If the line remains low, the receiver starts character assembly after a delay of one bit cycle. If the line is high, the receiver resumes start bit searching, interpreting the previously detected space as noise (figures 5.18 (a) and (b)).

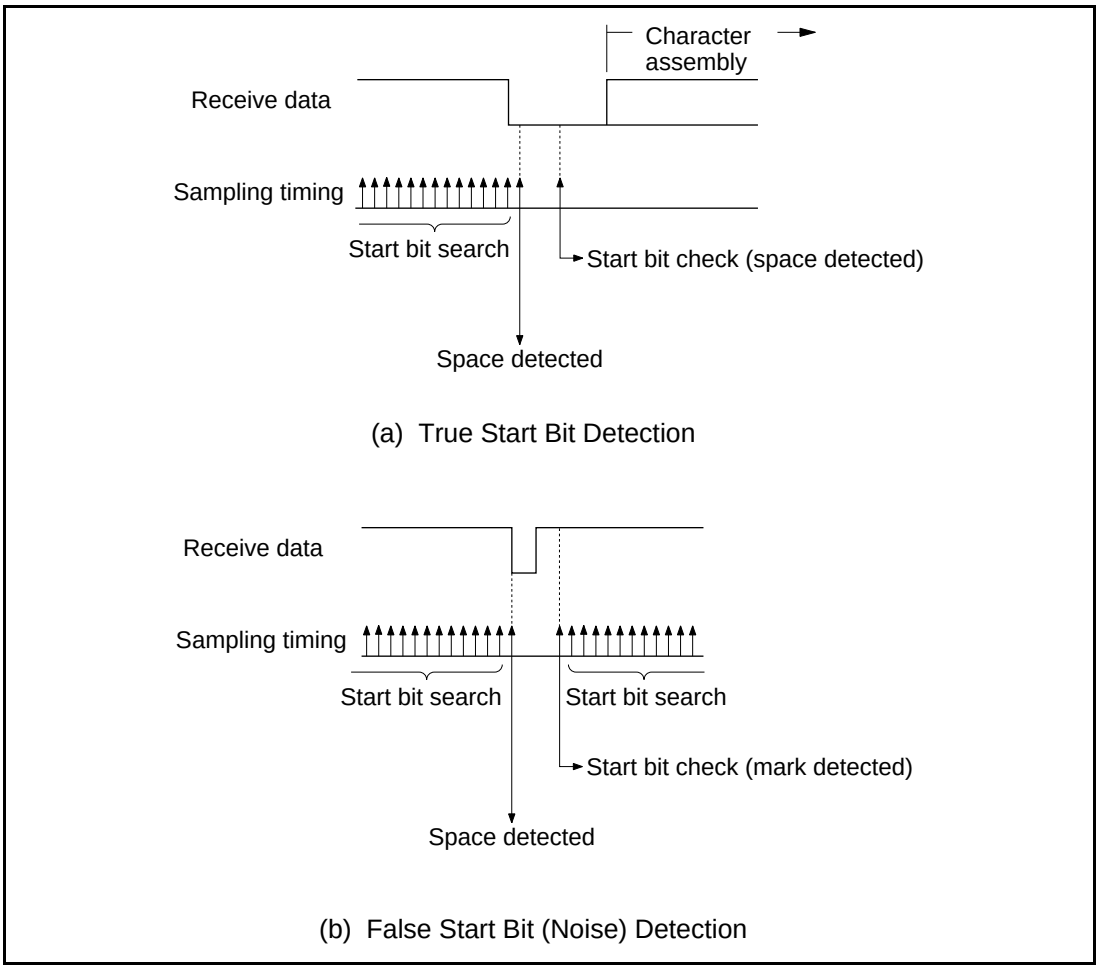


Figure 5.18 Start Bit Sampling

In the character assembly process, the receiver samples data every other bit cycle. On detecting the most significant bit (MSB) or the parity bit (if present), the receiver checks the stop bit after a delay of one bit cycle. If the RXD line is high (normal), the receiver begins start bit searching immediately. If the line is low (framing error), the receiver begins start bit searching after a delay of half a bit cycle.

In 1/16, 1/32, or 1/64 clock mode, the noise suppression function operates for start bit, character, parity bit, and stop bit sampling.

The noise suppression function operates by using the RXD line level that occurs in two of three sampling timings, which are the current sampling timing and the two preceding sampling timings (figure 5.19).

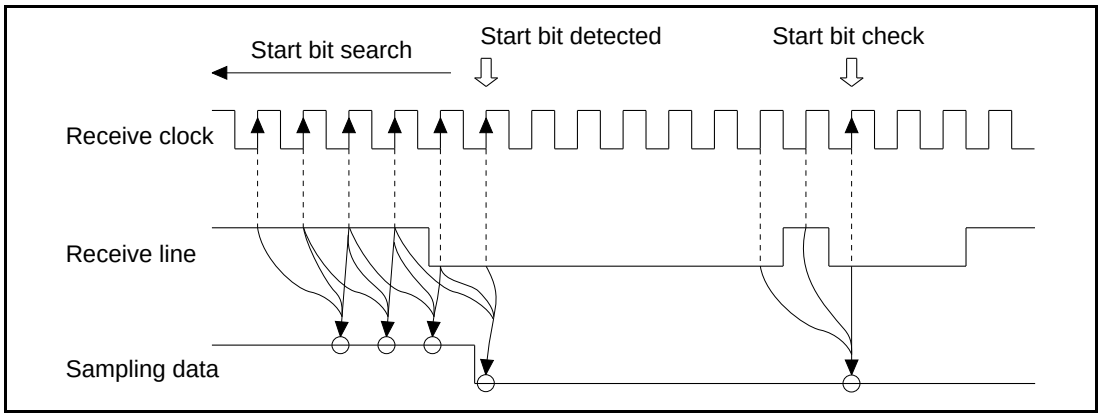


Figure 5.19 Noise Suppression Function

In asynchronous mode, the receivable character length is 8 to 5 bits, which is specified with the RXCHR1–RXCHR0 bits of MD1.

Figure 5.20 shows the receive data format. When the character length is 7 to 5 bits, the high-order bits are padded with 0s.

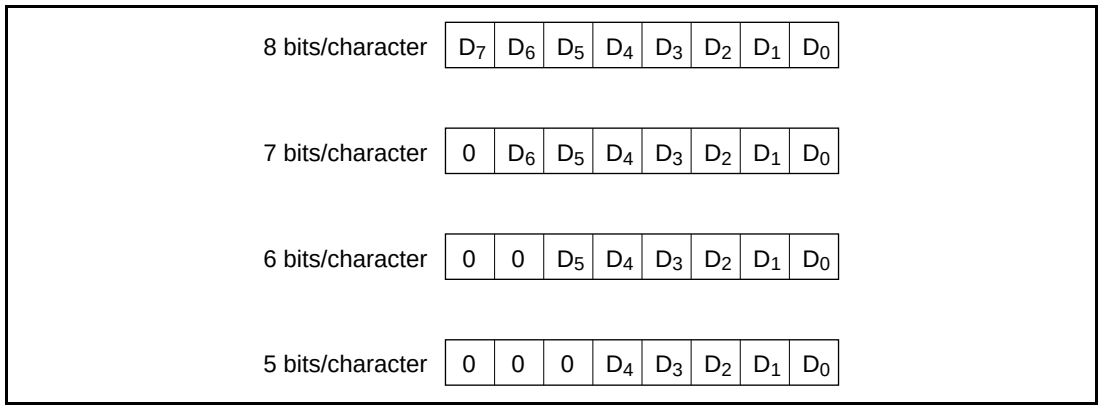


Figure 5.20 Receive Character Format

Parity/MP Bit: An even or odd parity bit or an MP bit can be appended to transmit/receive characters, as specified with the PMPM1–PMPM0 bits of MD1.

When an even parity bit is specified, the transmitter counts the number of 1s in the transmit character and appends a 0 if the number is even or a 1 if the number is odd. In this way, the total number of 1s actually transmitted should be even. The receiver checks whether or not the total number of 1s in the received character and parity bit is even.

Similarly, when an odd parity bit is specified, the value of the parity bit is set so that the total number of 1s transmitted should be odd.

When the MP bit is specified, an MP bit is appended to transmit and receive characters to enable multiprocessor communication support.

For details, see Multiprocessor Support.

Error Checking: Involves the following items.

- Parity check

  Received data is checked to see whether it has the proper parity.

  When even parity is specified and the total number of 1s in the received character and the parity bit is odd, the PE (parity error) bit of status register 2 (ST2) is set to 1 when the received data containing the parity error is ready to be read. The situation for odd parity is the same except that an even number of 1s triggers the error.

  For details on the PE bit, see section 5.2.11, MSCI Status Register 2 (ST2).

  Even if a parity error occurs, the data is received normally. However, the PE bit cannot be cleared even if the subsequent data causes no parity error. It can be cleared only when a 1 is written to the bit position or ST2 is reset.

  When the PE bit is set to 1, an interrupt request is generated (if enabled).
- Framing error

A space detected where a stop bit should be causes a framing error. The FRME bit of ST2 is set to 1 when the received data containing a framing error is ready to be read. Here, an interrupt request is generated (if enabled). Even if the stop bit length is 1.5 or 2 bits, only the first bit is checked.

For details on the FRME bit, see section 5.2.11, MSCI Status Register 2 (ST2).

A framing error does not stop the reception operation. In 1/1 clock mode, start bit searching resumes at the clock cycle following detection of the framing error. In 1/16, 1/32, or 1/64 clock mode, searching resumes after a delay of a half-bit cycle that skips invalid stop bit(s).

Once the FRME bit is set to 1 by a framing error, it cannot be cleared even if subsequent data causes no framing error. It can be cleared only when a 1 is written to the bit position or ST2 is reset.

- Overrun error

  An overrun error occurs when the receive buffer is full when new data is transferred.

  When an overrun error occurs, the new data overwrites the bottom stage of the receive buffer, erasing the previous data. At the same time, the bottom stage of the receive status FIFO is overwritten with the status (indicating an overrun) of the new data.

  The OVRN bit of ST2 is set to 1 when the overwritten data is ready to be read. Here, an interrupt request is generated (if enabled). For details on the OVRN bit, see section 5.2.11, MSCI Status Register 2 (ST2).

  Even if an overrun error occurs, subsequent data is received normally. However, the OVRN bit cannot be cleared even if the subsequent data causes no overrun error. It can be cleared only when a 1 is written to the bit position or ST2 is reset.

Break Transmission and Detection: When the transmitter must suspend data transmission, it transmits a break (space or low level).

Normally, the transmitter issues a break transmission request after completing the current character transmission. The transmitter must continue to send the break signal for one or more character cycles. Break transmission is controlled by the BRK bit of CTL. When this bit is set to 1, the TXD line goes low at the falling edge of the next transmit clock pulse. When this bit is cleared, the TXD line goes high at the falling edge of the next transmit clock pulse to cancel break transmission. At break transmission cancellation, the transmitter guarantees one or more bit cycles of high level to next start bit.

When break transmission is requested, the output data in the transmit shift register is lost, but the transmit buffer is not affected.

The receiver detects a break in the following procedure:

On receiving data whose data bits and parity bits are all 0s and contain a framing error, the receiver assumes it to be the start of a break, and sets the BRKD bit of ST1 to 1. Here, the null character containing the framing error is discarded (not transferred to the receive buffer).

If break transmission starts during character transmission, the break transmission must continue for two or more character cycles. The reason for this can be seen from figure 5.21, which shows break detection by the receiver.

On detecting a mark (high level) of a half-bit cycle or longer after detecting the start of a break, the receiver assumes it to be the end of a break, and sets the BRKE bit of ST1 to 1. (In 1/1 clock mode, detection of the first mark causes the break to end.)

When the BRKD or BRKE bit is set to 1, an interrupt request is generated (if enabled).

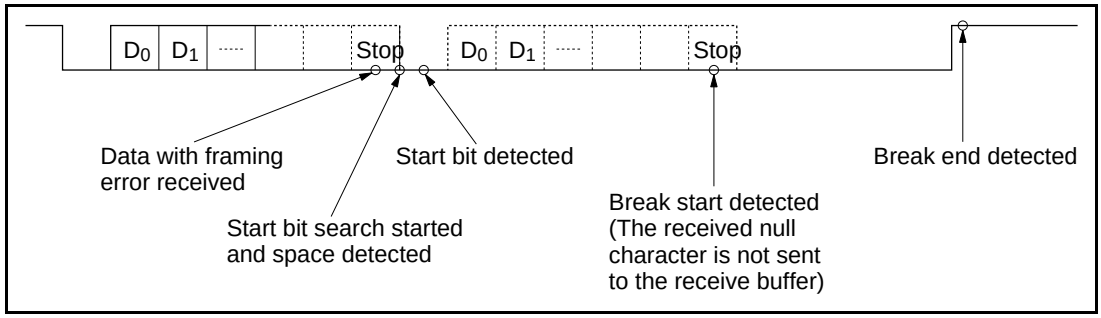


Figure 5.21 Break Detection by the Receiver

The transmitter generally transmits a break using the following procedure:

- Waits for the end of transmission (idle status)
- Writes a 1 to the BRK bit
- Waits one or more character cycles
- Writes a 0 to the BRK bit

Multiprocessor Support: The MSCI supports a function which specifies whether a specific terminal should receive or ignore data in character units. This is useful for communications between multiple terminals.

In multiprocessor mode, a character format is used in which an MP bit is appended instead of a parity bit. This format can be specified by the PMPM1–PMPM0 bits of MD1.

In multiprocessor mode, data should usually be transmitted with the MP bit set to 0. However, the user can set the MP bit to 1 by issuing an MP bit on command immediately before transferring the transmit data to the transmit buffer. This command is valid for only one data transmission after it is issued.

On the receiver side, the MP bit in the receive data is transferred to the receive buffer together with other status information. When the receive data is ready to be read, the value of the MP bit is set to ST2. Data whose MP bit = 0 can be ignored (not transferred to the receive buffer) by issuing a search MP bit command. This command is invalidated when data whose MP bit = 1 is received, and subsequent data is received in the normal manner.

For information on the MP bit on command and search MP bit command, see section 5.2.8, MSCI Command Register (CMD).

Figure 5.22 shows communications between multiprocessors by using the MP bit.

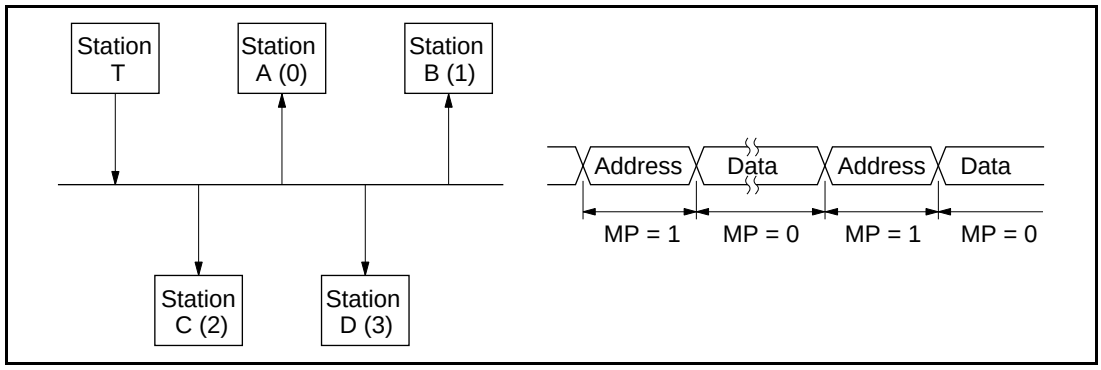


Figure 5.22 MP Bit Operation Example

In figure 5.22, T is a transmit station, and A, B, C, and D are receive stations. Receive stations A, B, C, and D are assigned addresses 0, 1, 2, and 3, respectively.

For transmitting data from T to B, transmit station T sends address (1), with MP bit set to 1, to the communication path. The receive stations all monitor the communications path. When they receive data with the MP bit = 1, they assume the data is a station address and compare it with their own address. In this example, the received data matches the address of station B. Station B now assumes that it is the destination of subsequent data with the MP bit = 0. Other receive stations A, C, and D, issue a search MP bit command and ignore the data with the MP bit = 0. Thus, the transmit station can send data to a specific receive station by transmitting the destination address, with the MP bit set to 1, and then transmitting the data, with the MP bit set to 0.

If the transmit station wants to send data to a different receive station, it transmits the new station address, with the MP bit set to 1, to clear the search MP bit command. The transmit station can use the same procedure as described above to communicate with any desired receive station.

5.3.2Byte Synchronous Mode

In byte synchronous mode, synchronization of characters is accomplished by adding a SYN character pattern to the beginning of the transmit or receive data.

MSCI byte synchronous mode supports mono-sync, bi-sync, and external synchronous modes. In mono- or bi-sync mode, a one- or two-byte SYN character pattern is added for synchronization, respectively. In external synchronous mode, the $\overline{\text{SYNC}}$ line is activated for synchronization. Byte synchronous mode is specified with the PRTCL2–PRTCL0 bits of mode register 0 (MD0).

The character format for byte synchronous mode is shown in figure 5.23.

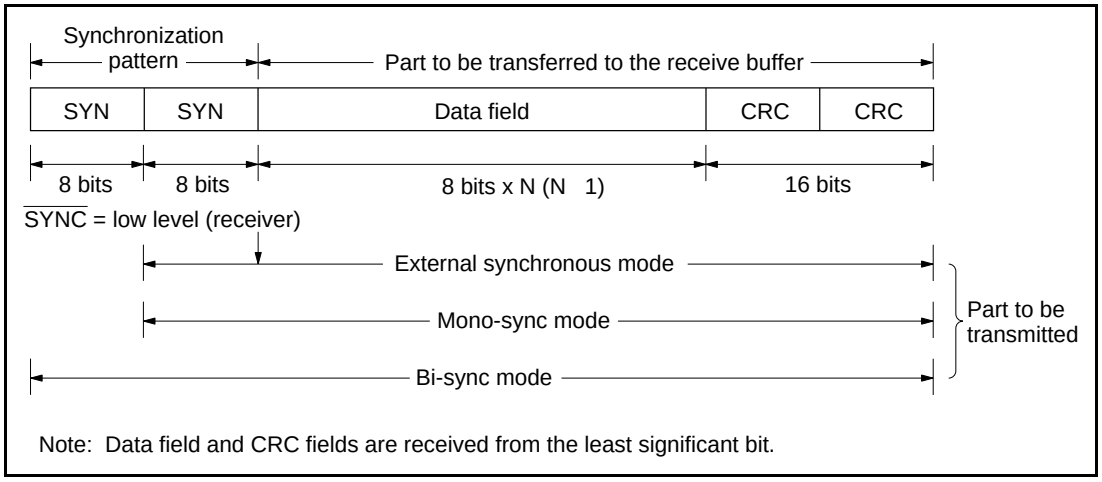


Figure 5.23 Character Format for Byte Synchronous Mode

The SYN character pattern lengths for transmission and reception in byte synchronous mode are listed in table 5.12.

Table 5.12 SYN Character Pattern Length in Byte Synchronous Mode

Synchronous Mode	For Transmission	For Reception
Mono-sync	1 byte	1 byte
Bi-sync	2 bytes	2 bytes
External synchronous	1 byte	$\overline{\text{SYNC}}$ line used for synchronization

The SYN character pattern is specified by synchronous/address registers 0 and 1 (SA0 and SA1).

To transmit a header preceding the SYN character pattern, write the header pattern to the idle pattern register (IDL), and delay data write to the transmit buffer. The transmitter keeps transmitting the header until data is written to the transmit buffer. (For details, see section 5.2.4, MSCI Control Register (CTL), section 5.2.18, MSCI Synchronous/Address Register 0 (SA0), section 5.2.19, MSCI Synchronous/Address Register 1 (SA1), and section 5.2.20, MSCI Idle Pattern Register (IDL).)

The receiver does not reestablish synchronization of the received data using SYN characters in the data field. The SYN characters in the data field are automatically deleted or loaded into the receive buffer according to the setting of the SYNCLD bit of CTL. (For details, see section 5.2.4, MSCI Control Register (CTL).)

Transmission Operation: Figure 5.24 is the state transition diagram for transmission in byte synchronous mode.

- TX disable state

  The transmitter is placed in TX disable state by a hardware reset, a channel reset, or a TX reset command. It is also placed in TX disable state when no data remains in the transmit buffer after a TX disable command is issued. In this state, the TXD line is high (mark), and the TXRDY bit of status register 0 (ST0) is cleared.
- Idle state

  The TX enable command sets the transmitter in the idle state from the TX disable state. In the idle state, the transmitter behaves according to the value of the IDLC bit of CTL: a high level (mark) is transmitted when IDLC is 0, or the contents of IDL are transmitted when IDLC is 1, via the TXD line. When the transmit data is written, the transmitter enters SYN1 transmit state.
- SYN1 transmit state

  The transmitter transmits the SYN character pattern set in SA1, and enters character transmit state in mono-sync or external synchronous mode, or SYN2 transmit state in bi-sync mode. (For details, see section 5.2.18, MSCI Synchronous/Address Register 0 (SA0), and section 5.2.19, MSCI Synchronous/Address Register 1 (SA1).)
- SYN2 transmit state

  When in bi-sync mode, the transmitter transmits the SYN character pattern in SA0 and enters character transmit state. The transmitter does not enter character transmit state when in mono-sync or external synchronous mode.
- Character transmit state

  The transmitter transmits data in FIFO order from the transmit buffer via the TXD line.
- CRC transmit state

The transmitter transmits a 16-bit CRC code. If data remains in the transmit buffer after transmission, the transmitter enters SYN1 wait state. If no data remains, it enters idle state. (CRC code is specified with the CRC1–CRC0 bits of MD0. Whether to perform CRC calculation and to send the result is specified with the CRCCC bit of MD0. For details, see section 5.2.1, MSCI Mode Register 0 (MD0).)

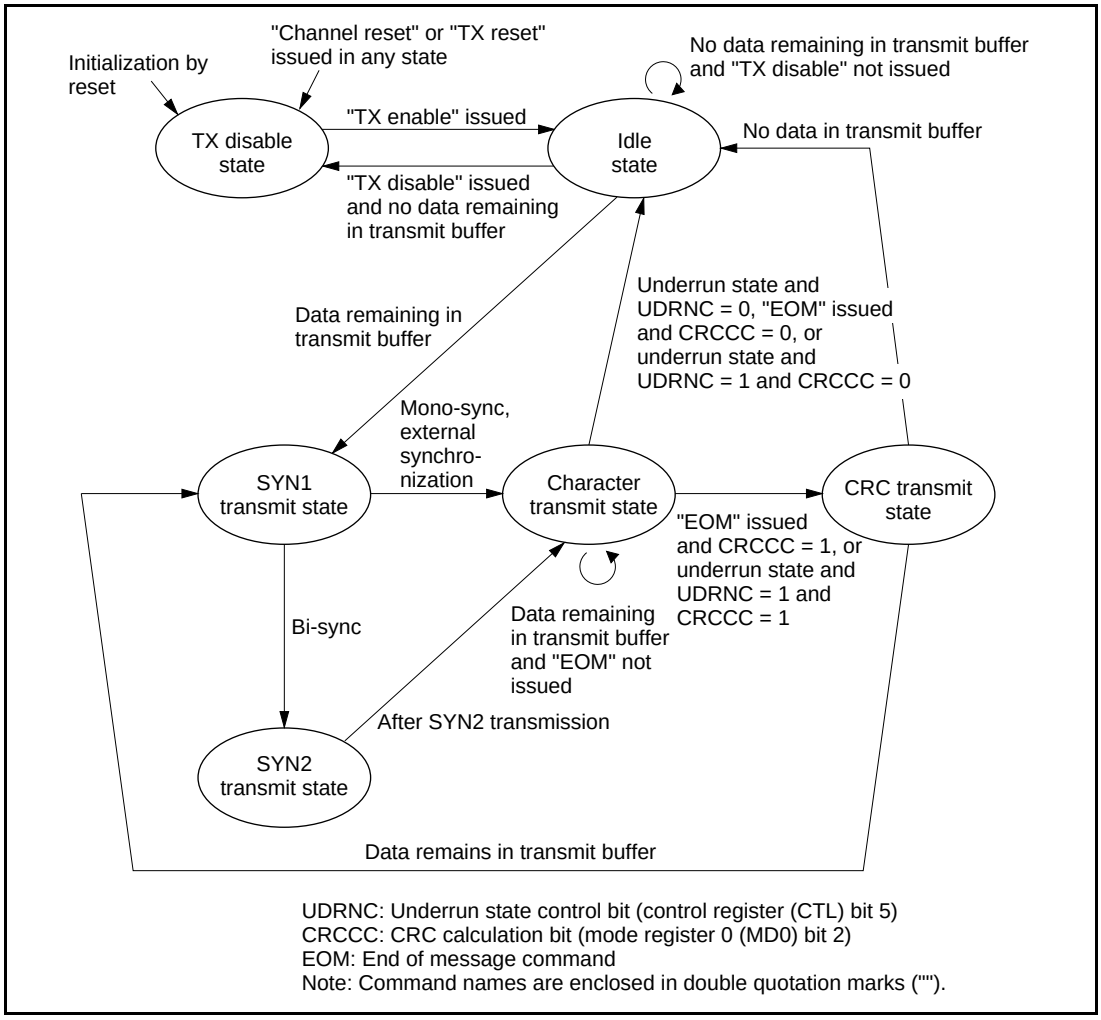


Figure 5.24 State Transition Diagram for Transmission in Byte Synchronous Mode

Reception Operation: Figure 5.25 is the state transition diagram for reception in byte synchronous mode.

- RX disable state

  The receiver is placed in RX disable state by a hardware reset, a channel reset, an RX reset, or an RX disable command. In this state, the receiver ignores the input from the RXD line and does not performs a reception operation.

- SYN1 wait state

  The receiver waits for the first byte of the SYN character pattern to establish a character boundary. If the received data matches the SYN character pattern set in SA0, the receiver enters character reception state in mono-sync mode or SYN2 wait state in bi-sync mode. In external synchronous mode, synchronization is established by the $\overline{\text{SYNC}}$ line input.

- SYN2 wait state

  The receiver waits for the second byte of the SYN character pattern in bisync mode only. If the received data matches the SYN pattern set in SA1, the receiver enters character receive state. If it does not match, the receiver enters SYN1 wait state. The receiver does not enter SYN1 wait state when in mono-sync or external synchronous mode.

- Character receive state

  The receiver transmits the received character to the receive buffer. The SYN character(s) in the data field may be optionally transmitted to the receive buffer, as specified with the SYNCLD bit of CTL. The receiver is placed in SYN1 wait state when a message reject command is issued.

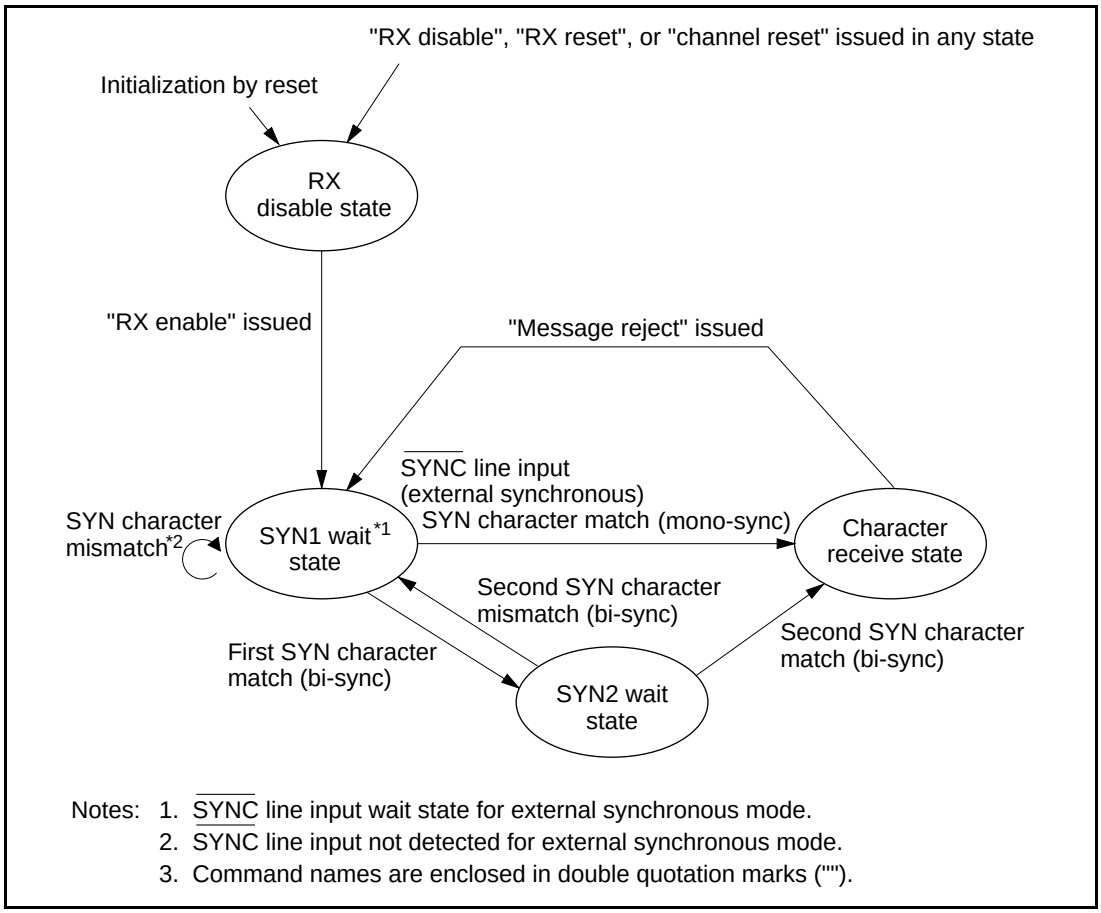


Figure 5.25 State Transition Diagram for Byte Synchronous Mode Reception

Error Checking: Involves the following items.

- CRC error and CRC code transmission

  The MSCI supports two CRC code types: CRC-16 and CRC-CCITT. The program can select the type to be used and the initial value (all 0s or 1s) with the CRC1–CRC0 bits of MD0.

  The CRC polynomial is $X^{16} + X^{15} + X^2 + 1$ for CRC-16, and $X^{16} + X^{12} + X^5 + 1$ for CRC-CCITT.

  The transmitter and receiver both have a CRC calculator.

  The TX CRC calculator is automatically initialized immediately before data field transmission. It can also be initialized when a TX CRC initialization command is issued.

  During transmission, SYN characters are excluded from the CRC calculation. Specific data can also be excluded (in character units) from the CRC calculation by a TX CRC calculation exclusion command. Use the CRCCC bit of MD0 and the end of message command to enable CRC code transmission. The CRC code is transmitted automatically when both the CRCCC bit and the UDRNC bit of CTL are set to 1 in underrun state. If an underrun error occurs while UDRNC or CRCCC is 0, the MSCI directly enters idle state without transmitting the CRC code. For details, see section 5.2.1, MSCI Mode Register 0 (MD0), section 5.2.4, MSCI Control Register (CTL), and section 5.2.8, MSCI Command Register (CMD).

  The RX CRC calculator is automatically initialized immediately before data field reception. It can also be initialized when an RX CRC initialization command is issued.

  During reception, the characters not to be input to the receive buffer, such as SYN characters, are excluded from the CRC calculation. Specific data can be also excluded (by character) from the CRC calculation by an RX CRC calculation exclusion command. The CRC code check is completed 15 system clock cycles after the character following the last checked character has entered the receive buffer. If a CRC calculation forcing command is issued and if the character following the last character does not enter the receive buffer, the check is completed 15 system clock cycles after the command has been issued. In either case, the CRC error status is valid until the next character enters the receive buffer.

  When a CRC error is detected, the CRCE bit of status register 2 (ST2) is set to 1. This generates an interrupt request (if enabled). For details, see section 5.2.11, MSCI Status Register 2 (ST2).

- Overrun error

  An overrun error occurs when the receive buffer is full when new data is transferred.

  When an overrun error occurs, the new data overwrites the top stage of the receive buffer, erasing the previous data. At the same time, the top stage of the status FIFO is overwritten with the status (indicating an overrun) of the new data.

  The OVRN bit of ST2 is set to 1 when the new data is ready to be read. This generates an interrupt request (if enabled).

  Even if an overrun error occurs, subsequent characters are received normally. However, the OVRN bit is not cleared even if the subsequent data causes no overrun error. It is cleared only when a 1 is written to the bit position or ST2 is reset.

- Underrun error

  An underrun error occurs when the transmit buffer is empty after data has been sent from the transmit shift register.

  When an underrun error occurs, the transmitter enters idle state. (The MSCI assumes an underrun error when the transmit shift register and transmit buffer are both empty and an end of message command has not been issued.) The transmitter then drives the TXD line high (when the IDLC bit of CTL is 0), or outputs an idle pattern (when the IDLC bit is 1). Here, the transmitter can transmit the CRC code before entering idle state if the UDRNC bit of CTL is set to 1. If an underrun error occurs when UDRNC or CRCCC is 0, the transmitter directly enters idle state without transmitting the CRC code.

  After entering idle state, the transmitter enters SYN1 transmit state when the UDRN bit is cleared and when data is written to the transmit buffer.

  When an underrun error occurs, the UDRN bit of ST1 is set to 1, and the TXRDY bit of ST0 is cleared. This generates an interrupt request (if enabled). The UDRN bit is cleared only when a 1 is written to the bit position or ST1 is reset.

Message End Operation: During transmission, the MSCI recognizes the end of message when it executes an end of message command. Also, the MSCI automatically assumes an end of message when an underrun error occurs while the UDRNC bit of CTL is 1.

When the CRCCC bit of MD0 is 1 when the message transmission is completed, the transmitter automatically transmits a CRC code and then enters idle state. When the CRCCC bit is 0 when the message transmission is completed, the transmitter enters idle state without CRC code transmission, and an interrupt request is generated (if enabled).

During reception, the receiver does not detect the end of message.

5.3.3 Bit Synchronous Mode

In bit synchronous mode, communication is performed using flags added to frame boundaries. This mode is specified with the PRTCL2–PRTCL0 bits of mode register 0 (MD0).

The message format for bit synchronous mode is shown in figure 5.26. The address (A) and control and information (C and I) fields are configured in byte units and are sent to the receive buffer. Except the frame check sequence (FCS) field, data is transmitted or received beginning with the least significant bit. The FCS field data is transmitted and received beginning with the most significant bit.

Residual bit frames cannot be transmitted. During reception, when residual bits exist at the end of receive data, the valid bits (residual bits) in the last character are justified to the high-order bit positions, generating undefined low-order bits. The undefined bits cannot be distinguished from valid bits. When a residual bit frame is received, the status of the last character indicates both the residual bit frame and end of receive frame. (This status is indicated by the EOM and RBIT bits of status register 2 (ST2).)

In bit synchronous mode, frame boundaries can only be detected by the flag pattern "01111110." To prevent the same data pattern as the flag, the SCA inserts or deletes zeros in the data strings. This function is called "zero insertion/deletion."

During transmission, the transmitter constantly monitors data strings between the opening and closing flags. If the transmitter detects five consecutive ones in a data string in the transmit buffer, it adds a zero to the end before transmitting the data string on the TXD line. For example, if the transmitter detects in the transmit buffer "11111111" and "11011111," it inserts one zero in each data string to transmit "111011111" and "110011111," respectively.

During reception, if the receiver detects one zero after five consecutive ones in a data string received from the RXD line, it deletes the zero from the data string before storing the data in the receive buffer. If the receiver detects six or more consecutive one's, it regards the data as a flag or abort frame and so performs the specified protocol without zero deletion.

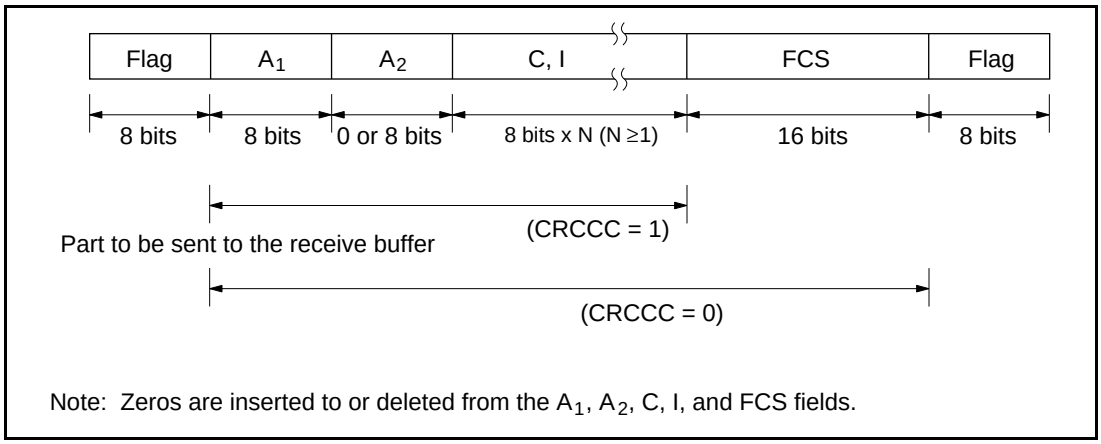


Figure 5.26 Message Format for Bit Synchronous Mode

Transmission Operation: Figure 5.27 is the state transition diagram for transmission in bit synchronous HDLC mode.

- TX disable state

  The transmitter is placed in TX disable state by a hardware reset, a channel reset, or a TX reset command. In this state, the TXD line is high (mark), and the TXRDY bit of status register 0 (ST0) is cleared.
- Idle state

  A transmit enable command sets the transmitter in idle state from TX disable state. In idle state, the transmitter behaves according to the value of the IDLC bit of the control register (CTL): repeatedly transmits high (mark) signals when IDLC is 0 or transmits the contents of the idle pattern register (IDL) when IDLC is 1, via the TXD line. When transmit data is written, the transmitter enters opening flag transmit state.
- Opening flag transmit state

  The transmitter transmits one flag and enters character transmit state.
- Character transmit state

  The transmitter sequentially transmits data from the transmit buffer.
- FCS transmit state

  The transmitter transmits FCS (CRC) data and enters the next state.
- Closing flag transmit state

  The transmitter transmits one flag and enters the next state. (When frames are sent in succession, they are automatically delimited by at least one closing flag and one opening flag.)
- Abort transmit state

  The transmitter transmits abort pattern 11111111 and enters the next state.

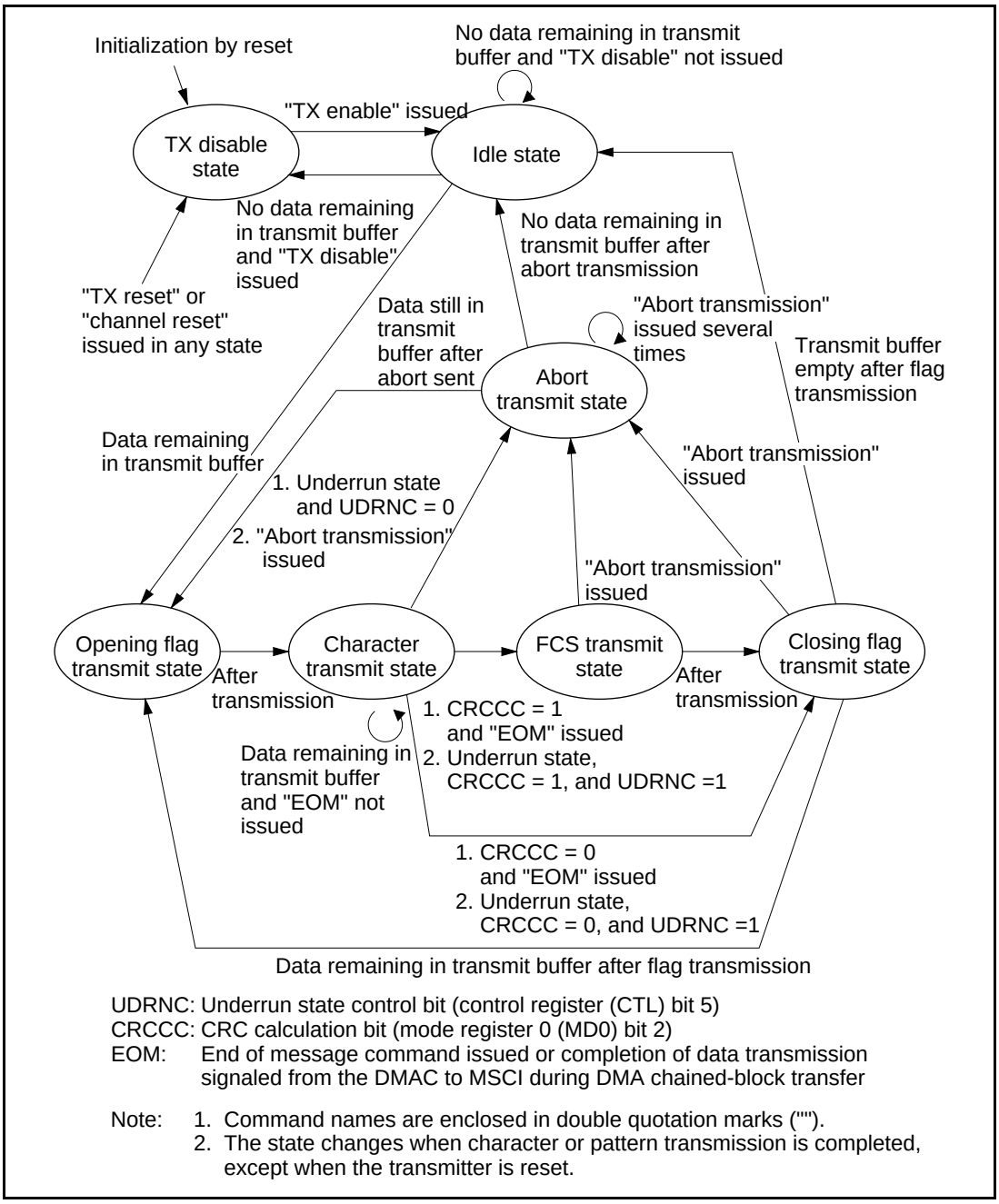


Figure 5.27 State Transition Diagram for Transmission in Bit Synchronous HDLC Mode

Reception Operation: Figure 5.28 is the state transition diagram for reception in bit synchronous mode.

- RX disable state

  The receiver is placed in RX disable state by a hardware reset, a channel reset, an RX reset, or an RX disable command. In this state, the receiver ignores the input from the RXD line, and does not perform a reception operation.

- Flag wait state

  The receiver waits for a flag pattern to compare it with the received bit string. (Successive frames which share opening and closing flags can be received normally.) On detecting a flag pattern, the receiver enters character wait state.

- Character wait state

  To detect a frame boundary, the receiver waits for a non-flag pattern while ignoring successive flags. On detecting a non-flag pattern, the receiver enters address field check state.

- Address field check state

  The receiver checks the address field to determine whether or not to receive the associated frame. When the address is identical to the present station address, the receiver enters character reception state. When the address is not identical to the present station address, the receiver enters flag wait state. In address field no-check mode, the receiver skips this check and enters character reception state immediately after character wait state. On detecting a flag within three character cycles after the address field check, the receiver assumes the received bit to be a short frame, and enters character wait state.

- Character receive state

  The receiver transmits the received character to the receive buffer. On detecting a flag in character receive state, the receiver transmits data up to and including the last character in the I field when the CRCCC bit of MD0 is 1, or transmits the FCS when the CRCCC bit is 0 to the receive buffer, and then enters character wait state.

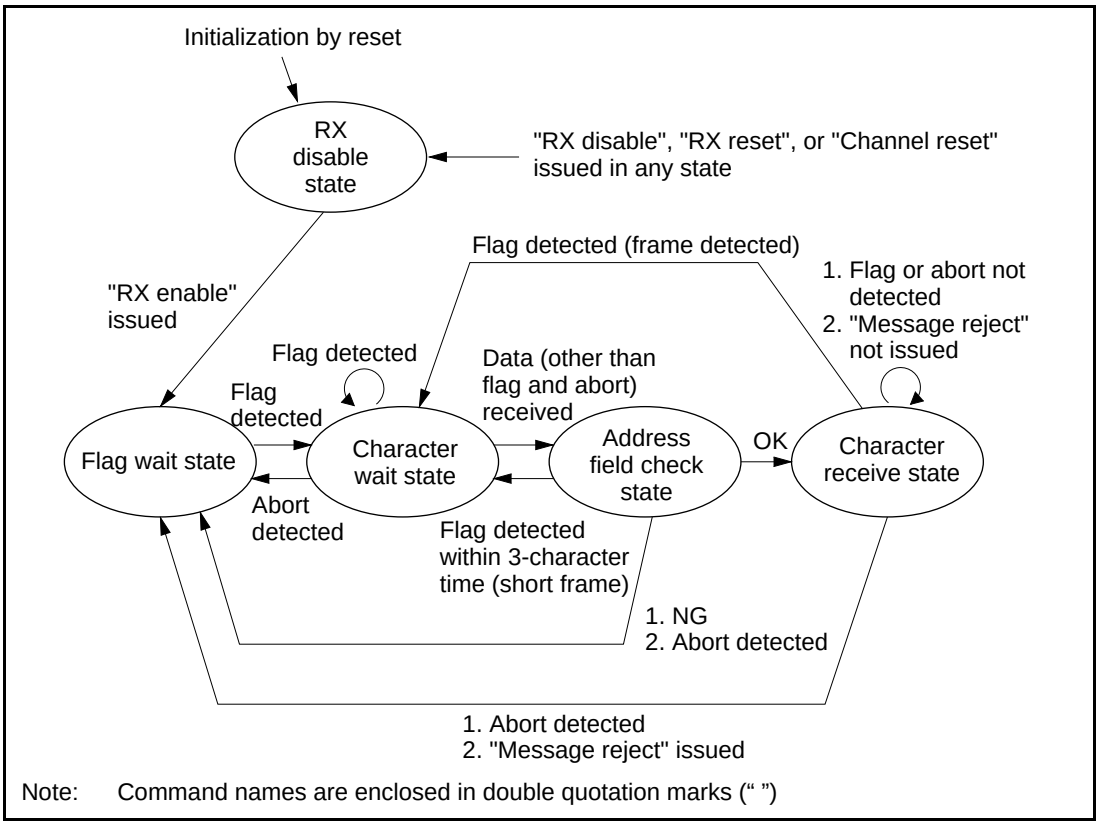


Figure 5.28 State Transition Diagram for Reception in Bit Synchronous Mode

Error Checking: Involves the following items.

- CRC errors

  In bit synchronous HDLC mode, CRC-CCITT is usually used. Its initial value is all 1s, which can be specified with the CRC1–CRC0 bits of MD0. (The CRC polynomial is $X^{16} + X^{12} + X^5 + 1$ for CRC-CCITT.)

  The transmitter and receiver both have a CRC calculator.

  The CRC calculator is automatically initialized immediately before the A field transmission or reception.

  During transmission, CRC calculation is carried out on the data in the A, C, and I fields before zero insertion.

  Use the CRCCC bit of MD0 and the end of message command to enable CRC code transmission. The CRC code is transmitted automatically when both the CRCCC bit and the UDRNC bit of CTL are set to 1 in underrun state. For details, see section 5.2.1, MSCI Mode Register 0 (MD0), section 5.2.4, MSCI Control Register (CTL), and section 5.2.8, MSCI Command Register (CMD).

During reception, CRC calculation is carried out on the 0-deleted data in the A, C, and I fields. The CRC code check is completed when the last character in the I field enters the receive buffer with the CRCCC bit of MD0 = 1. The error status is sent to the status FIFO associated with the character before being set to the CRCE bit of ST2. When the CRCE bit is set to 1, an interrupt request is generated (if enabled). When the CRCCC bit is 0, the CRCE bit is not set to 1.

- Overrun error

  An overrun error occurs when the receive buffer is full when new data is transferred.

  When an overrun error occurs, the new data overwrites the top stage of the receive buffer, erasing the previous data. At the same time, the top stage of the status FIFO is overwritten with the status (indicating an overrun) of the new data. The EOM bit is cleared as well when the new data is written.

  The OVRN bit of ST2 is set to 1 when the new data becomes ready to be read. This generates an interrupt request is generated (if enabled).

  Even if an overrun error occurs, subsequent characters are received normally. However, the OVRN bit is not cleared even if the subsequent data causes no overrun error. It can be cleared only when a 1 is written to the bit position or ST2 is reset.

- Underrun error

  An underrun error occurs when the transmit buffer is empty after data has been sent from the transmit shift register.

  When an underrun error occurs and abort transmission is enabled by the UDRNC bit of CTL, the transmitter enters idle state after sending an abort. (The MSCI assumes an underrun error when the transmit shift register and transmit buffer are both empty and an end of message command has not been issued.) In other cases, the MSCI assumes that an underrun error is an end of message and terminates the frame normally. Thus, the MSCI enters idle state after sending FCS and a flag.

  The UDRN bit of ST1 is set to 1 when an underrun error occurs. In this case, the transmit buffer is not full, but the TXRDY bit of ST0 is not set to 1 as long as the UDRN bit remains 1. This prevents the remaining data from being transmitted as a normal frame when an underrun occurs during DMA transfer.

  When the UDRN bit is set to 1, an interrupt request is generated (if enabled).

Message End Operation: During transmission, the MSCI recognizes the end of message when it executes an end of message command. Also, the MSCI automatically assumes an end of message for either a completed DMA-chained block transfer or an underrun error occurring when the UDRNC bit of CTL is 1.

The last character to be transmitted is the first character written to the transmit buffer after an end of message command is issued; it is the last character transferred in DMA-chained block transfer, and is the character transmitted immediately before the underrun in underrun state.

At message transmission completion, the MSCI enters closing flag transmit state when the CRCCC bit of MD0 is 0, or enters FCS transmit state when the CRCCC bit is 1.

During reception, the MSCI assumes a flag detected in character receive state to be the end of message.

When the CRCCC bit of MD0 is 1, characters up to and including the last character in the I field are sent to the receive buffer, and FCS is deleted. The receive frame end status and CRC error status associated with the last character are sent to the status FIFO and set to the EOM bit and CRCE bit of ST2 when the last character becomes ready to be read. At the same time, the internal DMAC is informed of the end of a frame, and an interrupt request is generated (if enabled).

When the CRCCC bit is 0, FCS is also sent to the receive buffer. In this case, its associated receive frame end status is transferred to the status FIFO. To enable this control, characters are sent to the receive buffer three character cycles after they are received. Thus, the last character in the I field and the FCS have not yet been sent to the receive buffer when the MSCI detects a closing flag.

Address Field Check: In bit synchronous mode, each data frame contains an address (A) field which specifies what secondary station(s) should receive the frame. The MSCI supports four address field check modes: address field no-check, single address 1, single address 2, and dual address (table 5.13).

Table 5.13 Address Field Check

Mode	Function
Address field no-check	Allows the MSCI to receive all frames
Single address 1	Allows the MSCI to receive only the frames whose A_1 field has the specified value or global address (FFH)
Single address 2	Allows the MSCI to receive only the frames whose A_2 field has the specified value or global address (FFH)
Dual address	Allows the MSCI to receive only the frames whose A_1 and A_2 fields have the specified values, global addresses (FFFFH), or group addresses (A_2 = specified value, A_1 = FFH)

The ADDRS1–ADDRS0 bits of MD1 select an address field check mode, and synchronous/ address registers 0 and 1 (SA0 and SA1) specify the address. For details, see section 5.2.2, MSCI Mode Register 1 (MD1), section 5.2.18, MSCI Synchronous/Address Register 0 (SA0), and section 5 2.19, MSCI Synchronous/Address Register 1 (SA1).

Short Frame Detection: On detecting a short frame, the MSCI acts according to the frame length, CRCCC bit value of MD0, and address field check mode as shown in table 5.14.

Table 5.14 MSCI Actions at Short Frame Detection

	Mode Settings			
	CRCCC Bit = 0		CRCCC Bit = 1	
Frame Length (excluding flag)	**Address Field No-Check Single Address 1**	**Single Address 2 Dual Address**	**Address Field No-Check Single Address 1**	**Single Address 2 Dual Address**
Bits 1–8	Sends no data to the receive buffer.	Sends no data to the receive buffer.	Sends no data to the receive buffer.	Sends no data to the receive buffer.
Bits 9–23	Sends a part of the data to the receive buffer. Appends the short frame status to the last character and sets the SHRT bit of ST2.	Sends a part of the data to the receive buffer. Appends the short frame status to the last character and sets the SHRT bit of ST2.	Sends no data to the receive buffer.	Sends no data to the receive buffer.
Bits 24–31	Sends a part of the data to the receive buffer. Appends the short frame status to the last character and sets the SHRT bit of ST2.	Sends a part of the data to the receive buffer. Appends the short frame status to the last character and sets the SHRT bit of ST2.	Sends a part of the data to the receive buffer. Appends the short frame status to the last character and sets the SHRT bit of ST2.	Sends a part of the data to the receive buffer. Appends the short frame status to the last character and sets the SHRT bit of ST2.
Bits 32–39	Receives the data as normal data.	Sends a part of the data to the receive buffer. Appends the short frame status to the last character and sets the SHRT bit of ST2.	Receives the data as normal data.	Sends a part of the data to the receive buffer. Appends the short frame status to the last character and sets the SHRT bit of ST2.
Bits 40 or more	Receives the data as normal data.	Receives the data as normal data.	Receives the data as normal data.	Receives the data as normal data.

Note: On detecting a short frame, the MSCI sets the SHRT bit of ST2 to 1. This automatically sets the EOM bit to 1, indicating the end of a receive frame. At this time, an interrupt request is generated (if enabled). Even if the MSCI detects a short frame, it does not set the SHRT bit to 1, if data is not transferred to the receive buffer.

Abort Transmission and Reception: During transmission, the MSCI enables abort transmission when an abort transmit command is issued. If abort transmission is enabled using the UDRNC bit of CTL (UDRNC = 0), the MSCI transmitter automatically enters abort transmit state when an underrun occurs.

This state causes the transmitter to transmit an abort pattern (eight 1s) to clear the transmit buffer. Thus, the contents of the transmit shift register and transmit buffer are lost. After transmitting the abort pattern, the MSCI enters idle state.

During reception, the MSCI assumes 01111111 (0 followed by seven 1s) as an abort. On detecting an abort, the receiver enters flag wait state, generating an interrupt request (if enabled).

If the receiver is in character receive state when the abort is detected, it performs the following additional operation:

When the CRCCC bit of MD0 is 0, data up to the position preceding 01111111 is sent to the receive buffer. When the CRCCC bit is 1, data up to the character being assembled at detection is sent to the receive buffer, and 16 bits of data preceding 01111111 are truncated. (This operation is the same as for receive frame end on flag detection, except that the ABT bit of ST2 is set to 1.)

5.4 Transmit/Receive Clock Sources

5.4.1 Overview

The MSCI transmit and receive clock sources are selected from the following:

- Transmit clock sources:
  - — TXC line input
  - — Transmit baud rate generator output
  - — Receive clock

  The transmit clock source is selected by the TXCS2–TXCS0 bits of the TX clock source register (TXS).
- Receive clock sources
  - — RXC line input
  - — Receive baud rate generator output
  - — RXC line input with noise suppressed by the ADPLL (the ADPLL operating clock = receive baud rate generator output)
  - — Clock extracted from the receive data by the ADPLL (the ADPLL operating clock = RXC line input or the receive baud rate generator output)

  The receive clock source is selected by the RXCS2–RXCS0 bits of the RX clock source register (RXS).

The internal baud rate generator (BRG) can provide independent outputs for transmission and reception by dividing the system clock. The internal ADPLL can extract clock from the receive data; suppress noise in the receive data; and suppress noise in the receive clock.

The ADPLL employs the receive BRG output or RXC line input for both clock extraction and noise suppression. The ADPLL uses the receive BRG output for receive clock noise suppression.

The MSCI clock sources are shown in figure 5.29.

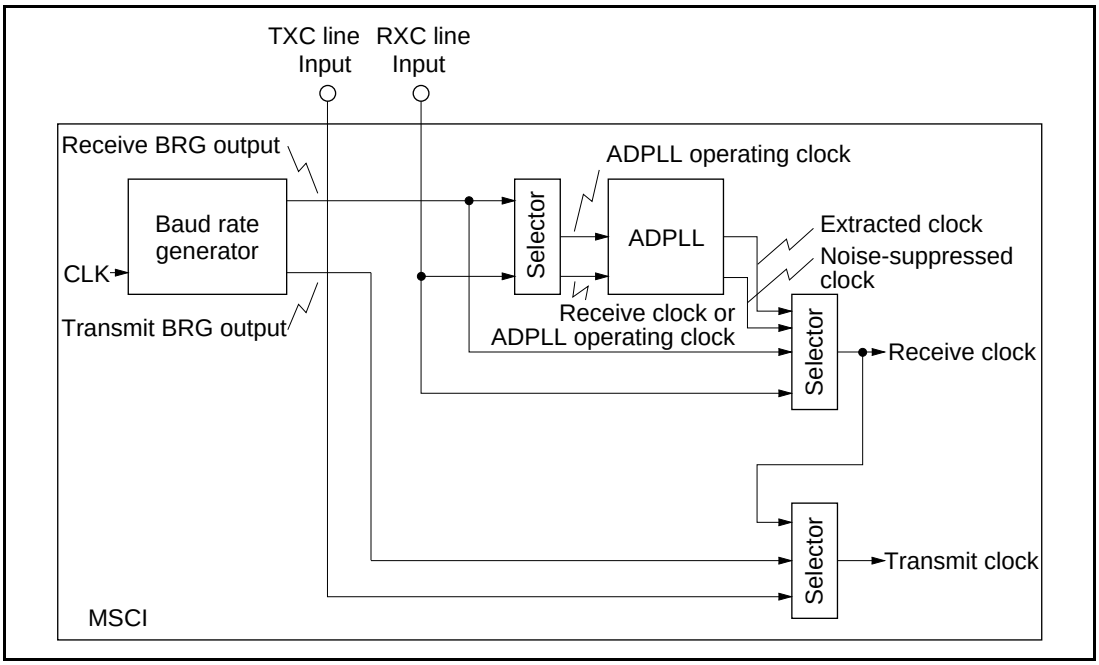


Figure 5.29 Selecting Transmit and Receive Clock Sources

5.4.2 Transmit Clock Sources

The transmit clock sources are shown in figure 5.30. When the transmit baud rate generator output serves as the transmit clock, the TXC line functions as the transmit clock output line.

The receive clock is used as the transmit clock when the clock is extracted by the ADPLL or is in bit synchronous loop mode.

In asynchronous mode, the actual bit rate is determined by the clock mode (1/1, 1/16, 1/32, or 1/64). In byte or bit synchronous mode, 1/1 clock mode is automatically selected. For details, see section 5.2.2, MSCI Mode Register 1 (MD1).

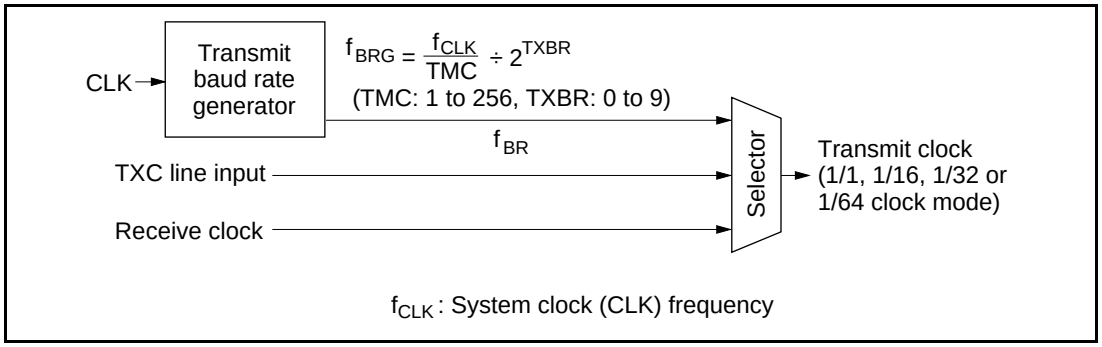


Figure 5.30 Transmit Clock Sources

5.4.3 Receive Clock Sources

The receive clock sources are shown in figures 5.31 (a), (b), and (c). When the RXC signal is not used as a clock source, the RXC line functions as the receive clock output line.

In asynchronous mode, the actual bit rate is determined by the clock mode (1/1, 1/16, 1/32, or 1/64). In byte or bit synchronous mode, 1/1 clock mode is automatically selected. For details, see section 5.2.2, MSCI Mode Register 1 (MD1).

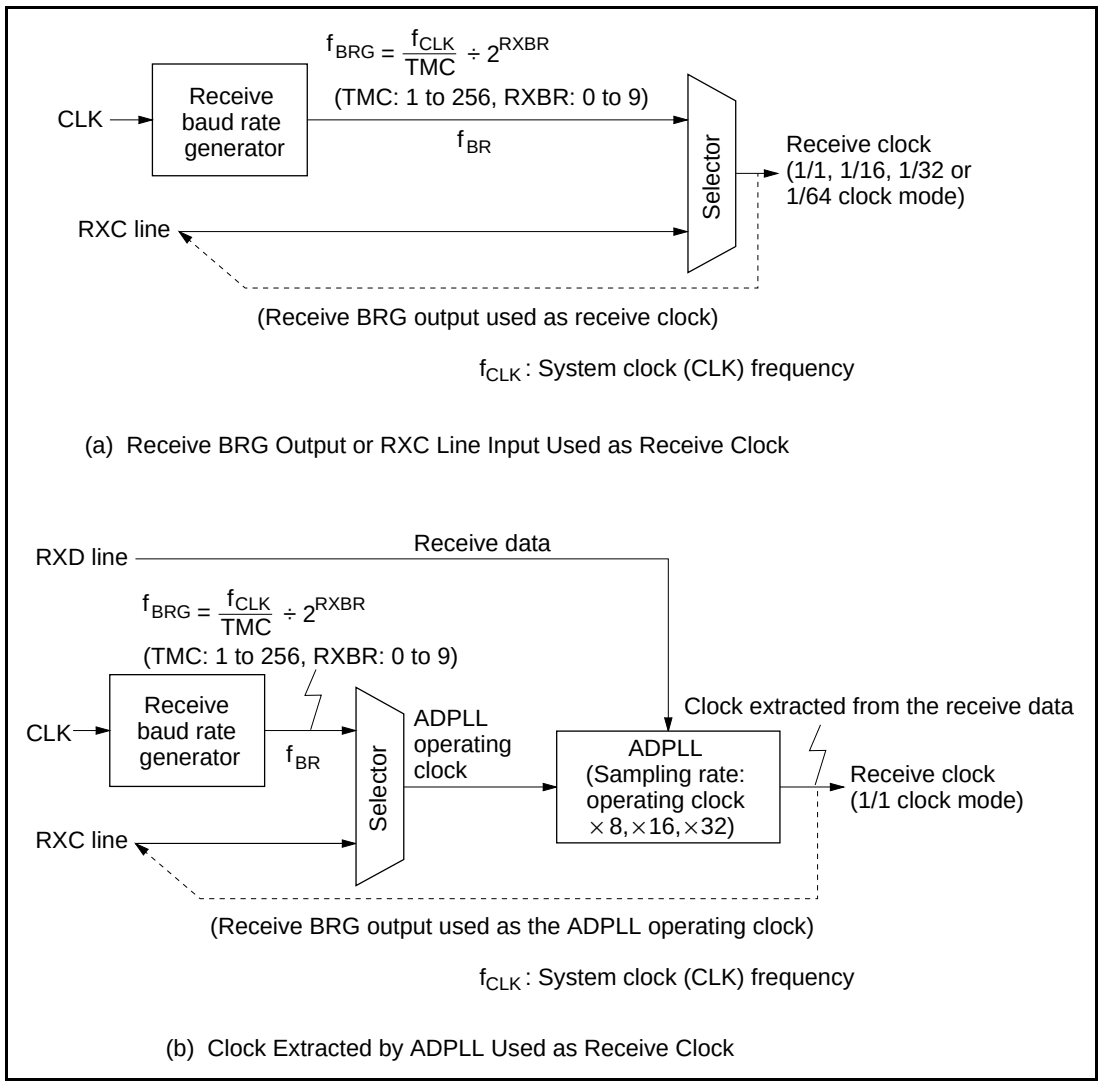


Figure 5.31 Receive Clock Sources

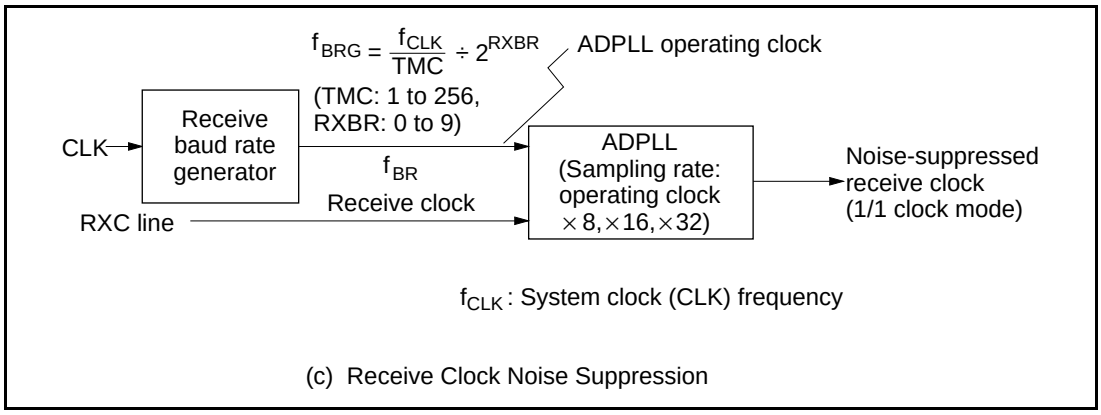


Figure 5.31 Receive Clock Sources (cont)

5.4.4 Baud Rate Generator

The output frequency of the baud rate generator for transmission and reception is obtained by the following equation:

$$f_{BRG} = \frac{f_{CLK}}{TMC} \div 2^{BR}$$

f_{BRG}: BRG output frequency
f_{CLK}: System clock frequency
TMC: Value (1–256) set in the time constant register (TMC)
BR: Value (0–9) set in the TXBR3–TXBR0 bits of TXS, or the RXBR3–RXBR0 bits of RXS

Frequencies determined by the above equation are independently output for transmission and reception from the baud rate generator.

5.4.5 ADPLL

In byte or bit synchronous mode, either of two receive clocks can be used for the MSCI: a clock extracted from the receive data by the ADPLL or the noise-suppressed RXC line input by ADPLL.

The ADPLL has the following operating modes: × 8, × 16, and × 32 (ratio of the ADPLL operating clock rate to the bit rate). In other words, the operating clock frequency must be 8, 16, or 32 times the bit rate, to use the ADPLL clock extraction function, regardless of the source of the operating clock. The DRATE1–DRATE0 bits of mode register 2 (MD2) selects the ADPLL operating mode.

5.5 ADPLL

5.5.1 Overview

The advanced digital PLL (ADPLL) extracts clock signals from the receive data and generates a decoding clock for the receive data.

The ADPLL features:

- Clock extraction from five transmission code types (figure 1.8)
  - — NRZ
  - — NRZI
  - — Manchester
  - — FM0
  - — FM1
- Selectable ratio of the ADPLL clock rate to the bit rate
  - — × 8
  - — × 16
  - — × 32
- Receive data noise suppression (see section 5.5.2, Operation)
- Receive clock noise suppression (see section 5.5.2, Operation)

Figure 5.32 is the block diagram of the ADPLL.

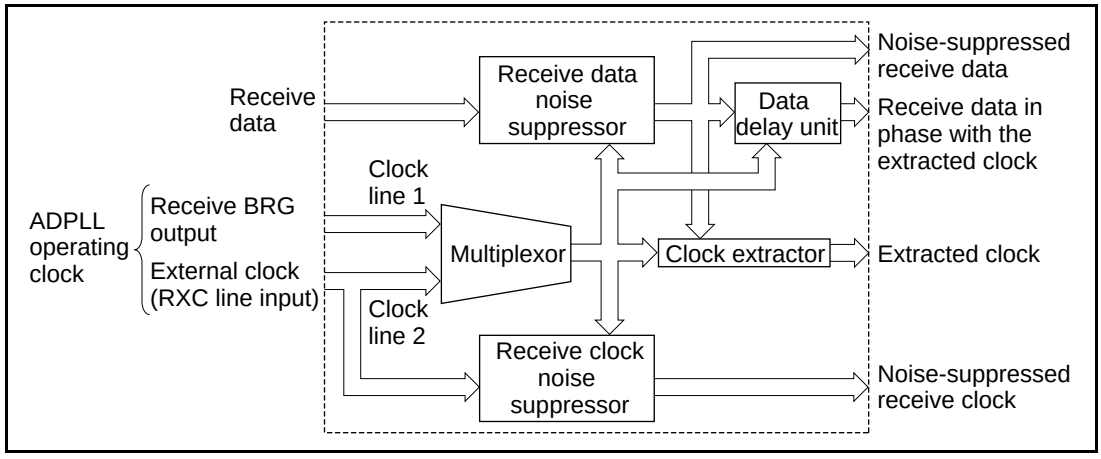


Figure 5.32 ADPLL Block Diagram

The ADPLL can perform either clock extraction from the receive data or noise suppression for the receive clock input from the RXC line. In both cases it suppresses the receive data noise.

The ADPLL receives the receive data and is supplied with the operating clock. The ADPLL has two clock input lines: one for the receive baud rate generator output, and the other for the RXC line input.

The ADPLL uses the receive baud rate generator output or an external clock (RXC line input) as the operating clock to extract the clock component from the receive data. The ADPLL operating clock, functioning as a common operating clock, is supplied to the receive data noise suppressor, clock extractor, and data delay unit. The ADPLL sends the extracted clock and the noise-suppressed receive data to the receiver. The extracted clock serves as the receive clock. When the output of the receive baud rate generator is used as the ADPLL operating clock, the RXC line outputs the receive clock. (ADPLL operation is controlled by the RXCS2–RXCS0 bits of the RX clock source register (RXS).)

The ADPLL uses the output of the receive baud rate generator as the operating clock to suppress noise for the receive clock input from the RXC line. The ADPLL operating clock, functioning as a common operating clock, is supplied to the noise suppressers for the receive clock and the receive data. In this case, the clock extractor does not operate. The ADPLL sends noise-suppressed receive data and the receive clock to the receiver.

The clock extraction from the receive data and noise suppression for the receive data and receive clock are synchronized with the ADPLL operating clock. The ratio of the ADPLL clock rate to the bit rate can be selected from × 8, × 16, and × 32 using the DRATE1–DRATE0 bits of mode register (MD2).

The relationship between the ADPLL clock and bit rates is shown in table 5.15.

Table 5.15 Relationship Between the ADPLL Operating Clock and Bit Rates

Function	ADPLL Operating Clock Source	Operating Mode	Ratio of ADPLL Operating Clock Rate to Bit Rate
Clock extraction from receive data and noise suppression for receive data	RXC line input Receive BRG output	× 8	8/1
		× 16	16/1
		× 32	32/1
Noise suppression for receive clock and receive data	Receive BRG output	× 8	8/1
		× 16	16/1
		× 32	32/1

The ADPLL can compensate the phase of the extracted clock pulses. If the extracted clock is skewed by one or more cycles from the receive data that was passed via the data delay unit, the

ADPLL automatically compensates it to within ±1 operating clock cycle until the clock phase and receive data phase are synchronized.

ADPLL specifications are shown in table 5.16. The transmission codes supported by the ADPLL are summarized in figure 1.8.

Table 5.16 ADPLL Specifications

No.	Item	Mode				Specification			Remarks
1	Maximum operating clock frequency	All				17.6 MHz			
2	Maximum bit rate	Operating mode			× 8	2.2 Mbps			
					× 16	1.1 Mbps			
					× 32	0.5 Mbps			
3	Maximum number of level transitions necessary for synchronization	Code type	NRZ		× 8	4			
					× 16	8			
					× 32	16			
			FM	Normal mode	× 8	4			
					× 16	8			
					× 32	16			
				Search mode		1			Sampling ratio must also be set
4	Receive data noise suppression	Noise suppression			On	Undefined	Off		
		Operating mode			× 8	x<1/8	1/8≤x<2/8	2/8≤x	
					× 16	x<2/16	2/16≤x<3/16	3/16≤x	
					× 32	x<4/32	4/32≤x<5/32	5/32≤x	

5	Receive clock noise suppression	Operating mode	× 8	x<1/8	1/8≤x<2/8	2/8≤x	Clock extractor does not function for receive clock noise suppres-sion
			× 16	x<2/16	2/16≤x<3/16	3/16≤x	
			× 32	x<4/32	4/32≤x<5/32	5/32≤x	
6	Maximum bit rate for receive clock noise suppression	Operating mode	× 8		1.25 Mbps		
			× 16		0.62 Mbps		
			× 32		0.31 Mbps		

(x: Noise width/1-bit cell width)

Note: The ADPLL enters search mode when an enter-search-mode command is issued. For details, see section 5.5.3, Enter-Search-Mode Command.

5.5.2 Operation

The ADPLL has two main functions: clock component extraction from noise-suppressed receive data, and receive clock noise suppression.

Clock Component Extraction from Receive Data: The flow of receive data and the ADPLL operating clock signals for clock extraction are shown in figure 5.33. Either the receive baud rate generator output from clock line 1 or the external clock (RXC line input) from clock line 2 can be used as the ADPLL clock.

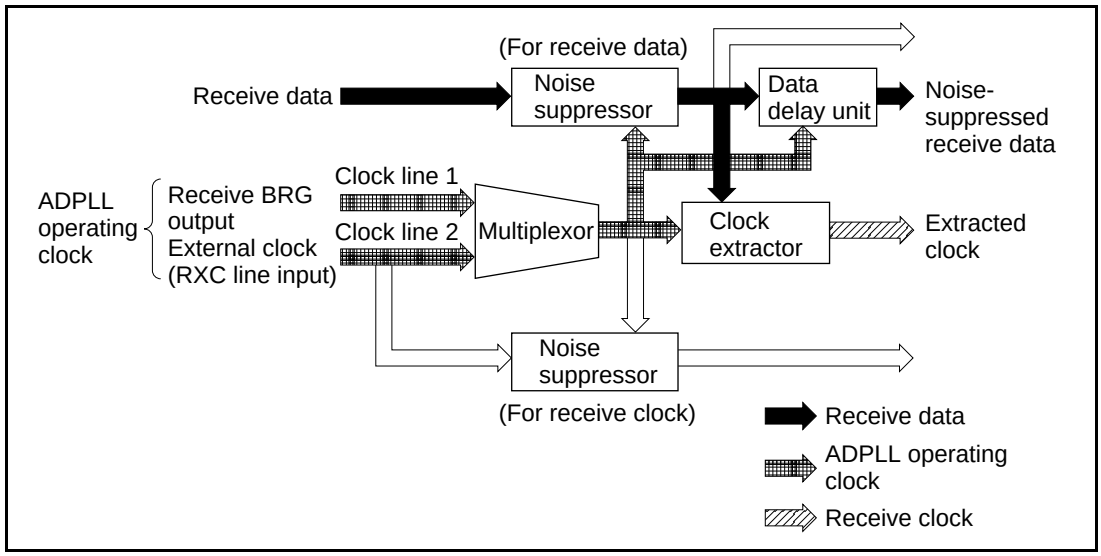


Figure 5.33 Data and Clock Signal Flow for Clock Extraction from Receive Data

The specific operation of the ADPLL are as follows:

- The receive data noise suppressor receives receive data and suppresses noise.
- The noise suppressor outputs the noise-suppressed receive data to the clock extractor and data delay unit.
- The data delay unit outputs the noise-suppressed receive data to the receiver, synchronizing the data with the extracted clock.
- The clock extractor extracts clock components from the noise-suppressed receive data and outputs the resulting clock signal.
- The ADPLL operating clock (the receive baud rate generator output or external clock) passes through the multiplexor to the clock extractor, receive data noise suppressor, and data delay unit.

The ADPLL outputs the noise-suppressed receive data and extracted clock, synchronizing their phases using the ADPLL phase compensation function. Phase compensation for the NRZ- and FM0-code receive data is shown in figures 5.34 and 5.35.

In the figures, the ADPLL outputs the noise-suppressed receive data from the noise suppressor to the data delay unit and clock extractor. The clock extractor samples the noise-suppressed receive data at the rising edge of the ADPLL operating clock pulse, and performs clock extraction.

The ADPLL compares the phases of the receive data and extracted clock at level transition points (T_S, T_{S-1}, T_{S-2}) in the receive data output from the data delay unit. If the two phases are skewed, the extracted clock cycle is lengthened or shortened by one ADPLL operating clock cycle. In the examples shown in figures 5.38 and 5.39 (operating mode = $\times 8$), this synchronization can be

established within a maximum of four transition points. (For FM type codes (FM0, FM1 and Manchester), synchronization can be established in one level transition by issuing the enter search mode command.)

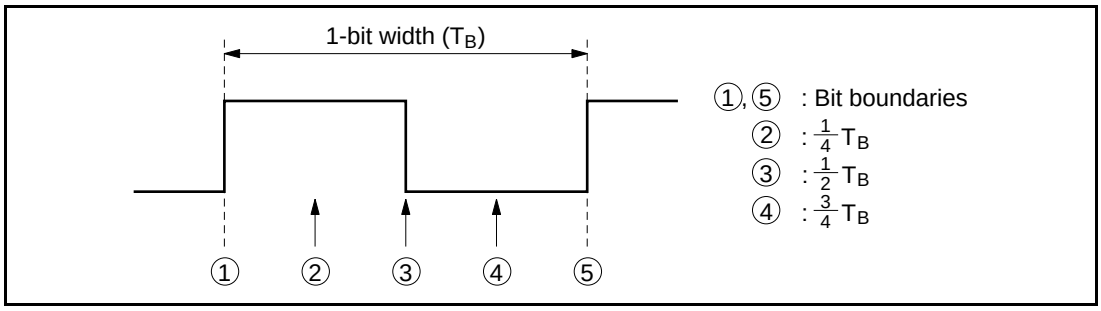
The relationship between the extracted clock and the receive data bit cell depends on the receive data code type. For NRZ and NRZI codes, the rising edge of the extracted clock pulse is located at the midpoint of the data bit cell width that is output from the data delay unit. For FM0, FM1, and Manchester codes, the rising edge of the extracted clock is located at the 1/4 point of the data bit cell width that is output from the data delay unit. This applies also to operating modes × 16 and × 32 except that the maximum number of level transition points required for synchronization is 8 and 16, respectively.

Phase compensation functions for NRZ codes and FM codes are summarizes in table 5.17.

Table 5.17 Phase Compensation for NRZ Codes and FM Codes

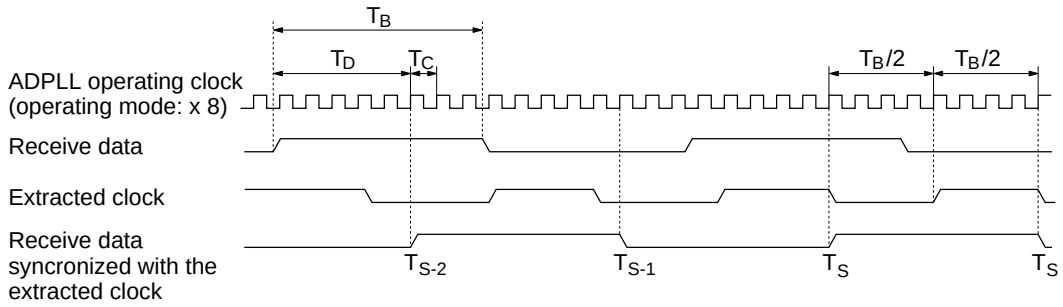
Codes	Receive Data Transition Points	Phase Compensation
NRZ NRZI	Bit boundary to $1/2T_B$	–1 ADPLL operating clock
	$-1/2T_B$ to bit boundary	+1 ADPLL operating clock
FM0 FM1	Bit boundary to $1/4T_B$	–1 ADPLL operating clock (Note)
	$-3/4T_B$ to bit boundary	+1 ADPLL operating clock (Note)
	Others	No phase compensation
Manchester	$1/2T_B$ to $3/4T_B$	–1 ADPLL operating clock (Note)
	$1/4T_B$ to $1/2T_B$	+1 ADPLL operating clock (Note)
	Others	No phase compensation

T_B: One bit cycle of receive data



Note: An enter search mode command is automatically issued; the ADPLL automatically enters search mode and attempts to reestablish synchronization at the next transition point if the

ADPLL detects no level transition in FM-code receive data in the successive two bit cycles, or "windows" (from the bit boundary to 1/4 T_B or from 3/4 T_B to the bit boundary), and the CLMD bit of MSCI status register 1 (ST1) is set to 1.



T_B: One receive data bit time

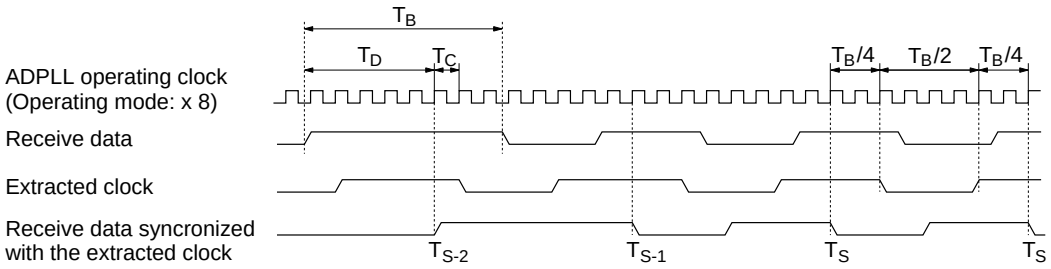
T_C: One ADPLL operating clock cycle

T_D: Delay time between the receive data input to the ADPLL and the receive data after passing the noise suppressor and data delay unit

T_{S-1} and T_{S-2}: Receive data level transitions after noise suppression

T_S: Synchronized transitions of noise-suppressed receive data after noise suppression.

Figure 5.34 NRZ Receive Data Phase Compensation in Operating Mode × 8



T_B: One receive data bit time

T_C: One ADPLL operating clock cycle

T_D: Delay time between the receive data input to the ADPLL and the receive data after passing the noise suppressor and data delay unit

T_{S-1} and T_{S-2}: Receive data level transitions after noise suppression

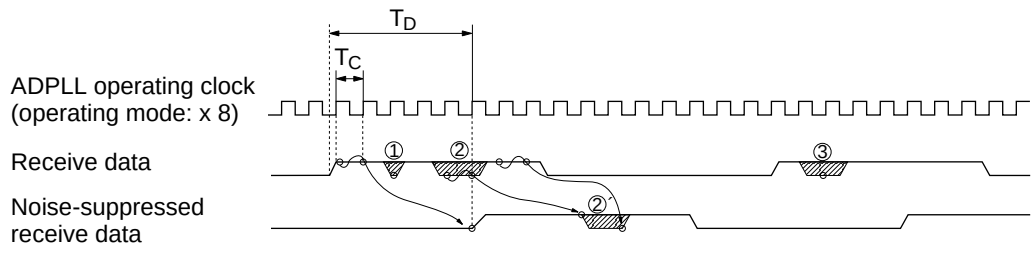
T_S: Synchronized transitions of noise-suppressed receive data after noise suppression.

Figure 5.35 FM0 Receive Data Phase Compensation in Operating Mode × 8

The receive data noise suppression timing in the noise suppressor is shown in figure 5.36. NRZ code receive data is used in this example. The same basic timing also applies to other codes.

The ADPLL samples receive data at the rising edge of the ADPLL operating clock pulse. In operating mode × 8, the same receive data level sampled twice in succession is considered valid data. (The same data level sampled three times in succession in operating mode × 16 and five times in succession in operating mode × 32 is considered valid data.) All other sampled data is suppressed as noise.

Å, Ç, and É in the figure correspond to "On", "Off", and "Undefined" in No. 4 of table 5.16, ADPLL Specifications. É is suppressed as noise since the same level cannot be sampled twice in succession.



T_C: One ADPLL operating clock cycle

T_D: Delay time between the receive data input to the ADPLL and the receive data after passing the noise suppressor and data delay unit

°: Receive data sampling points. The receive data is sampled at the rising edge of the ADPLL clock pulse.

Figure 5.36 Noise Suppression in the Receive Data Noise Suppressor in Operating Mode × 8

Receive Clock Noise Suppression: The flow of receive data, the ADPLL operating clock signal, and the receive clock signal for noise suppression are shown in figure 5.37.

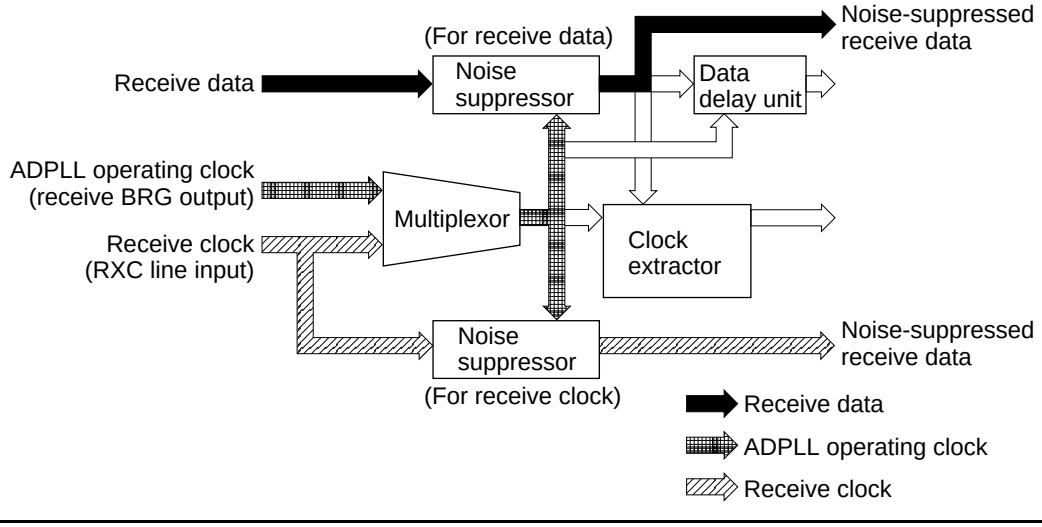


Figure 5.37 Data and Clock Signal Flow for Receive Clock Noise Suppression

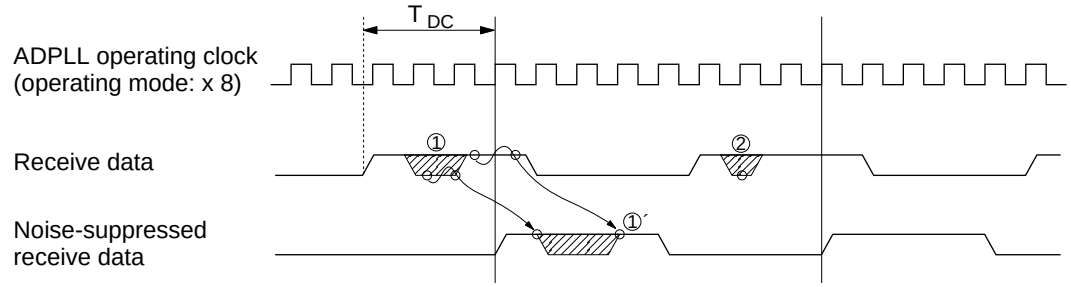
The specific operations of the ADPLL are as follows:

- The receive data noise suppressor receives receive data and outputs the noise-suppressed receive data.
- The ADPLL operating clock is supplied to the receive data noise suppressor and the receive clock noise suppressor via the multiplexor.
- The receive clock noise suppressor outputs the noise-suppressed receive clock .

Noise suppression timing in the receive clock noise suppressor is shown in figure 5.38. In this example, operating mode × 8 is used. The same basic timing applies to other modes except for the number of successive sampling times. The ADPLL samples the receive clock at the rising edge of the ADPLL operating clock pulse. In operating mode × 8, the same receive data level sampled twice in succession is considered valid data. (The same data level sampled three times in succession in operating mode × 16 and five times in succession in operating mode × 32 is considered valid data.) All other sampled data is suppressed as noise. If noise occurs around the rising or falling edges of the receive clock pulses, the rising or falling edges of the noise-suppressed receive clock pulses may be shifted forward or backward. The maximum shift widths in × 8, × 16, and × 32 modes are 2, 3, and 5 ADPLL operating clock cycles, respectively.

Å and Ç in the figure correspond to "Off" and "On" in No. 5 of table 5.16, ADPLL Specifications.

Receive data noise is suppressed as described in Clock Component Extraction from Receive Data above.



T_{DC}: Delay time between the input receive clock and the receive clock after passing the noise suppressor.

Figure 5.38 Noise Suppression in the Receive Clock Noise Suppressor

5.5.3 Notes on Usage

Synchronization patterns: By issuing an enter search command, FM-coded receive data can be synchronized after only one transition. This command is effective in all operating modes (× 8, × 16, or × 32).

When issuing an enter search mode command, for correct synchronization, use the following synchronization patterns:

- FM0 11111111
- FM1 00000000
- Manchester 10101010 or 01010101

Transmission Encoding and Timing

NRZ-type codes: With NRZ-type encoding (NRZ or NRZI), all transitions in the transmit data on the TXD line occur at falling edges of the transmit clock input or output on the TXC line. Receive data is latched from the RXD line on rising edges of the receive clock input or output on the RXC line. Figure 5.39 shows the timing.

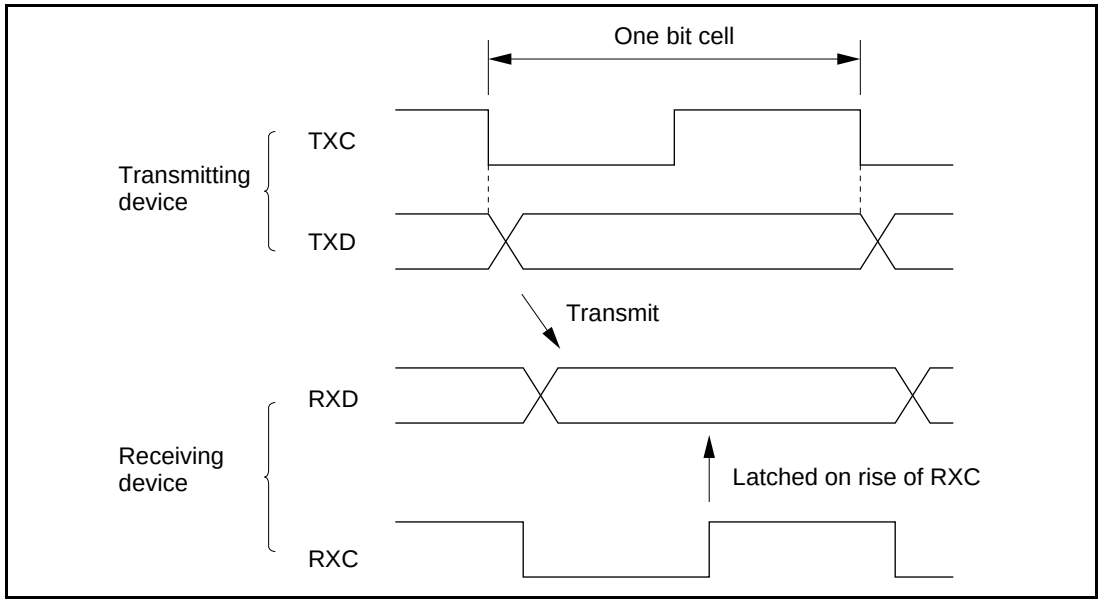


Figure 5.39 Transmit and Receive Timing for NRZ-Type Codes

FM-type codes: With FM-type encoding (FM0, FM1, or Manchester), transitions in the transmit data occur at the beginning and center of the bit cells, as shown in figure 1.8. With FM0 and FM1, transitions at the beginning of bit cells occur at the rising edges of the transmit clock input or output on the TXC line. Transitions at the centers of bit cells occur at the falling edges of the transmit clock. With Manchester, transitions at the beginning of bit cells occur at the falling edge of the transmit clock input or output on the TXC line. Transitions at the centers of bit cells occur at the rising edges of the transmit clock.

- (a) FM0 and FM1
- (b) Manchester

Figure 5.40 shows the timing.

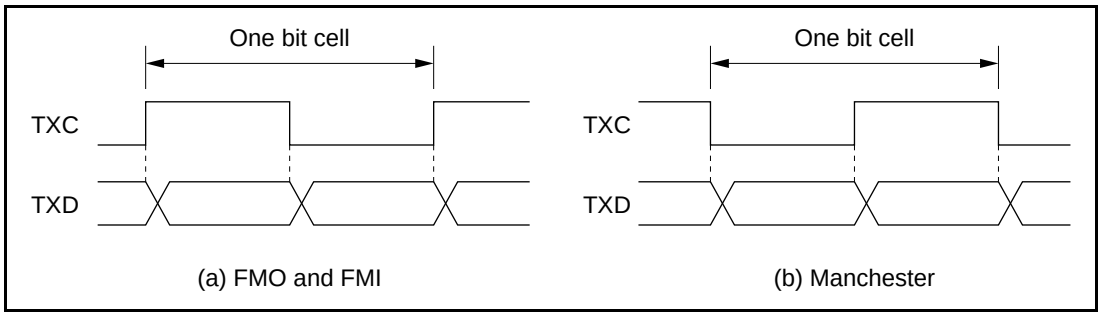


Figure 5.40 Transmit Timing for FM-Type Codes

When the transmit clock (TXC) is generated by the internal baud rate generator, if BR = 0 and TMC > 2, then as shown in table 5.12, the duty cycle of TXC is not 50%, so the duty cycle of the signal on TXD is not 50%. When the receiving device inputs this signal on its RXD line, the ADPLL does not extract the clock or sample the data correctly. For this reason, transmitter settings with BR = 0 and TMC > 2 should be avoided.

When an FM-type code is received, normally the ADPLL is used to extract the clock component from the RXD input, then the data is sampled using the extracted receive clock. There is accordingly no need to supply a receive clock on the RXC line, but an operating clock must be supplied to the ADPLL.

It is possible to receive FM-type encoded data using a receive clock input via RXC, without using the ADPLL. In this case the receive data is sampled on the rising edges of the RXC receive clock, as in reception of NRZ-type encoded data, so attention must be paid to the phase relationship between RXC and RXD. With FM0 or FM1 encoding, the data can be received by latching the value in the second half of the bit cell. For Manchester encoding, the data can be received by latching the value in the first half of the bit cell. Figure 5.41 shows these timing relationships. Since they differ from the timing shown in figure 5.40, in communication between two SCA's, an external circuit must adjust the phase relationship between the transmit clock and transmit data.

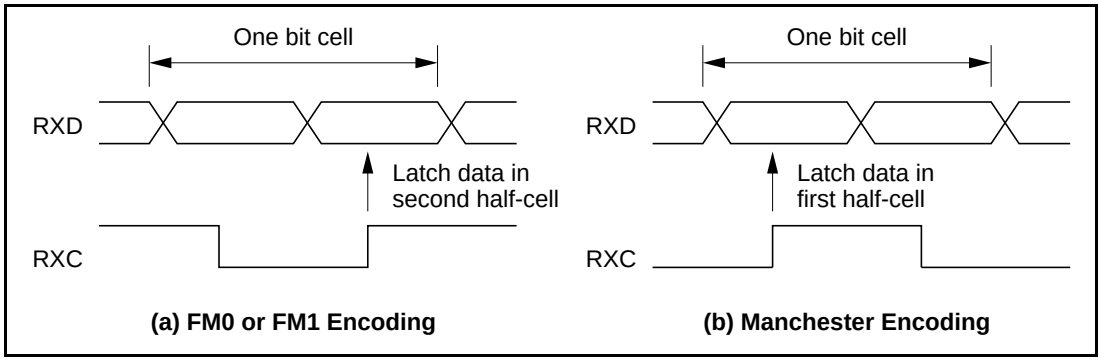


Figure 5.41 Receive Timing for FM-Type Codes

Notes on Clock Extraction: NRZ-type codes differ from FM-type codes in that the data does not include a clock component. Accordingly, when the ADPLL is used to receive NRZ-type encoded data by extracting the clock from the data, receive data including a level transition on the RXD line must be supplied periodically to ensure that the ADPLL does not lose synchronization.

Table 5.18 gives precautions necessary for each type of encoding in each protocol mode.

Table 5.18 Notes on Clock Extraction

Class	Code	Protocol Mode	Description
NRZ-type	NRZ, NRZI	Byte synchronous mode	Ensure that $t_0 < t_{ADPLL}$ (NRZ only) and $t_1 < t_{ADPLL}$. Before transmitting SYN characters, transmit an appropriate synchronization pattern in the idle state to synchronize the ADPLL. (See note 1.)
		Bit synchronous mode	Ensure that $t_0 < t_{ADPLL}$ (NRZ only) and that (6 clock cycles) $< t_{ADPLL}$ (because a flag has six consecutive 1s). Before transmitting an opening flag, transmit an appropriate synchronization pattern in the idle state to synchronize the ADPLL. (See note 1.)
FM-type	FM0, FM1, Manch-ester	Byte synchronous mode, bit synchronous mode	In the idle state, have the receiver receive a synchronization pattern, and issue the enter search mode command to synchronize the ADPLL. (See note 2.)

t_0: Maximum interval containing consecutive 0 data

t_1: Maximum interval containing consecutive 1 data

t_{ADPLL}: Minimum interval in which ADPLL synchronization can be lost by receiving consecutive data at the same level.

Notes: 1. See table 5.16 for the number of transition points needed in the synchronization pattern.
2. For further information about ADPLL synchronization with an FM-type code, see section 5.4.5, "Notes on Use."

ADPLL Receive Margin: Table 5.19 indicates the theoretical ADPLL receive margin (the tolerable bit distortion and bit rate distortion).

As shown in figure 5.42, t0 is the width of one bit in the ideal waveform, and it is the width in the actual waveform. T_0 and T are the ideal and actual time occupied by an arbitrary number of bits.

Compared with the × 8 operating mode, the × 32 operating mode samples each bit of input on the RXD line more often, so the bit margin is higher, but less phase compensation is applied each time by the ADPLL, so the bit rate margin is lower.

Table 5.19 ADPLL Receive Margin (theoretical values; see note 1)

Code Type	Operating Mode	Bit Margin $(t - t_0)/t_0$	Bit Rate Margin $(t - t_0)/t_0$
NRZ-type	× 8	±37.5%	$\pm (12.5 + (t_0/T_0) \times 37.5)$ %
	× 16	±43.7%	$\pm (6.2 + (t_0/T_0) \times 43.7)$ %
	× 32	±46.8%	$\pm (3.1 + (t_0/T_0) \times 46.8)$ %
FM-type	× 8	±25.0%	$\pm (12.5 + (t_0/T_0) \times 25.0)$ %
	× 16	±37.5%	$\pm (6.2 + (t_0/T_0) \times 37.5)$ %
	× 32	±43.7%	$\pm (3.1 + (t_0/T_0) \times 43.7)$ %

Notes: 1. Values in this table are theoretical. They do not guarantee the performance of a device.
2. The operating mode is the ratio of the ADPLL operating clock frequency to the bit rate, as selected by bits DRATE1–0 in MSCI mode register 2.
3. If T_0 is sufficiently long in comparison to t_0, then since t_0/T_0 is approximately zero, the second term in the bit rate margin formula can be ignored and the first term can be used as the average bit rate margin.

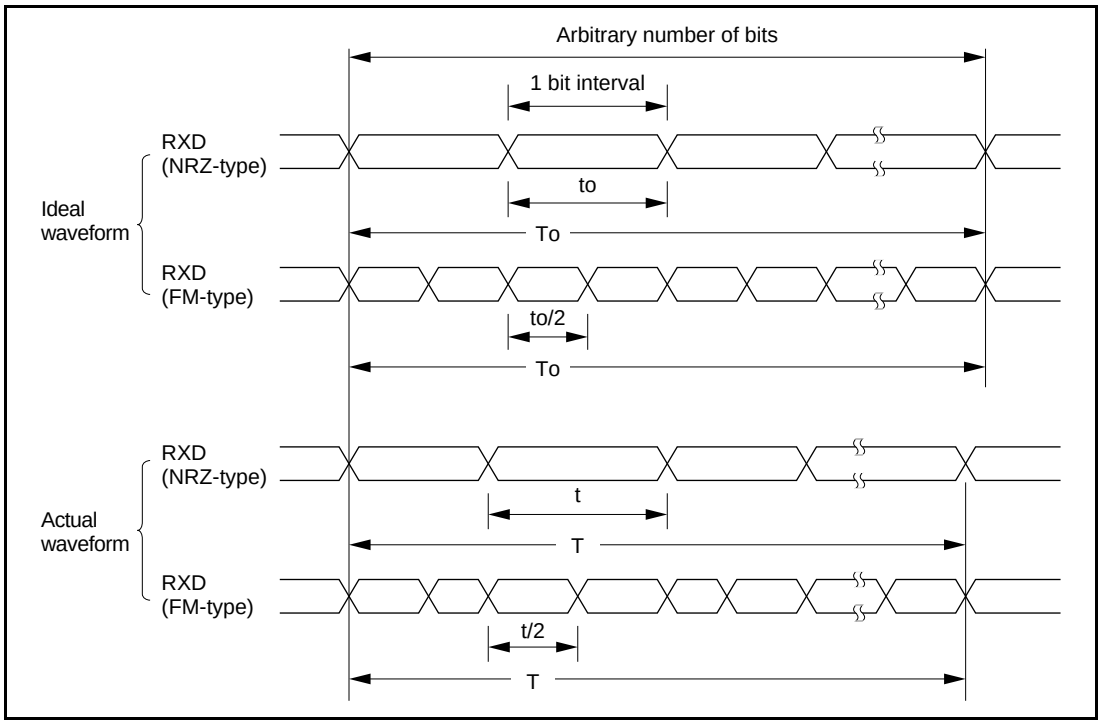


Figure 5.42 RXD Input Waveform

5.6 Baud Rate Generator

5.6.1 Overview

The MSCI uses an internal baud rate generator (BRG) to generate the MSCI transmit/receive clock. The BRG has the following main features:

- Output clock frequency range from f_{CLK} to $f_{CLK}/2^{17}$ (2^{17} = 131,072).
  (f_{CLK}: System clock frequency). When $f_{BRG} = f_{CLK}$, BRG output cannot be obtained from the TXD or RXD lines.
- Frequency accuracy within ±0.5% for any frequency range from $f_{CLK}/100$ to $f_{CLK}/2^{17}$.
  $|f - f_{BRG}| \leq 50 \div$ set value in the time constant register (TMC) (%), where f is the target frequency and f_{BRG} is the BRG output frequency set to the value closest to f ($f_{CLK} \geq f \geq f_{CLK}/2^{17}$).
  Independent transmit and receive frequencies can be specified as 2^n (where n is a positive integer).

Figure 5.43 is the baud rate generator block diagram.

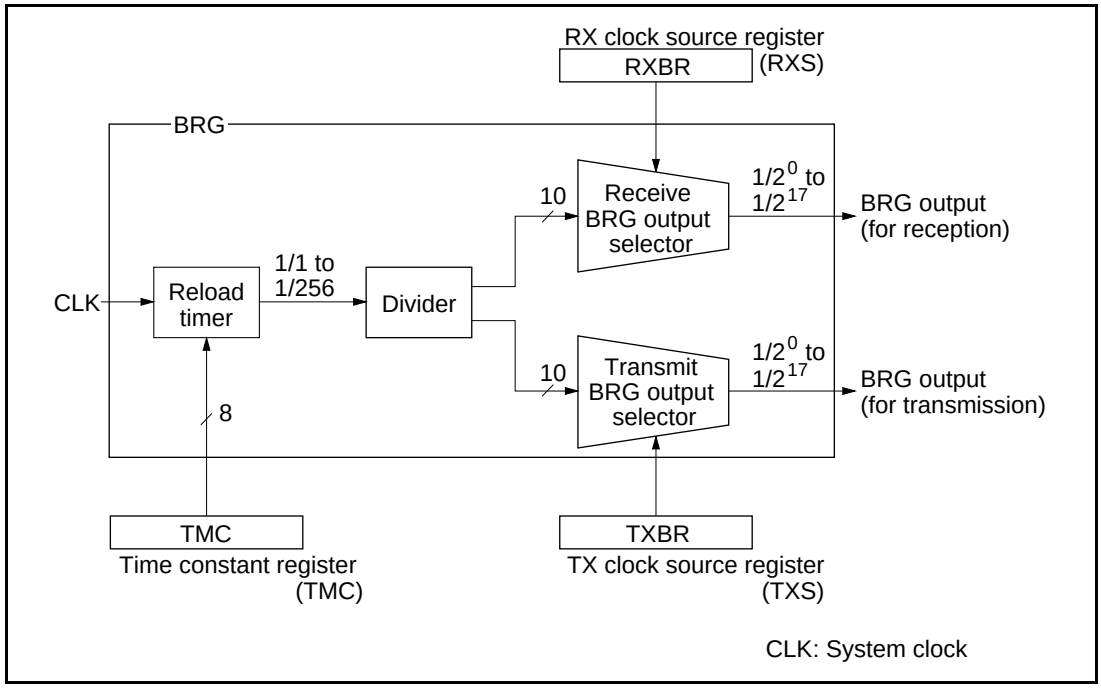


Figure 5.43 Baud Rate Generator Block Diagram

5.6.2 Functions

The MSCI baud rate generator generates clock pulses according to the settings of the TMC7–TMC0 bits of the time constant register (TMC), the TXBR3–TXBR0 bits of the TX clock source register (TXS), and the RXBR3–RXBR0 bits of the RX clock source register (RXS).

TMC is an 8-bit register for specifying the value to be loaded into the reload timer in the baud rate generator. The reload timer is decremented based on the system clock CLK, and outputs a high-level signal for one clock cycle each time the reload timer value equals 1. Thus, the timer outputs a high-level signal for one clock cycle each time the number of system clock cycles specified with the TMC7–TMC0 bits of TMC elapses, as shown in figure 5.44. Zero specified by TMC is assumed to be 256, and when 1 is specified, the output will be the same as the system clock frequency.

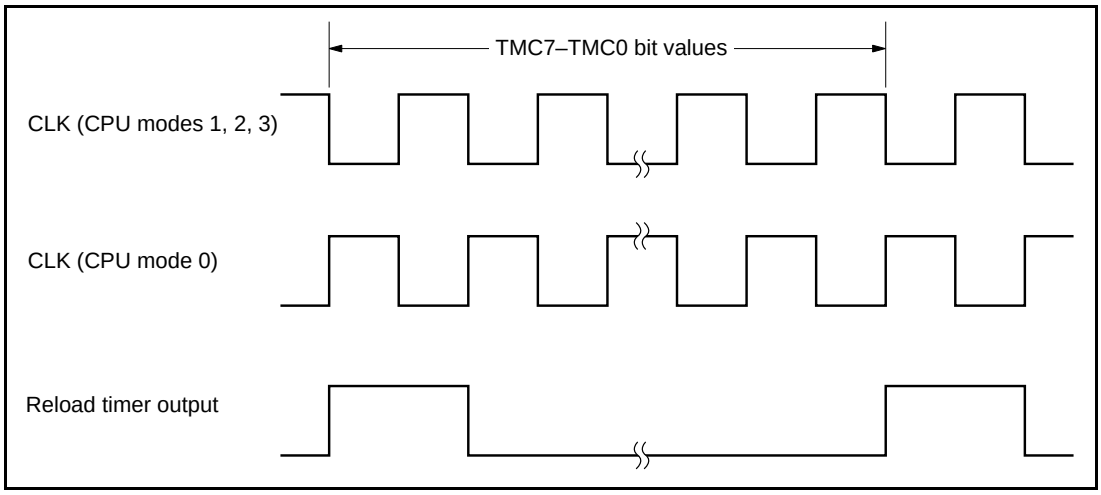


Figure 5.44 Reload Timer Output

The reload timer output is input to the frequency divider. The transmit frequency division ratio is specified with the TXBR3–TXBR0 bits of TXS and the receive frequency division ratio with the RXBR3–RXBR0 bits of RXS.

In addition, the TXCS2–TXCS0 bits of TXS and RXCS2–RXCS0 bits of the RXS specify whether or not to supply the output clock to the MSCI transmitter and receiver, respectively. The BRG output can be used for the transmit/receive clock or for the ADPLL operating clock. For details on these specifications, see sections 5.2.4, MSCI Control Register (CTL), 5.2.5, MSCI RX Clock Source Register (RXS), and 5.2.6, MSCI TX Clock Source Register (TXS).

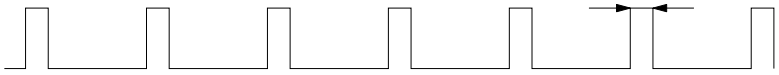
The relationship between the register set values and the generated clock frequency is given below.

$$f_{BRG} = \frac{f_{CLK}}{} \div 2^{BR}$$

f_{BRG}: Transmit (receive) BRG output frequency
f_{CLK}: System clock frequency (frequency equal to f_{BRG} can be used only for the ADPLL operating clock)
TMC: Value (1–256) set in TMC
BR: Value (0–9) of TXBR3–TXBR0 bits of TXS or RXBR3–RXBR0 bits of RXS

Table 5.20 gives widths and duty ratios (pulse width to pulse period) of BRG output clock waveforms along with the corresponding register set values.

Table 5.20 BRG Output Waveform and Register Set Values

Set Value		
BR	**TMC**	**Waveform**
1–9	—	Duty ratio = 50% 
0	≠ 1	Duty ratio = 50% when TMC = 2 Duty ratio ≠ 50% when TMC ≠ 2 Pulse width is 1 system clock cycle 
	= 1	Duty ratio = 50% Cycle width is 1 system clock cycle 

BR: Value of bits 3–0 of the TXS or RXS
TMC: Value of bits 7–0 of the TMC

5.6.3 Register Set Values and Bit Rates

Asynchronous Mode: In asynchronous mode, the bit rate is selected with TMC7–TMC0 bits of the time constant register (TMC), the TXBR3–TXBR0 bits of the TX clock source register (TXS),

the RXBR3–RXBR0 bits of the RX clock source register (RXS), and the BRATE7–BRATE6 bits of mode register 1 (MD1).

Typical register set values and bit rates are listed in table 5.21.

Table 5.21 Register Set Values and Bit Rates in Asynchronous Mode

Bit Rate (bps)	f_{CLK} (MHz)							
	1.7898				2.4576			
	TMC	BR	CM	Deviation (%)	TMC	BR	CM	Deviation (%)
38400	—	—	—	—	1	1	1/32	0.00
19200	—	—	—	—	1	1	1/64	0.00
9600	—	—	—	—	1	2	1/64	0.00
4800	—	—	—	—	1	3	1/64	0.00
2400	47	0	1/16	–0.83	1	4	1/64	0.00
1200	93	0	1/16	–0.25	1	5	1/64	0.00
600	93	0	1/32	–0.25	1	6	1/64	0.00
300	93	0	1/64	–0.25	1	7	1/64	0.00
150	93	1	1/64	–0.25	1	8	1/64	0.00
110	127	1	1/64	0.10	175	1	1/64	–0.25

Bit Rate (bps)	f_{CLK} (MHz)							
	3.072				4			
	TMC	BR	CM	Deviation (%)	TMC	BR	CM	Deviation (%)
38400	5	0	1/16	0.00	—	—	—	—
19200	5	0	1/32	0.00	13	0	1/16	0.16
9600	5	0	1/64	0.00	13	0	1/32	0.16
4800	5	1	1/64	0.00	13	0	1/64	0.16
2400	5	2	1/64	0.00	13	1	1/64	0.16
1200	5	3	1/64	0.00	13	2	1/64	0.16
600	5	4	1/64	0.00	13	3	1/64	0.16
300	5	5	1/64	0.00	13	4	1/64	0.16
150	5	6	1/64	0.00	13	5	1/64	0.16
110	109	2	1/64	0.08	71	3	1/64	0.03

TMC: Value of the TMC7–TMC0 bits of TMC

BR: Value of the TXBR3–TXBR0 bits of TXS or the RXBR3–RXBR0 bits of RXS

CM: Value of the BRATE1–BRATE0 bits of MD1 (clock mode in asynchronous mode (bit rate/clock frequency))

Table 5.21 Register Set Values and Bit Rates in Asynchronous Mode (cont)

Bit Rate (bps)	f_{CLK} (MHz)							
	4.608				4.9152			
	TMC	BR	CM	Deviation (%)	TMC	BR	CM	Deviation (%)
38400	—	—	—	—	1	1	1/64	0.00
19200	15	0	1/16	0.00	1	2	1/64	0.00
9600	15	0	1/32	0.00	1	3	1/64	0.00
4800	15	0	1/64	0.00	1	4	1/64	0.00
2400	15	1	1/64	0.00	1	5	1/64	0.00
1200	15	2	1/64	0.00	1	6	1/64	0.00
600	15	3	1/64	0.00	1	7	1/64	0.00
300	15	4	1/64	0.00	1	8	1/64	0.00
150	15	5	1/64	0.00	1	9	1/64	0.00
110	41	4	1/64	–0.22	175	2	1/64	–0.25

Bit Rate (bps)	f_{CLK} (MHz)							
	6				6.144			
	TMC	BR	CM	Deviation (%)	TMC	BR	CM	Deviation (%)
38400	—	—	—	—	5	0	1/32	0.00
19200	—	—	—	—	5	0	1/64	0.00
9600	39	0	1/16	0.16	5	1	1/64	0.00
4800	39	0	1/32	0.16	5	2	1/64	0.00
2400	39	0	1/64	0.16	5	3	1/64	0.00
1200	39	1	1/64	0.16	5	4	1/64	0.00
600	39	2	1/64	0.16	5	5	1/64	0.00
300	39	3	1/64	0.16	5	6	1/64	0.00
150	39	4	1/64	0.16	5	7	1/64	0.00
110	213	2	1/64	0.03	109	3	1/64	0.08

TMC: Value of the TMC7–TMC0 bits of TMC

BR: Value of the TXBR3–TXBR0 bits of TXS or the RXBR3–RXBR0 bits of RXS

CM: Value of the BRATE1–BRATE0 bits of MD1 (clock mode in asynchronous mode (bit rate/clock frequency))

Table 5.21 Register Set Values and Bit Rates in Asynchronous Mode (cont)

Bit Rate (bps)	f_{CLK} (MHz)							
	8				9.216			
	TMC	BR	CM	Deviation (%)	TMC	BR	CM	Deviation (%)
38400	13	0	1/16	0.16	15	0	1/16	0.00
19200	13	0	1/32	0.16	15	0	1/32	0.00
9600	13	0	1/64	0.16	15	0	1/64	0.00
4800	13	1	1/64	0.16	15	1	1/64	0.00
2400	13	2	1/64	0.16	15	2	1/64	0.00
1200	13	3	1/64	0.16	15	3	1/64	0.00
600	13	4	1/64	0.16	15	4	1/64	0.00
300	13	5	1/64	0.16	15	5	1/64	0.00
150	13	6	1/64	0.16	15	6	1/64	0.00
110	71	4	1/64	0.03	41	5	1/64	– 0.22

Bit Rate (bps)	f_{CLK} (MHz)							
	9.8304				10			
	TMC	BR	CM	Deviation (%)	TMC	BR	CM	Deviation (%)
38400	2	1	1/64	0.00	—	—	—	—
19200	2	2	1/64	0.00	—	—	—	—
9600	2	3	1/64	0.00	65	0	1/16	0.16
4800	2	4	1/64	0.00	65	0	1/32	0.16
2400	2	5	1/64	0.00	65	0	1/64	0.16
1200	2	6	1/64	0.00	65	1	1/64	0.16
600	2	7	1/64	0.00	65	2	1/64	0.16
300	2	8	1/64	0.00	65	3	1/64	0.16
150	2	9	1/64	0.00	65	4	1/64	0.16
110	175	3	1/64	–0.25	89	4	1/64	– 0.25

TMC: Value of the TMC7–TMC0 bits of TMC

BR: Value of the TXBR3–TXBR0 bits of TXS or the RXBR3–RXBR0 bits of RXS

CM: Value of the BRATE1–BRATE0 bits of MD1 (clock mode in asynchronous mode (bit rate/clock frequency))

Table 5.21 Register Set Values and Bit Rates in Asynchronous Mode (cont)

Bit Rate (bps)	f_{CLK} (MHz)			
	12			
	TMC	BR	CM	Deviation (%)
38400	—	—	—	—
19200	39	0	1/16	0.16
9600	39	0	1/32	0.16
4800	39	0	1/64	0.16
2400	39	1	1/64	0.16
1200	39	2	1/64	0.16
600	39	3	1/64	0.16
300	39	4	1/64	0.16
150	39	5	1/64	0.16
110	213	3	1/64	0.03

TMC: Value of the TMC7–TMC0 bits of TMC

BR: Value of the TXBR3–TXBR0 bits of TXS or the RXBR3–RXBR0 bits of RXS

CM: Value of the BRATE1–BRATE0 bits of MD1 (clock mode in asynchronous mode (bit rate/clock frequency))

Byte synchronous/Bit synchronous mode: In byte or bit synchronous mode, the bit rate is selected with the TMC7–TMC0 bits of TMC, the TXBR3–TXBR0 bits of TXS, and the RXBR3–RXBR0 bits of RXS.

Typical register set values and bit rates are listed in table 5.22.

Table 5.22 Register Set Values and Bit Rates in Byte Synchronous/Bit Synchronous Mode

Bit Rate (bps)	f_{CLK} (MHz)								
	2.4576			3.072			4		
	TMC	BR	Deviation (%)	TMC	BR	Deviation (%)	TMC	BR	Deviation (%)
38400	32	1	0.00	40	1	0.00	52	1	0.16
19200	32	2	0.00	40	2	0.00	52	2	0.16
9600	32	3	0.00	40	3	0.00	52	3	0.16
4800	32	4	0.00	40	4	0.00	52	4	0.16
2400	32	5	0.00	40	5	0.00	52	5	0.16
1200	32	6	0.00	40	6	0.00	52	6	0.16
600	32	7	0.00	40	7	0.00	52	7	0.16
300	32	8	0.00	40	8	0.00	52	8	0.16

Bit Rate (bps)	f_{CLK} (MHz)								
	4.608			4.9152			6		
	TMC	BR	Deviation (%)	TMC	BR	Deviation (%)	TMC	BR	Deviation (%)
38400	60	1	0.00	64	1	0.00	78	1	0.16
19200	60	2	0.00	64	2	0.00	78	2	0.16
9600	60	3	0.00	64	3	0.00	78	3	0.16
4800	60	4	0.00	64	4	0.00	78	4	0.16
2400	60	5	0.00	64	5	0.00	78	5	0.16
1200	60	6	0.00	64	6	0.00	78	6	0.16
600	60	7	0.00	64	7	0.00	78	7	0.16
300	60	8	0.00	64	8	0.00	78	8	0.16

TMC: Value of the TMC7–TMC0 bits of TMC

BR: Value of the TXBR3–TXBR0 bits of TXS or the RXBR3–RXBR0 bits of RXS

Table 5.22 Register Set Values and Bit Rates in Byte Synchronous/Bit Synchronous Mode (cont)

Bit Rate (bps)	f_{CLK} (MHz)								
	6.144			8			9.216		
	TMC	BR	Deviation (%)	TMC	BR	Deviation (%)	TMC	BR	Deviation (%)
38400	80	1	0.00	104	1	0.16	120	1	0
19200	80	2	0.00	104	2	0.16	120	2	0
9600	80	3	0.00	104	3	0.16	120	3	0
4800	80	4	0.00	104	4	0.16	120	4	0
2400	80	5	0.00	104	5	0.16	120	5	0
1200	80	6	0.00	104	6	0.16	120	6	0
600	80	7	0.00	104	7	0.16	120	7	0
300	80	8	0.00	104	8	0.16	120	8	0

Bit Rate (bps)	f_{CLK} (MHz)								
	9.8304			10			12		
	TMC	BR	Deviation (%)	TMC	BR	Deviation (%)	TMC	BR	Deviation (%)
38400	128	1	0	130	1	0.16	156	1	0.16
19200	128	2	0	130	2	0.16	156	2	0.16
9600	128	3	0	130	3	0.16	156	3	0.16
4800	128	4	0	130	4	0.16	156	4	0.16
2400	128	5	0	130	5	0.16	156	5	0.16
1200	128	6	0	130	6	0.16	156	6	0.16
600	128	7	0	130	7	0.16	156	7	0.16
300	128	8	0	130	8	0.16	156	8	0.16

TMC: Value of the TMC7–TMC0 bits of TMC

BR: Value of the TXBR3–TXBR0 bits of TXS or the RXBR3–RXBR0 bits of RXS

5.7 Interrupts

5.7.1 Interrupt Types and Sources

The MSCI can issue four types of interrupt requests: TXRDY, RXRDY, TXINT, and RXINT.

These interrupts are initiated with the status bits (bits 7, 6, 1, and 0) of status register 0 (ST0) and are enabled/disabled with the enable bits (bits 7, 6, 1, and 0) of interrupt enable register 0 (IE0).

The TXINT and RXINT interrupts are also assigned with status bits and corresponding enable bits for each source. The status bit and its enable bit are ANDed for each interrupt source. The interrupt sources are indicated by the TXINT bit (bit 7) or RXINT bit (bit 6) of ST0, regardless of the values of the TXINTE bit (bit 7) or RXINTE bit (bit 6) of IE0.

5.7.2 Interrupt Clear

The methods for clearing each interrupt are given below.

- TXRDY interrupt

  Write data to the transmit buffer until the data byte count in the buffer becomes equal to or greater than TXF + 1, (TXF is the value specified with TX ready control register 1 (TRC1)), or disable the transmitter. A channel reset or a TX reset command will also clear this interrupt.
- RXRDY interrupt

  Read data from the receive buffer until it becomes empty. A channel reset or an RX reset command will also clear this interrupt.
- TXINT interrupt

  Write a 1 to each status bit. When the interrupt source is an idle transmitter, write transmit data to the transmit buffer.
- RXINT interrupt

  Write a 1 to each status bit. When the interrupt source is a parity/MP or CRC error, read receive data from the receive buffer. In bit synchronous mode, ST2 bit values are transferred to the frame status register (FST), and ST2 is reset when the last character to be transferred has been read from the receive buffer at completion of frame transfer. Table 5.23 shows interrupt types, sources, and clearing procedures.

Table 5.23 Interrupts, Interrupt Sources, and Clearing Procedures

Interrupt Type	Interrupt Status Bit	Enable Bit	Interrupt Source	Source Status Bit	Enable Bit	Clearing Procedure*[1]
TXRDY interrupt	TXRDY	TXRDYE	TX ready	—	—	1
RXRDY interrupt	RXRDY	RXRDYE	RX ready	—	—	2
TXINT interrupt	TXINT	TXINTE	(1)Underrun error	UDRN	UDRNE	3
			(2)Transmitter idle	IDL	IDLE	
			(3)$\overline{\text{CTS}}$ line level transition	CCTS	CCTSE	
RXINT interrupt	RXINT	RXINTE	(1)SYN pattern detection/flag detection	SYNCD/ FLGD	SYNCDE/ FLGDE	4
			(2)$\overline{\text{DCD}}$ line level transition	CDCD	CDCDE	
			(3)Break start detection/abort detection	BRKD/ ABTD	BRKDE/ ABTDE	
			(4)Break end detection/idle start detection	BRKE/ IDLD	BRKEE/ IDLDE	
			(5)Receive frame end (ST2)	EOM*[2]	EOME	
			(6)Parity or MP bit = 1/short frame detection	PMP/ SHRT*[2]	PMPE/ SHRTE	
			(7)Parity error/abort end frame detection	PE/ABT*[2]	PEE/ABTE	
			(8)Framing error detection/residual bit frame detection	FRME/ RBIT*[2]	FRMEE/ RBITE	

Table 5.23 Interrupts, Interrupt Sources, and Clearing Procedures (cont)

Interrupt Type	Interrupt Status Bit	Enable Bit	Interrupt Source	Source Status Bit	Enable Bit	Clearing Procedure*[1]
RXINT interrupt	RXINT	RXINTE	(9)Overrun error	OVRN*[2]	OVRNE	4
			(10)CRC error	CRCE*[2]	CRCEE	
			(11)End of message (FST)	EOMF	EOMFE	
			(12)Two-clock missing detection	CLMD	CLMDE	

Clearing procedure 1: Write data to the transmit buffer until the data byte count in the buffer becomes equal to or greater than TXF + 1, (TXF is the value specified with TX ready control register 1 (TRC1)), or disable the transmitter.

Clearing procedure 2: Read data from the receive buffer until it becomes empty.

Clearing procedure 3: —(1), (3) : Write a 1 to each status bit.

—(2):Write data to the transmit buffer to place the transmitter in other state.

Clearing procedure 4: —(1) – (12): Write a 1 to each status bit.

—PMP: Read data from the receive buffer to enable reading the next data*[3].

—CRCE: Automatically cleared when the CRC calculation result is normal*[4].

Notes: 1. The RXINT interrupt source can also be cleared by a channel reset or an RX reset command. The TXRDY and TXINT interrupt sources can also be cleared by a channel reset or a TX reset command.

2. Status register 2 (ST2) bit values are transferred to the frame status register (FST) and ST2 is reset when the last character has been read from the receive buffer at completion of receive frame transfer.

3. In CPU mode 1, the PMP bit is cleared when the parity/MP bit of the next data is 0, (when the next data becomes ready to be read). In CPU modes 0, 2, and 3, this bit is cleared when the parity/MP bit of the next two bytes of data are both 0 (when the next two bytes of data become ready to be read).

4. CRC calculation result can be read from the CRCE bit when the CRCCC bit of mode register 0 (MD0) is 1. For details on the setting/resetting timing of the CRCE bit, see Error Checking, in section 5.3.2, Byte Synchronous Mode; and Error Checking, in section 5.3.3, Bit Synchronous Mode.

5.7.3 Interrupt Enable Conditions

The conditions for the TXRDY, RXRDY, TXINT, and RXINT interrupt requests are listed below.

- TXRDY interrupt request condition
  TXRDY = TXRDY • TXRDYE
- RXRDY interrupt request condition
  RXRDY = RXRDY • RXRDYE
- TXINT interrupt request condition
  TXINT = TXINT • TXINTE
  where, TXINT = UDRN • UDRNE + IDL • IDLE + CCTS • CCTSE
- RXINT interrupt request condition
  RXINT = RXINT • RXINTE
  where, RXINT = (SYNCD/FLGD) • (SYNCDE/FLGDE) + CDCD • CDCDE
  + (BRKD/ABTD) • (BRKDE/ABTDE)
  + (BRKE/IDLD) • (BRKEE/IDLDE) + EOM • EOME
  + (PMP/SHRT) • (PMPE/SHRTE) + (PE/ABT) • (PEE/ABTE)
  + (FRME/RBIT) • (FRMEE/RBITE) + OVRN • OVRNE
  + CRCE • CRCEE + EOMF • EOMFE
  + CLMD • CLMDE

See figure 1.25 for the relationship between interrupt requests, their status bits, and enable bits of each register.

5.8 Reset Operation

When the MSCI is reset, (1) the receiver and transmitter are disabled, (2) the transmit/receive buffers are cleared, (3) the input/output lines (RXC and TXC) are set for input, (4) the output lines (TXD and $\overline{\text{RTS}}$) are inactivated, and (5) all the internal registers are initialized.

In addition, (1) asynchronous mode with 1 stop bit, 8-bit character length, 1/1 clock rate, and without parity is selected; (2) full-duplex communication with NRZ code is selected; (3) the transmit/receive status bits and interrupt enable bits are cleared; (4) the TXC line input serves as the transmit clock and the RXC line input as the receive clock; (5) the ADPLL and baud rate generator are initialized.

Section 6 Direct Memory Access Controller (DMAC)

6.1 Overview

The HD64570 has a four on-chip direct memory access controller channels (DMAC channels 0–3), which support chained-block transfer. Channel 0 is connected to the MSCI channel 0 receiver, channel 1 to the MSCI channel 0 transmitter, channel 2 to the MSCI channel 1 receiver, and channel 3 to the MSCI channel 1 transmitter (figure 1.14). Other than the connection destination, the specifications for the four channels are identical.

6.1.1 Functions

The on-chip DMAC supports the following DMA transfer modes: single-block transfer (single address) and chained-block transfer (single address). The features and functions of each mode are summarized as follows:

Single-Block Transfer Mode (Single Address): Data is transferred in byte units from the MSCI to memory via DMAC channels 0 and 2, or from memory to the MSCI via DMAC channels 1 and 3.

- Up to 64 Kbytes of data transferred
- Up to 16 Mbytes of memory addresses directly accessed
- Interrupt generation at DMA transfer completion
- Maximum data transfer rate of 11.1 Mbytes/s (at 16.7-MHz operation without wait states inserted)

Chained-Block Transfer Mode (Single Address): When the MSCI is in bit synchronous mode, data is transferred from the MSCI to memory via DMAC channels 0 and 2, or from the MSCI to memory via DMAC channels 1 and 3. Successive single or multi-frame transfers can be made by writing and reading data to/from buffers in memory.

- Interrupt generation at DMA transfer completion or frame transfer completion
- Maximum data transfer rate of 11.1 Mbytes/s (at 16.7-MHz operation without wait states inserted)

The priority of channels 0–3 is program-selectable in either transfer mode above.

6.1.2 Configuration and Operation

The configuration of each DMAC channel is shown in figure 1.3.

The DMAC supports single-block transfer mode (single address) and chained-block transfer mode, both of which control DMA transfers between the on-chip MSCI and memory.

In either mode, a DMA transfer is initiated by a transfer request received when the DMA is enabled after the DMAC's internal registers have been loaded with the required values in DMA initial state.

Single-Block Transfer Mode: The DMAC transfers one word or one byte of data between memory and the MSCI in each memory read or memory write cycle, using the single addressing mode. After having transferred the specified number of bytes (up to 64 Kbytes), the DMAC returns to DMA initial state.

Because the DMAC channels 0 and 2 are hardwired to the MSCI receivers and channels 1 and 3 to the MSCI transmitters, the transfer direction for each channel is fixed: from the MSCI to memory for channels 0 and 2, and from memory to the MSCI for channels 1 and 3.

Transfer requests are generated by a request signal indicating the status of the MSCI receive/transmit buffers.

Chained-Block Transfer Mode: When the MSCI is in bit synchronous mode, the DMAC transfers one word or one byte of frame-bounded data between memory and the MSCI in each memory read or memory write cycle, using the single addressing mode. After having transferred frame(s), the DMAC returns to DMA initial state. Note that normal operation is not guaranteed for chained-block transfer mode initiated in modes other than bit synchronous mode.

The transfer direction for each channel is fixed: from the MSCI to memory for channels 0 and 2, and from memory to the MSCI for channels 1 and 3.

In this mode, it is always necessary to assign the required buffers and descriptors in memory before transfer operations, regardless of the transfer direction. The user may assign as many buffers as required, linking the buffers in a chain form with the descriptors. Thus, the user must load the starting address of the buffer and the next descriptor into each descriptor.

For an MSCI-to-memory transfer, loading the necessary values into the DMAC registers and then enabling the DMA causes the DMAC to write data sequentially to the receive buffer in memory. For a memory-to-MSCI transfer. The same events cause the DMAC to read the data sequentially. Even while the DMA is enabled, buffers whose contents have already been read/written can be released and used for new data. This enables transfer of successive data frames.

Transfer requests are generated by an internal request signal indicating the status of the MSCI receive/transmit buffers.

6.2 Registers

6.2.1 Channels 0, 2: Destination Address Register (DAR: DARL, DARH, DARB)/Buffer Address Register (BAR: BARL, BARH, BARB) Channels 1, 3: Buffer Address Register (BAR: BARL, BARH, BARB)

One set of three 8-bit subregisters, serving as the destination address register (DAR) or buffer address register (BAR) depending on the transfer mode, is provided for each of channels 0, 1, 2, and 3 (figure 6.1).

Single-Block Transfer Mode: In single-block transfer mode, these subregisters serve as the destination address register (DAR: DARL, DARH, DARB) for specifying the destination address to which data is to be transferred. DARB, DARH, and DARL specify bits 23–16, 15– 8, and 7–0 of the 24-bit destination address, respectively. This register can directly access a maximum of 16 Mbytes of memory space.

This register must be set in DMA initial state. (The DMAC has the following operation states: initial, enable, and halt states. For details, refer to section 6.2.11, DMA Command Register (DCR).)

After reset, the value of this register is undefined.

Chained-Block Transfer Mode: In chained-block transfer mode, these subregisters serve as the buffer address registers (BAR: BARL, BARH, BARB) for indicating the address of the data in the buffer currently being accessed. BARB, BARH, and BARL specify bits 23–16, 15– 8, and 7–0 of the 24-bit memory address currently being accessed, respectively. MPU cannot write to this register in this mode.

After reset, the value of these registers is undefined.

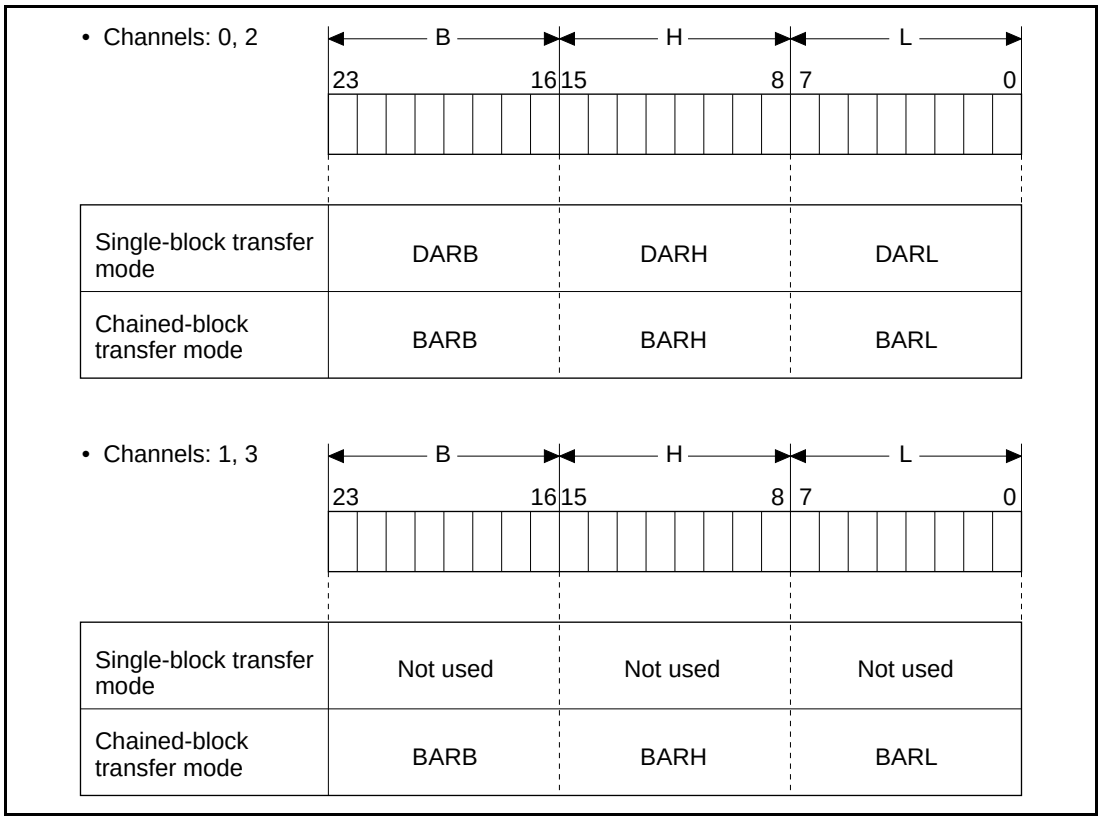


Figure 6.1 Destination Address Register/Buffer Address Register

6.2.2 Channels 0, 2: Chain Pointer Base (CPB)
Channels 1, 3: Source Address Register (SAR: SARL, SARH, SARB)/Chain Pointer Base (CPB)

One set of three 8-bit subregisters, serving as the source address registers (SAR: SARL, SARH, SARB) or chain pointer base (CPB) depending on the transfer mode, is provided for each of channels 0, 1, 2, and 3 (figure 6.2).

Single-Block Transfer Mode: In single-block transfer mode, the three 8-bit sub-registers serve as the source address registers (SAR: SARL, SARH, SARB) for specifying the 24-bit source address of the data to be transferred. SARB, SARH, and SARL specify bits 23–16, 15–8, and 7–0 of the source address, respectively. This register can directly access a maximum of 16 Mbytes of memory space.

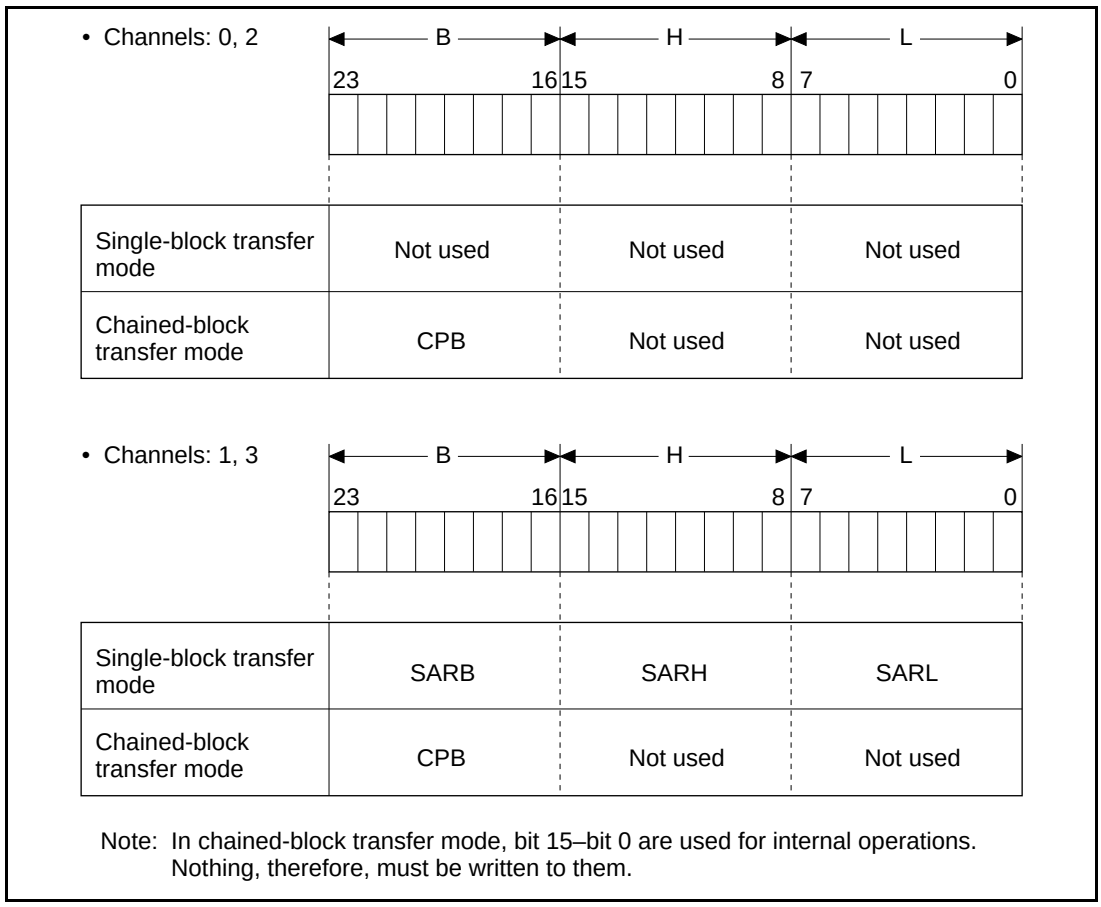
This register must be set in DMA initial state.

After reset, the value of this register is undefined.

Chained-Block Transfer Mode: In chained-block transfer mode, the 8-bit subregister composed of bit 23–bit 16 serves as the chain pointer base (CPB) for specifying the high-order eight bits of the 24-bit descriptor address. When the high-order eight bits are specified, the 64-Kbytes memory space is used as the descriptor area.

This register must be set in DMA initial state.

After reset, the value of these registers is undefined.



Note: In chained-block transfer mode, bit 15–bit 0 are used for internal operations. Nothing, therefore, must be written to them.

Figure 6.2 Source Address Register/Chain Pointer Base

6.2.3 Current Descriptor Address Register (CDA: CDAL, CDAH)

One set of two 8-bit subregisters, serving as the current descriptor address register (CDA: CDAL, CDAH), is provided for each of channels 0, 1, 2, and 3 (figure 6.3).

Single-Block Transfer Mode: In single-block transfer mode, these subregisters are not used. Their contents have no effect on operation.

Chained-Block Transfer Mode: In chained-block transfer mode, these subregisters serve as the current descriptor address register (CDA: CDAL, CDAH). This register must be initialized to the low-order 16 bits of the 24-bit starting address of the descriptor that indicates the first buffer to be written or read. Later, after a DMA transfer is initiated, the initial value is updated to the starting address of the next descriptor by the DMAC when the buffers are switched. The high-order eight bits of the descriptor are specified by the chain pointer base (CPB) and are not updated by the DMAC.

This register can be read even when a DMA is enabled. For reading this register in byte units, read CDAL first. Values read from CDAH are those it contained when CDAL was read.

This register must be set in DMA initial state.

After reset, the value of this register is undefined.

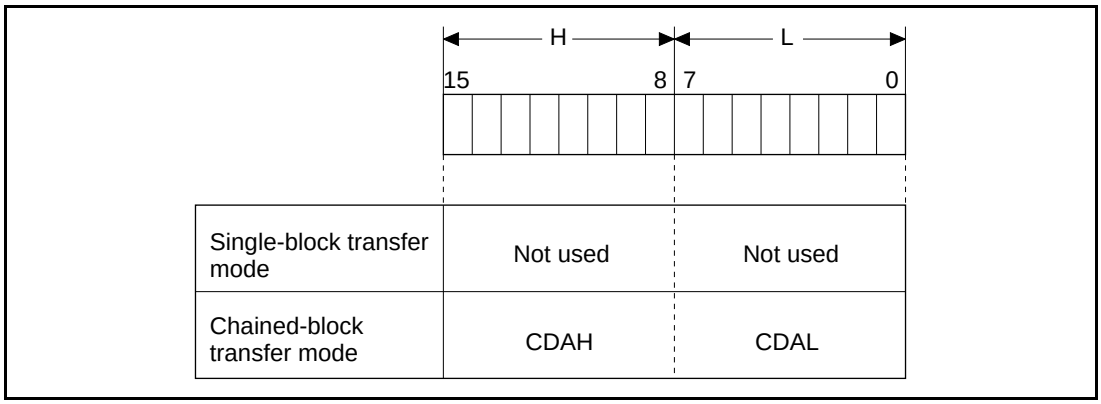


Figure 6.3 Current Descriptor Address Register

6.2.4 Error Descriptor Address Register (EDA: EDAL, EDAH)

One set of two 8-bit sub-registers, serving as the error descriptor address register (EDA: EDAL, EDAH), is provided for each of channels 0, 1, 2, and 3 (figure 6.4).

Single-Block Transfer Mode: In single-block transfer mode, these subregisters are not used. Their contents have no effect on operation.

Chained-Block Transfer Mode: In chained-block transfer mode, these subregisters serve as the error descriptor address register (EDA: EDAL, EDAH). This register must be initialized to the low-order 16 bits of the 24-bit starting address of the descriptor that indicates the buffer next to the last buffer to be written or read. When the value of the current descriptor address register (CDA) matches that of EDA, chained-block transfer is terminated. The high-order eight bits of the descriptor are specified by the chain pointer base (CPB).

This register can be written by the MPU even while a DMA is enabled. For writing this register in byte units, write EDAL first. When EDAH is written, EDAL and EDAH are updated simultaneously.

After reset, the value of this register is undefined.

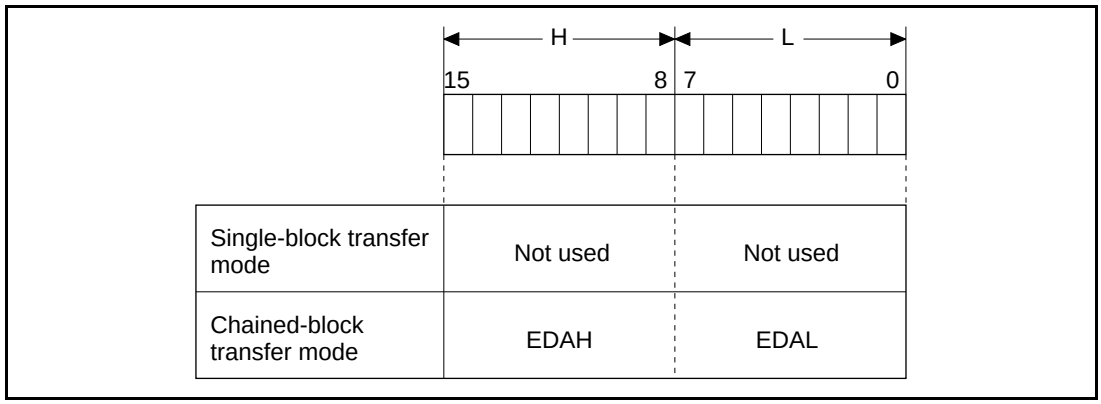


Figure 6.4 Error Descriptor Address Register

6.2.5 Receive Buffer Length Register (BFL: BFLL, BFLH)

One set of two 8-bit subregisters, serving as the receive buffer length register (BFL: BFLL, BFLH), is provided for each of channels 0 and 2 (figure 6.5).

Single-Block Transfer Mode: In single-block transfer mode, these subregisters are not used. Their contents have no effect on operation.

Chained-Block Transfer Mode: In chained-block transfer mode, these subregisters serve as the receive buffer length register (BFL: BFLL, BFLH). This register specifies the buffer length in memory in byte units only in MSCI-to-memory chained-block transfer mode.

This register must be set in DMA initial state. In chained-block transfer mode, the receive buffer length register (BFL) must not be 1 in CPU modes 0, 2, and 3, since data may be transferred in word units.

The above restrictions do not apply to CPU mode 1 or transmission operation.

After reset, the value of this register is undefined.

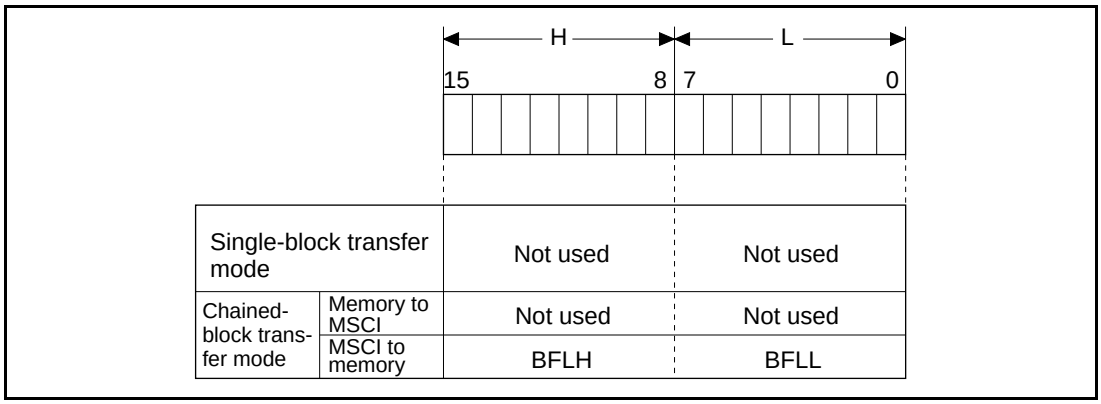


Figure 6.5 Receive Buffer Length Register

6.2.6 Byte Count Register (BCR: BCRL, BCRH)

One set of two 8-bit subregisters, serving as the byte count register (BCR: BCRL, BCRH), is provided for each of channels 0, 1, 2, and 3 (figure 6.6).

Single-Block Transfer Mode: In single-block transfer mode, BCR specifies the number of bytes to be transferred (up to 64 Kbytes). The BCR value is decremented by 1 each time one byte of data is transferred by the DMAC or decremented by 2 each time one word of data is transferred. The transfer operation terminates when the value becomes 0000H. If 0000H is set as the initial value, 64 Kbytes of data will be transferred.

This register must be set in DMA initial state. In single-block transfer mode, the byte counter register (BCR) must not be 1 in CPU modes 0, 2, and 3, since data may be transferred in word units. To transmit/receive only one byte of data, data must be directly written to or read from the MSCI TX/RX buffer register (TRB), instead of using the on-chip DMAC.

The above restrictions do not apply to CPU mode 1.

After reset, the value of this register is undefined.

Chained-Block Transfer Mode: In this mode, BCR indicates the number of bytes remaining in the buffer currently being accessed. When the BCR value becomes 0000H, read/write access to the current buffer terminates, and the next buffer becomes available. At this time, the BCR value is updated, either to the byte length stored in the descriptor data length for a memory-to-MSCI transfer (transmission: buffer read), or to the value of the receive buffer length register (BFL) for an MSCI-to-memory transfer (reception: buffer write).

The MPU cannot write to this register in this mode.

After reset, the value of this register is undefined.

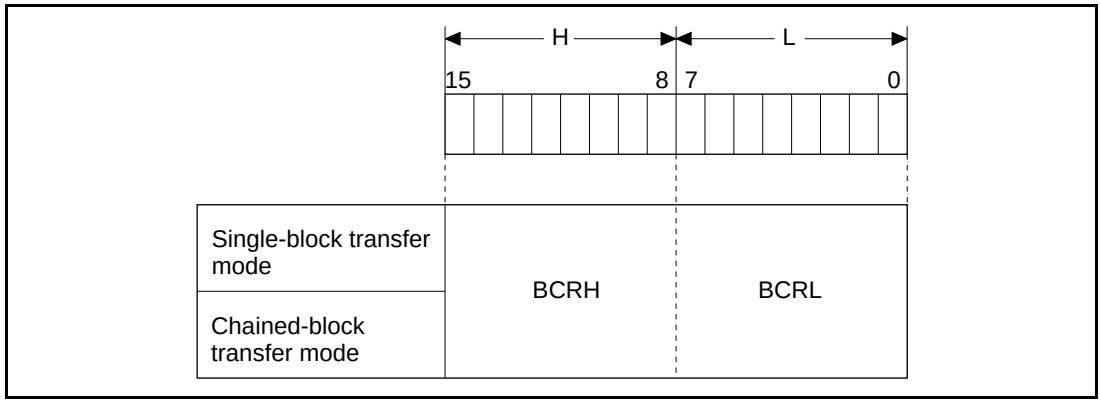
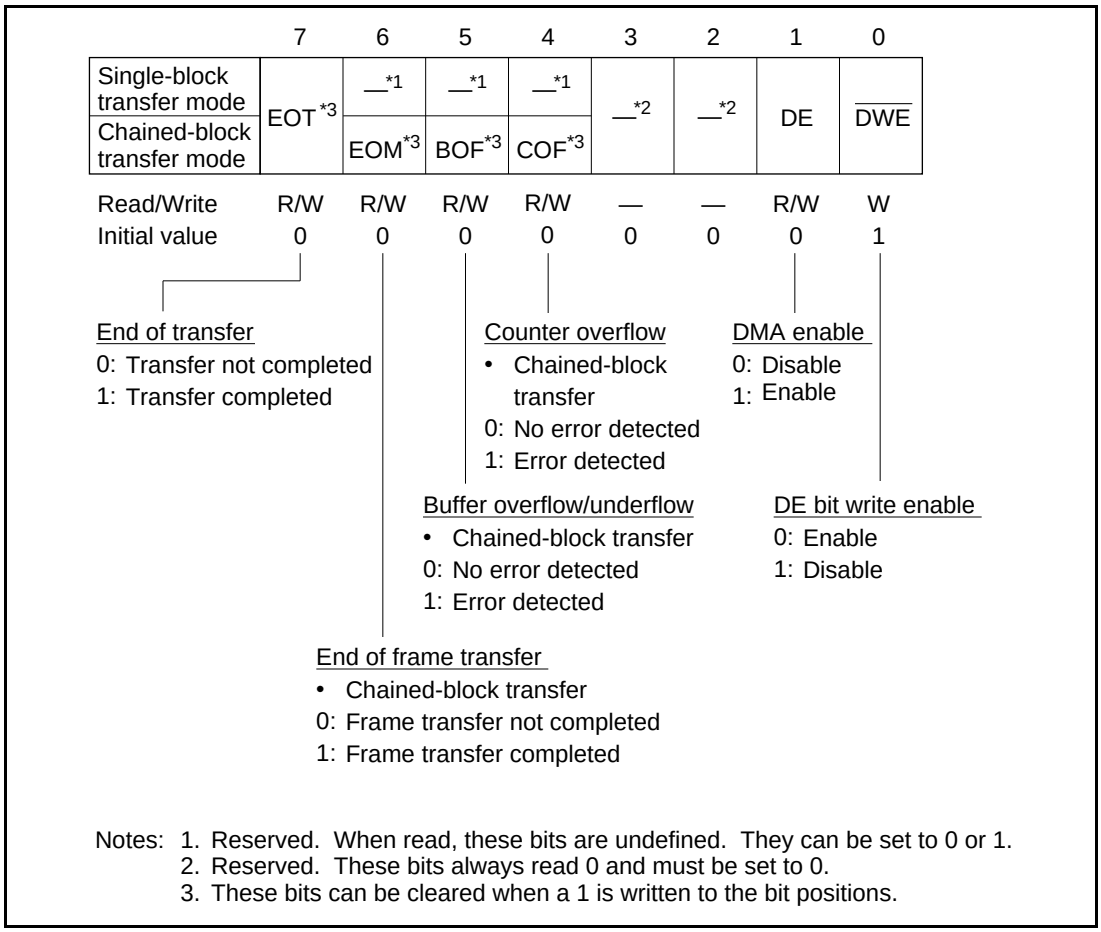


Figure 6.6 Byte Count Register

6.2.7 DMA Status Register (DSR)

The DMA status register (DSR), provided for each of channels 0, 1, 2, and 3, indicates the status of a DMA transfer. This register also enables or disables each DMAC channel.



Notes: 1. Reserved. When read, these bits are undefined. They can be set to 0 or 1.
2. Reserved. These bits always read 0 and must be set to 0.
3. These bits can be cleared when a 1 is written to the bit positions.

Bit 7 (EOT: End of Transfer): A 1 indicates that the transfer operation by the DMAC has been completed normally, in either single-block transfer or chained-block transfer mode. See section 6.4.1, Overview, for the conditions governing DMA normal completion.

This bit is cleared when a 1 is written to the bit position.

When this bit and the EOTE bit of the DMAC interrupt enable register (DIR) are both 1, the DMAC generates an interrupt request (DMIB). For details, see section 6.2.10, DMA Interrupt Enable Register (DIR).

Bit 6 (EOM: End of Frame Transfer): The function of this bit is described below.

- Single-block transfer mode
  Reserved. When read, the value of this bit is undefined. This bit can be set to 0 or 1.
- Chained-block transfer mode
  An EOM bit of 1 indicates that a transfer of one frame has been completed normally.

This bit is cleared when a 1 is written to the bit position while the frame end interrupt counter (FCT) is disabled.

While FCT is enabled and its value is not 0000, the EOM bit remains 1. (See section 6.2.8, DMA Mode Register (DMR)). At this time, when a 1 is written to this bit, the counter is decremented. When the counter value becomes 0000, this bit is set to 0. (While FCT is enabled and the EOM bit is 0, a 1 must not be written to this bit .) The EOM bit is also cleared when a frame end interrupt counter clear command is issued.

When an FCT overflow occurs, FCT is reset to 0000 and the EOM bit is set to 1. The EOM bit can be cleared by a frame end interrupt counter clear command specified by the DMA command register (DCR).

When this bit and the EOME bit of DIR are both 1, the DMAC generates an interrupt request (DMIB). See the description on the CNTE bit in section 6.2.8, DMA Mode Register, for more detail.

Bit 5 (BOF: Buffer Overflow/Underflow): The function of this bit is described below.

- Single-block transfer mode
  Reserved. When read, the value of this bit is undefined. This bit can be set to 0 or 1.
- Chained-block transfer mode
  The BOF bit is set to 1 to indicate that a buffer overflow or underflow occurs in the DMAC. In this mode, a buffer overflow is defined as the condition when a transfer request is issued by the MSCI during MSCI-to-memory transfer (reception) while the value of the current descriptor address register (CDA) and that of the error descriptor address register (EDA) are the same. A buffer underflow is defined as the condition when a transfer request is issued by the MSCI during memory-to-MSCI transfer (transmission) while CDA and EDA have the same values.

This bit is cleared when a 1 is written to the bit position.

When this bit and the BOFE bit of DIR are both 1, the DMAC generates an interrupt request (DMIA). For details, see section 6.2.10, DMA Interrupt Enable Register (DIR).

Bit 4 (COF: Counter Overflow): The function of this bit is described below.

- Single-block transfer mode
  Reserved. When read, the value of this bit is undefined. This bit can be set to 0 or 1.
- Chained-block transfer mode
  The COF bit indicates an overflow in FCT; this bit is set to 1 when a frame transfer is completed after the FCT value becomes 1111. At this time, the FCT value is reset to 0000.

This bit is cleared when a 1 is written to the bit position.

When this bit and the COFE bit of DIR are both 1, the DMAC generates an interrupt request (DMIA). For details, see section 6.2.10, DMA Interrupt Enable Register (DIR).

Bits 3–2: Reserved. These bits always read 0 and must be set to 0.

Bit 1 (DE: DMA Enable): Enables or disables the corresponding DMA channel in either single-block transfer mode or chained-block transfer mode as follows:

DE = 0: Disables the DMA channel 0, 1, 2, or 3

DE = 1: Enables the DMA channel 0, 1, 2, or 3

To write a value to the DE bit, a 0 must be written to the $\overline{\text{DWE}}$ bit at the same time. Transfer starts when the request is issued while this bit is 1.

When the DMA transfer end condition is satisfied, the DE bit of the corresponding channel is automatically cleared. For the DMA transfer end conditions, see section 6.4.1, Overview.

The DMAC enters halt state when a 0 is written to this bit during a transfer.

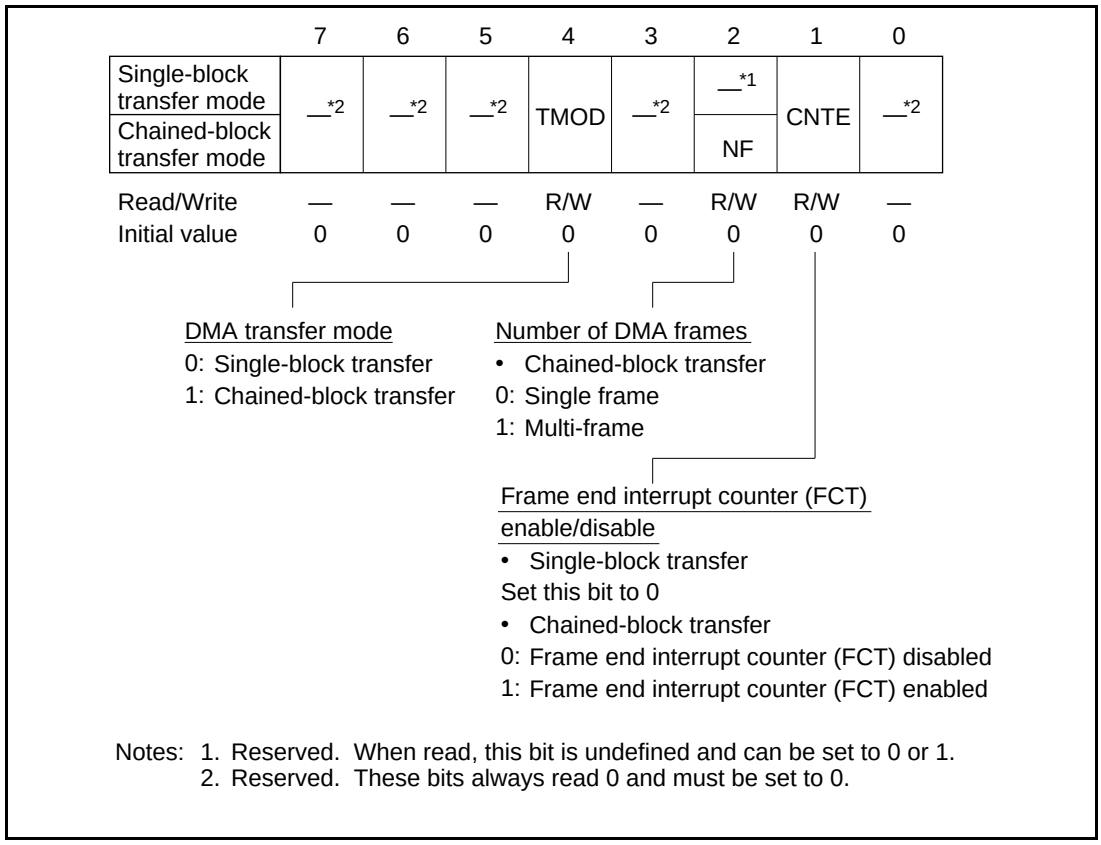
Bit 0 (DWE: DE Bit Write Enable): Enables write operation to the DMA enable (DE) bit in either single-block transfer mode or chained-block transfer mode. To write a value to the DE bit, a 0 must be written to the $\overline{\text{DWE}}$ bit at the same time. Since the value of this bit is not retained, a 0 must be written to the $\overline{\text{DWE}}$ bit each time any value is written to the DE bit.

When read, this bit always reads 1.

6.2.8 DMA Mode Register (DMR)

The DMA mode register (DMR), provided for each of channels 0, 1, 2, and 3, specifies DMA transfer mode and number of DMA frames (single or multiple). This register also enables or disables the frame end interrupt counter (FCT).

This register must be set in DMA initial state.



Bits 7–5: Reserved. These bits always read 0 and must be set to 0.

Bit 4 (TMOD: DMA Transfer Mode): Specifies the DMAC operation mode in either single-block transfer mode or chained-block transfer mode as follows. This bit is reset to 0.

TMOD = 0: Specifies single-block transfer mode

TMOD = 1: Specifies chained-block transfer mode

Bit 3: Reserved. This bit always reads 0 and must be set to 0.

Bit 2 (NF: Number of DMA Frames): The function of this bit is described below.

- Single-block transfer mode
  Reserved. When read, the value of this bit is undefined. This bit can be set to 0 or 1.
- Chained-block transfer mode
  The NF bit specifies either single- or multi-frame chained-block transfer mode as follows. This bit is reset to 0.
  NF = 0: Specifies single-frame mode
  NF = 1: Specifies multi-frame mode

Bit 1 (CNTE : Frame End Interrupt Counter Enable/Disable): The function of this bit is described below.

- Single-block transfer mode
  The CNTE bit must be set to 0.
- Chained-block transfer mode
  The CNTE bit enables or disables the frame end interrupt counter (FCT) as follows. See section 6.2.7, DMA Status Register (DSR), and section 6.2.9, Frame End Interrupt Counter (FCT). This bit is reset to 0.
  CNTE = 0: Disables FCT
  CNTE = 1: Enables FCT

Bit 0: Reserved. This bit always reads 0 and must be set to 0.

6.2.9 Frame End Interrupt Counter (FCT)

The frame end interrupt counter (FCT), provided for each of channels 0, 1, 2, and 3, counts the unprocessed interrupt requests which have occurred during multi-frame chained-block transfer. This is a 4-bit read-only counter.

	7	6	5	4	3	2	1	0
Single-block transfer mode	—*1	—*1	—*1	—*1	—*2	—*2	—*2	—*2
Chained-block transfer mode					FCT3	FCT2	FCT1	FCT0
Read/Write	—	—	—	—	R	R	R	R
Initial value	0	0	0	0	0	0	0	0

Frame end interrupt counter (FCT) value

Notes: 1. Reserved. These bits always read 0.
2. Reserved. When read, these bits are undefined.

Bits 7–4: Reserved. These bits always read 0.

Bits 3–0 (FCT3–FCT0: Frame End Interrupt Counter Value): The function of these bits is described below.

- Single-block transfer mode

  Reserved. When read, the value of these bits is undefined.
- Chained-block transfer mode

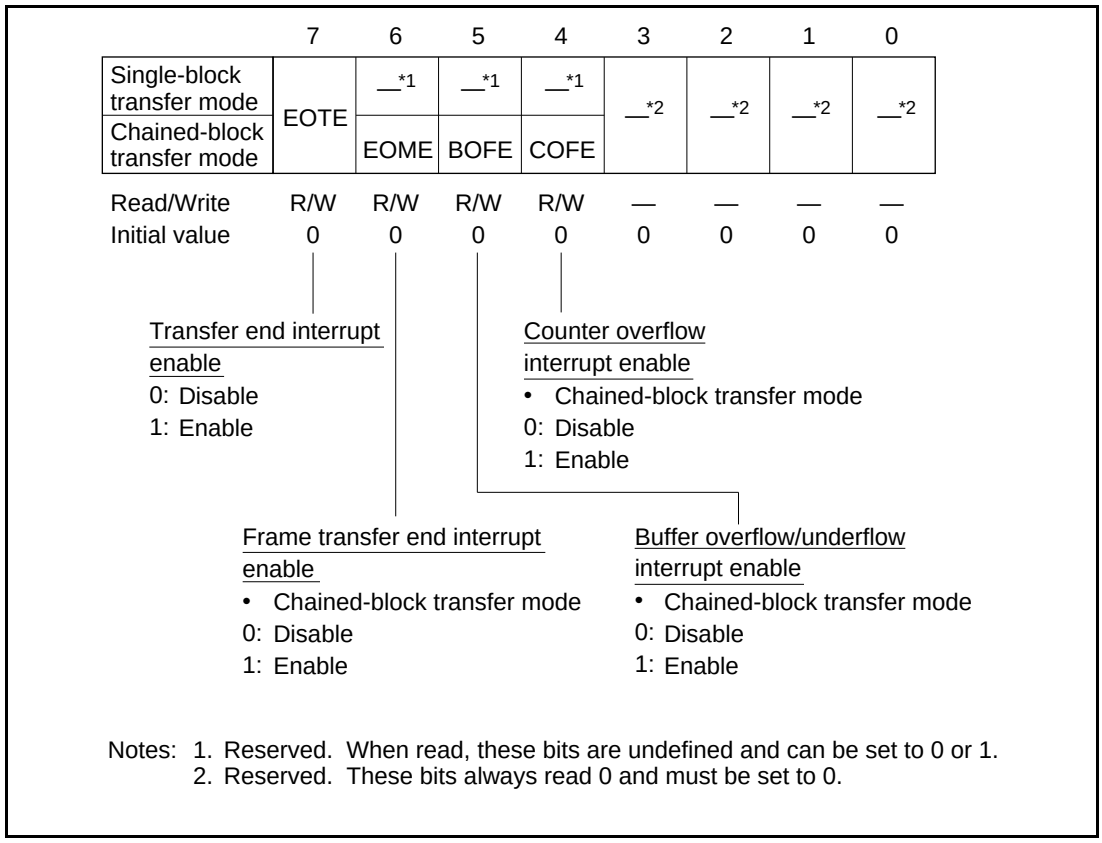
  In multi-frame chained-block transfer mode, the DMAC can request a DMIB interrupt (frame end interrupt) at completion of each frame. (The DMAC remains enabled and successive interrupts can occur.) If the transfer of successive requested frames is completed before the MPU executes the interrupt processing routine, some interrupt requests might remain unprocessed. The frame interrupt counter (FCT) counts such interrupts.

FCT is enabled or disabled by the CNTE bit of DMA mode register (DMR). For details, see section 6.2.8, DMA Mode Register (DMR).

The EOM bit of the DMA status register (DSR) remains 1 when the FCT value is not 0000. When a 1 is written to the EOM bit, the FCT value is decremented. (While FCT is enabled and its value is 0000, the EOM bit of DSR must not be set to 1.) If frame transfer continues after the FCT value reaches 1111, the DMAC terminates transfer operation when the next frame transfer has been completed. At this time, the COF bit of DSR is set to 1. If the COFE bit of the DMA interrupt enable register (DIR) is 1, the DMAC generates a counter overflow interrupt (DMIA). Here, the FCT value reaches 0000, and the EOM bit is set to 1. The EOM bit can be cleared by a frame end interrupt counter clear command specified by the DMA command register (DCR). For commands, see section 6.2.11, DMA Command Register (DCR).

6.2.10 DMA Interrupt Enable Register (DIR)

The DMA interrupt enable register (DIR), provided for each of channels 0, 1, 2, and 3, enables or disables transfer end interrupts, frame transfer end interrupts, buffer overflow/underflow interrupts, and counter overflow interrupts.



Notes: 1. Reserved. When read, these bits are undefined and can be set to 0 or 1.
2. Reserved. These bits always read 0 and must be set to 0.

Bit 7 (EOTE: Transfer End Interrupt Enable): Enables or disables a DMA normal end interrupt (DMIB) caused by the EOT bit of DSR in either single-block transfer mode or chained-block transfer mode as follows.

EOTE = 0: Disables an interrupt (DMIB) caused by the EOT bit

EOTE = 1: Enables an interrupt (DMIB) caused by the EOT bit

Bit 6 (EOME: Frame Transfer End Interrupt Enable): The function of this bit is described below.

- Single-block transfer mode

Reserved. When read, the value of this bit is undefined. This bit can be set to 0 or 1.

- Chained-block transfer mode

  The EOME bit enables or disables a DMA frame end interrupt (DMIB) caused by the EOM bit of DSR as follows:

  EOME = 0: Disables an interrupt (DMIB) caused by the EOM bit

  EOME = 1: Enables an interrupt (DMIB) caused by the EOM bit

Bit 5 (BOFE: Buffer Overflow/Underflow Interrupt Enable): The function of this bit is described below.

- Single-block transfer mode

  Reserved. When read, the value of this bit is undefined. This bit can be set to 0 or 1.

- Chained-block transfer mode

  The BOFE bit enables or disables a buffer overflow/underflow interrupt (DMIA) caused by the BOF bit of DSR as follows:

  BOFE = 0: Disables an interrupt (DMIA) caused by the BOF bit

  BOFE = 1: Enables an interrupt (DMIA) caused by the BOF bit

Bit 4 (COFE: Counter Overflow Interrupt Enable): The function of this bit is described below.

- Single-block transfer mode

  Reserved. When read, the value of this bit is undefined. This bit can be set to 0 or 1.

- Chained-block transfer mode

  The COFE bit enables or disables a counter overflow interrupt (DMIA) caused by the COF bit of DSR as follows:

  COFE = 0: Disables an interrupt (DMIA) caused by the COF bit

  COFE = 1: Enables an interrupt (DMIA) caused by the COF bit

Bits 3–0: Reserved. These bits always read 0 and must be set to 0.

6.2.11 DMA Command Register (DCR)

The DMA command register (DCR), provided for each of channels 0, 1, 2, and 3, issues a software abort or a frame end interrupt counter clear command to the DMAC. This register always reads 00H.

	7	6	5	4	3	2	1	0
Single-block transfer mode / Chained-block transfer mode	—*1	—*1	—*1	—*1	—*1	—*1	DCMD1	DCMD0
Read/Write	—	—	—	—	—	—	W	W
Initial value	—	—	—	—	—	—	—	—

Command specification*2
01: Software abort
10: Frame end interrupt counter cleared
Others: Reserved

Notes: 1. Reserved. These bits always read 0 and must be set to 0.
2. These commands must not be issued when the corresponding DMAC channel is enabled (DE = 1). No values other than those shown here (01H and 02H) must be written to these bits.

Bits 7–2: Reserved. These bits always read 0 and can be set to 0.

Bits 1–0 (DCMD1–DCMD0: Command): The function of these bits in either single-block transfer mode or chained-block transfer mode is described below.

DCDM1, DCDM0 = 0, 1: Issues a software abort command; initializes the corresponding DMAC channel (figure 6.7). All DMAC registers retain their previous values.

DCDM1, DCDM0 = 1, 0: Issues a frame end interrupt clear command; clears the frame end interrupt counter (FCT) of the corresponding DMAC channel to 0000 and the EOM bit of the DMA status register (DSR) to 0.

Other settings: Inhibited

If the DMAC is disabled by software (the DE bit of DSR cleared) for a new operation, the DMAC must be initialized by a software abort command. This is necessary because the DMAC retains its internal state even after being disabled. However, if the DMAC was disabled when the transfer end conditions were satisfied, a software abort command is not necessary.

The state transition diagram for three operating modes of the DMAC (initial state, enable state, and halt state) is shown in figure 6.7.

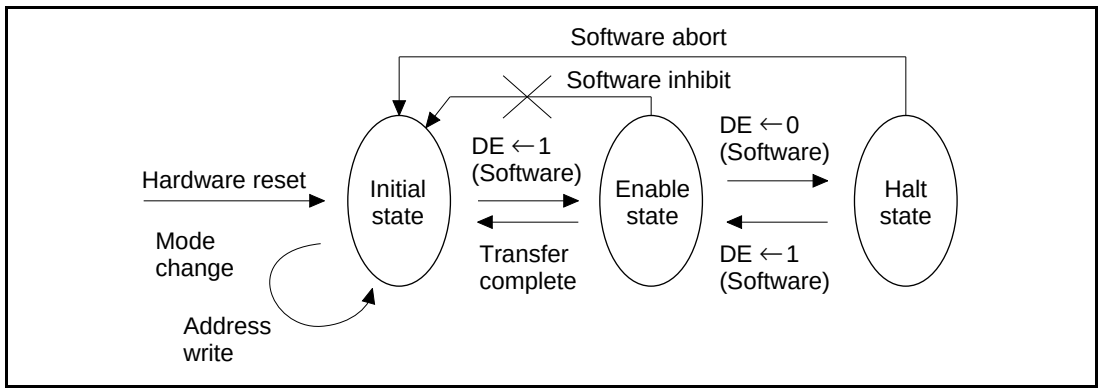


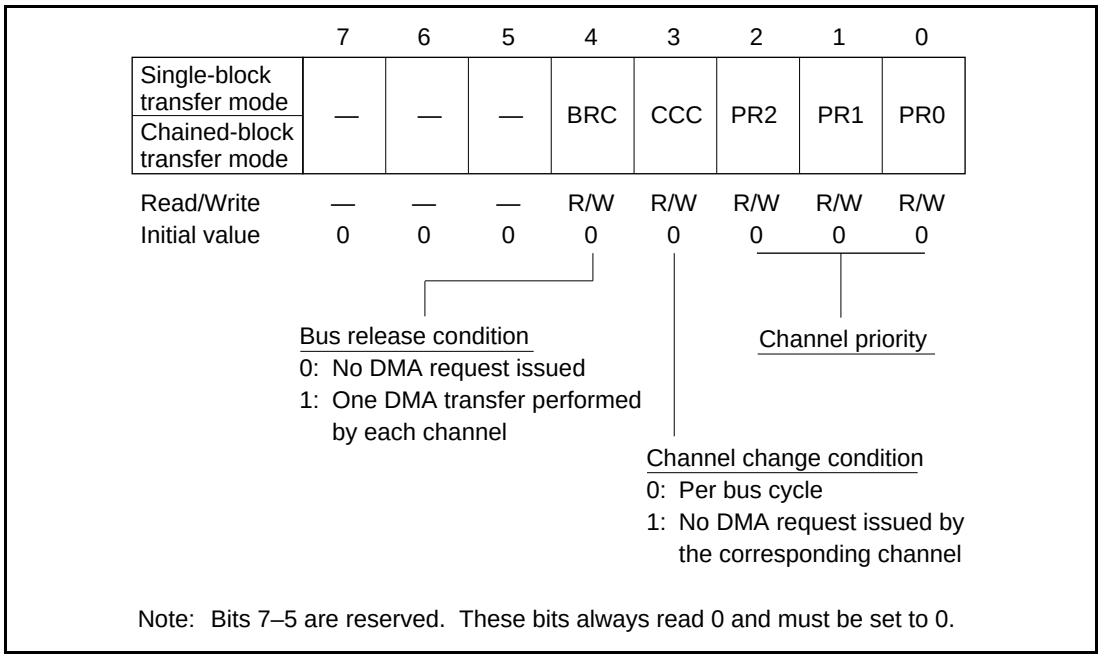
Figure 6.7 Software Abort and DMAC Operation

When the DE bit of DSR is cleared by the MPU while the DMAC is enabled, the DMAC enters halt state. Issuing a software abort command at this time causes the DMAC to enter initial state. In this case, the DMA mode register (DMR), DSR, FCT, and DMA interrupt enable register (DIR) retain their previous values. Note that software cannot cause the DMAC to enter initial state directly from enable state.

Mode, address, and data length must not be changed during DMAC halt or enable state. If it is necessary to change these values, the DMAC must be initialized by a software abort command in advance. However, after a DMA transfer is completed (see table 6.3, DMAC Operating Modes, in section 6.4.1, Overview), the DMAC is automatically placed in initial state, and no software abort commands are necessary.

6.2.12 DMA Priority Control Register (PCR)

The DMA priority control register (PCR), shared by channels 0, 1, 2, and 3, specifies channel priority. When multiple channels request a DMA transfer, the channel given the highest priority can use the bus. This register can be accessed only in byte units.



Note: Bits 7–5 are reserved. These bits always read 0 and must be set to 0.

Bits 7–5: Reserved. These bits always read 0 and must be set to 0.

Bit 4 (BRC: Bus Release Condition): Specifies the condition for the SCA to release the bus control obtained in either single-block transfer mode or chained-block transfer mode as follows.

BRC = 0: The SCA releases the bus control when all DMA transfer requests have been processed.

BRC = 1: The SCA releases the bus control when every DMAC channel has performed one DMA transfer according to the priority specified by the PR2–PR0 bits (channel priority bits) of PCR. In this case, a channel releases the bus control when it has performed one DMA transfer, and the bus control is given to a channel that has not performed a DMA transfer. Bus control is switched between channels according to the CCC bit (channel change condition bit) of PCR. The SCA also releases the bus when all DMA transfer requests have been processed, even when not all DMAC channels have performed one DMA transfer.

Bit 3 (CCC: Channel Change Condition): Specifies the condition for switching bus control between channels in either single-block transfer mode or chained-block transfer mode as follows. Bus control changes immediately after T_3 or T_i state of each cycle (transmit/receive data transfer cycle or the cycles shown in figure 6.22).

CCC = 0: One channel releases the bus to another channel at each cycle

CCC = 1: One channel releases the bus to another channel when all DMA requests for the channel have been processed

Bits 2–0 (PR2–PR0: Channel Priority): Specify channel priority on the bus control in either single-block transfer mode or chained-block transfer mode as follows.

PR2, PR1, PR0 = 0, 0, 0: Priority = Channel 0 > channel 1 > channel 2 > channel 3

PR2, PR1, PR0 = 0, 0, 1: Priority = Channel 2 > channel 3 > channel 0 > channel 1

PR2, PR1, PR0 = 0, 1, 0: Priority = Channel 0 > channel 2 > channel 1 > channel 3

PR2, PR1, PR0 = 0, 1, 1: Priority = Channel 1 > channel 3 > channel 0 > channel 2

PR2, PR1, PR0 = 1, ×, ×: Priority = Channel 0 ‡ channel 1 ‡ channel 2 ‡ channel 3 ‡ channel 0 (rotation) ("×" indicates either 0 or 1)

DMA transfers by multiple channels with PR2 = 1, BRC = 1, and CCC = 1 is described below.

1. When the SCA has obtained bus control, channels that request their first DMA transfer are serviced, according to the priority specified.

2. When a different channel requests its first DMA transfer during the DMA transfer initiated in step 1:

a. If a channel with a lower priority requests its first DMA transfer when a channel with a higher priority is performing its DMA transfer, the specified priority is obeyed.

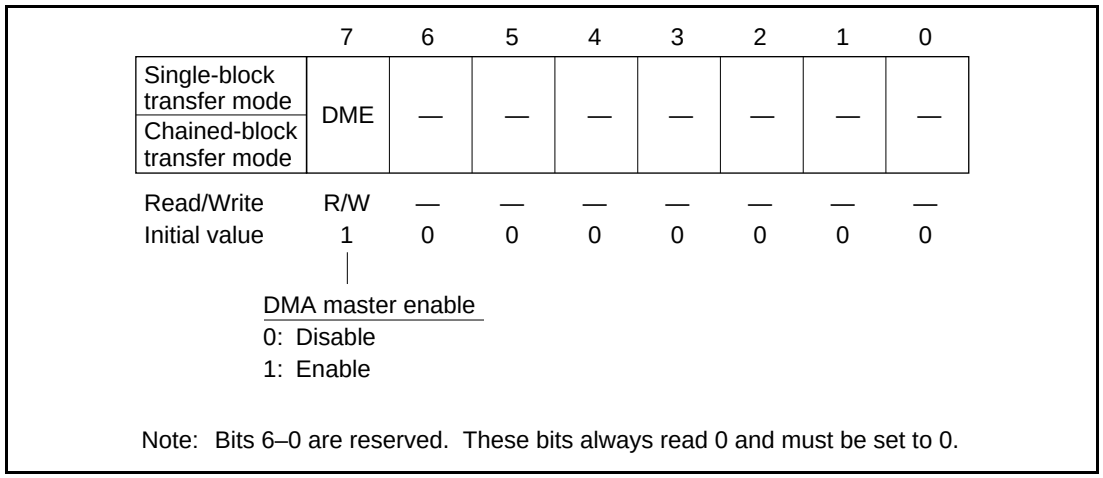
b. If a channel with a higher priority requests its first DMA transfer when a channel with a lower priority is performing its first DMA transfer, the higher priority channel must wait until the channel with the lowest priority in step 1 has completed its DMA transfer.

a or b applies also when multiple channels request their first DMAs.

3. The SCA releases the bus when every channel that requested a first DMA transfer has completed its DMA transfer, except when a channel requests a second DMA transfer before the above procedure is completed. In this case, the SCA repeats the above procedure, beginning with the step when the SCA obtains bus control.

6.2.13 DMA Master Enable Register (DMER)

The DMA master enable register (DMER), shared by channels 0, 1, 2, and 3, enables or disables DMA master operation. This register can be accessed only in byte units.



Note: Bits 6–0 are reserved. These bits always read 0 and must be set to 0.

Bit 7 (DME: DMA Master Enable): Enables or disables channels 0, 1, 2, or 3 in either single-block transfer mode or chained-block transfer mode as follows.

DME = 0: Disables all channels

DME = 1: Enables channel(s) depending on the DE bit of the DMA status register (DSR) of each channel

After reset, the value of the DME bit is 1.

Bits 6–0: Reserved. These bits always read 0 and must be set to 0.

6.3 Descriptors

In chained-block transfer mode, transmit/receive data is stored in buffers in system memory. Each buffer has a descriptor indicating the buffer attributes. The buffers are linked by these descriptors.

6.3.1 Memory-to-MSCI Chained-Block Transfer Mode (Transmission)

Descriptors and buffers in system memory for memory-to-MSCI chained-block transfer mode are shown in figure 6.8.

Each descriptor consists of a 16-bit chain pointer (CP), 24-bit buffer pointer (BP), 16-bit data length field (DL), and 8-bit status field (ST). These fields are allocated to system memory in byte units. The descriptor format is shown in figure 1.22. Detailed descriptions of these fields are given below.

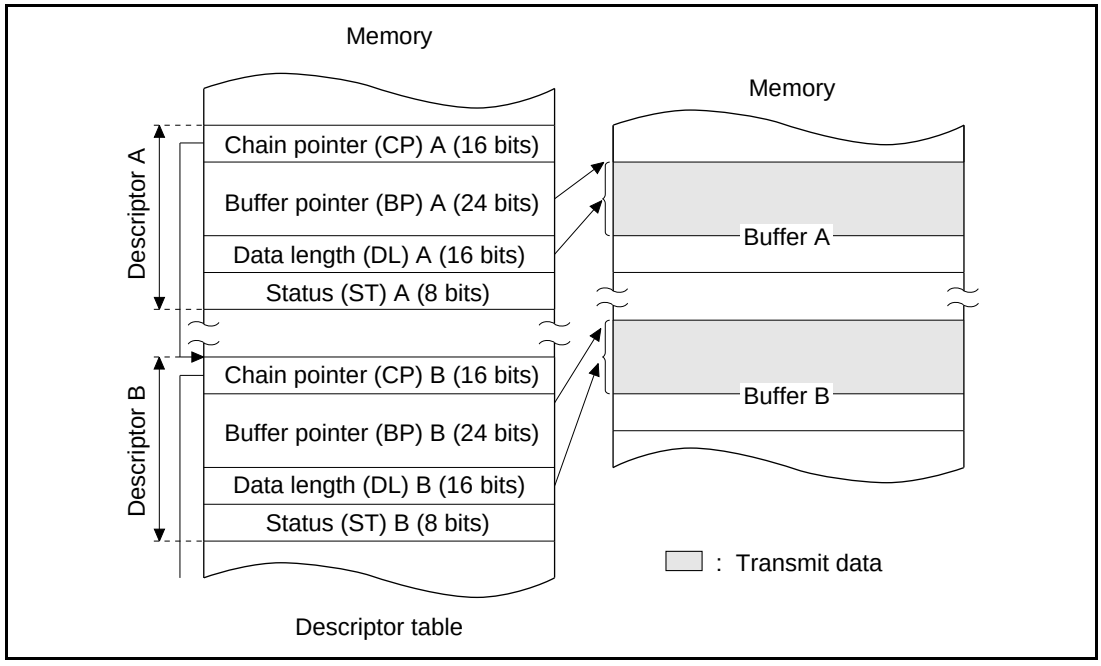


Figure 6.8 Descriptors and Buffers in Memory-to-MSCI Chained-Block Transfer Mode

Chain Pointer (CP) (16 Bits): Specifies the low-order 16 bits of the 24-bit start address of the next descriptor. The high-order eight bits are specified by the chain pointer base (CPB). The chain pointer value is loaded into the current descriptor address register (CDA) at buffer switching.

Buffer Pointer (BP) (24 Bits): Specifies the start address of the buffer corresponding to the descriptor. The BP value is loaded into the buffer address register (BAR) at the start of transfer or at buffer switching.

Data Length (DL) (16 Bits): Specifies the data length in the buffer corresponding to the descriptor in byte units. The DL value is loaded into the byte count register (BCR) at the start of transfer or at buffer switching.

This field is controlled by the MPU in memory-to-MSCI chained-block transfer mode.

Status (ST) (8 Bits): Indicates a frame transfer end or DMA transfer end for buffer data corresponding to the descriptor.

This field is controlled by the MPU in memory-to-MSCI chained-block transfer mode.

ST configuration for memory-to-MSCI chained-block transfer mode (transmission) is shown in table 6.1.

Table 6.1 Status Configuration (transmission)

Bit	Function
7	EOM
6	Not used
5	Not used
4	Not used
3	Not used
2	Not used
1	Not used
0	EOT

Note: Status bits 6–1 are not used in memory-to-MSCI chained-block transfer mode.

The functions of bit 7 and bit 0 are described below.

Bit 7 (EOM: End of Message): Indicates whether or not a frame transfer ends in the buffer corresponding to the descriptor.

EOM = 0: Indicates that no frame ends in the buffer corresponding to the descriptor

EOM = 1: Indicates that a frame ends in the buffer corresponding to the descriptor

Bit 0 (EOT: End of Transfer): Specifies whether or not to terminate DMA transfer after the current frame is transferred in multi-frame transfer mode.

EOT = 0: Does not terminate transfer

EOT = 1: Terminates transfer

6.3.2 MSCI-to-Memory Chained-Block Transfer Mode (Reception)

Descriptors and buffers in system memory for MSCI-to-memory chained-block transfer mode are shown in figure 6.9.

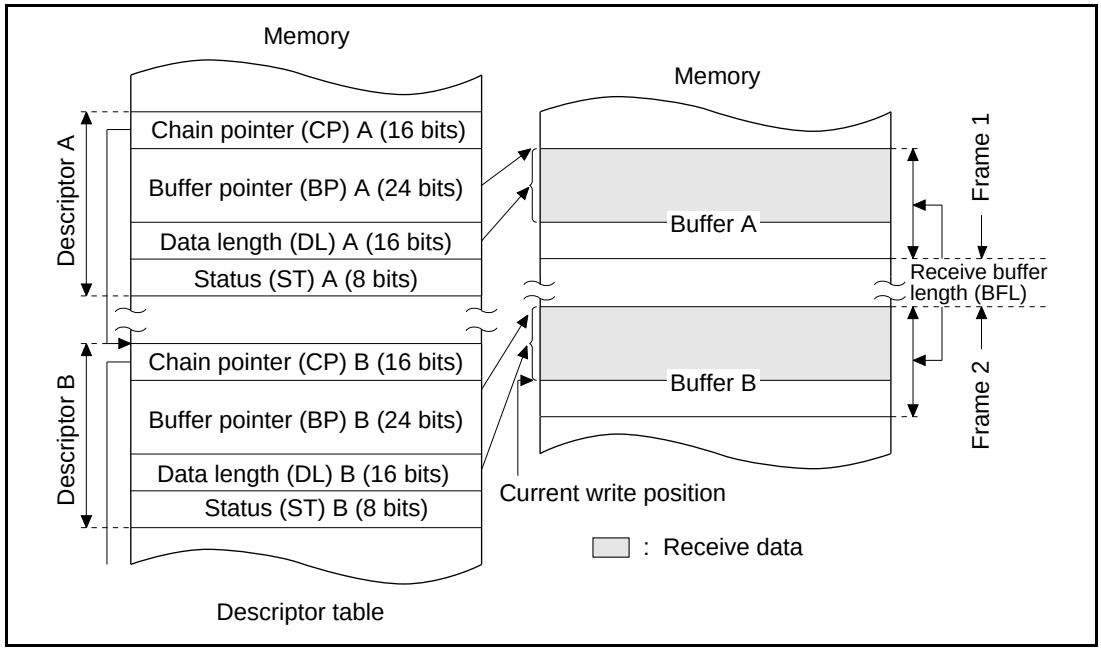


Figure 6.9 Descriptors and Buffers in MSCI-to-Memory Chained-Block Transfer Mode

The descriptor format for MSCI-to-memory chained-block transfer mode is the same as that shown in figure 6.8. Detailed descriptions of these fields are given below.

Chain Pointer (CP) (16 Bits): Specifies the low-order 16 bits of the 24-bit start address of the next descriptor. The high-order eight bits are specified by the chain pointer base (CPB). The chain pointer value is loaded into the current descriptor address register (CDA) at buffer switching.

Buffer Pointer (BP) (24 Bits): Specifies the start address of the buffer corresponding to the descriptor. The BP value is loaded into the buffer address register (BAR) at the start of transfer or at buffer switching.

Data Length (DL) (16 Bits): Specifies the data length in the buffer corresponding to the descriptor in byte units.

This field is controlled by the DMAC in MSCI-to-memory chained-block transfer mode. After received data is loaded into the buffer, the DMAC loads the byte count of the data into this field.

Status (ST) (8 Bits): Indicates the status of the data in the buffer corresponding to the descriptor.

After data is loaded into the buffer, the DMAC loads the status of the data into this field.

This field is controlled by the DMAC in MSCI-to-memory chained-block transfer mode.

ST configuration for MSCI-to-memory chained-block transfer mode (reception) is shown in table 6.2.

Table 6.2 Status Configuration (reception)

Bit	Function
7	EOM
6	Short frame
5	Abort
4	Residual bit
3	Overrun
2	CRC
1	Not used
0	Not used

When a frame ends in the buffer corresponding to the descriptor, ST bit 7 to bit 0 are loaded with the MSCI frame status register (FST) value, which is set immediately after the MSCI transmits the end of frame from the receive buffer to the data bus. (For bit 7 to bit 0, see sections 5.2.11, MSCI Status Register 2 (ST2), and 5.2.13, MSCI Frame Status Register (FST). When no frame ends in the buffer corresponding to the descriptor and if the buffer is switched during one frame, ST bit 7 to bit 0 are cleared.

6.4 Operating Modes

6.4.1 Overview

The DMAC supports single-block transfer mode (single address) and chained-block transfer mode (single address). Each transfer mode is summarized in table 6.3.

Single-block transfer mode is available in asynchronous, byte synchronous, and bit synchronous modes. Chained-block transfer mode is available only in bit synchronous mode. (Normal operation is not guaranteed in asynchronous or byte synchronous mode.)

The DMAC supports byte transfer in CPU mode 1 (8-bit MPUs of the HD64180 family) and word transfer in CPU modes 0, 2, and 3 (16-bit MPUs). In CPU modes 0, 2, and 3, the DMAC begins transferring a word of data after transferring one byte of data when the start address of the data buffer in memory is odd.

Table 6.3 DMAC Operating Modes

Operating Mode*[1]			Single-Block Transfer Mode (Single Address)		Chained-Block Transfer Mode (Single Address)			
					Memory to MSCI		MSCI to Memory	
			Memory to MSCI	MSCI to Memory	Single Frame Transfer	Multi-Frame Transfer	Multi-Frame Transfer	Multi-Frame Transfer
Requesting source			MSCI					
Data transfer unit			Single block		Single frame	Multi-frame	Single frame	Multi-frame
Bus mode			Started by a request from the MSCI					
			A request from the MSCI is level sensitive					
Minimum transfer states/byte (Word)			3 states					
Operation	Source address		Specified by the source address register (SAR)	MSCI receiver	Specified by the buffer address register (BAR)		MSCI receiver	
	Destination address		MSCI transmitter	Specified by the destination address register (DAR)	MSCI transmitter		Specified by BAR	
	Transfer end condition	Normal end	The number of bytes of data specified in the byte count register (BCR) has been transferred		One frame has been transferred	The frame specified by the descriptor status field (ST) has been transferred	One frame has been transferred	—
		Error end	—		A DMA transfer request is issued when the error descriptor address register (EDA) and current descriptor address register (CDA) match			
			—		Frame end interrupt counter (FCT) overflows when it is enabled			
Available MSCI modes			Asynchronous, byte synchronous, or bit synchronous		Bit synchronous*[2]			

Notes: 1. The operating mode is specified using the AMOD and TMOD bits of the DMA mode register (DMR). For details, see section 6.2.8, DMA Mode Register (DMR).

2. Normal operation is not guaranteed in asynchronous or byte synchronous mode.

6.4.2 Memory-to/from-MSCI Single-Block Transfer Mode

Operation: The HD64570 allows single-block transfers (single address) from the MSCI to memory via DMAC channels 0 and 2, and from memory to the MSCI via DMAC channels 1 and 3.

In MSCI-to-memory single-block transfer mode, the destination start address and transfer byte count must be set in the destination address register (DAR) and byte count register (BCR), respectively, in DMAC channels 0 and 2. Similarly, in memory-to-MSCI single-block transfer mode, the source start address and transfer byte count must be set in the source address register (SAR) and BCR, respectively, in the DMAC channels 1 and 3.

Single-block transfer between memory and the MSCI is shown in figure 6.10. As shown in the figure, in MSCI-to-memory transfer mode, as many bytes of data as specified by BCR are DMA-transferred in byte units from the MSCI receiver to the memory address specified by DAR of channels 0 and 2. Similarly, in memory-to-MSCI transfer mode, as many bytes of data as specified by BCR are DMA-transferred in byte units from the memory address specified by SAR of channels 1 and 3 to the MSCI transmitter.

During transfer, the BCR value is decremented by 1 each time the DMAC has transferred one byte of data, and is decremented by 2 each time the DMAC has transferred one word of data. When the BCR value reaches 0000H, the DMAC terminates data transfer and enters initial state. At this time, the DMAC generates an interrupt (if enabled). Note that when the BCR value reaches 0001H, the DMAC transfers one byte of data instead of one word of data, decrementing the BCR value to 0000H.

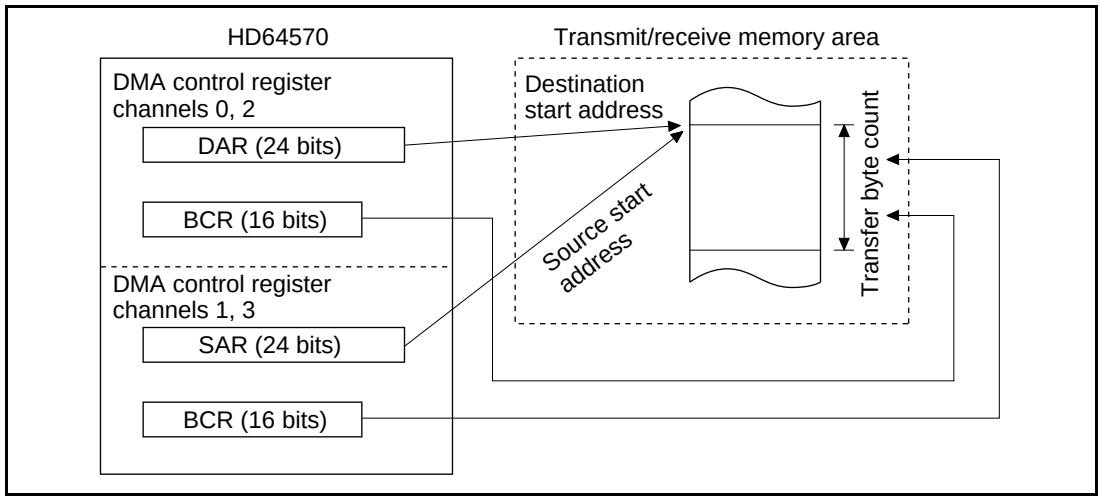


Figure 6.10 Memory-to/from-MSCI Single-Block Transfer Mode

Register Setting: To start a memory-to/from-MSCI single-block transfer, follow the steps below starting with the DMA in its initial state. (Steps 1 to 3 may be completed in any order.)

1. For memory-to-MSCI transfers, load the memory start address of the source into SAR. For MSCI-to-memory transfers, load the memory start address of the destination into DAR.

2. Load the transfer byte count into BCR.

3. Clear the TMOD and CNTE bits of the DMA mode register (DMR) to specify single-block transfer mode.

4. After steps 1 to 3, set the DE bit of the DMA status register (DSR) to 1 to start DMA operation.

External Bus Timing: The external bus timing in memory-to-MSCI single-block transfer mode is shown in figure 6.11 and that in MSCI-to-memory single-block transfer mode is shown in figure 6.12. In the figures, wait states (T_W) are inserted between T_2 and T_3. In memory-to/from-MSCI single-block transfer mode, one byte of data transfer (CPU mode 1) or one word of data transfer (CPU modes 0, 2, and 3) is completed within one memory read or write cycle. Accordingly, high-speed DMA transfer is possible.

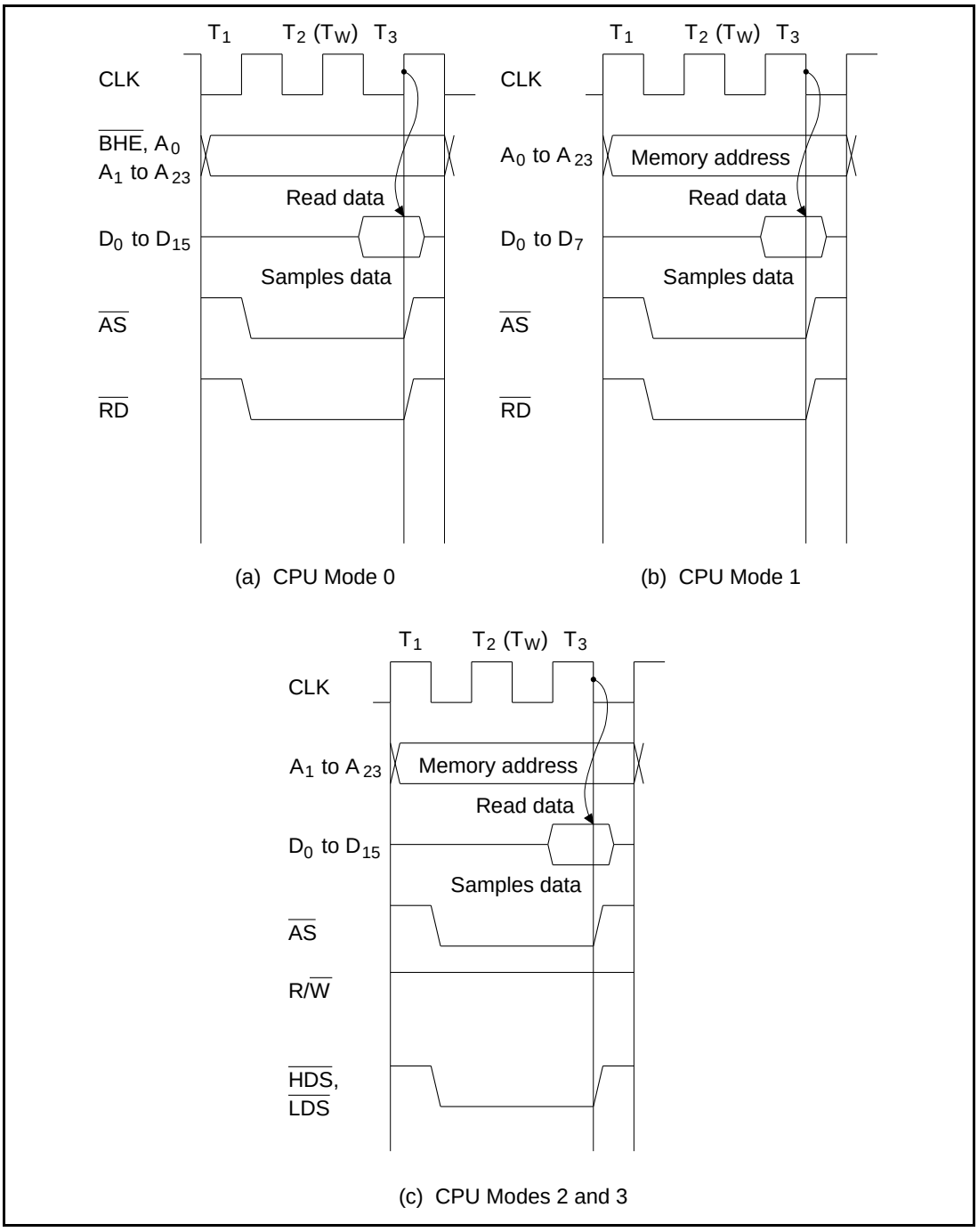


Figure 6.11 External Bus Timing in Memory-to-MSCI Single-Block Transfer Mode

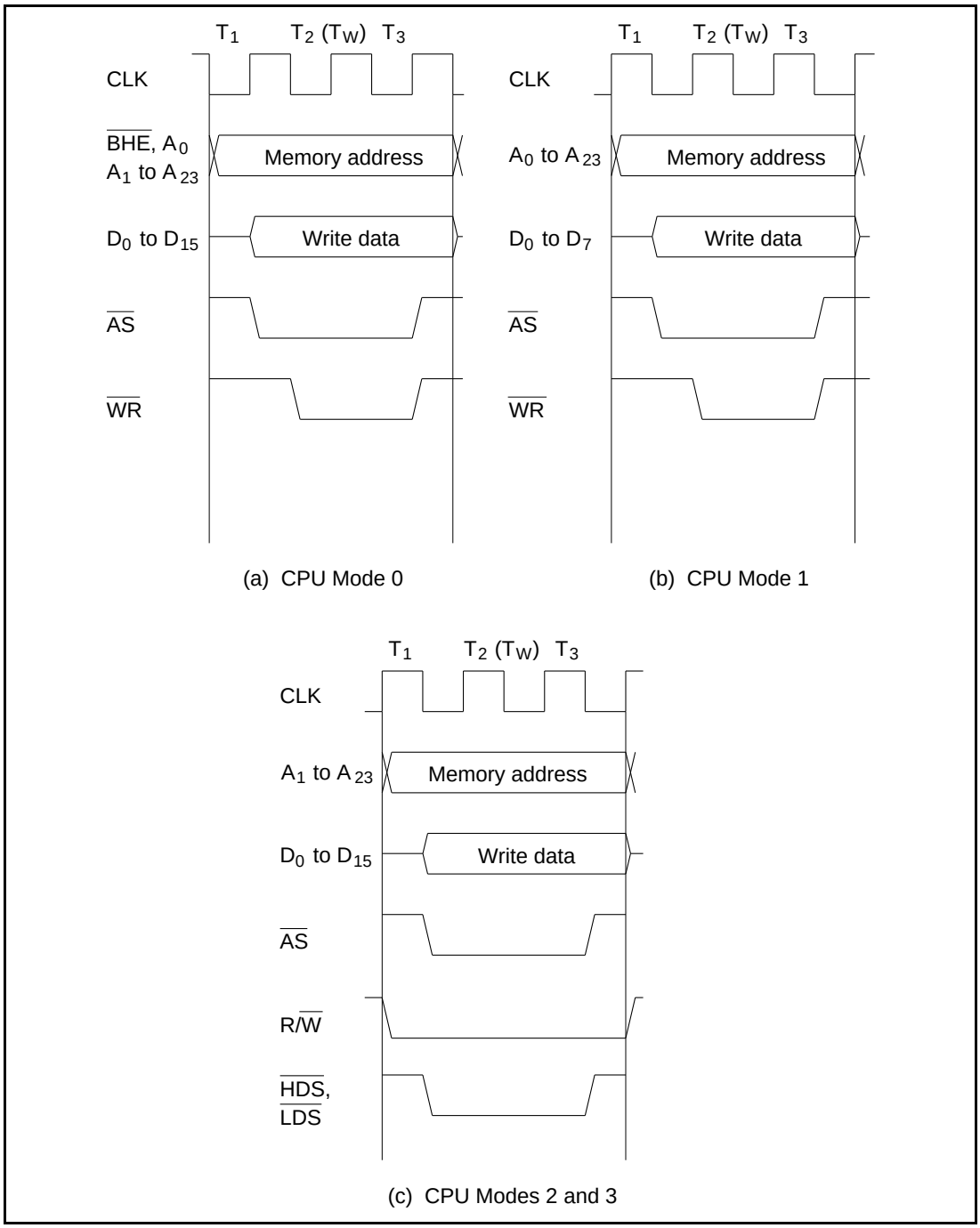


Figure 6.12 External Bus Timing in MSCI-to-Memory Single-Block Transfer Mode

Keep the following in mind about the timing in this transfer mode.

- Transfer requests are issued using the MSCI signal.
- Wait states can be inserted between T_2 and T_3 states in each bus cycle (memory read cycle and memory write cycle), using the WAIT line or the wait controller registers.
- One T_i clock cycle is inserted before the first byte or word is transferred.

6.4.3 Memory-to-MSCI Chained-Block Transfer Mode

Operation: In memory-to-MSCI chained-block transfer mode, frame-bounded data is DMA-transferred in byte or word units from a system memory buffer to the MSCI in bit synchronous mode. Transfer requests are initiated by the MSCI internal signal. Note that chained-block transfer mode is not available with the MSCI operated in asynchronous or byte synchronous mode.

Memory-to-MSCI transfer employs DMAC channels 1 and 3. For this transfer mode, follow the steps below starting with the DMA in its initial state. (Steps 1 to 5 may be completed in any order.)

1. Specify chained-block transfer mode with the DMA mode register (DMR).
2. Load the high-order eight bits of the 24-bit descriptor address into the chain pointer base (CPB). Since the CPB value is fixed during operation, descriptors can be assigned to any consecutive 64-Kbyte area in system memory.
3. Load the low-order 16 bits of the start address of the descriptor, which indicates the buffer next to the last transmit buffer, into the error descriptor address register (EDA).
4. Load the low-order 16 bits of the start address of the descriptor, which indicates the first transmit buffer, into the current descriptor address register (CDA).
5. Initialize the chain pointer (CP), buffer pointer (BP), data length (DL), and status (ST) in each descriptor.
6. After steps 1 to 5, set the DE bit of the DMA status register (DSR) to 1. DMA operation starts when the DMAC obtains the bus control.

Memory-to-MSCI chained-block transfer mode is shown in figure 6.13.

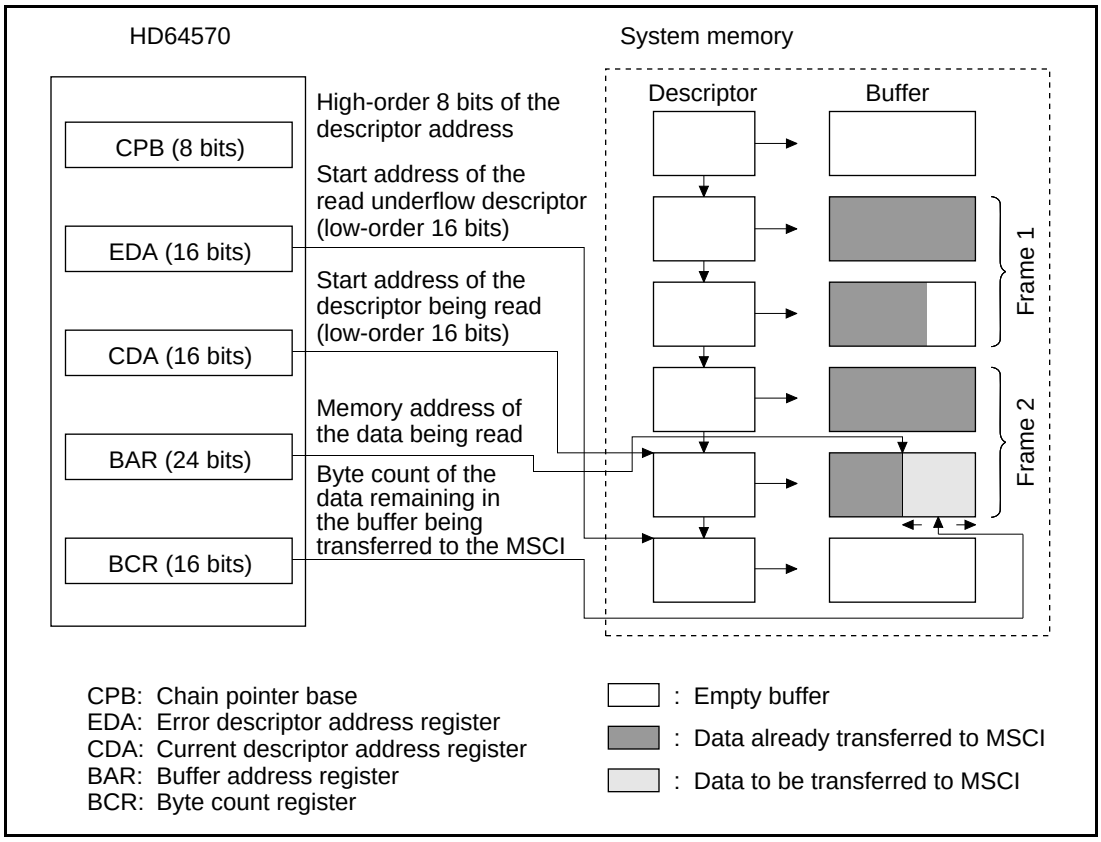


Figure 6.13 Memory-to-MSCI Chained-Block Transfer Mode

The operation flow in memory-to-MSCI chained-block transfer mode is shown in figure 6.13. As shown in the figure, a DMA transfer starts with loading the contents of the descriptor specified by CPB and CDA into the SCA internal registers. The DMAC then transfers data to the MSCI transmitter from the buffer corresponding to the descriptor specified by CPB and CDA. At this time, the DMAC writes the 24-bit memory address of the buffer currently being read to the buffer address register (BAR) and the number of bytes remaining unread in the buffer to the byte count register (BCR). When data transfer starts, the DMAC writes the BP value of the corresponding descriptor to BAR and the data length (DL) value of the corresponding descriptor to BCR.

The BAR value is incremented by 1 or 2 each time one byte or one word of data is transferred, respectively. Similarly, the BCR value is decremented by 1 or 2 each time one byte or one word of data is transferred, respectively. When the BCR value reaches 0000H, the DMAC terminates data transfer and updates the CDA value to indicate the start address of the next descriptor (buffer switching), after which data is read from the buffer specified by the descriptor. In this way, the DMAC transfers data from the buffers specified by the descriptor by updating the descriptors.

In chained-block transfer mode, since the DMAC transfers data in frame units, different frame data cannot be saved in the same buffer. If a buffer contains the end of a frame, the EOM bit of the status field (ST) of the descriptor specifying the buffer must be set to 1. In single-frame transfer mode, the DMAC terminates DMA transfer after transferring the end of the frame in the buffer and updating the CDA value. The descriptor, with the EOT bit of ST set to 1, notifies the DMAC of the completion of data transfer after data is transferred from the specified buffer. This notification indicates the completion of multi-frame transfer.

At completion of frame or DMA transfer, the DMAC issues interrupt DMIB (if enabled).

EDA must initially contain the low-order 16 bits of the address of the descriptor indicating the first buffer which contains no transmit data. In this case, if data has been written to the buffer specified by the descriptor, the MPU can update the EDA value to indicate the start address of the descriptor indicating the next empty buffer. (EDA can be written even while DMA is enabled.) This allows transmit data to be added and modified while DMA is enabled.

When the CDA and EDA values are equal and a transfer request is issued, the DMAC terminates data transfer and issues interrupt DMIA (if enabled).

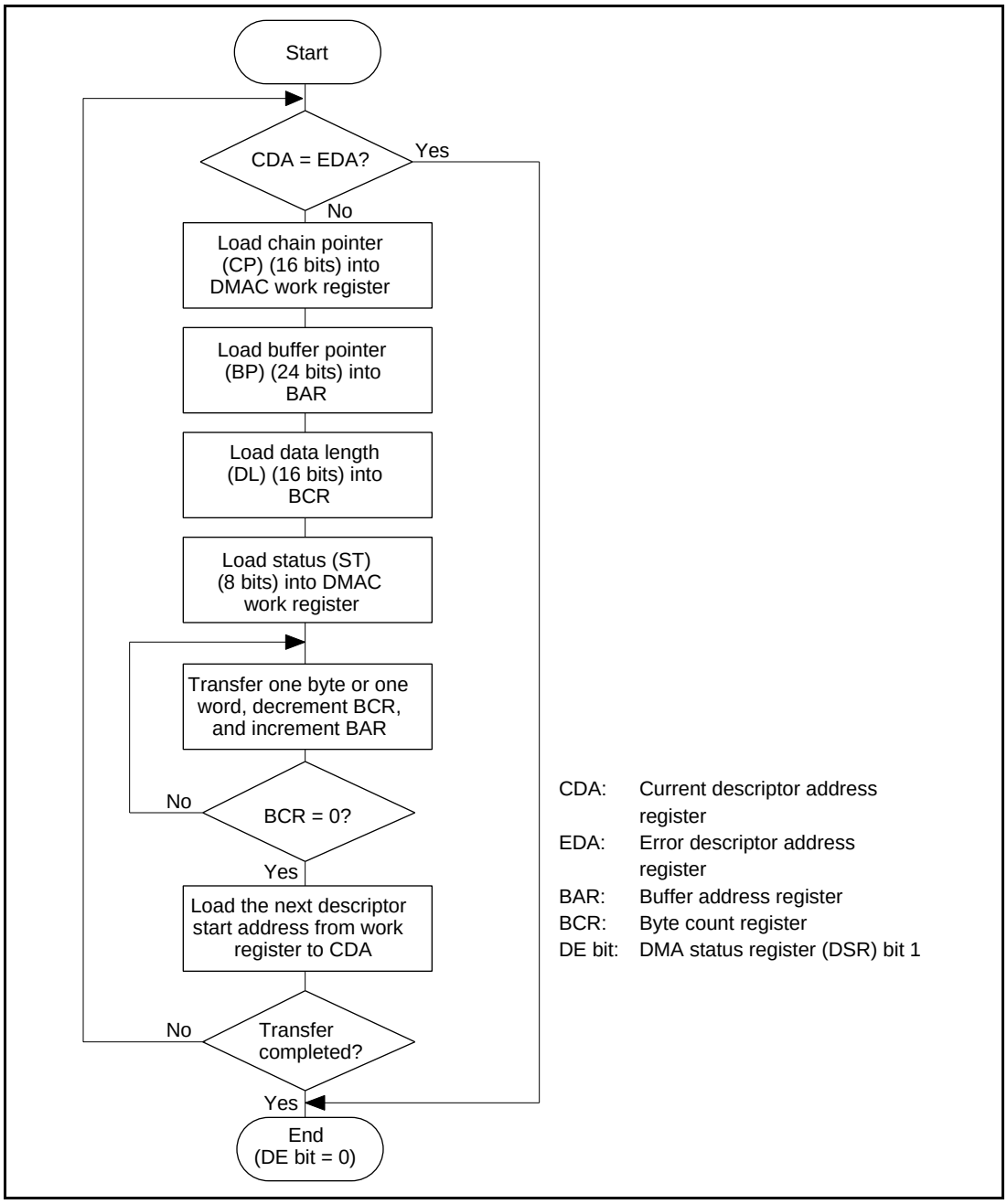


Figure 6.14 Operation Flow in Memory-to-MSCI Chained-Block Transfer Mode

The functions of the registers used in memory-to-MSCI chained-block transfer mode are listed in table 6.4. As can be seen from the table, in memory-to-MSCI chained-block transfer mode, either a single-frame transfer or multi-frame transfer can be selected. In single-frame transfer mode,

transfer is completed within one frame, after which the DMAC enters initial state. Here, the DE bit of DSR is automatically cleared. When the DE bit is set to 1 again, the DMAC restarts operation.

Table 6.4 Control Registers Used in Memory-to-MSCI Chained-Block Transfer Mode (transmission)

Register Name	Chain Pointer Base (CPB)	Error Descriptor Address Register (EDA)	Current Descriptor Address Register (CDA)
Number of bits	8	16	16
Function	Specifies the high-order 8 bits of the 24-bit descriptor start address.	Specifies the low-order 16-bits of the start address of the descriptor corresponding to the buffer following the last transmit buffer.	Specifies the low-order of the descriptor corresponding to the first transmit buffer. This address is updated by the DMAC during buffer chaining.
Role in DMAC operation	—	—	After the DMAC starts, it loads the low-order 16 bits of the start address of the descriptor corresponding to the buffer being transferred into the MSCI.
		Transfer ends when a transfer request is issued while the EDA and CDA match. An interrupt is generated, if enabled.	
Register update	Under MPU control.	Under MPU control.	When the current buffer read is completed, the next descriptor start address is automatically loaded into this register.
Register updated by the MPU	Initialized before transmission.	Loaded with the start address of the descriptor indicating the buffer following the last buffer containing transmit data. To add transmit data during a transmission, load the start address of the descriptor indicating the next buffer to be written.	The start address of the descriptor indicating the first buffer containing transmit data is loaded before transmission starts.

Table 6.4 Control Registers Used in Memory-to-MSCI Chained-Block Transfer Mode (transmission) (cont)

Register Name	Receive Buffer Length (BFL)	Byte Count Register (BCR)	Buffer Address Register (BAR)
Number of bits	16	16	24
Function	—	Specifies the byte count of the data to be transferred to the MSCI. Writing to this register by the MPU is inhibited.	Specifies the system memory address of the data being transferred to the MSCI. Writing to this register by the MPU is inhibited.
Role in DMAC operation	—	When the contents of this register equal 0000H, reading from the current buffer is completed.	When a transfer request is issued, data is read from the address specified by this register.
Register update	—	The contents of this register are decremented each time one byte or one word is read. When the buffer is switched, the byte length specified by the descriptor is loaded.	The contents of this register are incremented each time one byte or one word is read. When the buffer is switched, the next buffer start address is loaded.
Register updated by the MPU	—	—	—

Table 6.5 shows a memory-to-MSCI chained-block single-frame transfer using four descriptors and four buffers. In this example, data is not added to the buffers during transmission. As described in the table, DMA operation of frame 1 ends after steps 1 to 5, when the DMAC enters DMA initial state. The transfer control register value is retained and thus DMA transfer of frame 2 subsequently starts when the DE bit is set to 1. When frame 2 is completed, the CDA and EDA contents are equal. Accordingly, the DMAC transfers no data, even if an additional request is issued from the MSCI, and generates an interrupt DMIA (if enabled).

Table 6.5 Memory-to-MSCI Chained-Block Single-Frame Transfer Mode (no transmit data added during transmission)

Step	DMAC Operation	MPU Operation	CDA Value	EDA Value	DE Bit Value	Note
1	—	A_0 ‡ CDA A_3 ‡ EDA 1 ‡ DE bit	A_0	A_3	1	Specifies the first buffer containing data to be transmitted using CDA, and specifies the next to last buffer using EDA. (see figure 6.15.)
2	Reads data from buffer 0	—	A_0	A_3	1	
3	A_1 ‡ CDA	—	A_1	A_3	1	
4	Reads data from buffer 1	—	A_1	A_3	1	
5	A_2 ‡ CDA 0 ‡ DE bit	—	A_2	A_3	0	Clears the DE bit after the transfer of one frame. When a 1 is written to the DE bit, the DMAC can accept a transfer request.
6	—	1 ‡ DE bit	A_2	A_3	1	
7	Reads data from buffer 2	—	A_2	A_3	1	
8	A_3 ‡ CDA 0 ‡ DE bit	—	A_3	A_3	0	When a 1 is written to the DE bit, and a transfer request is issued, the DMAC generates a DMIA interrupt. (see figure 6.15.)

A_n: Start address of each descriptor
CDA: Current descriptor address register
EDA: Error descriptor address register
DE bit: Bit 1 of the DMA status register (DSR)

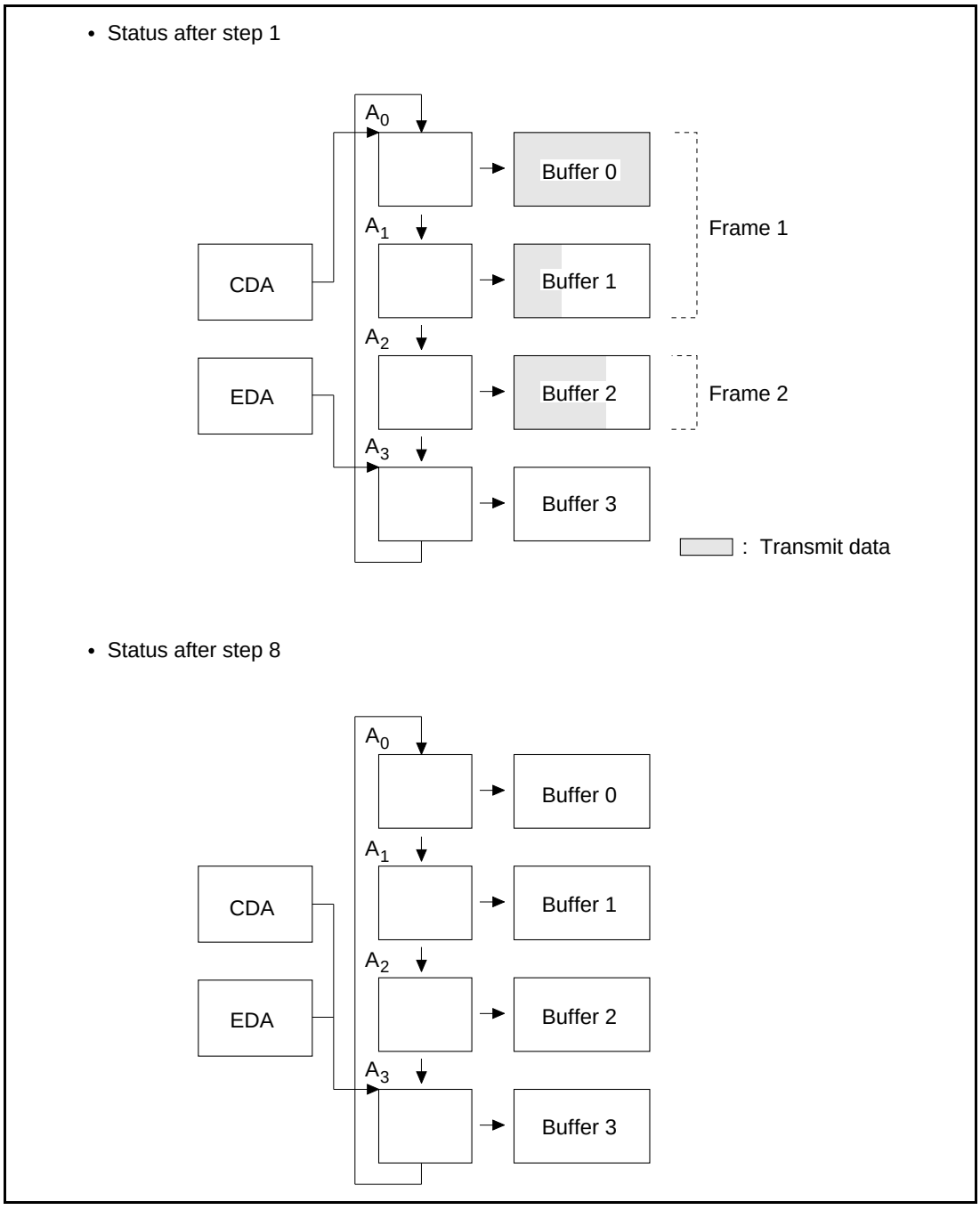


Figure 6.15 Memory-to-MSCI Chained-Block Single-Frame Transfer

Table 6.6 shows a memory-to-MSCI chained-block multi-frame transfer using four descriptors and four buffers. In this example, data is added to the buffer during transmission. As described in the table, after steps 1 and 2, the MPU writes additional transmit data to buffers 2 and 3 and at the same time updates EDA to the start address of the descriptor indicating buffer 0. In this way, the DMAC transfers the data in buffers 2 and 3 after the data in buffer 1. Since the DMAC remains enabled after one frame has been transferred in multi-frame transfer mode, some frame end interrupts (DMIB) remain unprocessed. The number of unprocessed interrupts is stored in the frame end interrupt counter (FCT). When the FCT value is 1111 and frame transfer continues, a counter overflow error occurs and the DMAC terminates data transfer after transmitting the current frame. The FCT value is then reset to 0000, and a DMIA interrupt is generated (if enabled). For details, see sections 6.2.8, DMA Mode Register (DMR), and 6.2.9, Frame End Interrupt Counter (FCT).

Table 6.6 Memory-to-MSCI Chained-Block Multi-Frame Transfer Mode (transmit data added during transmission)

Step	DMAC Operation	MPU Operation	CDA Value	EDA Value	DE Bit Value	Note
1	—	A_0 ‡ CDA A_2 ‡ EDA 1 ‡ DE bit	A_0	A_2	1	Specifies the buffer containing data to be transmitted using CDA (see figure 6.16)
2	Reads data from buffer 0	—	A_0	A_2	1	
3	A_1 ‡ CDA	—	A_1	A_2	1	
4	—	Loads transmit data into buffer 2 A_3 ‡ EDA	A_1	A_3	1	Adds transmit data to the buffer, and rewrites EDA.
5	—	Loads transmit data into buffer 3 A_0 ‡ EDA	A_1	A_0	1	
6	Reads data from buffer 1	—	A_1	A_0	1	
7	A_2 ‡ CDA	—	A_2	A_0	1	(see figure 6.16)

A_n: Start address of each descriptor
CDA: Current descriptor address register
EDA: Error descriptor address register
DE bit: Bit 1 of the DMA status register (DSR)

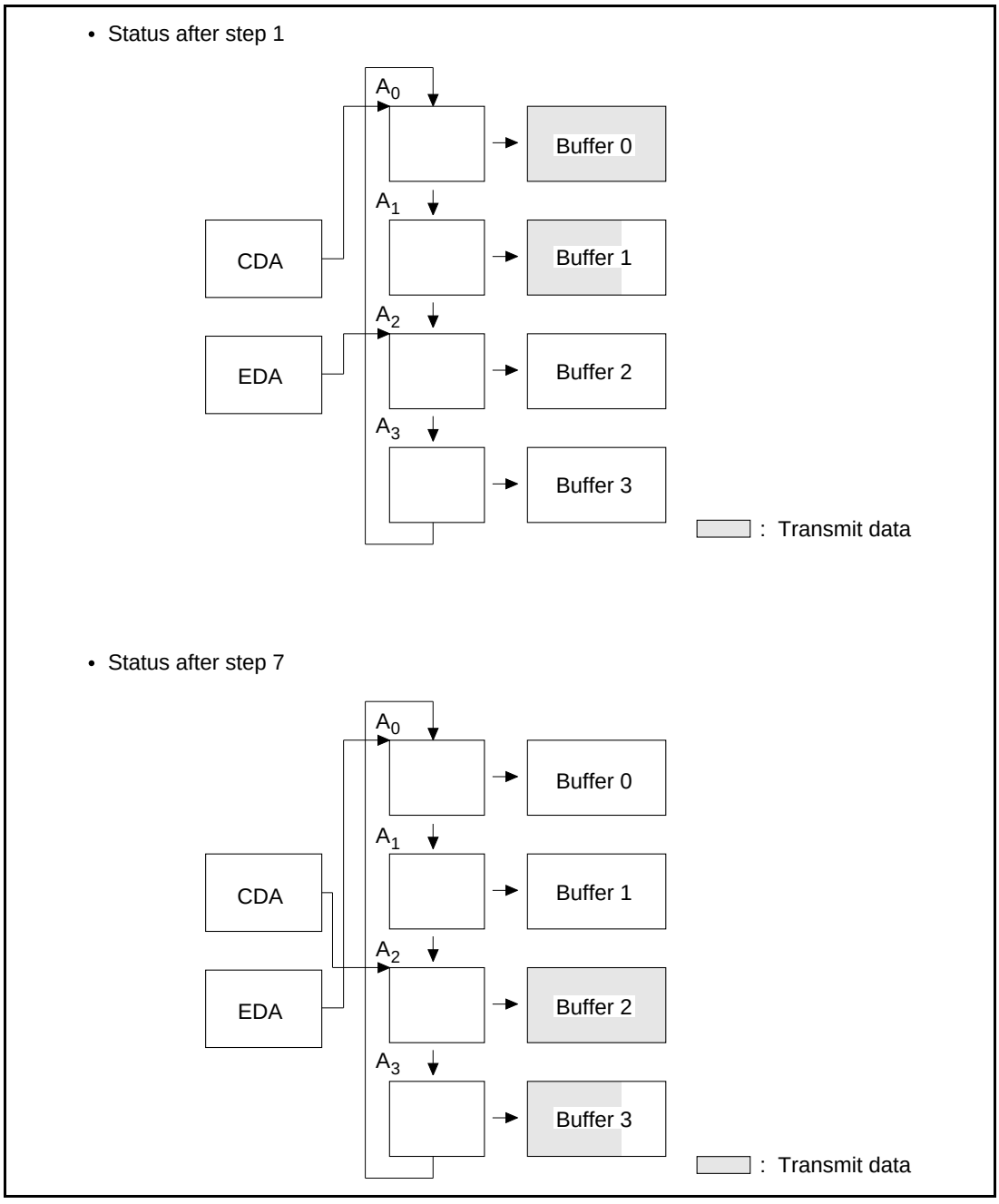


Figure 6.16 Memory-to MSCI Chained-Block Multi-Frame Transfer

Register and Descriptor Setting: To start a memory-to-MSCI chained-block transfer, follow the steps below starting with the DMA in its initial state. (Steps 1 to 6 may be completed in any order.)

1. Create any desired number of descriptors anywhere in the system memory area (64 Kbytes or less), using the MPU. Note that since the high-order eight bits of the 24-bit address are specified by CPB, the high-order eight bits are common to the same 64-Kbyte area. Specify a 16-bit chain pointer (CP), 24-bit buffer pointer (BP), 16-bit data length (DL), and the EOM and EOT bits of ST in each descriptor. (Descriptors may be specified in DMA halt state.)
2. Set the TMOD bit of DMR to 1.
3. Clear the NF bit to 0 of DMR for single-frame transfer, and set the NF bit to 1 for multi-frame transfer.
4. Load the high-order eight bits of the 24-bit descriptor address into CPB.
5. Load the low-order 16 bits of the start address of the descriptor corresponding to the buffer next to the last transmit buffer into EDA.
6. Load the start address of the descriptor corresponding to the first transmit buffer into CDA.
7. After steps 1 to 6, set the DE bit of DSR to 1 to start DMA operation.

External Bus Timing: In memory-to-MSCI chained-block transfer mode, one byte or one word of data is transferred within one memory read cycle. The memory read cycle timing is the same as that in memory-to-MSCI single-block transfer mode shown in figure 6.12.

Prior to the start of DMA transfer and at buffer switching, this transfer mode requires several set-up cycles for the DMAC to perform a read operation on a descriptor and other operations, as shown in figure 6.17. In the figure, 20 states (CPU modes 0, 2, and 3) or 32 states (CPU mode 1) are inserted before the read operation of the transmit data. At buffer switching, one internal state (in the middle of a frame) or five states (at the end of a frame) are inserted. (These states are indicated by “*2”) This is followed by a read operation of the next descriptor.

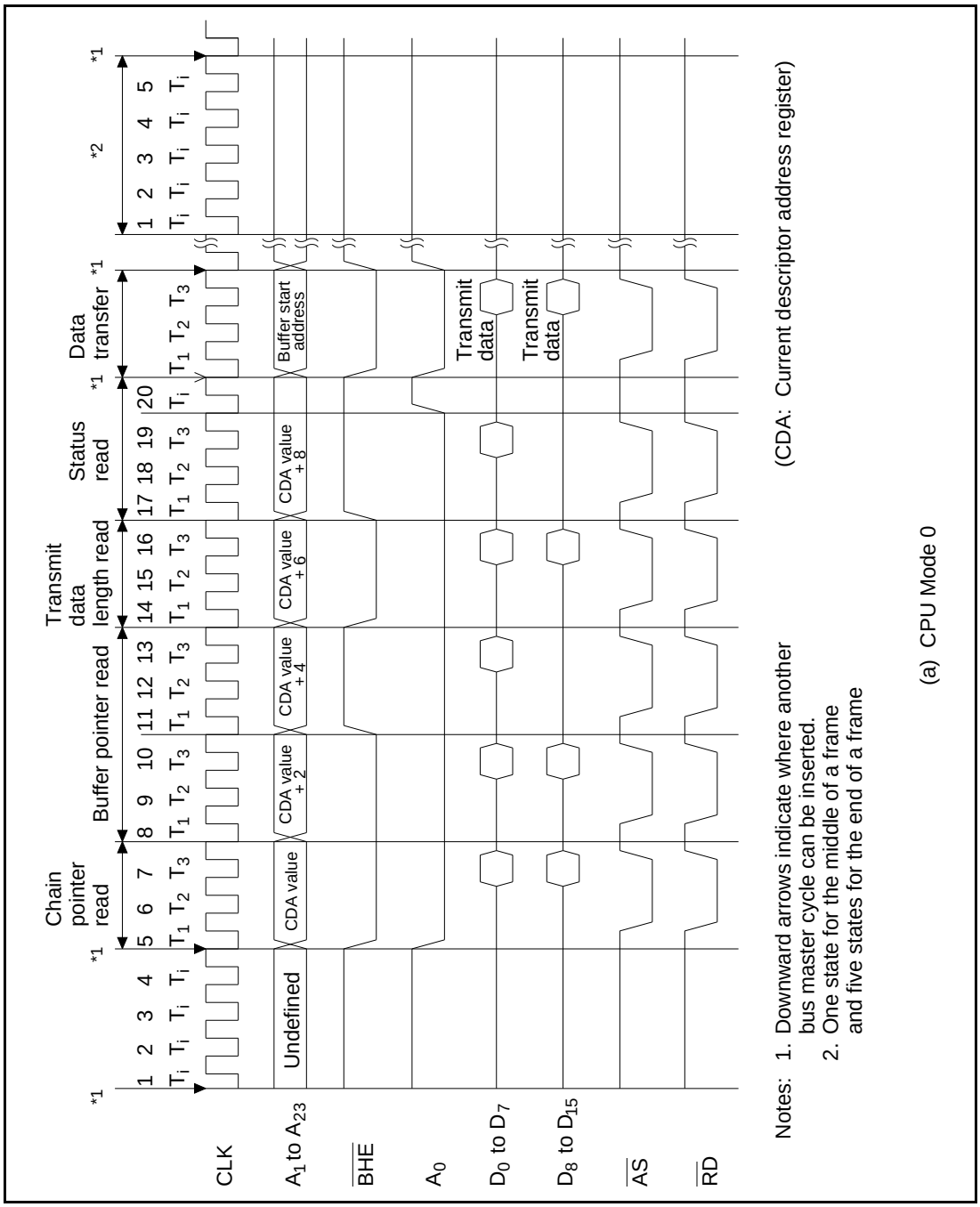


Figure 6.17 Transfer Start and Buffer Switching Timing in Memory-to-MSCI Chained-Block Transfer Mode

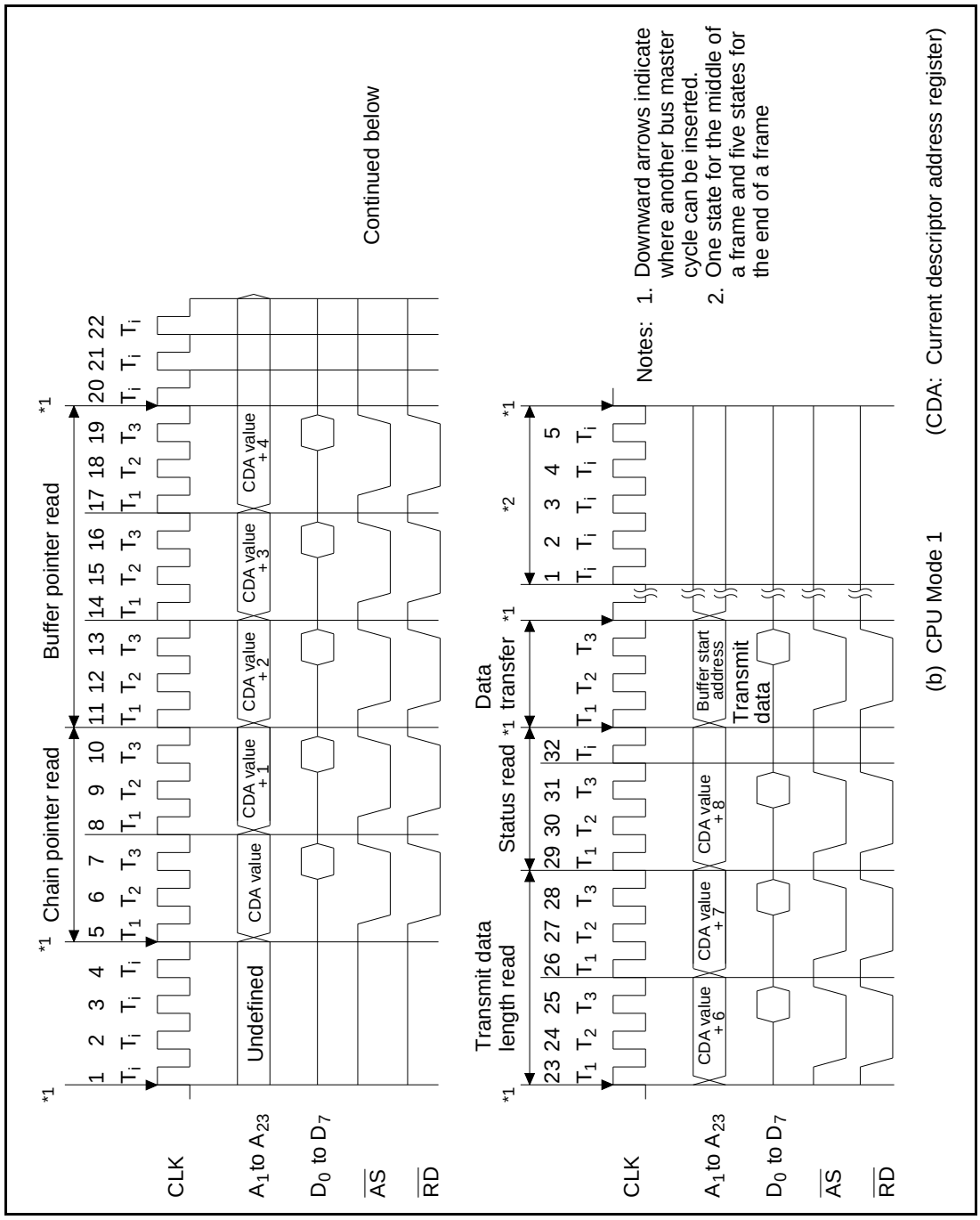


Figure 6.17 Transfer Start and Buffer Switching Timing in Memory-to-MSCI Chained-Block Transfer Mode (cont)

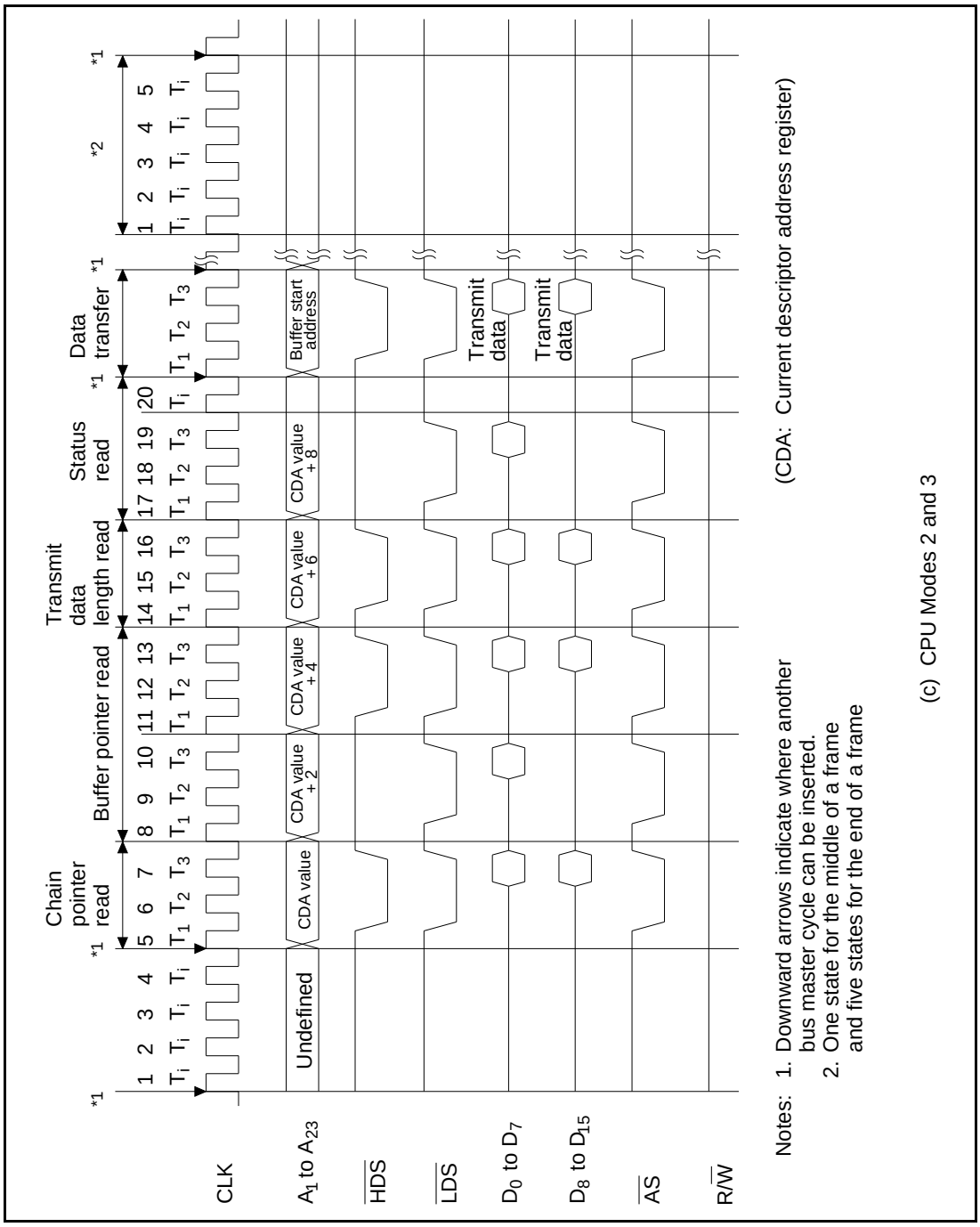


Figure 6.17 Transfer Start and Buffer Switching Timing in Memory-to-MSCI Chained-Block Transfer Mode (cont)

6.4.4 MSCI-to-Memory Chained-Block Transfer Mode

Operation: In MSCI-to-memory chained-block transfer mode, frame-bounded data is DMA-transferred from the MSCI receiver (in bit synchronous mode) to a system memory buffer. Transfer requests are initiated by the MSCI internal signal. Note that chained-block transfer mode is not available with the MSCI operated in asynchronous or byte synchronous mode.

MSCI-to-memory transfer employs DMAC channels 0 and 2. For this transfer mode, follow the steps below starting with the DMA in its initial state. (Steps 1 to 6 may be completed in any order.)

1. Specify chained-block transfer mode with the DMA mode register (DMR).
2. Load the high-order eight bits of the 24-bit descriptor address into the chain pointer base (CPB). Since the CPB value is fixed during operation, descriptors can be assigned to any consecutive 64-Kbyte area in system memory.
3. Load the low-order 16 bits of the start address of the descriptor, which indicates the buffer next to the last receive buffer, into the error descriptor address register (EDA).
4. Load the low-order 16 bits of the start address of the descriptor, which indicates the first receive buffer, into the current descriptor address register (CDA).
5. Load the buffer length in byte units into the receive buffer length (BFL). (This value is shared by all buffers.)
6. Initialize the chain pointer (CP) and buffer pointer (BP) in each descriptor.
7. After steps 1 to 6, set the DE bit of the DMA status register (DSR) to 1 to start a DMA operation.

MSCI-to-memory chained-block transfer mode is shown in figure 6.18.

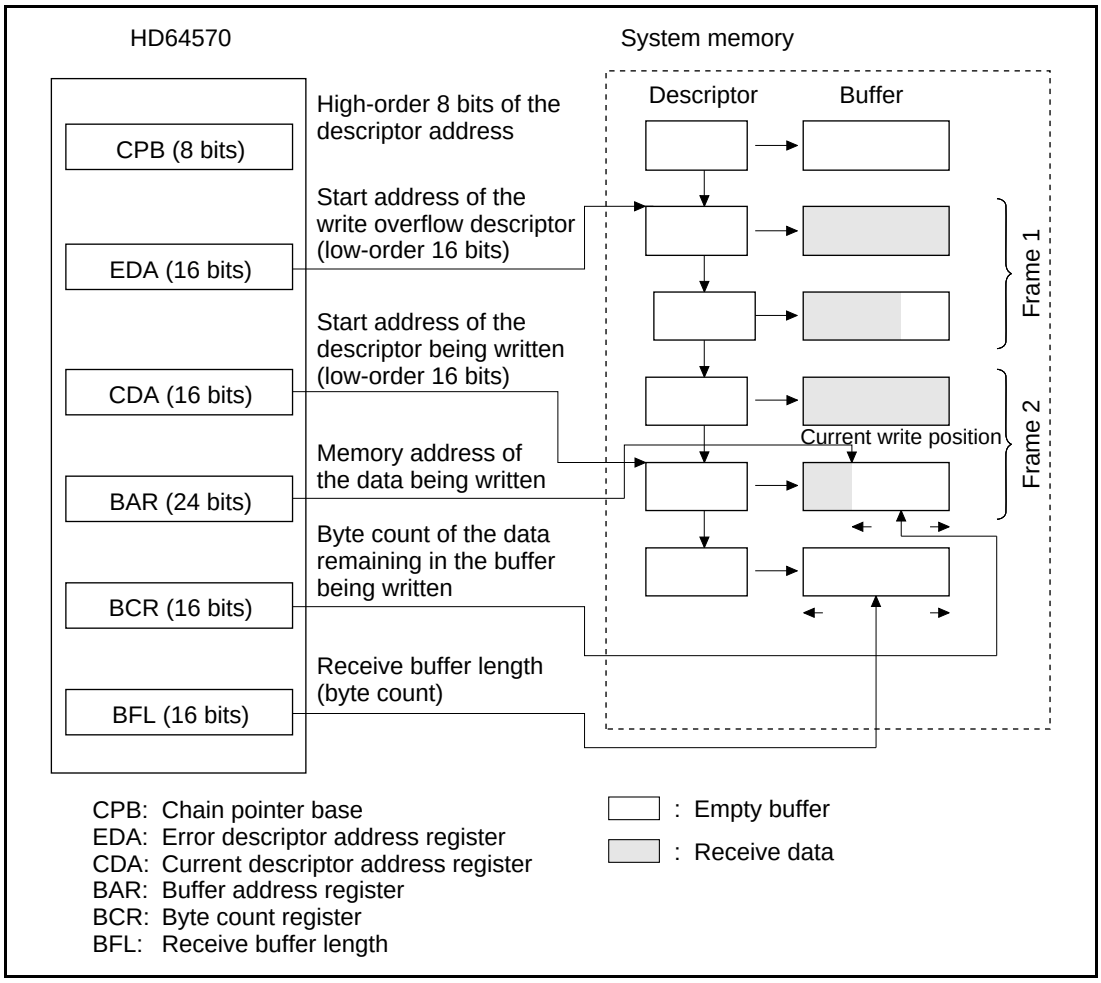


Figure 6.18 MSCI-to-Memory Chained-Block Transfer

The operation flow in MSCI-to-memory chained-block transfer mode is shown in figure 6.19. As shown in the figure, the DMAC transfers data from the MSCI receiver to the buffer corresponding to the descriptor specified by CPB and CDA. At this time, the DMAC writes the 24-bit memory address of the buffer currently being written to the buffer address register (BAR) and the number of bytes remaining unwritten in the buffer to the byte count register (BCR). When data transfer starts, the DMAC writes the BP value of the corresponding descriptor to BAR and the BFL value to BCR.

The BAR value is incremented by 1 or 2 each time one byte or one word of data is transferred, respectively. Similarly, the BCR value is decremented by 1 or 2 each time one byte or one word of data is transferred, respectively. When the BCR value reaches 0000H, the DMAC terminates data transfer, writes the receive data length to the descriptor, and updates the CDA value to indicate the start address of the next descriptor (buffer switching). The DMAC, at that time,

updates BAR and BCR by writing the BP value of the descriptor to BAR, and the BFL value of the descriptor to BCR. In this way, the DMAC transfers data to the buffers specified by the descriptors by updating the descriptors.

On detecting the end of a frame in the buffer currently being written, the DMAC immediately switches the buffer, and writes the MSCI frame status register (FST) value, which is stored immediately after the data transfer, into the status (ST) field of the corresponding descriptor. (At this time, the DMAC also writes data length (DL) to the descriptor.) In single-frame transfer mode, the DMAC terminates data transfer after updating the CDA value. In multi-frame transfer mode, the DMAC switches the buffer and updates the CDA, BAR, and BCR values, after which the DMAC starts writing data to the next buffer.

At completion of frame transfer, the DMAC issues interrupt DMIB (if enabled).

EDA must initially contain the low-order 16 bits of the address of the descriptor indicating the first buffer that is disabled for receive data writing. In this case, buffers can be accessed if the EDA value is updated, even while DMA is enabled. At this time, EDA must be loaded with the start address of the descriptor indicating the buffer next to the last write buffer.

When the CDA and EDA values are equal and a transfer request is issued, the DMAC terminates data transfer and issues interrupt DMIA (if enabled).

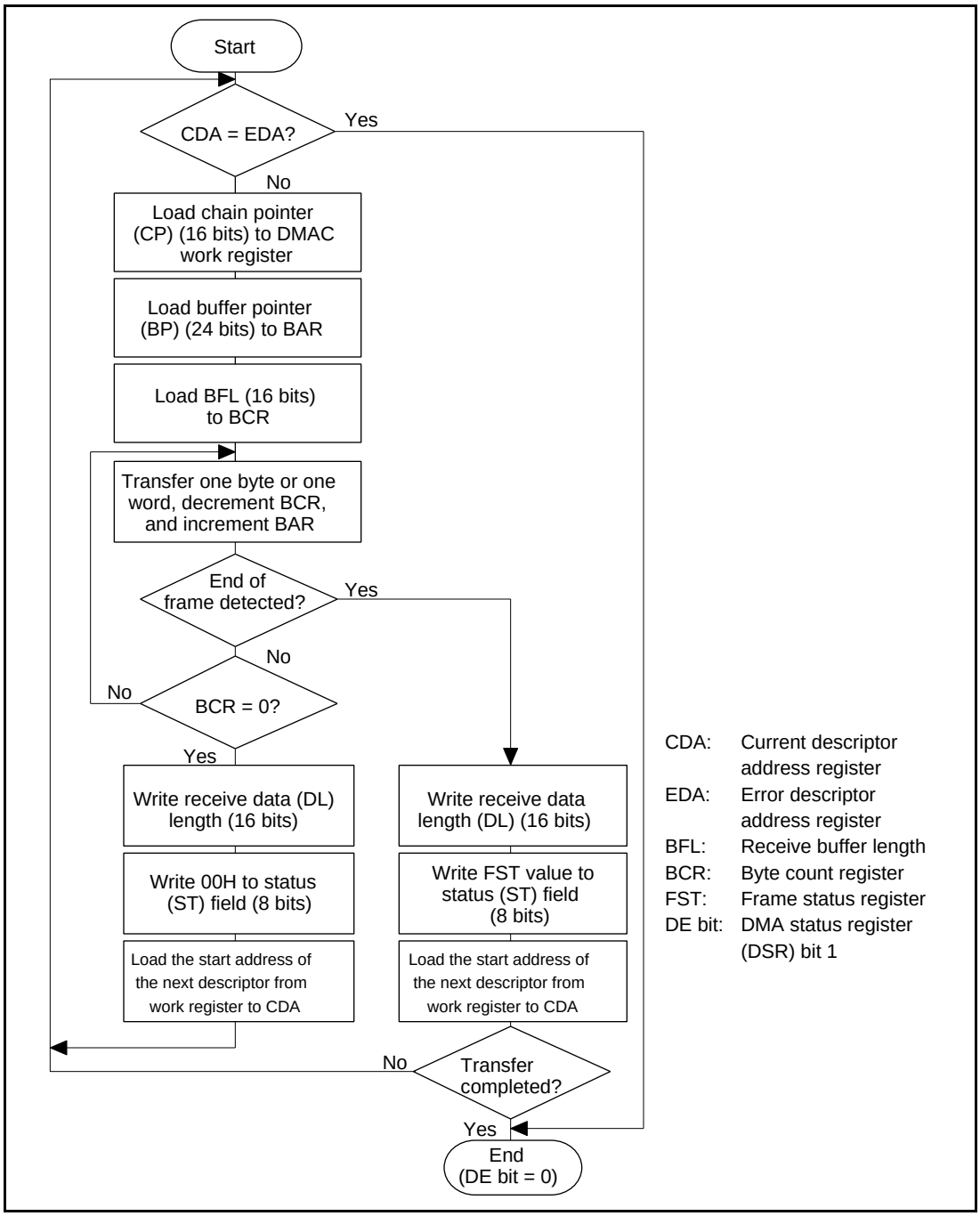


Figure 6.19 Operation Flow in MSCI-to-Memory Chained-Block Transfer Mode

The functions of the registers used in MSCI-to-memory chained-block transfer mode are shown in table 6.7. As can be seen from the table, in MSCI-to-memory chained-block transfer mode, either single-frame transfer or multi-frame transfer can be selected. In single-frame transfer mode, transfer is completed within one frame, after which the DMAC enters initial state. Here, the DE bit of DSR is automatically cleared. When the DE bit is set to 1 again, the DMAC restarts operation. In multi-frame transfer mode, the DMAC subsequently transfers frames of data if a request is issued from the MSCI. When the CDA and EDA values match, the DMAC terminates data transfer, even if an additional transfer request has been issued.

Table 6.7 Control Registers Used in MSCI-to-Memory Chained-Block Transfer Mode (reception)

Item	Chain Pointer Base (CPB)	Error Descriptor Address Register (EDA)	Current Descriptor Address Register (CDA)
Number of bits	8	16	16
Function	Specifies the high-order 8 bits of the 24-bit descriptor start address.	Indicates the low-order 16 bits of the start address of the descriptor following the descriptor indicating the last write-enabled buffer.	Specifies the low-order 16 bits of the start address of the descriptor corresponding to the first receive buffer. This address is updated by the DMAC during buffer chaining.
Role in DMAC operation	—	—	When the DMAC begins receive operation, indicates the low-order 16 bits of the start address of the descriptor corresponding to the buffer being written.
		Transfer ends when a transfer request is issued while the EDA and CDA match. An interrupt, if enabled, is generated.	
Register update	Under MPU control.	Under MPU control.	When the current buffer write is completed, the next descriptor start address is automatically loaded into this register.

Table 6.7 Control Registers Used in MSCI-to-Memory Chained-Block Transfer Mode (reception) (cont)

Register Name	Receive Buffer Length (BFL)	Byte Count Register (BCR)	Buffer Address Register (BAR)
Register updated by the MPU	Initialized before reception.	Loaded with the start address of the descriptor indicating the buffer following the last write buffer. When releasing the buffer, this register indicates the start address of the descriptor for the buffer following the one being released.	When reception begins, indicates the start address of the descriptor which indicates the buffer to be written.
Number of bits	16	16	24
Function	Indicates the buffer length in bytes.	Indicates the byte count of the data remaining in the buffer waiting to be written to memory. Writing to this register by the MPU is prohibited.	Indicates the system memory address of the data being loaded into the buffer. Writing to this register by the MPU is prohibited.
Role in DMAC operation	—	When the contents of this register equal 0000H, writing to the current buffer stops.	When a transfer request is issued, data is loaded into the address specified by this register.
Register update	Under MPU control.	The contents are decremented each time one byte or one word is written. When the buffer is switched, the BFL value is loaded.	The contents are incremented each time one byte or one word is written. When the buffer is switched, the next buffer start address is loaded.
Register updated by the MPU	Initialized.	—	—

Table 6.8 shows a typical MSCI-to-memory chained-block multi-frame transfer using four descriptors and four buffers. In this example, after a transfer begins, CDA is updated and then the CDA initial value is written to EDA since transfer is disabled when CDA and EDA are equal. As a result, the write-enabled buffer size is maximized. In this example, the CDA and EDA values match after frame 2 has been transferred (step 9). At this time, any additional transfer request is disabled and interrupt DMIA is generated (if enabled).

Table 6.8 MSCI-to-Memory Chained-Block Multi-Frame Transfer Mode (normal reception operation)

Step	DMAC Operation	MPU Operation	CDA Value	EDA Value	DE Bit Value	Note
1	—	A_2 ‡ CDA A_1 ‡ EDA 1 ‡ DE bit	A_2	A_1	1	Specifies the buffer where receive data is to be written using CDA and EDA (figure 6.20)
2	Writes data to buffer 2	—	A_2	A_1	1	
3	A_3 ‡ CDA	A2 ‡ EDA	A_3	A_2	1	Writes A_2 to EDA to reserve the maximum buffer size
4	Writes data to buffer 3	—	A_3	A_2	1	
5	A_0 ‡ CDA	—	A_0	A_2	1	
	→					
8	Writes data to buffer 1	—	A_1	A_2	1	
9	A_2 ‡ CDA	—	A_2	A_2	1	If another write request is issued in this state, the DMAC generates a DMIA interrupt (figure 6.20)

A_n: Start address of each descriptor
CDA: Current descriptor address register
EDA: Error descriptor address register
DE bit: Bit 1 of the DMA status register (DSR)

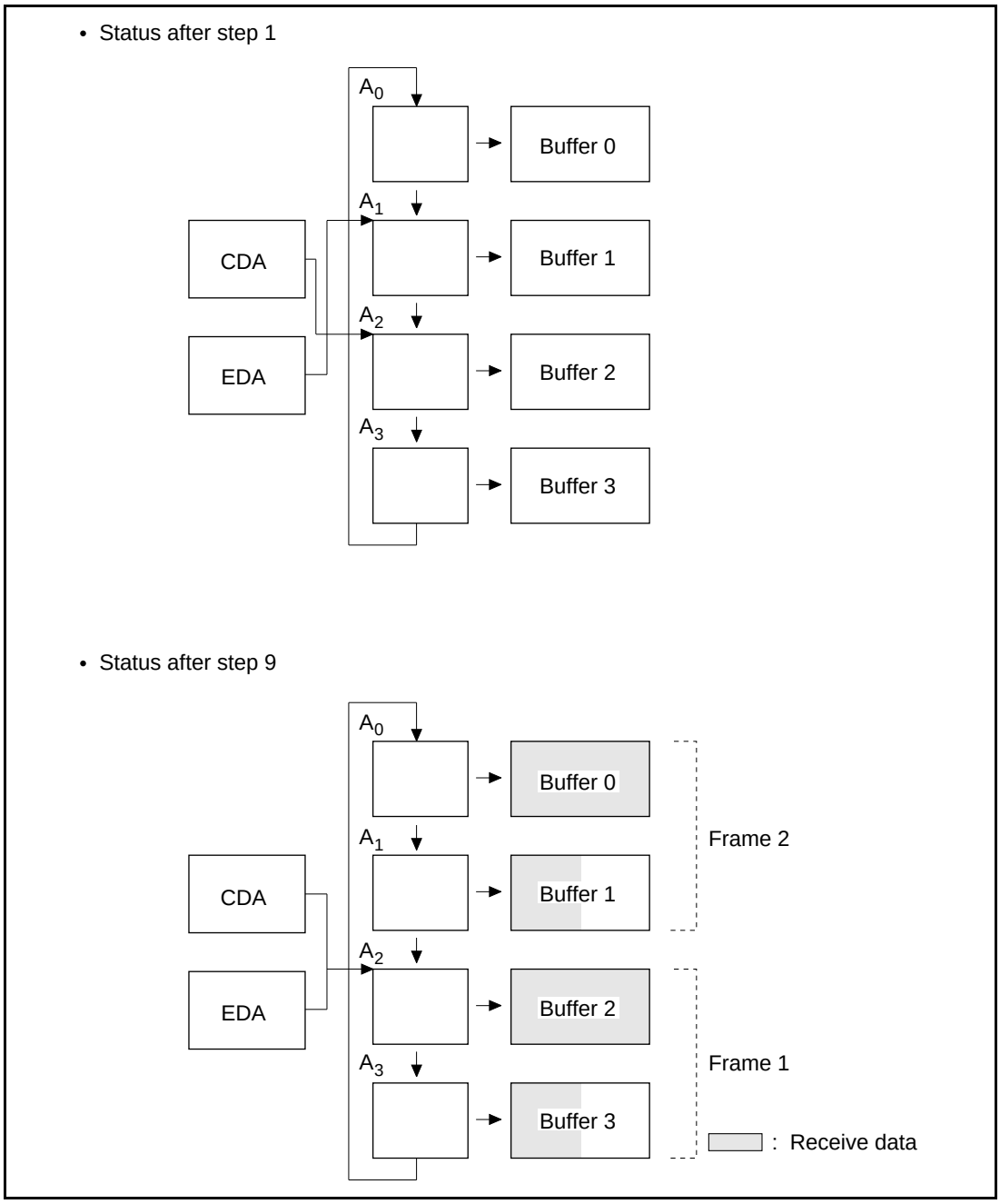


Figure 6.20 MSCI-to Memory Chained-Block Multi-Frame Transfer (normal reception operation)

Table 6.9 shows another example of MSCI-to-memory multi-frame transfer using four descriptors and four buffers. In this example, to rewrite a buffer, the received data stored in the buffer is moved to another area during reception operations, and EDA is updated. Steps 1 to 7 are the same as those in table 6.8. Since the DMAC remains enabled after one frame has been transferred in multi-frame transfer mode, some frame end interrupts (DMIB) might remain unprocessed. The number of unprocessed interrupts is stored in the frame end interrupt counter (FCT). When the FCT value is 1111 and frame transfer continues, a counter overflow error occurs, and the DMAC terminates data transfer after transmitting the current frame. The FCT value is then reset to 0000, and DMIA interrupt is generated (if enabled). For details, see sections 6.2.8, DMA Mode Register (DMR), and 6.2.9, Frame End Interrupt Counter (FCT).

Table 6.9 MSCI-to-Memory Chained-Block Multi-Frame Transfer Mode (part of a buffer released during reception operation)

Step	DMAC Operation	MPU Operation	CDA Value	EDA Value	DE Bit Value	Note
1	—	A_1 ‡ CDA A_0 ‡ EDA 1 ‡ DE bit	A_1	A_0	1	Specifies the buffer where the receive data is to be written using the CDA (figure 6.21)
2	Writes data to buffer 1	—	A_1	A_0	1	
3	A_2 ‡ CDA	A_1 ‡ EDA	A_2	A_1	1	Writes A_1 to EDA to reserve the maximum buffer size
4	Writes data to buffer 2	—	A_2	A_1	1	
5	A_3 ‡ CDA	—	A_3	A_1	1	
6	Writes data to buffer 3	—	A_3	A_1	1	
7	A_0 ‡ CDA	—	A_0	A_1	1	
8	—	Transfers data from buffers 1 and 2 to another area	A_0	A_1	1	After transferring receive data to another area, the MPU rewrites EDA to release the buffer (figure 6.21)
9	—	A_3 ‡ EDA	A_0	A_3	1	
10	Writes data to buffer 0	—	A_0	A_3	1	

A_n: Start address of each descriptor

CDA: Current descriptor address register

EDA: Error descriptor address register

DE bit: Bit 1 of the DMA status register (DSR)

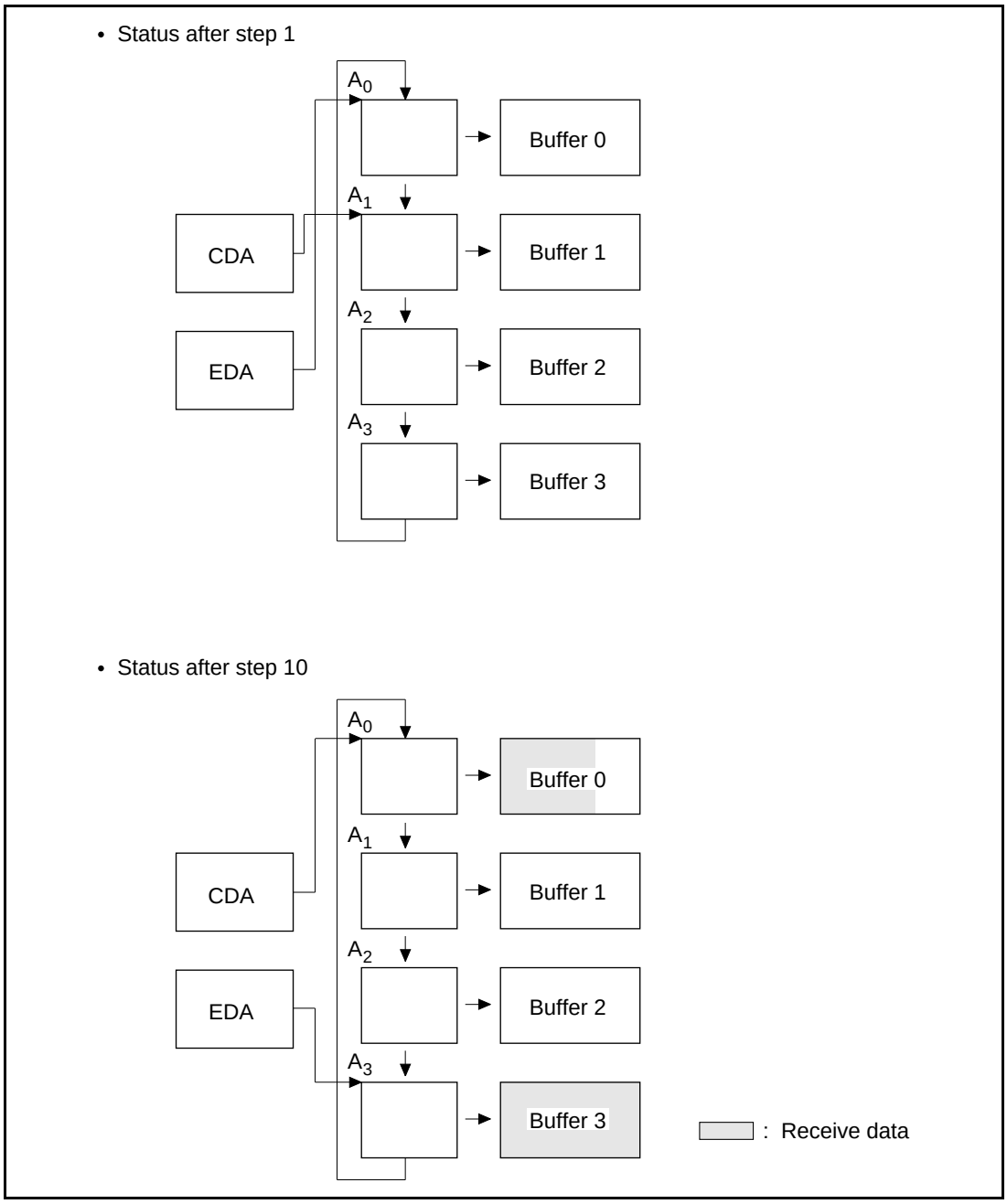


Figure 6.21 MSCI-to-Memory Chained-Block Multi-Frame Transfer (part of a buffer released during reception operation)

Register and Descriptor Setting: To start an MSCI-to-memory chained-block transfer, follow the steps below starting with the DMA in its initial state. (Steps 1 to 7 may be completed in any order.)

1. Create any desired number of descriptors anywhere in the system area (64 Kbytes or less), using the MPU. Note that since the high-order eight bits of the 24-bit address are specified by CPB, the high-order eight bits are common to the same 64-Kbyte area. Specify a 16-bit chain pointer (CP) and a 24-bit buffer pointer (BP) in each descriptor. (Descriptors may be specified in DMA halt state.)
2. Set the TMOD bit of DMR to 1.
3. Clear the NF bit of DMR to 0 for single-frame transfer, and set the NF bit to 1 for multi-frame transfer.
4. Load the high-order eight bits of the 24-bit descriptor address into CPB.
5. Load the low-order 16 bits of the start address of the descriptor corresponding to the buffer next to the last write-enabled buffer into EDA.
6. Load the start address of the descriptor corresponding to the first receive buffer into CDA.
7. Load the buffer length in byte units into BFL. (This value is shared by all buffers.)
8. After steps 1 to 7, set the DE bit of DSR to 1 to start DMA operation.

External Bus Timing: In MSCI-to-memory chained-block transfer mode, one byte or one word of data is transferred within one memory write cycle. The memory write cycle timing is the same as that in MSCI-to-memory single-block transfer mode shown in figure 6.11.

Prior to the start of DMA transfer and at buffer switching, this transfer mode requires several set-up cycles for the DMAC to perform a read operation on a descriptor and other operations, as shown in figure 6.22. In the figure, 18 states (CPU modes 0, 2, and 3) or 23 states (CPU mode 1) are inserted before the start of a DMA transfer. At buffer switching, 8 states (CPU modes 0, 2, and 3) or 11 states (CPU mode 1) indicated by “*3” are inserted to write receive data length (DL) and status (ST) fields in the descriptor. This is followed by a read operation on the next descriptor.

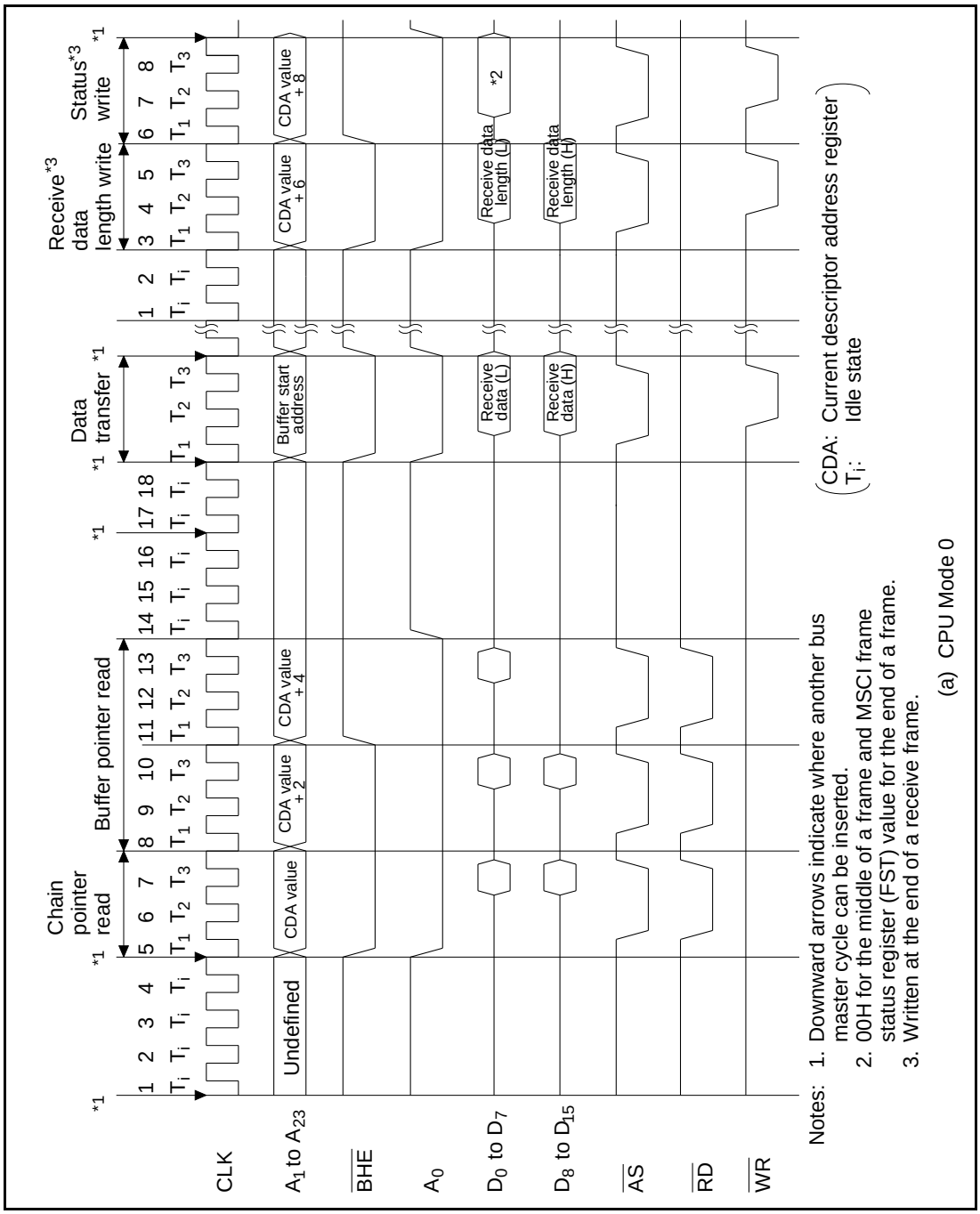


Figure 6.22 Transfer Start and Buffer Switching Timing in MSCI-to-Memory Chained-Block Transfer Mode

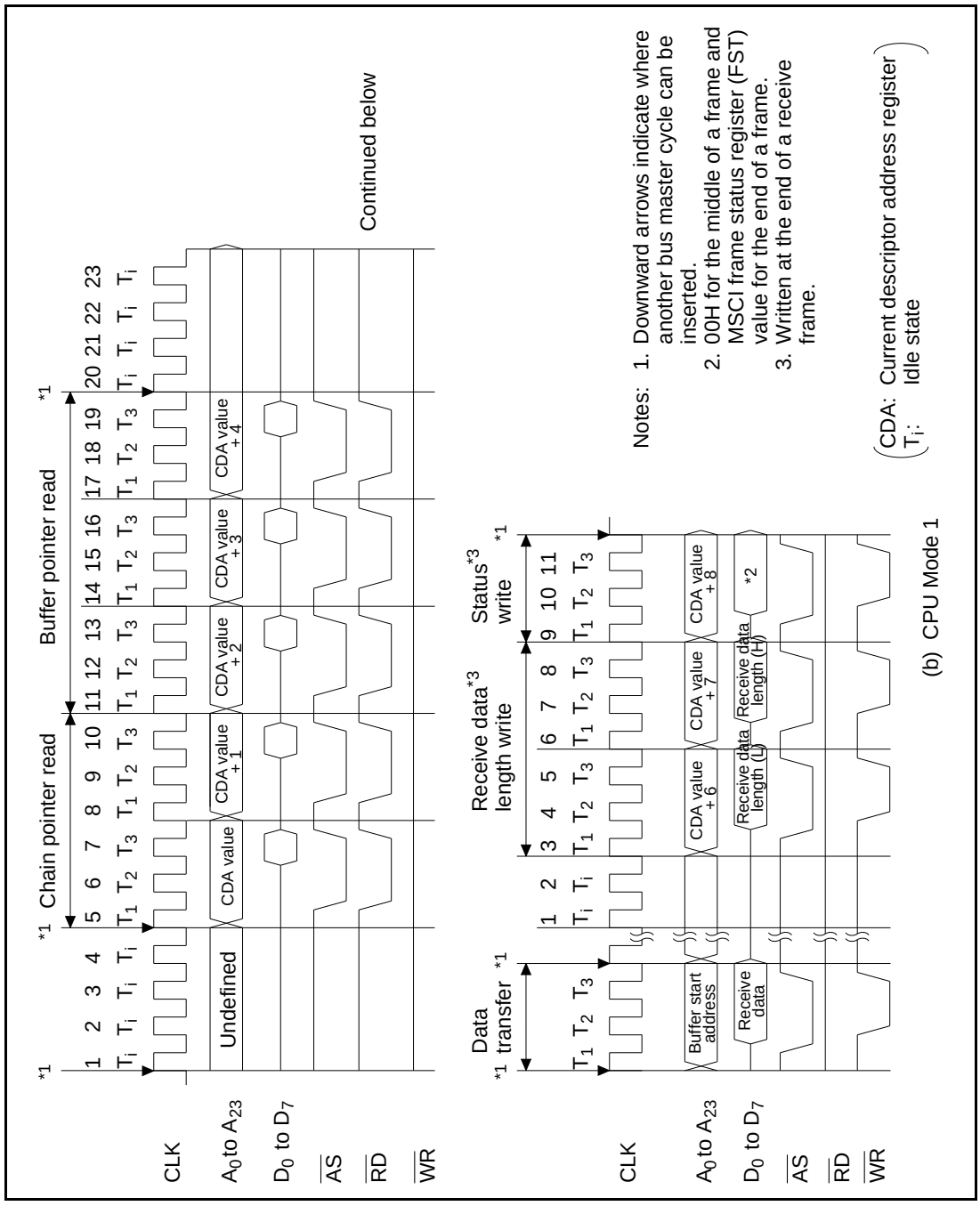


Figure 6.22 Transfer Start and Buffer Switching Timing in MSCI-to-Memory Chained-Block Transfer Mode (cont)

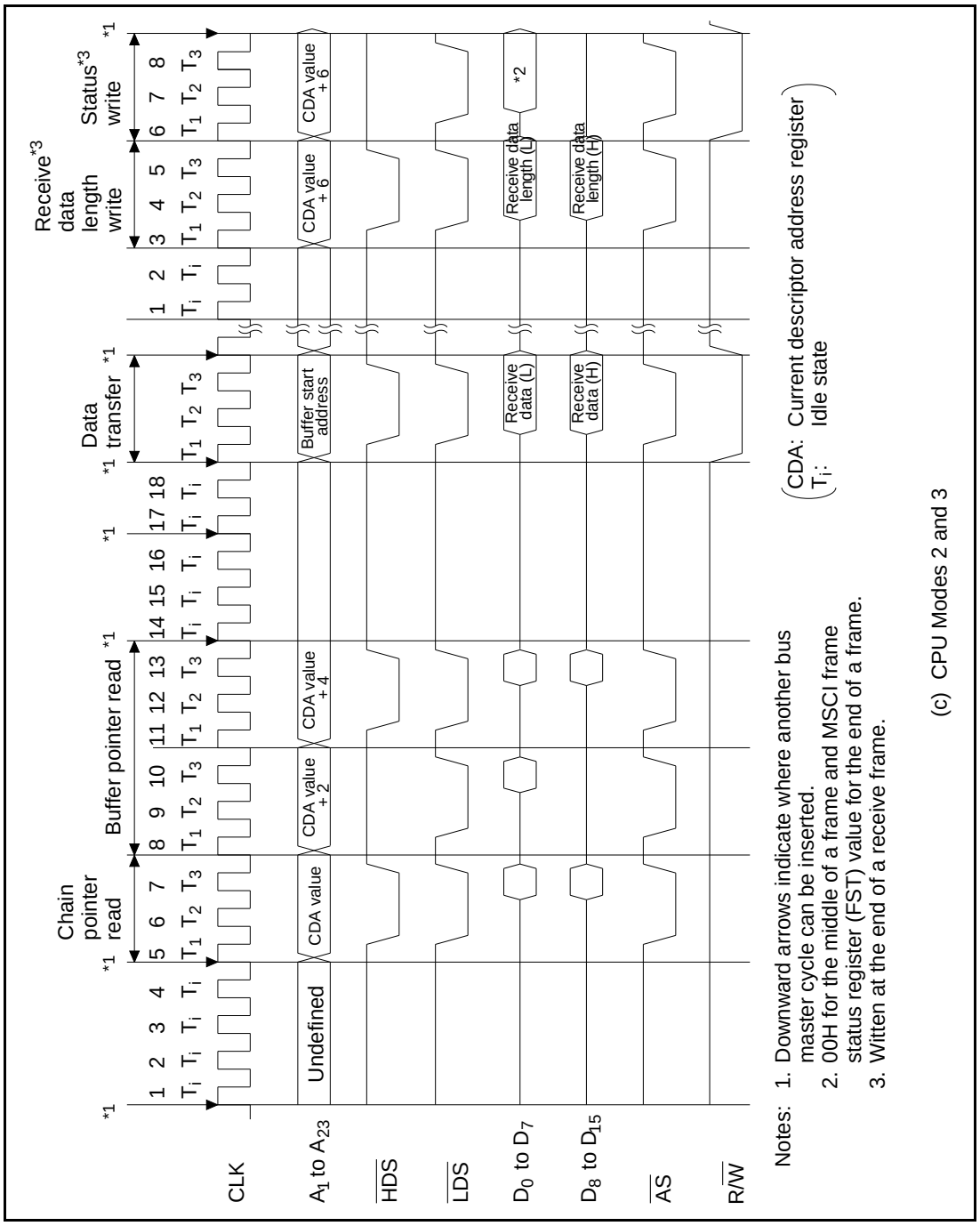


Figure 6.22 Transfer Start and Buffer Switching Timing in MSCI-to-Memory Chained-Block Transfer Mode (cont)

6.4.5 DMAC Characteristics

Tables 6.10 and 6.11 list the DMAC characteristics in different modes.

Table 6.10 DMAC Characteristics in CPU Modes 0, 2, and 3*[1]

Mode	Transfer Direction	DMA Transfer Rate (states/word)	DMA Transfer Set-Up Time*[2]	DMAC Buffer Switching Time
Single-block transfer mode	Memory to MSCI	3	—	—
	MSCI to memory	3	—	—
Chained-block transfer mode	Memory to MSCI (transmission)	3	20*[3]	21/25*[4]
	MSCI to memory (reception)	3	18*[5]	26*[6]

Table 6.11 DMAC Characteristics in CPU Mode 1*[1]

Mode	Transfer Direction	DMA Transfer Rate (states/byte)	DMA Transfer Set-Up Time*[2]	DMAC Buffer Switching Time
Single-block transfer mode	Memory to MSCI	3	—	—
	MSCI to memory	3	—	—
Chained-block transfer mode	Memory to MSCI (transmission)	3	32*[7]	33/37*[8]
	MSCI to memory (reception)	3	23*[9]	34*[10]

Notes: 1 memory cycle = 3 states

Internal states are used for SCA internal operations.

1. Units are states unless otherwise specified. The values shown here are valid when no wait state is inserted.
2. Before entering a data transfer cycle, the DMAC requires some set-up time to read the first descriptor.
3. 20 states = 5 memory cycles (15 states) + 5 internal states
4. 21 states = 5 memory cycles (15 states) + 6 internal states (in the middle of a frame)
   25 states = 5 memory cycles (15 states) + 10 internal states (at the end of a frame)
5. 18 states = 3 memory cycles (9 states) + 9 internal states
6. 26 states = 5 memory cycles (15 states) + 11 internal states
7. 32 states = 8 memory cycles (24 states) + 8 internal states
8. 33 states = 8 memory cycles (24 states) + 9 internal states (in the middle of a frame)
   37 states = 8 memory cycles (24 states) + 13 internal states (at the end of a frame)
9. 23 states = 5 memory cycles (15 states) + 8 internal states
10. 34 states = 8 memory cycles (24 states) + 10 internal states

6.5 Interrupts

The DMAC can issue DMIA (error) and DMIB (normal end) interrupt requests to the MPU. These requests are indicated by the DMA status register (DSR) and are enabled or disabled by the DMA interrupt enable register (DIR). Table 6.12 lists interrupt types, interrupt sources, and clearing their procedures.

Table 6.12 Interrupt Types, Interrupt Sources, and Clearing Procedures

Type	Source	Status Bit	Enable Bit	Clearing Procedure
Error interrupt (DMIA)*[1]	FCT overflow (the number of unprocessed interrupts ≥ 16)	COF	COFE	Write a 1 to the status bit
	Buffer underrun/overrun (EDA value = CDA value and a new transfer request issued)	BOF	BOFE	Write a 1 to the status bit
Normal end interrupt (DMIB)*[1]	Frame transfer completion in chained-block transfer mode*[2]	EOM	EOME	1. Write a 1 to the status bit*[3] 2. Issue a frame end interrupt counter clear command
	DMA transfer completion	EOT	EOTE	Write a 1 to the status bit

FCT: Frame end interrupt counter

CDA: Current descriptor address register

EDA: Error descriptor address register

Notes: 1. Interrupts, once issued, continue to be requested also in DMA initial state or halt state.

2. An interrupt issued at the end of a 1-frame transfer in chained-block multi-frame transfer mode does not signal the end of a transfer.

3. When FCT is enabled and the FCT value is not 0000, the EOM bit is set to 1. For details, see sections 6.2.7, DMA Status Register (DSR), 6.2.9, Frame End Interrupt Counter (FCT), and 6.2.11, DMA Command Register (DCR).

6.6 Reset Operation

When the DMAC is reset, the following steps occur.

- The DMAC enters DMA initial state
- Channel priority becomes 0 > 1 > 2 > 3
- The value of the transfer control registers for specifying addresses and that of the DMA command register (DCR) become undefined
- The DMA status register (DSR), DMA mode register (DMR), frame end interrupt counter (FCT), and DMA interrupt enable register (DIR) are initialized as follows:
  - Operating mode is single-block transfer mode
  - Interrupt status bits and enable bits are cleared
  - The FCT value is cleared and FCT is disabled

6.7 Precautions

- The DMAC registers must be initialized in DMA initial state. When DMAC operation is suspended by software with the DE bit set to 0, the DMAC retains its previous operation status. Thus, to initiate a new operation, the software abort command must be issued to initialize the status. However, when DMAC operation is terminated with transfer completion conditions satisfied, the software abort command is not necessary. For details, see section 6.2.11, DMA Command Register (DCR).
- The DMAC must be disabled in system stop mode.

Section 7 Timer

7.1 Overview

7.1.1 Functions

The HD64570 incorporates a timer with four identically functioning channels 0, 1, 2, and 3.

This timer has the following functional features:

- 16-bit reloadable data
- Operates on a base clock (BC) (ϕ clock internally divided by eight)
  — Increment intervals in the range of $BC/2^0$–$BC/2^7$
- Interrupt issued when a counter value matches a specified value

7.1.2 Configuration and Operation

Figure 1.4 shows the timer block diagram.

In this timer, the timer up-counter (TCNT) increments based on the specified clock signal. When the TCNT value matches the specified value in the timer constant register (TCONR), an interrupt is generated, if enabled. For details on interrupt timing, see section 7.4, Interrupt. Here, the TCNT value is cleared to 0000H, and incrementation restarts from 0000H. For details on timer increment timing, see section 7.3.1, Timer Increment Timing.

7.2 Registers

7.2.1 Timer up-counter (TCNT: TCNTH, TCNTL)

The timer up-counter (TCNT), provided for each of channels 0, 1, 2, and 3, increments based on the clock signal specified by the ECKS2−ECKS0 bits of the timer expand prescale register (TEPR). For information regarding clock selection, see section 7.2.4, Timer Expand Prescale Register (TEPR).

The MPU can read/write TCNT without affecting TCNT operation. When the TCNT value was changed during incrementing, time t between the start of the TCNT write cycle and the start of TCNT incrementing is $c \leq t \leq n \times 8 + c - 1$, where c is 4, 5, 6, or 5 in CPU mode 0, 1, 2, or 3, respectively.

The TCNT value is initialized to 0000H after its value matches the value in the timer constant register (TCONR).

TCNTH

	7	6	5	4	3	2	1	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0
Timer	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}

TCNTL

	7	6	5	4	3	2	1	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0
Timer	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}

CPU Mode	C
Mode 0	4
Mode 1	5
Mode 2	6
Mode 3	5

Note: Initial values are the same in system stop mode and after reset.

7.2.2 Timer Constant Register (TCONR: TCONRH, TCONRL)

The timer constant register (TCONR), provided for each of channels 0, 1, 2, and 3, specifies timer output timing. The TCONR value is constantly compared with the timer up-counter (TCNT) value. When they match, the CMF bit of timer control status register (TCSR) is set to 1, and an interrupt is generated, if enabled. Here, TCNT is cleared and resumes incrementing from 0000H. (For details on timing, see section 7.3.2, Output Timing.) In this way, periodic interrupts can be generated without software overhead. TCONR is initialized to FFFFH at reset or in system stop mode.

TCONRH

	7	6	5	4	3	2	1	0
Read/Write	W	W	W	W	W	W	W	W
Initial value	1	1	1	1	1	1	1	1
Timer constant	2^{15}	2^{14}	2^{13}	2^{12}	2^{11}	2^{10}	2^{9}	2^{8}

TCONRL

	7	6	5	4	3	2	1	0
Read/Write	W	W	W	W	W	W	W	W
Initial value	1	1	1	1	1	1	1	1
Timer constant	2^{7}	2^{6}	2^{5}	2^{4}	2^{3}	2^{2}	2^{1}	2^{0}

Note: TCONR is a write-only register. It always reads 0000H.

7.2.3 Timer Control/Status Register (TCSR)

The timer control/status register (TCSR), provided for each of channels 0, 1, 2, and 3, requests interrupts and controls the timer up-counter (TCNT) operation.

	7	6	5	4	3	2	1	0
Bit name	CMF	ECMI	—	TME	—	—	—	—
Read/Write	R	R/W	—	R/W	—	—	—	—
Initial value	0	0	0	0	0	0	0	0

Compare match flag

0: TCNT and TCONR are not equal
1: TCNT and TCONR are equal

Timer enable

0: Stops incrementing
1: Starts incrementing

CMF interrupt enable

0: Disable
1: Enable

Note: Bit 5 and bits 3–0 are reserved.
These bits always read 0 and must be set to 0.

Bit 7 (CMF: Compare Match Flag): Indicates whether or not the TCNT value matches the timer constant register (TCONR) value. This bit is cleared when TCNT is read after TCSR. Other instructions can be inserted between the TCSR and TCNT read instructions. This bit is also cleared at reset or in system stop mode.

CMF = 0: Indicates that the TCNT and TCONR values do not match.

CMF = 1: Indicates that the TCNT and TCONR values match. An interrupt request (T0IRQ, T1IRQ, T2IRQ, or T3IRQ) is generated when the ECMI bit (bit 6) has been set.

Bit 6 (ECMI: CMF Interrupt Enable): Enables or disables an interrupt request initiated by the CMF bit. This bit is cleared at reset.

ECMI = 0: Disables an interrupt request initiated by the CMF bit

ECMI = 1: Enables an interrupt request initiated by the CMF bit

Bit 5: Reserved. This bit always reads 0 and must be set to 0.

Bit 4 (TME: Timer Enable): Starts or stops TCNT operation. This bit is cleared at reset or in system stop mode.

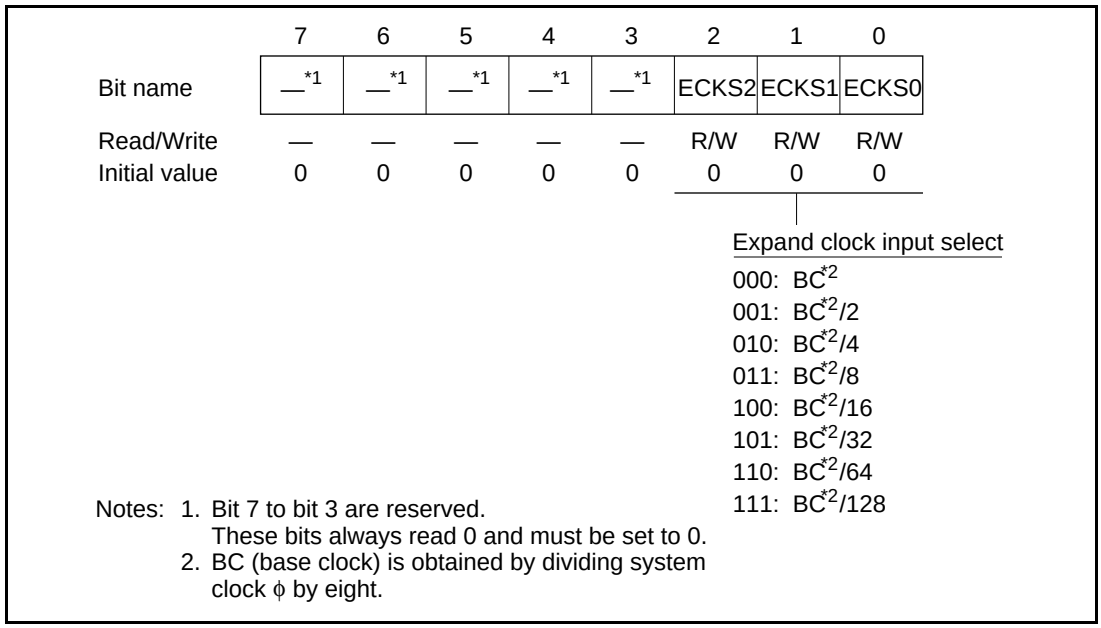
TME = 0: Stops TCNT, retaining the current TCNT value.
(TCNT resumes incrementing from the retained value, when TME is again set to 1.)

TME = 1: Starts TCNT.

Bits 3–0: Reserved. These bits always read 0 and must be set to 0.

7.2.4 Timer Expand Prescale Register (TEPR)

The timer expand prescale register (TEPR), provided for each of channels 0, 1, 2, and 3, selects the expanded clock input for the timer up-counter (TCNT).



Bits 7–3: Reserved. These bits always read 0 and must be set to 0.

Bits 2–0 (ECKS2–ECKS0: Expand Clock Input Select): Selects the TCNT clock as shown below. These bits are cleared at reset.

ECKS2, ECKS1, ECKS0 = 0, 0, 0: TCNT clock rate = BC

ECKS2, ECKS1, ECKS0 = 0, 0, 1: TCNT clock rate = BC/2

ECKS2, ECKS1, ECKS0 = 0, 1, 0: TCNT clock rate = BC/4

ECKS2, ECKS1, ECKS0 = 0, 1, 1: TCNT clock rate = BC/8

ECKS2, ECKS1, ECKS0 = 1, 0, 0: TCNT clock rate = BC/16

ECKS2, ECKS1, ECKS0 = 1, 0, 1: TCNT clock rate = BC/32

ECKS2, ECKS1, ECKS0 = 1, 1, 0: TCNT clock rate = BC/64

ECKS2, ECKS1, ECKS0 = 1, 1, 1: TCNT clock rate = BC/128

7.3 Operation Timing

7.3.1 Timer Increment Timing

Figure 7.1 shows the timer increment timing when the counter operating rate is BC. Incrementing is initiated when a 1 is written to the TME bit of the timer constant/status register (TCSR), after the timer up-counter (TCNT) and the timer constant register (TCONR) have been set.

When the TCNT and TCONR values match, the CMF bit of TCSR is set to 1, and an interrupt (T0IRQ, T1IRQ, T2IRQ, or T3IRQ), if enabled, is generated. The CMF bit can be cleared when TCNT is read after TCSR. (Other instructions can be inserted between the TCSR and TCNT read instructions.) Here, TCNT is initialized to 0000H, and incrementing restarts. TCNT can be written even during incrementing. In this case, incrementing restarts from the newly written value.

When the TME bit is cleared during incrementing, TCNT stops incrementing, retaining its current contents. When the TME bit is again set to 1, incrementing resumes from the retained value.

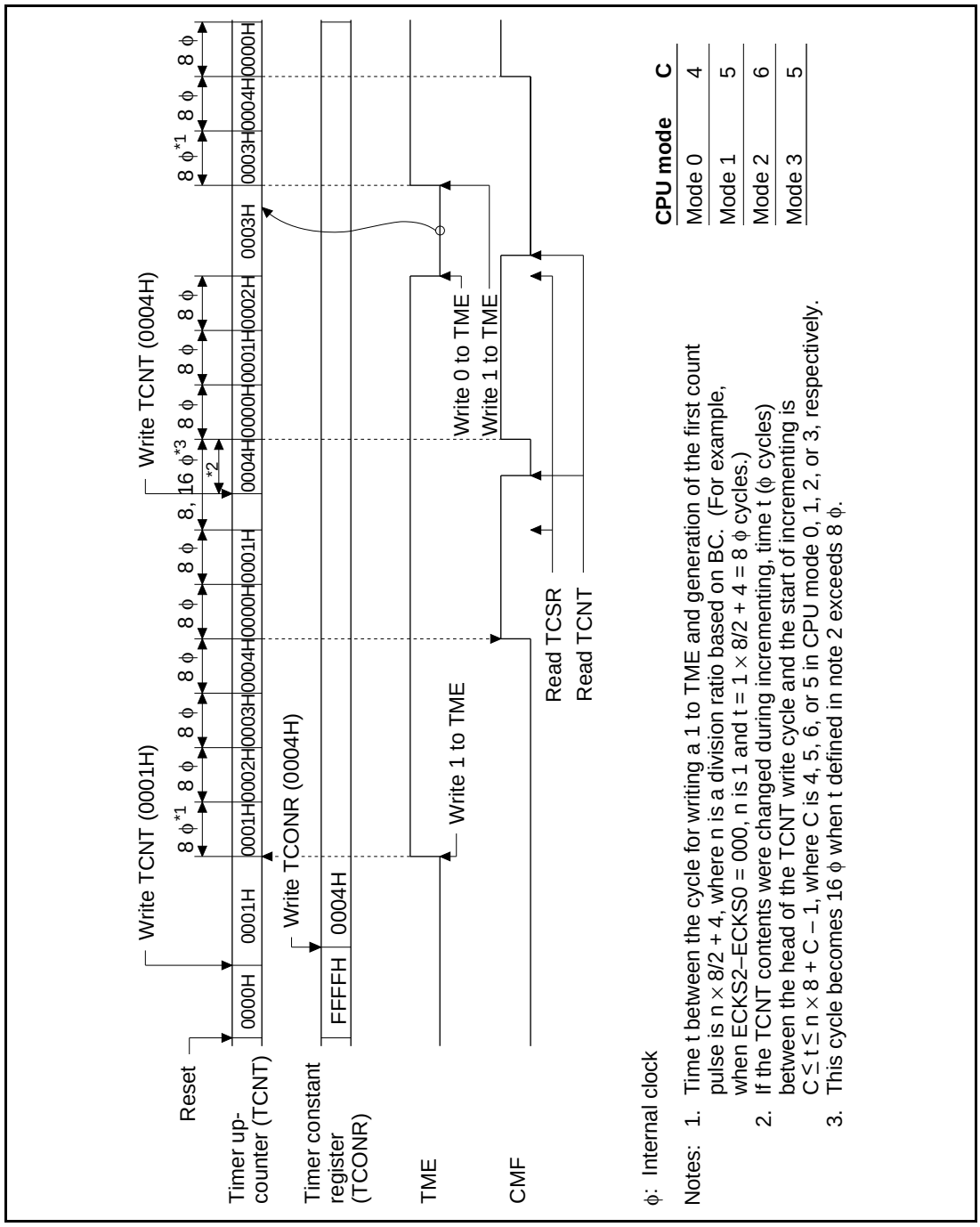


Figure 7.1 Timer Increment Timing (when the counter operating rate is BC)

Figure 7.2 shows the timer increment timing when the counter operating rate is BC/4.

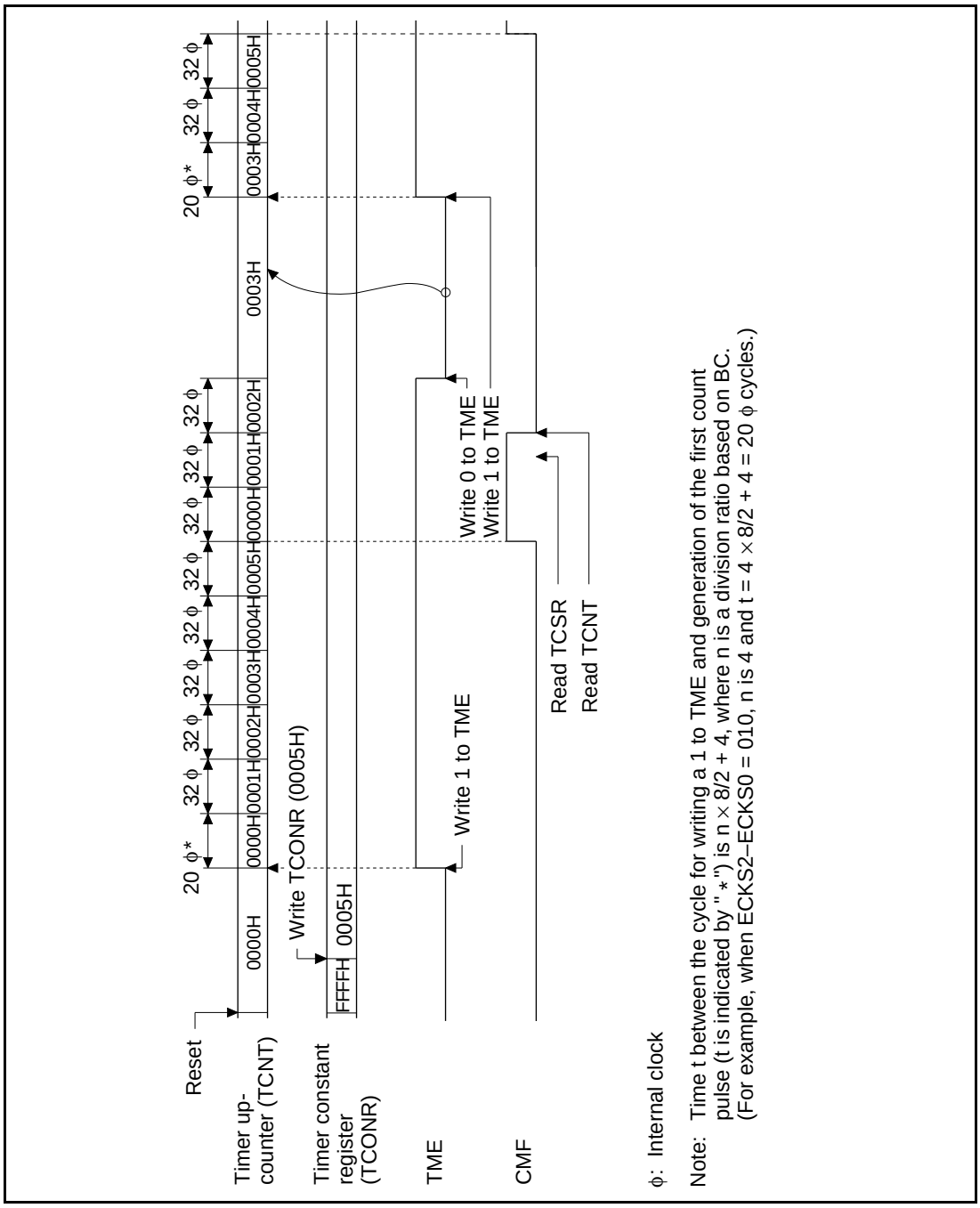


Figure 7.2 Timer Increment Timing (when the counter operating rate is BC/4)

7.3.2 Output Timing

Figure 7.3 shows the timer output change timing. When the timer up-counter (TCNT) and the timer constant register (TCONR) values match and TCNT is subsequently initialized to 0000H, the CMF bit of the timer control/status register (TCSR) is set to 1, one ϕ clock cycle later.

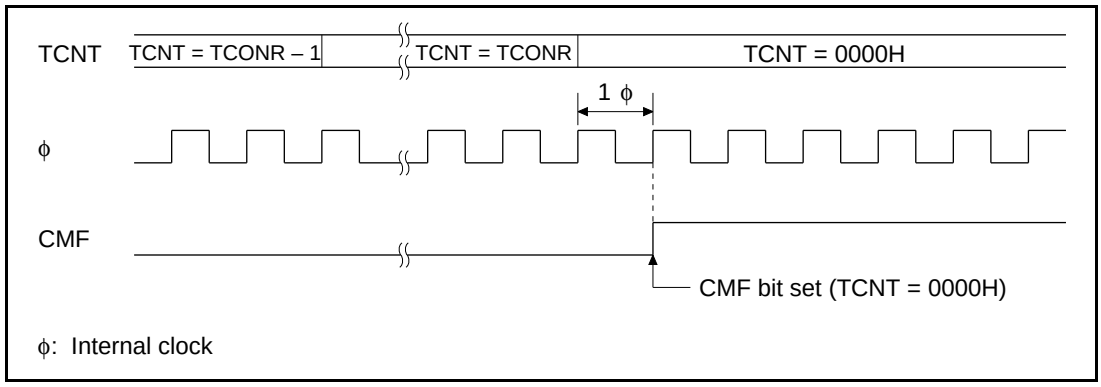


Figure 7.3 Timer Output Timing

7.4 Interrupt

When the timer up-counter (TCNT) and the timer constant register (TCONR) values match, the CMF bit of the timer control/status register (TCSR) is set to 1. Here, an interrupt is generated, if enabled. (Interrupts initiated by the CMF bit are enabled or disabled by the ECMI bit of TCSR.)

Figure 7.4 shows interrupt timing.

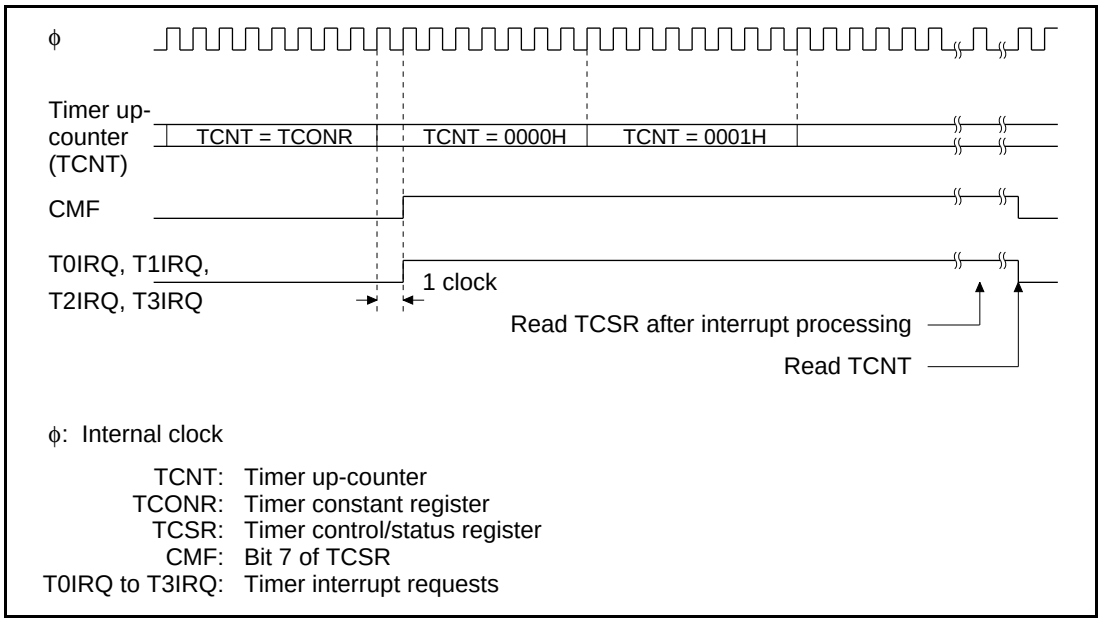


Figure 7.4 Interrupt Timing (when the counter operating rate is BC)

7.5 Operation in System Stop Mode

In system stop mode, the following events occur:

- The CMF and TME bits of the timer constant/status register (TCSR) are cleared.
- TCSR and the timer expand prescale register (TEPR) retain their current contents, except the CMF and TME bits of TCSR.
- The timer up-counter (TCNT) stops and is initialized to 0000H.
- No interrupt request is generated.

System stop mode is canceled by $\overline{\text{RESET}}$ input; TEPR is cleared simultaneously.

7.6 Reset Operation

The timers are initialized at reset as follows:

- The timer control/status register (TCSR) and the timer expand prescale register (TEPR) are initialized to 0000H.
- The timer up-counter (TCNT) stops and is initialized to 0000H.
- The timer constant register (TCONR) is initialized to FFFFH.
- No interrupt request is generated.

7.7 Precautions

When using the timer, keep the following in mind:

- Clear the TME bit of the timer control/status register (TCSR) before changing the timer operating clock.
- Reserved bits of the TCSR and the timer expand prescale register (TEPR) always read 0.

Section 8 Wait Controller

8.1 Overview

8.1.1 Functions

The HD64570 incorporates a wait controller, which extends DMA bus cycles by inserting wait states. This allows access to low-speed memory devices.

The wait controller has the following functional features:

- Wait states can be inserted using either the WAIT line (hardware) or a register (software).
- Insertion of 0 to 7 wait states is independently specified when each of three different memory areas is accessed.

8.1.2 Configuration and Operation

Figure 1.5 shows the wait controller block diagram. The wait controller consists of one wait control unit, one set of wait control registers (WCRL, WCRM, and WCRH), and one set of physical address boundary registers (PABR0 and PABR1).

Wait state insertion using the WAIT line is accomplished by driving the WAIT line high (active). Wait state insertion using the register is accomplished by loading WCRL, WCRM, and WCRH with the number of wait states to be inserted.

Wait states are inserted between states T_2 and T_3 of a DMA bus cycle.

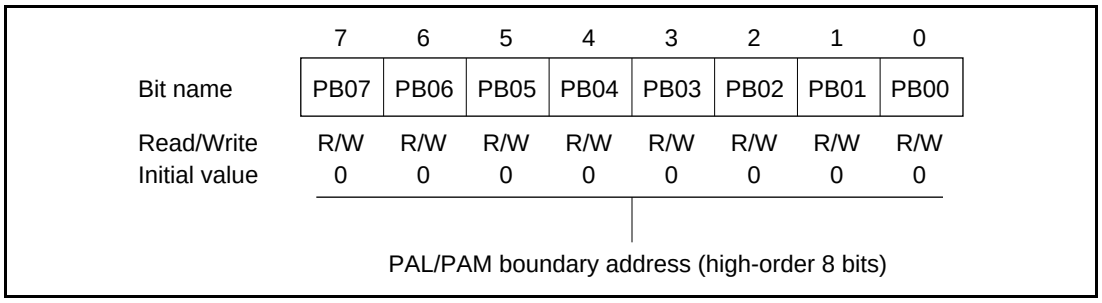
The memory space can be partitioned into three memory areas by the boundary addresses loaded into PABR0 and PABR1. The number of wait states inserted when each of these areas is accessed can be specified independently for each area.

8.2 Registers

8.2.1 Physical Address Boundary Registers 0, 1 (PABR0, PABR1)

The physical address boundary registers 0 and 1 (PABR0 and PABR1) specify the boundaries which divide the memory space into three areas.

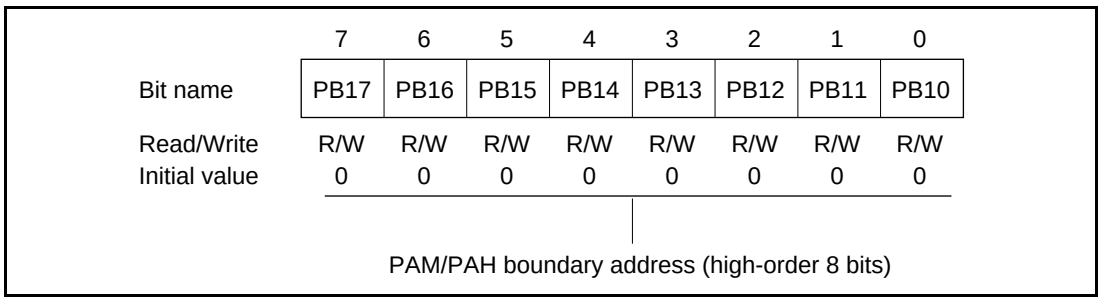
Physical Address Boundary Register 0 (PABR0): Specifies the high-order eight bits of the boundary address between the physical address low area (PAL) and the physical address middle area (PAM). This address is the lower limit address of PAM.



This register can specify only the high-order eight bits (A_{23}–A_{16}) of the boundary address; the remaining low-order 16 bits (A_{15}–A_0) are fixed to 0000H. Thus, each area is specified in 64-Kbyte units.

When PABR0 is set to 00H, the boundary is at the top of the memory space.

Physical Address Boundary Register 1 (PABR1): Specifies the high-order eight bits of the boundary address between PAM and the physical address high area (PAH). This address is the lower limit address of PAH.



This register can specify only the high-order eight bits (A_{23}–A_{16}) of the boundary address; the remaining low-order 16 bits (A_{15}–A_0) are fixed to 0000H. Thus, each area is specified in 64-Kbyte units.

When PABR1 is set to 00H, the boundary is at the top of the memory space.

Boundary Address Setting Examples: The memory space is usually divided into three areas: PAL, PAM, and PAH, as shown in figure 8.1. The boundary between PAL and PAM (the high-order eight bits of the lower limit address of PAM) is specified by PABR0, and that between PAM and PAH (the high-order eight bits of the lower limit address of PAH) is specified by PABR1, in 64-Kbyte units. In the figure, PABR0 and PABR1 are set to 01H and 40H, respectively. In this case, each memory area is specified as follows:

PAH: FFFFFFH (upper limit address) to 400000H (lower limit address)

PAM: 3FFFFFH (upper limit address) to 010000H (lower limit address)

PAL: 00FFFFH (upper limit address) to 000000H (lower limit address)

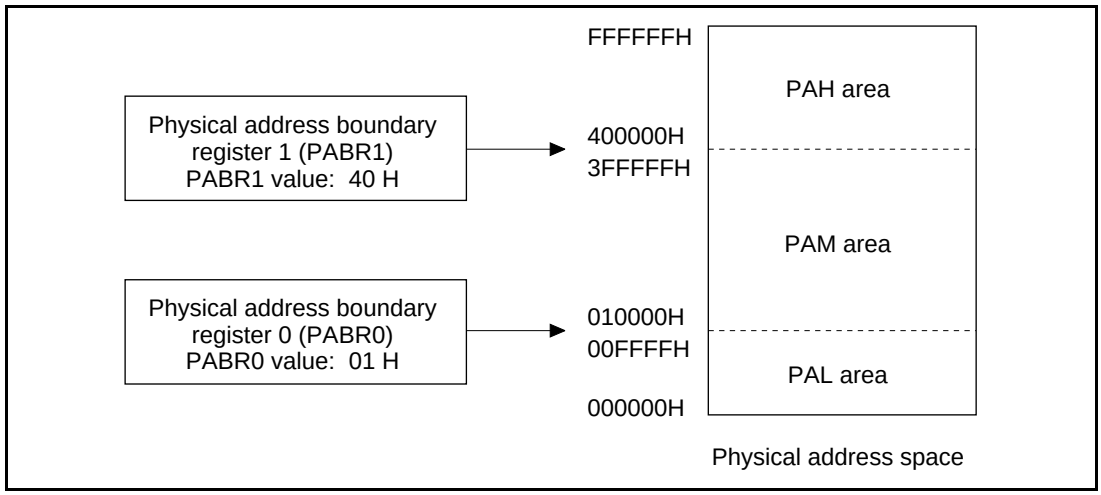


Figure 8.1 Memory Space Partitioned by PABR0 and PABR1

When either PABR0 or PABR1 is set to 00H, the boundary is at the top of the memory space. Accordingly, when PABR1 is set to 00H and PABR0 is set to 01H, each area is specified as follows:

PAH: —

PAM: FFFFFFH (upper limit address) to 010000H (lower limit address)

PAL: 00FFFFH (upper limit address) to 000000H (lower limit address)

Note that the memory space consists of only PAL and PAM because the PAM upper limit address is FFFFFFH.

Figures 8.2 (a) to 8.2 (d) show examples of when the physical address space is not partitioned, when it is partitioned into PAM and PAL, into PAH and PAL, and into three areas (PAH, PAM, and PAL), respectively.

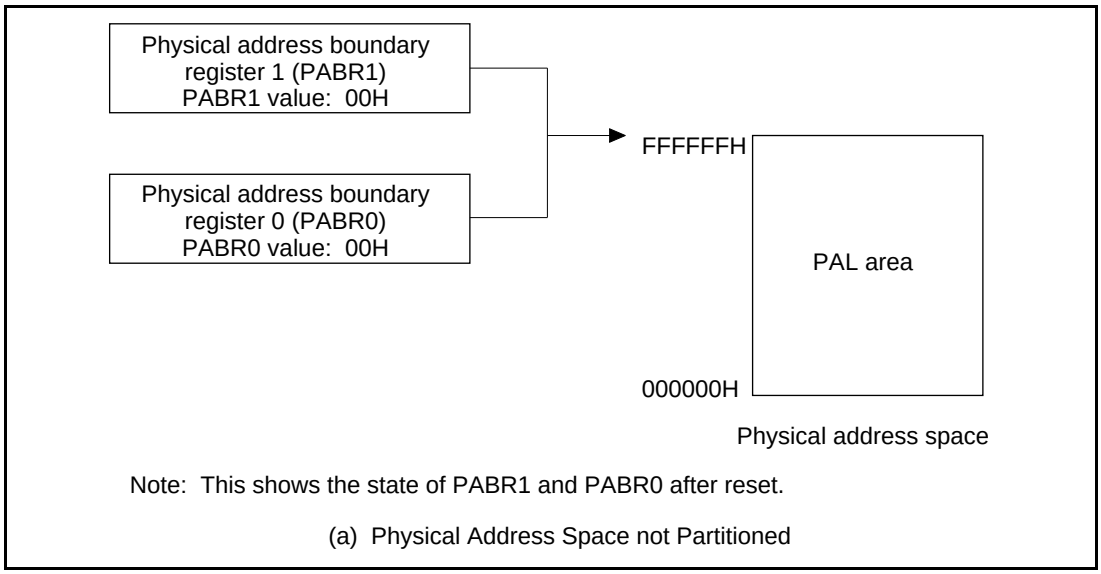


Figure 8.2 Boundary Address Setting Examples

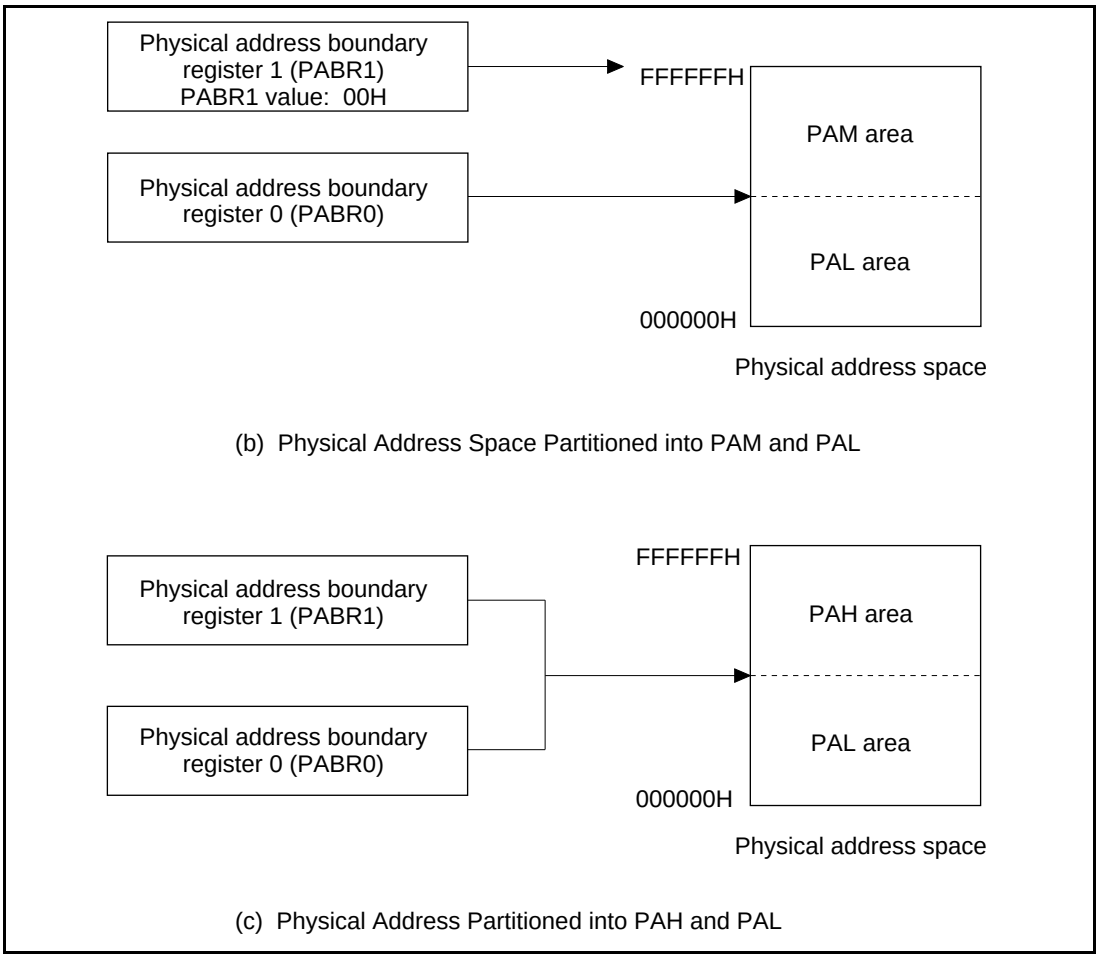


Figure 8.2 Boundary Address Setting Examples (cont)

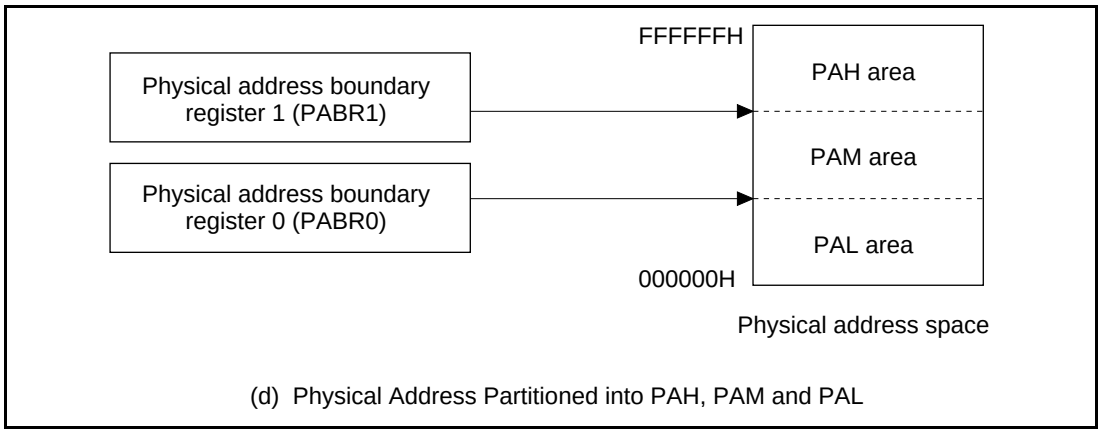


Figure 8.2 Boundary Address Setting Examples (cont)

Precautions: Normal operation is not guaranteed if the boundary specified by PABR0 is higher than that specified by PABR1. An example of this type is shown in figure 8.3. (Setting PABR0 to 00H and PABR1 to a value other than 00H may also disable normal operation.)

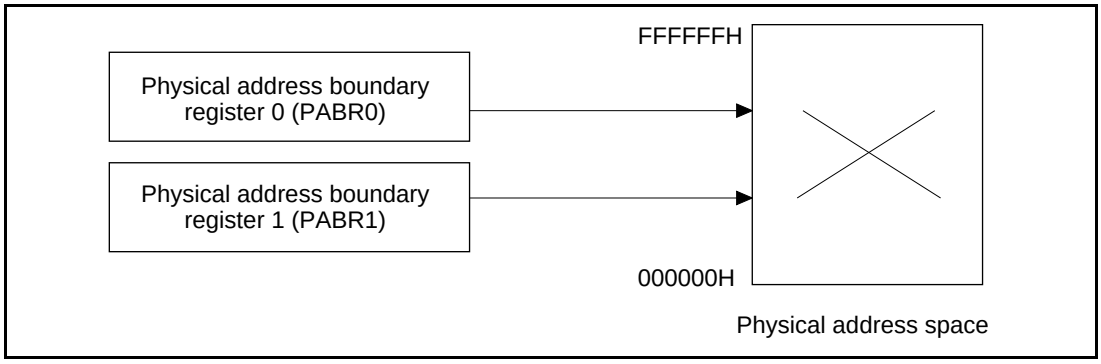
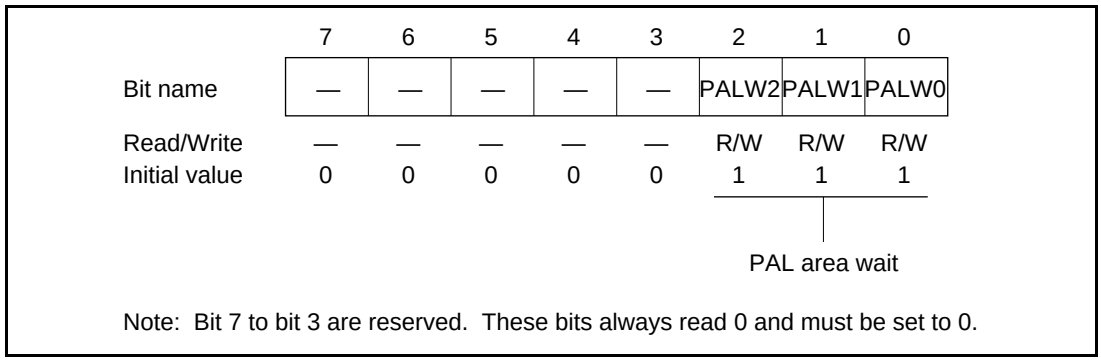


Figure 8.3 Incorrect Boundary Address Setting Example

8.2.2 Wait Control Registers L, M, H (WCRL, WCRM, WCRH)

Wait control registers WCRL, WCRM, and WCRH specify the number of wait states to be inserted each physical address areas and PAL, PAM, PAH.

Wait Control Register L (WCRL): Specifies the number of wait states to be inserted in a memory cycle when the PAL area is accessed.



Note: Bit 7 to bit 3 are reserved. These bits always read 0 and must be set to 0.

Bits 7–3: Reserved. These bits always read 0 and must be set to 0.

Bits 2–0 (PALW2–PALW0: PAL Area Wait): The functions of these bits are described below.

PALW2, PALW1, PALW0 = 0, 0, 0: Number of wait states = 0

PALW2, PALW1, PALW0 = 0, 0, 1: Number of wait states = 1

PALW2, PALW1, PALW0 = 0, 1, 0: Number of wait states = 2

PALW2, PALW1, PALW0 = 0, 1, 1: Number of wait states = 3

PALW2, PALW1, PALW0 = 1, 0, 0: Number of wait states = 4

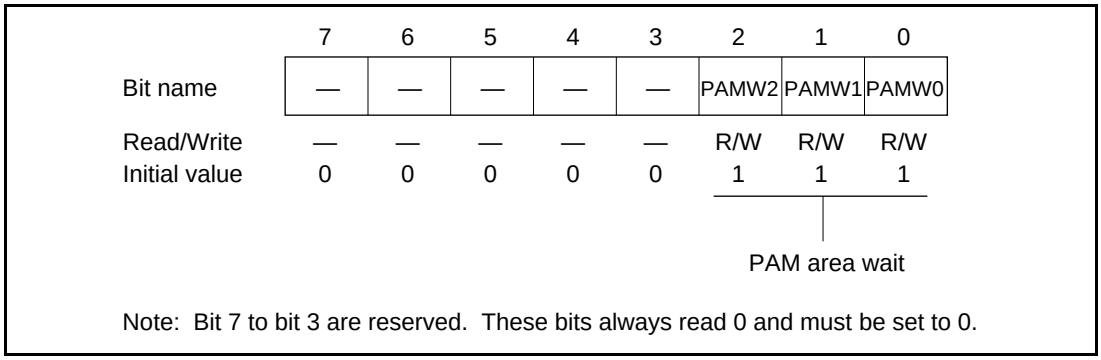
PALW2, PALW1, PALW0 = 1, 0, 1: Number of wait states = 5

PALW2, PALW1, PALW0 = 1, 1, 0: Number of wait states = 6

PALW2, PALW1, PALW0 = 1, 1, 1: Number of wait states = 7

Note that PALW2, PALW1, and PALW0 are initialized to (1, 1, 1) at reset.

Wait Control Register M (WCRM): Specifies the number of wait states to be inserted in a memory cycle when the PAM area is accessed.



Note: Bit 7 to bit 3 are reserved. These bits always read 0 and must be set to 0.

Bits 7–3: Reserved. These bits always read 0 and must be set to 0.

Bits 2–0 (PAMW2–PAMW0: PAM Area Wait): The function of these bits are described below.

PAMW2, PAMW1, PAMW0 = 0, 0, 0: Number of wait states = 0

PAMW2, PAMW1, PAMW0 = 0, 0, 1: Number of wait states = 1

PAMW2, PAMW1, PAMW0 = 0, 1, 0: Number of wait states = 2

PAMW2, PAMW1, PAMW0 = 0, 1, 1: Number of wait states = 3

PAMW2, PAMW1, PAMW0 = 1, 0, 0: Number of wait states = 4

PAMW2, PAMW1, PAMW0 = 1, 0, 1: Number of wait states = 5

PAMW2, PAMW1, PAMW0 = 1, 1, 0: Number of wait states = 6

PAMW2, PAMW1, PAMW0 = 1, 1, 1: Number of wait states = 7

Note that PAMW2, PAMW1, and PAMW0 are initialized to (1, 1, 1) at reset.

Wait Control Register H (WCRH): Specifies the number of wait states to be inserted in a memory cycle when the PAH area is accessed.

	7	6	5	4	3	2	1	0
Bit name	—	—	—	—	—	PAHW2	PAHW1	PAHW0
Read/Write	—	—	—	—	—	R/W	R/W	R/W
Initial value	0	0	0	0	0	1	1	1

PAH area wait

Note: Bits 7–3 are reserved. These bits always read 0 and must be set to 0.

Bits 7–3: Reserved. These bits always read 0 and must be set to 0.

Bits 2–0 (PAHW2–PAHW0: PAH Area Wait): The functions of these bits are described below.

PAHW2, PAHW1, PAHW0 = 0, 0, 0: Number of wait states = 0

PAHW2, PAHW1, PAHW0 = 0, 0, 1: Number of wait states = 1

PAHW2, PAHW1, PAHW0 = 0, 1, 0: Number of wait states = 2

PAHW2, PAHW1, PAHW0 = 0, 1, 1: Number of wait states = 3

PAHW2, PAHW1, PAHW0 = 1, 0, 0: Number of wait states = 4

PAHW2, PAHW1, PAHW0 = 1, 0, 1: Number of wait states = 5

PAHW2, PAHW1, PAHW0 = 1, 1, 0: Number of wait states = 6

PAHW2, PAHW1, PAHW0 = 1, 1, 1: Number of wait states = 7

Note that PAHW2, PAHW1, and PAHW0 are initialized to (1, 1, 1) at reset.

8.3 Operation

8.3.1 Wait State Insertion Using the WAIT Line

Wait states can be inserted between states T_2 and T_3 of states T_1–T_3 of a DMA bus cycle, using the WAIT line.

In wait state insertion using the WAIT line, when the WAIT line is driven high, a wait state (T_W) is inserted between states T_2 and T_3 (while the WAIT line maintains high). When the WAIT line is driven low, the cycle advances to state T_3.

Figure 8.4 shows the wait state insertion timing using the WAIT line. The WAIT line level is sampled at the falling edge of the system clock (CLK) pulse in state T_2 or T_W in CPU modes 1, 2, and 3, and is sampled at the rising edge in CPU mode 0. Each time the high level of the WAIT line is sampled at the falling edge (rising edge in CPU mode 0) of the CLK pulse in T_W state, another T_W state is inserted.

An unlimited number of wait states can be inserted. (When more wait states are requested by the register than by the WAIT line, the T_W states requested by the register are inserted.)

Note that, for driving the WAIT line signal high, the set-up time and hold time for the falling edge (rising edge in CPU mode 0) of the CLK pulse must be accounted for by synchronizing it to the rising edge (falling edge in CPU mode 0) of the CLK pulse. If not, correct operation is not guaranteed.

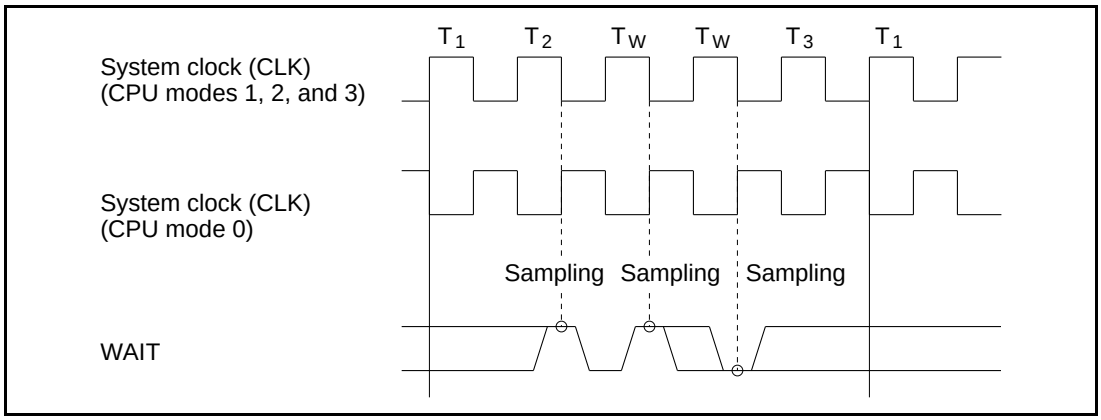


Figure 8.4 Wait State Insertion Timing Using the WAIT Line

8.3.2 Wait State Insertion Using the Register

Wait states can be inserted in a DMA bus cycle, using wait control registers WCRL, WCRM, and WCRH, eliminating the need for an external circuit. The optimum number of wait states can be inserted into a DMA bus cycle by software, according to the memory used. Figure 1.28 shows an example of dividing the memory space for interfacing three different types of memory. In this example, any desired number of wait states can be independently specified for each of the three types of memory. (When more wait states are requested using the WAIT line than those using the register, as many T_W states as requested by the WAIT line are inserted.) Physical address boundaries for dividing the memory space into three memory areas are specified by the physical address boundary registers 0 and 1 (PABR0 and PABR1). For details, see section 8.2.1, Physical

Address Boundary Registers 0, 1 (PABR0, PABR1). The number of wait states to be inserted in each memory area is specified by wait control registers WCRL, WCRM, and WCRH. For details, see section 8.2.2, Wait Control Registers L, M, H (WCRL, WCRM, WCRH).

8.4 Operation in System Stop Mode

When the wait controller stops in system stop mode, the current register contents are retained.

8.5 Reset Operation

At reset, the wait controller stops and its registers are initialized as follows:

- The wait control registers (WCRL, WCRM, and WCRH) are initialized so that the maximum number of wait states are inserted.
- The physical address boundary registers (PABR0 and PABR1) are initialized to 00H. This results in the physical address space consisting of only PAL. Accordingly, the number of wait states specified by WCRL is inserted in a DMA cycle.

8.6 Precautions

If wait state insertion is simultaneously requested by the register and WAIT line, the number of wait states specified by the register are inserted. If the WAIT line later requests more wait states than the register, the additional wait states are then inserted.

Section 9 Application Examples

9.1 Application Examples

9.1.1 Serial Data Transfer by MPU and DMAC

Transfer of Transmit Data: Three different types of data transfer are described below.

- Polling

  The MPU determines data write timing to the transmit buffer by monitoring the TXRDY bit of status register 0 (ST0). In this case, the TXRDY interrupt must be disabled.
- Interrupt

  The MPU writes data to the transmit buffer when receiving a TXRDY interrupt. The TXRDY interrupt is issued when the TXRDYE bit of interrupt enable register 0 (IE0) is set to 1 in TX ready state (specified by TX ready control registers 0 and 1 (TRC0 and TRC1)). In this case, the on-chip DMAC must be disabled for transfer requests.
- DMA transfer

  The on-chip DMAC controls data write operation to the transmit buffer using the DMA transfer request signal. This signal is issued when the TXRDY bit is set to 1. In this case, TRC0 must be set to a large enough value to prevent underrun errors, and the TXRDY interrupt must be disabled.

Transfer of Receive Data: Three different types of data transfer are described below.

- Polling

  The MPU determines data read timing from the receive buffer by monitoring the RXRDY bit of ST0. In this case, the RXRDY interrupt must be disabled.
- Interrupt

  The MPU reads data from the receive buffer when receiving an RXRDY interrupt. The RXRDY interrupt is enabled when the RXRDYE bit of IE0 is set to 1 in RX ready state (specified by RX ready control register (RRC)). In this case, the on-chip DMAC must be disabled for transfer requests.
- DMA transfer

  The on-chip DMAC controls data read operations from the receive buffer using the DMA transfer request signal. This signal is issued when the RXRDY bit is set to 1. In this case, the RXRDY interrupt must be disabled.

9.1.2 Transmission by Programmed I/O (Bi-Sync Mode)

Initialization: An example of an initialization program is given below.

CMD	fl	21H	Resets channel.
MD0	fl	44H	Specifies bi-sync mode.
			Disables the auto-enable function.
			Specifies CRC-16 mode, and presets to all 0s.
MD2	fl	00H	Specifies the NRZ code.
			Specifies full-duplex mode.
CTL	fl	11H	Specifies idle pattern transmission.
			Specifies RTS line high-level output.
TRC0	fl	00H	TXRDY bit = 1 when the transmit buffer is empty.
TRC1	fl	00H	TXRDY bit = 0 when the transmit buffer is not empty.
TXS	fl	00H	Specifies TXC line input for the transmit clock.
IE0	fl	82H	Enables TXINT interrupts.
			Enables TXRDY interrupts.
IE1	fl	80H	Enables underrun interrupts.
SA0	fl	16H	Specifies a SYN character.
SA1	fl	16H	Specifies a SYN character.
IDL	fl	XXH	Specifies a leading pad or SYN character.
CMD	fl	02H	Enables transmission.
TRB	fl	Transmit data	Transmits a leading pad, and a SYN character, followed by transmit data.

CMD: Command register
MD0: Mode register 0
MD2: Mode register 2
CTL: Control register
TRC0: TX ready control register 0
TRC1: TX ready control register 1
TXS: TX clock source register
IE0: Interrupt enable register 0
IE1: Interrupt enable register 1
SA0: Synchronous/address register 0
SA1: Synchronous/address register 1
IDL: Idle pattern register

Transmit Processing Routine: Two examples of transmission processing routines are given in figures 9.1 and 9.2.

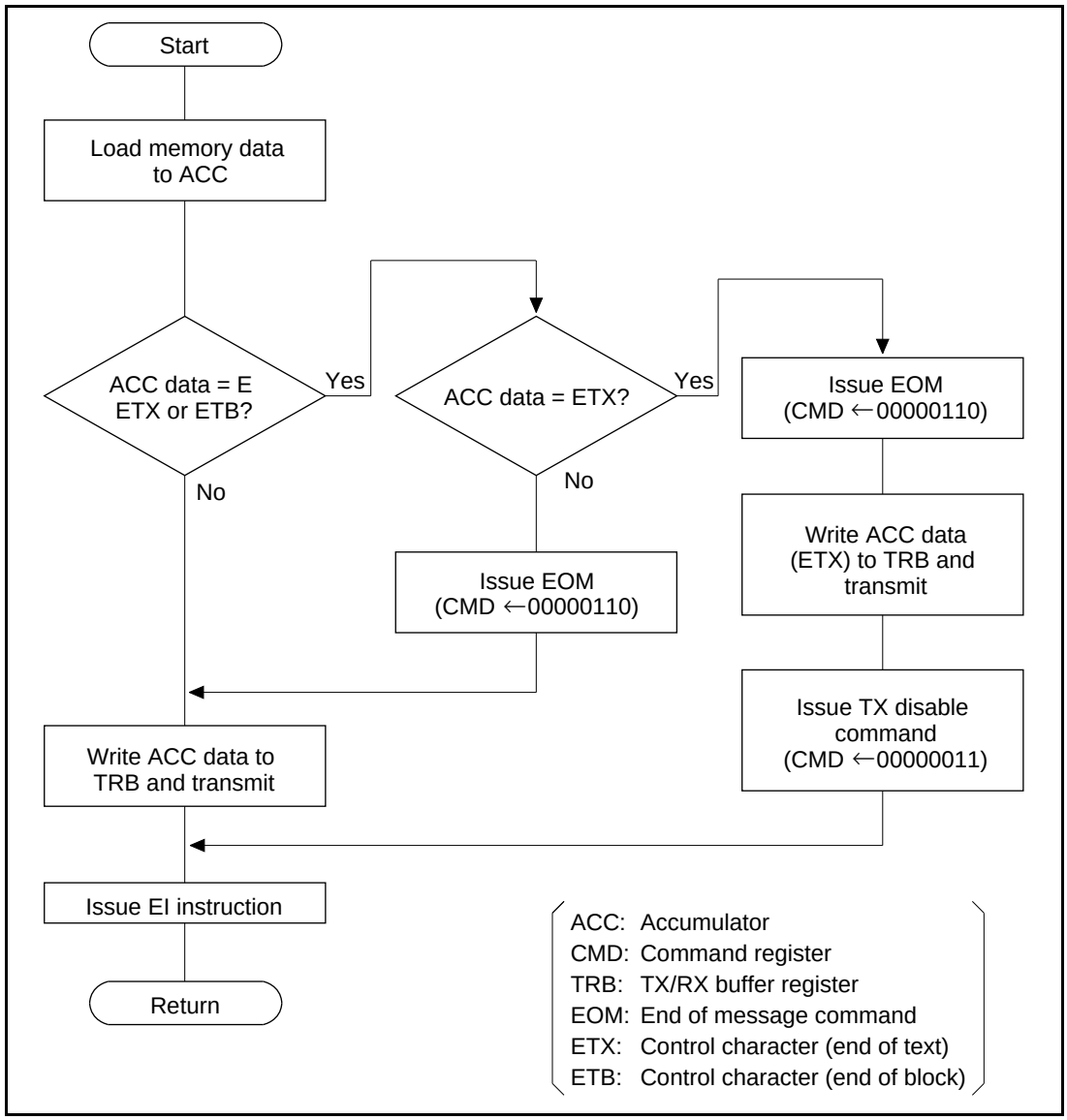


Figure 9.1 TXRDY Interrupt Processing Routine (using HD64180)

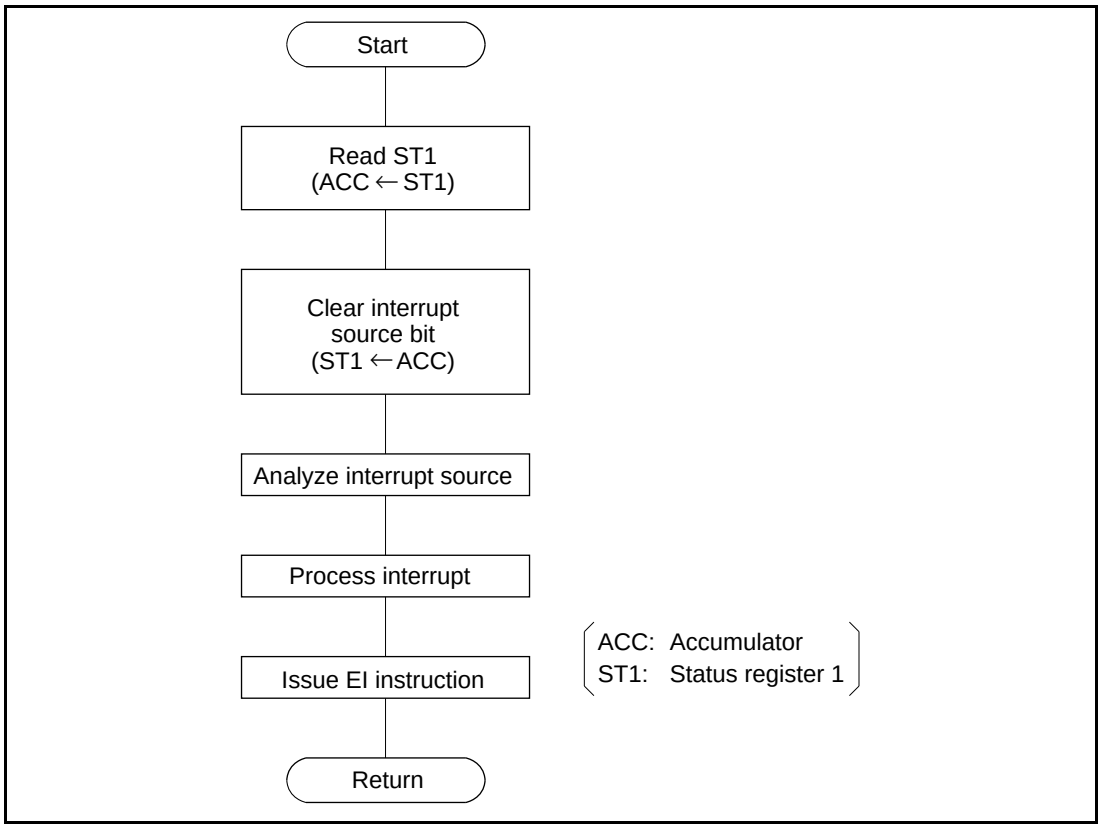


Figure 9.2 TXINT Interrupt Processing Routine (using HD64180)

9.1.3 Reception by Programmed I/O (Bi-Sync Mode)

Initialization: An example of an initialization program is given below.

CMD	fl	21H	Resets channel.
MD0	fl	44H	Specifies bi-sync mode.
			Disables the auto-enable function.
			Specifies CRC-16 mode, and presets to all 0s.
MD2	fl	00H	Specifies the NRZ code.
			Specifies full-duplex mode.
CTL	fl	05H	Specifies SYN character load.
RRC	fl	00H	RXRDY = 1 when the receive buffer is not empty.
RXS	fl	00H	Specifies RXC line input for the receive clock.
IE0	fl	41H	Enables RXINT interrupts.
			Enables RXRDY interrupts.

IE1 fl 10H..... Enables SYNCD interrupts.
IE2 fl 08H..... Enables overrun interrupts.
SA0 fl 16H..... Specifies a SYN character.
SA1 fl 16H..... Specifies a SYN character.
CMD fl 12H..... Enables reception.

CMD: Command register
MD0: Mode register 0
MD2: Mode register 2
CTL: Control register
RRC: RX ready control register
RXS: RX clock source register
IE0: Interrupt enable register 0
IE1: Interrupt enable register 1
IE2: Interrupt enable register 2
SA0: Synchronous/address register 0
SA1: Synchronous/address register 1

Reception Processing Routine: Two examples of reception processing routines are given in figures 9.3 and 9.4.

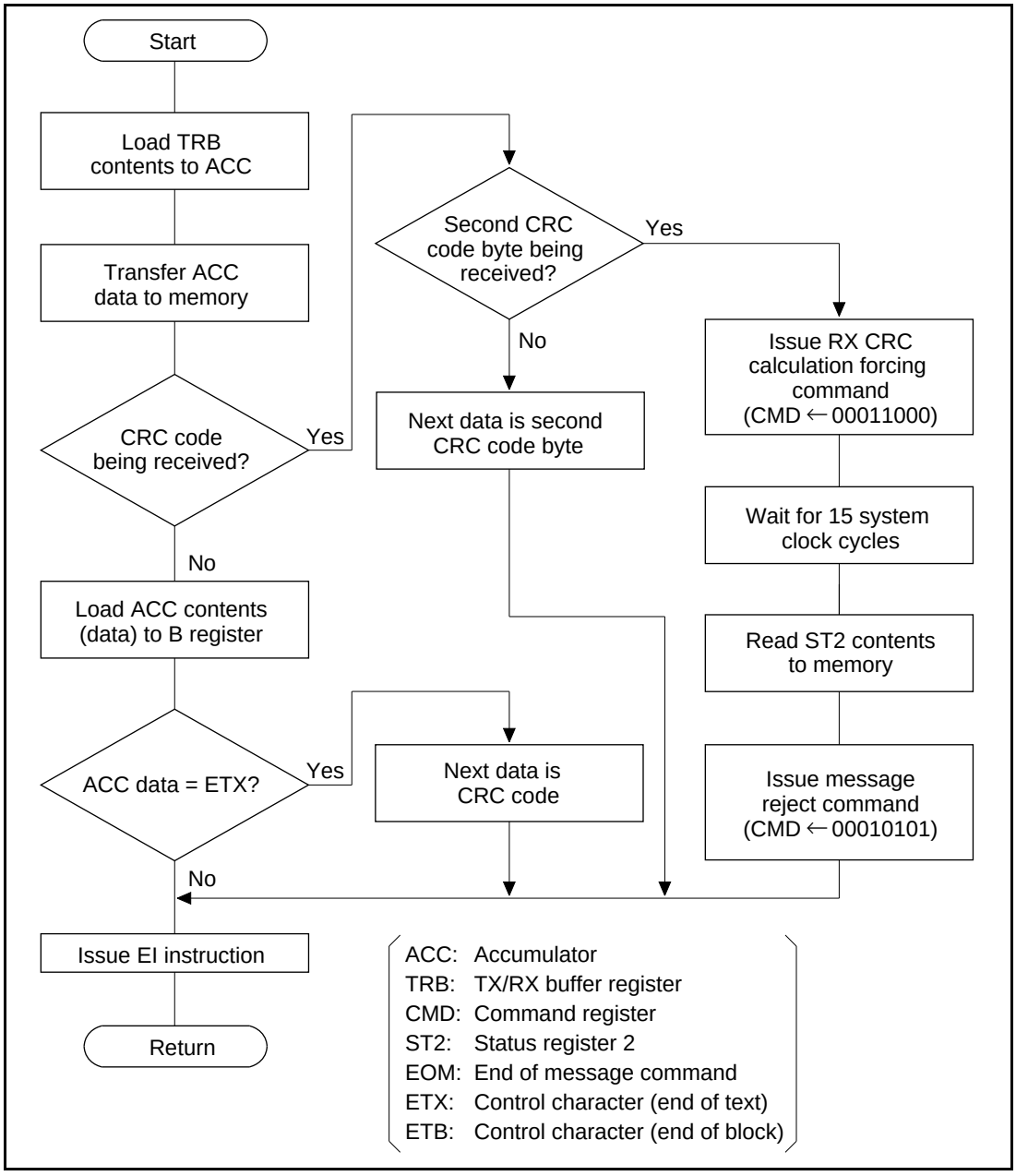


Figure 9.3 RXRDY Interrupt Processing Routine (using HD64180)

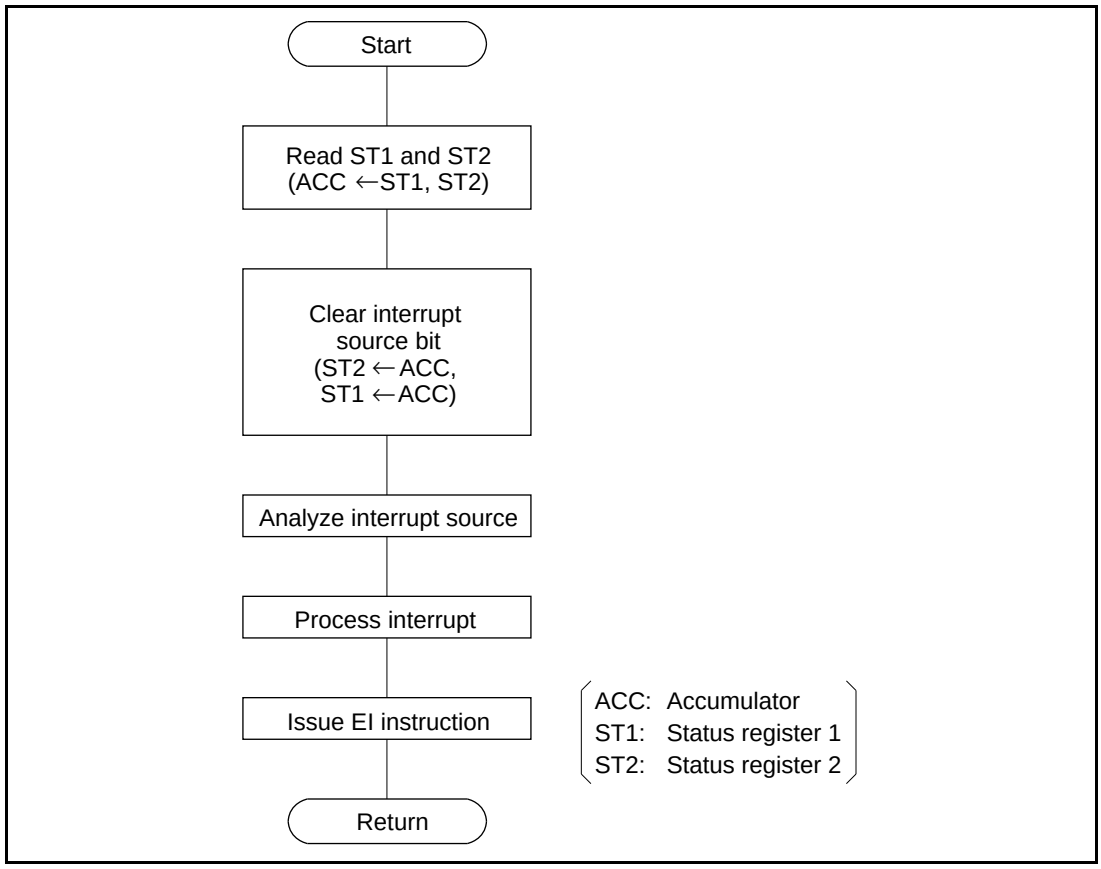


Figure 9.4 RXINT Interrupt Processing Routine (using HD64180)

9.1.4 Transmission in DMA Chained-Block Transfer Mode (Bit Synchronous HDLC Mode)

Initialization: An example of an initialization program is given below.

CMD	fl	21H	Resets channel.
MD0	fl	87H	Specifies bit synchronous HDLC mode. Specifies CRC-CCITT mode, and presets to all 1s.
MD2	fl	00H	Specifies the NRZ code. Specifies full-duplex mode.
CTL	fl	11H	Specifies idle pattern transmission. Specifies RTS line high-level output.
TRC0	fl	1FH	TXRDY bit = 1 when the transmit buffer is not full.
TRC1	fl	1FH	TXRDY bit = 0 when the transmit buffer is full.

TXS	fl	00H	Specifies TXC line input for transmit clock.
IE0	fl	80H	Enables TXINT interrupts.
IE1	fl	80H	Enables underrun interrupts.
IDL	fl	XXH	Specifies a leading pad or flag pattern (sets DMAC register).
CMD	fl	02H	Enables transmission.

CMD: Command register
MD0: Mode register 0
MD2: Mode register 2
CTL: Control register
TRC0: TX ready control register 0
TRC1: TX ready control register 1
TXS: TX clock source register
IE0: Interrupt enable register 0
IE1: Interrupt enable register 1
IDL: Idle pattern register

Transmission Processing Routine: An example of a transmission processing routine is given in figure 9.5.

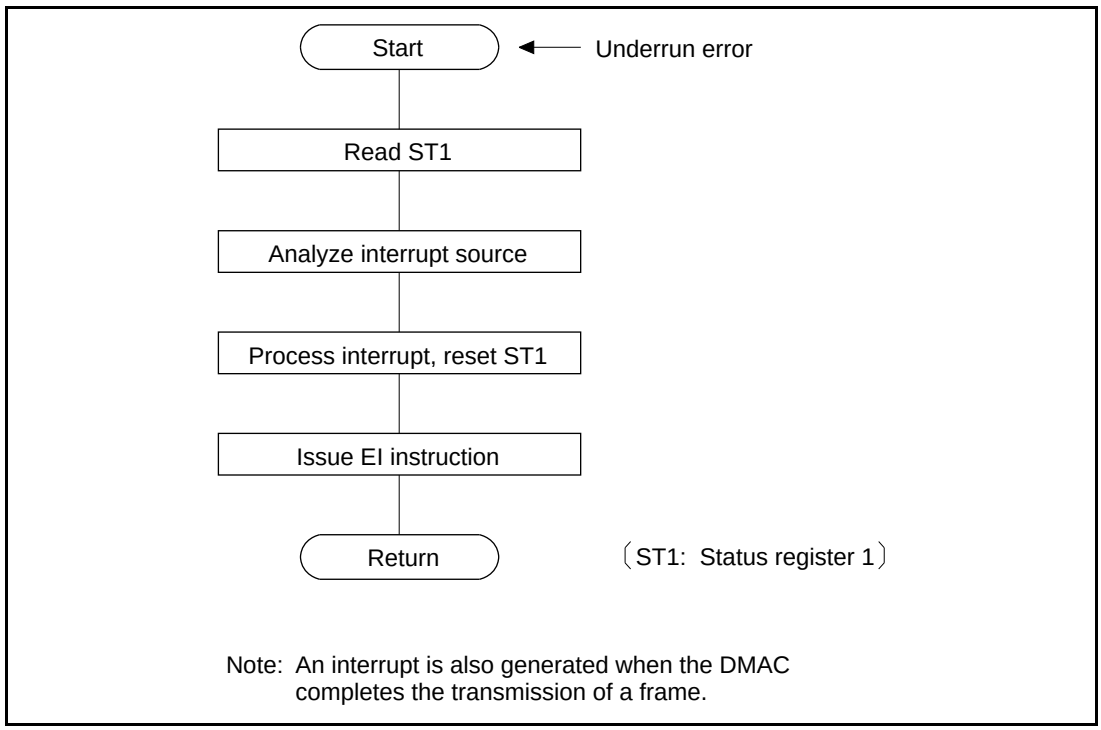


Figure 9.5 TXINT Interrupt Processing Routine (using HD64180)

9.1.5 Reception in DMA Chained-Block Transfer Mode (Bit Synchronous HDLC Mode)

Initialization: An example of an initialization program is given below.

CMD fl 21H..... Resets channel.
MD0 fl 87H..... Specifies bit synchronous HDLC mode.
Specifies CRC-CCITT mode, and presets to all 1s.
MD1 fl 40H..... Specifies single address 1.
MD2 fl 00H..... Specifies the NRZ code.
Specifies full-duplex mode.
CTL fl 01H..... Specifies FCS no-load.
RRC fl 00H..... RXRDY = 1 when the receive buffer is not empty.
RXS fl 00H..... Specifies RXC line input for the receive clock.
IE0 fl 40H..... Enables RXINT interrupts.
IE1 fl 03H..... Enables abort detection interrupts.

Enables idle detection interrupts.

SA0 fl XXH ..... Specifies secondary station address. (Sets DMAC registers.)

CMD fl 02H ..... Enables reception.

CMD: Command register
MD0: Mode register 0
MD2: Mode register 2
CTL: Control register
RRC: RX ready control register
RXS: RX clock source register
IE0: Interrupt enable register 0
IE1: Interrupt enable register 1
SA0: Synchronous/address register 0

Reception Processing Routine: An example of a reception processing routine is given in figure 9.6.

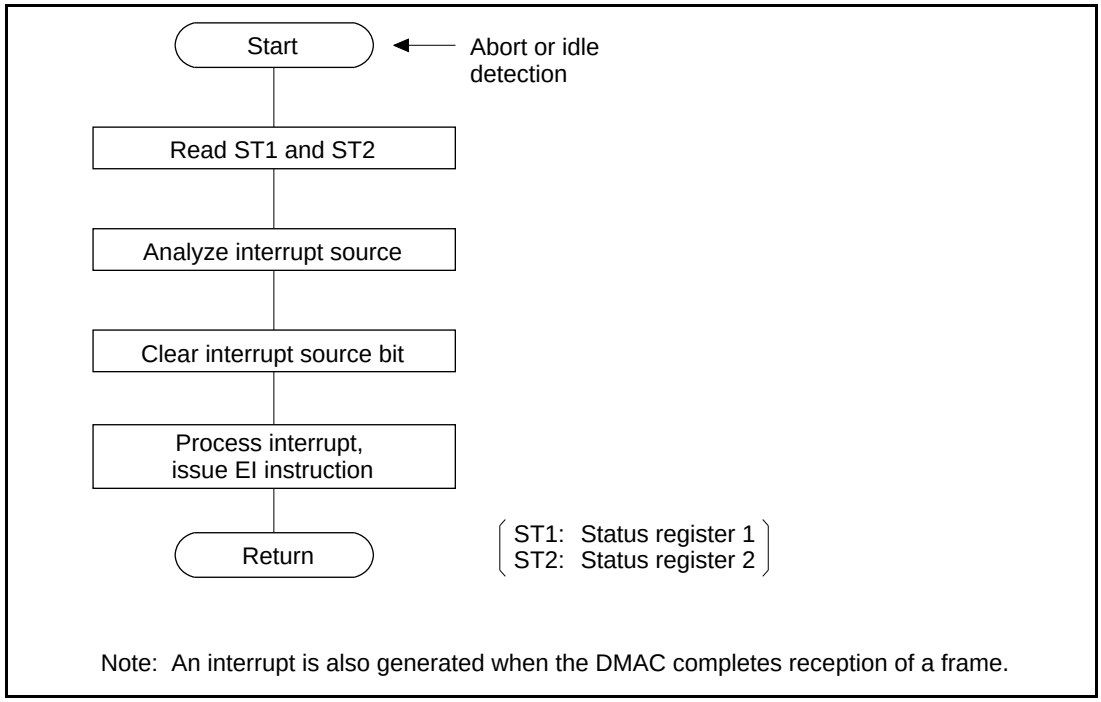


Figure 9.6 RXINT Interrupt Processing Routine (using HD64180)

9.2 Application Circuits

9.2.1 System Configuration Example

A typical system configuration incorporating the SCA is shown in figure 9.7.

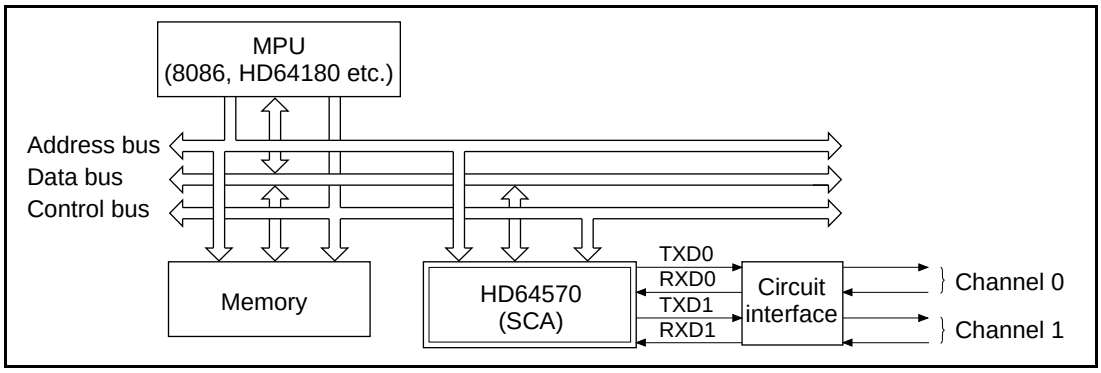


Figure 9.7 System Configuration Incorporating the SCA

9.2.2 Bus Arbitration Block

The SCA $\overline{\text{BUSREQ}}$ (HOLD) signal indicates a bus request, but not bus acquisition. Bus acquisition is indicated by the $\overline{\text{BUSY}}$ signal. When connecting the SCA to the MPU that uses $\overline{\text{BUSREQ}}$ and $\overline{\text{BUSACK}}$ to arbitrate the bus:

1. Input the result of ORing the SCA $\overline{\text{BUSREQ}}$ signal with the $\overline{\text{BUSY}}$ signal to the MPU $\overline{\text{BUSREQ}}$ line.
2. Input the result of ANDing the MPU $\overline{\text{BUSACK}}$ signal with the SCA $\overline{\text{BUSREQ}}$ signal to the SCA $\overline{\text{BUSACK}}$ line.

This is shown in figure 9.8. Possible bus masters are one MPU and one SCA; no other bus masters are considered. For more details of bus arbitration for a specific MPU, refer to figure 9.9.

Note that if the SCA $\overline{\text{BUSREQ}}$ and $\overline{\text{BUSACK}}$ signals and the MPU $\overline{\text{BUSREQ}}$ and $\overline{\text{BUSACK}}$ signals are connected to each other, respectively, malfunction occurs if the SCA $\overline{\text{BUSREQ}}$ signal is activated after it has been temporarily inactivated for one clock pulse. This is because the SCA, mistakenly determining that it has acquired the bus because the $\overline{\text{BUSACK}}$ signal is not inactivated, starts a DMA transfer. At the same time, the MPU, also mistakenly determining that it has acquired the bus, starts using the bus and inactivates the $\overline{\text{BUSACK}}$ signal. As a result, the SCA and the MPU use the bus at the same time, causing a malfunction.

To securely inform the HD64180 that the SCA has acquired the bus, connect the OR between the SCA $\overline{\text{BUSREQ}}$ and $\overline{\text{BUSY}}$ signals to the HD64180 $\overline{\text{BUSREQ}}$ pin. To securely inform the SCA that the HD64180 has released the bus, also connect the result of ANDing the SCA $\overline{\text{BUSREQ}}$-synchronous signal with the HD64180 $\overline{\text{BUSACK}}$ signal to the SCA $\overline{\text{BUSACK}}$ pin. Note that there are no wait cycles during DMA transfers from the SCA.

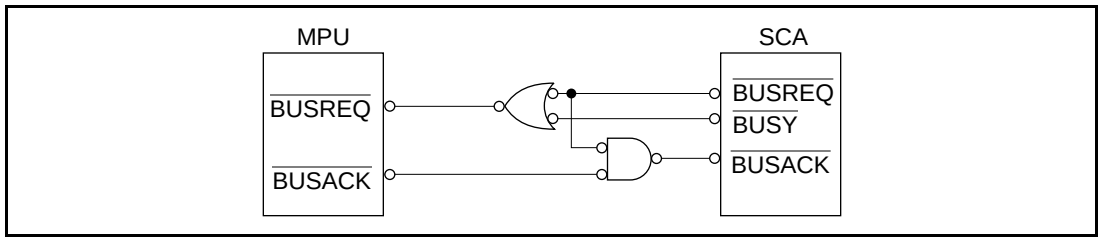


Figure 9.8 BUSREQ Control Circuit

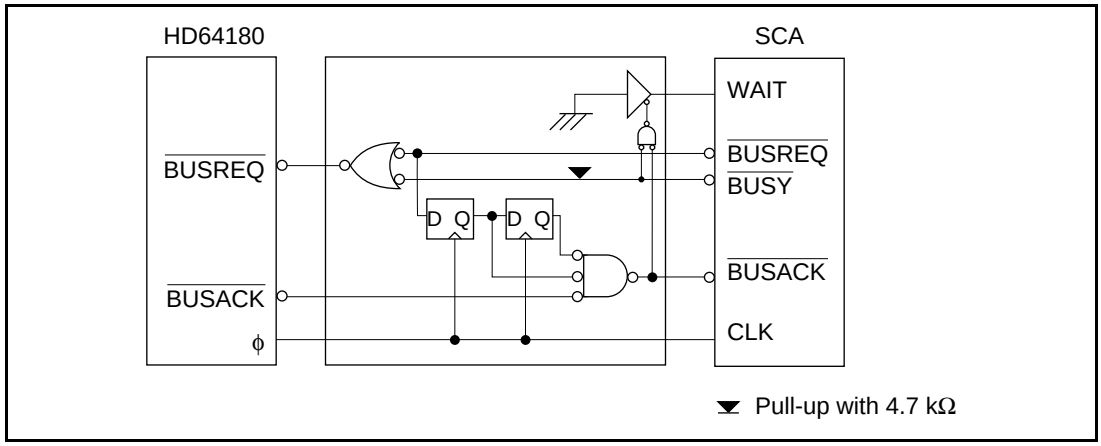


Figure 9.9 Diagrams of the Bus Arbitration Circuit

Section 10 Electrical Characteristics

10.1 Electrical Characteristics of HD64570CP and HD64570F

10.1.1 Absolute Maximum Ratings

Table 10.1 Absolute Maximum Ratings

Item	Symbol	Rating	Unit
Supply voltage	V_{CC}	–0.3 to +7.0	V
Input voltage	V_{in}	–0.3 to V_{CC} +0.3	V
Operating temperature	T_{opr}	–20 to +75	°C
Storage temperature	T_{stg}	–55 to +150	°C

Caution: Permanent damage to the HD64570 may result if it is subjected to conditions that exceed the absolute maximum ratings. To assure normal operation, the following conditions should be satisfied: $V_{SS} \leq V_{in} \leq V_{CC}$

10.1.2 DC Characteristics

Table 10.2 DC Characteristics

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –20 to +75°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Conditions
Input high level voltage for $\overline{\text{RESET}}$ and CLK	V_{IH1}	V_{CC} – 0.6	—	V_{CC} + 0.3	V	
Input high level voltage for pins other than $\overline{\text{RESET}}$ and CLK	V_{IH2}	2.0	—	V_{CC} + 0.3	V	
Input low level voltage for $\overline{\text{RESET}}$ and CLK	V_{IL1}	–0.3	—	0.6	V	
Input low level voltage for pins other than $\overline{\text{RESET}}$ and CLK	V_{IL2}	–0.3	—	0.8	V	
Output high level voltage for all output pins	V_{OH}	2.4	—	—	V	I_{OH} = – 200 µA
		V_{CC} – 1.2	—	—		I_{OH} = – 20 µA
Output low level voltage for all output pins	V_{OL}	—	—	0.45	V	I_{OL} = 2.2 mA
Input leakage current	I_{IL}	—	—	1.0	µA	V_{in} = 0.5 to V_{CC} – 0.5
Three-state leakage current	I_{TL}	—	—	1.0	µA	V_{in} = 0.5 to V_{CC} – 0.5
Current consumption(Note) (normal operation)	I_{CC}	—	60	120	mA	f = 10 MHz
Current consumption(Note) (system stop mode)		—	2	5	mA	f = 10 MHz
Pin capacitance	Cp	—	—	20	pF	V_{in} = 0V, f = 1 MHz, Ta = 25°C

Note: V_{IH} min = V_{CC} – 1.0 V, V_{IL} max = 0.8 V (when no output pins are loaded)

10.1.3 AC Characteristics

Table 10.3 CPU Mode 0 Slave Mode Bus Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –20 to +75°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
$\overline{CS}$ set-up time	t_{CSS}	30	—	—	ns	Figure 10.1
$\overline{CS}$ hold time 1	t_{CSH1}	20	—	—	ns	
$\overline{CS}$ hold time 2	t_{CSH2}	0	—	—	ns	
Address set-up time	t_{ADS}	30	—	—	ns	
Address hold time	t_{ADH}	0	—	—	ns	
$\overline{RD}$ active set-up time	t_{RDS1}	30	—	—	ns	
$\overline{RD}$ inactive set-up time	t_{RDS2}	30	—	—	ns	
$\overline{RD}$ inactive hold time	t_{RDH1}	10	—	—	ns	
$\overline{RD}$ active hold time	t_{RDH2}	0	—	—	ns	
$\overline{WR}$ active set-up time	t_{WRS1}	30	—	—	ns	
$\overline{WR}$ inactive set-up time	t_{WRS2}	30	—	—	ns	
$\overline{WR}$ inactive hold time	t_{WRH1}	10	—	—	ns	
$\overline{WR}$ active hold time	t_{WRH2}	0	—	—	ns	
WAIT active delay time	t_{WTD1}	—	—	50	ns	
WAIT inactive delay time	t_{WTD2}	—	—	50	ns	
Read data active delay time	t_{DBD1}	—	—	65	ns	
Read data hold time	t_{DBD2}	10	—	—	ns	
Read data floating delay time	t_{DBZ}	—	—	60	ns	
Write data set-up time	t_{DBS}	25	—	—	ns	
Write data hold time	t_{DBH}	20	—	—	ns	

Notes: 1. The CLK timing is the same in this mode and DMA mode. See table 10.7.
2. For the measurement conditions of AC characteristics, see figure 9.25.

Table 10.4 CPU Mode 1 Slave Mode Bus Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –20 to +75°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Address set-up time	t_{ADS}	30	—	—	ns	Figure 10.2
Address hold time	t_{ADH}	0	—	—	ns	
$\overline{CS}$ set-up time	t_{CSS}	30	—	—	ns	
$\overline{CS}$ hold time	t_{CSH}	0	—	—	ns	
$\overline{RD}$ active set-up time	t_{RDS1}	30	—	—	ns	
$\overline{RD}$ inactive set-up time	t_{RDS2}	30	—	—	ns	
$\overline{RD}$ inactive hold time	t_{RDH1}	10	—	—	ns	
$\overline{RD}$ active hold time	t_{RDH2}	0	—	—	ns	
$\overline{WR}$ active set-up time	t_{WRS1}	30	—	—	ns	
$\overline{WR}$ inactive set-up time	t_{WRS2}	30	—	—	ns	
$\overline{WR}$ inactive hold time	t_{WRH1}	10	—	—	ns	
$\overline{WR}$ active hold time	t_{WRH2}	0	—	—	ns	
WAIT active delay time	t_{WTD1}	—	—	50	ns	
WAIT inactive delay time	t_{WTD2}	—	—	60	ns	
Read data active delay time	t_{DBD1}	—	—	60	ns	
Read data hold time	t_{DBD2}	6	—	—	ns	
Read data floating delay time	t_{DBZ}	—	—	60	ns	
Write data set-up time	t_{DBS}	25	—	—	ns	
Write data hold time	t_{DBH}	20	—	—	ns	

Note: The CLK timing is the same in this mode and DMA mode. See table 10.8.

Table 10.5 CPU Mode 2 Slave Mode Bus Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –20 to +75°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Address set-up time	t_{ADS}	30	—	—	ns	Figure 10.3
Address hold time	t_{ADH}	0	—	—	ns	
$\overline{AS}$ set-up time	t_{ASS}	30	—	—	ns	
$\overline{AS}$ hold time 1	t_{ASH1}	0	—	—	ns	
$\overline{AS}$ hold time 2	t_{ASH2}	0	—	—	ns	
$\overline{CS}$ set-up time	t_{CSS}	30	—	—	ns	
$\overline{CS}$ hold time 1	t_{CSH1}	0	—	—	ns	
$\overline{CS}$ hold time 2	t_{CSH2}	0	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ active set-up time	t_{DSS1}	30	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ inactive set-up time	t_{DSS2}	30	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ inactive hold time	t_{DSH1}	10	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ active hold time	t_{DSH2}	0	—	—	ns	
R/$\overline{W}$ set-up time	t_{RWS}	30	—	—	ns	
R/$\overline{W}$ hold time 1	t_{RWH1}	0	—	—	ns	
R/$\overline{W}$ hold time 2	t_{RWH2}	0	—	—	ns	
WAIT inactive delay time	t_{WTD1}	—	—	50	ns	
WAIT active delay time	t_{WTD2}	—	—	60	ns	
Read data active delay time	t_{DBD1}	—	—	60	ns	
Read data hold time	t_{DBD2}	10	—	—	ns	
Read data floating delay time	t_{DBZ}	—	—	60	ns	
Write data set-up time	t_{DBS}	25	—	—	ns	
Write data hold time	t_{DBH}	20	—	—	ns	
Write data WAIT hold time	t_{DBWH}	0	—	—	ns	

Note: The CLK timing is the same in this mode and DMA mode. See table 10.9.

Table 10.6 CPU Mode 3 Slave Mode Bus Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –20 to +75°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Address set-up time	t_{ADS}	30	—	—	ns	Figure 10.4
Address hold time	t_{ADH}	0	—	—	ns	
$\overline{AS}$ set-up time	t_{ASS}	30	—	—	ns	
$\overline{AS}$ hold time	t_{ASH}	0	—	—	ns	
$\overline{CS}$ set-up time	t_{CSS}	30	—	—	ns	
$\overline{CS}$ hold time	t_{CSH}	0	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ active set-up time	t_{DSS1}	30	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ inactive set-up time	t_{DSS2}	30	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ inactive hold time	t_{DSH1}	10	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ active hold time	t_{DSH2}	0	—	—	ns	
R/$\overline{W}$ set-up time	t_{RWS}	30	—	—	ns	
R/$\overline{W}$ hold time 1	t_{RWH1}	0	—	—	ns	
R/$\overline{W}$ hold time 2	t_{RWH2}	0	—	—	ns	
WAIT inactive delay time	t_{WTD1}	—	—	50	ns	
WAIT active delay time	t_{WTD2}	—	—	50	ns	
Read data active delay time	t_{DBD1}	—	—	60	ns	
Read data hold time	t_{DBD2}	10	—	—	ns	
Read data floating delay time	t_{DBZ}	—	—	60	ns	
Write data set-up time	t_{DBS}	25	—	—	ns	
Write data hold time	t_{DBH}	20	—	—	ns	
Write data WAIT hold time	t_{DBWH}	0	—	—	ns	

Note: The CLK timing is the same in this mode and DMA mode. See table 10.9.

Table 10.7 CPU Mode 0 Master Mode Bus Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –20 to +75°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Clock cycle time	t_{CLCL}	100	—	2000	ns	Figure 10.5, figure 10.6
Clock high-level pulse width	t_{CHCL}	40	—	—	ns	
Clock low-level pulse width	t_{CLCH}	40	—	—	ns	
Clock fall time	t_{CL2CL1}	—	—	10	ns	
Clock rise time	t_{CH1CH2}	—	—	10	ns	
Address delay time	t_{CLAV}	—	—	55	ns	
Address set-up time	t_{AVAL}	20	—	—	ns	
$\overline{AS}$ active delay time	t_{CHLL}	—	—	50	ns	
$\overline{RD}$ active delay time	t_{CLRL}	—	—	50	ns	
Address hold time	t_{LLAX}	10	—	—	ns	
$\overline{AS}$ inactive delay time	t_{CLLH}	—	—	50	ns	
$\overline{RD}$ inactive delay time	t_{CLRH}	—	—	50	ns	
Data read set-up time	t_{DVCL}	25	—	—	ns	
Data read hold time	t_{RDX}	0	—	—	ns	
WAIT set-up time	t_{RYLCL}	30	—	—	ns	
WAIT inactive set-up time	t_{RYHCH}	30	—	—	ns	
WAIT hold time	t_{CHRYX}	30	—	—	ns	
Write data floating delay time	t_{CHDX}	—	—	60	ns	
$\overline{WR}$ active delay time	t_{CVCTV}	—	—	50	ns	
Write data delay time	t_{CLDV}	—	—	60	ns	
Write data set-up time	t_{DVWL}	15	—	—	ns	
$\overline{WR}$ inactive delay time	t_{CVCTX}	—	—	55	ns	
$\overline{WR}$ pulse width	t_{WLWH}	110	—	—	ns	
Write data hold time	t_{WHDX}	10	—	—	ns	
$\overline{AS}$ high-level pulse width	t_{ASWH}	70	—	—	ns	
$\overline{AS}$ low-level pulse width	t_{ASWL}	80	—	—	ns	

Table 10.8 CPU Mode 1 Master Mode Bus Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –20 to +75°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Clock cycle time	t_{CYC}	100	—	2000	ns	Figure 10.7
Clock high-level pulse width	t_{CHW}	40	—	—	ns	
Clock low-level pulse width	t_{CLW}	40	—	—	ns	
Clock fall time	t_{cf}	—	—	10	ns	
Clock rise time	t_{cr}	—	—	10	ns	
Address delay time	t_{AD}	—	—	55	ns	
Address set-up time	t_{AS}	20	—	—	ns	
$\overline{AS}$ delay time 1	t_{ASD1}	—	—	50	ns	
$\overline{RD}$ delay time 1	t_{RDD1}	—	—	50	ns	
Address hold time	t_{AH}	10	—	—	ns	
$\overline{AS}$ delay time 2	t_{ASD2}	—	—	50	ns	
$\overline{RD}$ delay time 2	t_{RDD2}	—	—	50	ns	
Data read set-up time	t_{DRS}	25	—	—	ns	
Data read hold time	t_{DRH}	5	—	—	ns	
WAIT set-up time	t_{WS}	30	—	—	ns	
WAIT hold time	t_{WH}	30	—	—	ns	
Write data floating delay time	t_{WDZ}	—	—	60	ns	
$\overline{WR}$ delay time 1	t_{WRD1}	—	—	50	ns	
Write data delay time	t_{WDD}	—	—	60	ns	
Write data set-up time	t_{WDS}	15	—	—	ns	
$\overline{WR}$ delay time 2	t_{WRD2}	—	—	55	ns	
$\overline{WR}$ pulse width	t_{WRP}	110	—	—	ns	
Write data hold time	t_{WDH}	10	—	—	ns	
$\overline{AS}$ high-level pulse width	t_{ASWH}	70	—	—	ns	
$\overline{AS}$ low-level pulse width	t_{ASWL}	80	—	—	ns	

Table 10.9 CPU Mode 2, 3 Master Mode Bus Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –20 to +75℃ unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Clock cycle time	t_{CYC}	100	—	2000	ns	Figure 10.8, figure 10.9
Clock high-level pulse width	t_{CH}	40	—	—	ns	
Clock low-level pulse width	t_{CL}	40	—	—	ns	
Clock fall time	t_{cf}	—	—	10	ns	
Clock rise time	t_{cr}	—	—	10	ns	
Address delay time 1	t_{AD1}	—	—	60	ns	
Set-up time from $\overline{AS}$	t_{ASS}	15	—	—	ns	
$\overline{AS}$ delay time	t_{ASD}	—	—	50	ns	
$\overline{HDS}$, $\overline{LDS}$ delay time 1	t_{DSD1}	—	—	50	ns	
Hold time from $\overline{AS}$ 1	t_{ASH1}	10	—	—	ns	
Hold time from $\overline{AS}$ 2	t_{ASH2}	10	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ delay time 3	t_{DSD3}	—	—	55	ns	
Read data set-up time	t_{RDS}	25	—	—	ns	
Read data hold time	t_{RDH}	20	—	—	ns	
WAIT set-up time	t_{WTS}	30	—	—	ns	
WAIT hold time	t_{WTH}	30	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ delay time 2	t_{DSD2}	—	—	50	ns	
$\overline{HDS}$, $\overline{LDS}$ low-level pulse width	t_{DSW}	110	—	—	ns	
Write data delay time	t_{WDD}	—	—	60	ns	
Write data set-up time	t_{WDS}	15	—	—	ns	
Write data hold time	t_{WDH}	10	—	—	ns	
Write data floating delay time	t_{WDZ}	—	—	60	ns	
$\overline{AS}$ high-level pulse width	t_{ASW1}	70	—	—	ns	
Read data strobe hold time	t_{RDHX}	0	—	—	ns	

Table 10.10 Interrupt Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –20 to +75°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
$\overline{\text{INT}}$ delay time	t_{IRD}	—	—	50	ns	Figure 10.10, figure 10.11
$\overline{\text{INTA}}$ active set-up time	t_{IAS1}	30	—	—	ns	
$\overline{\text{INTA}}$ inactive set-up time	t_{IAS2}	30	—	—	ns	
WAIT inactive delay time	t_{IWD1}	—	—	50	ns	
WAIT active delay time	t_{IWD2}	—	—	50	ns	
Vector data delay time	t_{IDBD1}	—	—	65	ns	
Vector data hold time	t_{IDBD2}	10	—	—	ns	
Vector data floating delay time	t_{IDBZ}	—	—	60	ns	

Table 10.11 Bus Arbitration Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –20 to +75°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
HOLD delay time	t_{HLDD}	—	—	55	ns	Figure 10.12
HOLDA set-up time	t_{HLAS}	30	—	—	ns	
$\overline{\text{BEO}}$ delay time	t_{BEOD}	—	—	50	ns	Figure 10.12, figure 10.13
$\overline{\text{BUSY}}$ delay time	t_{BSYD}	—	—	60	ns	
$\overline{\text{BUSY}}$ set-up time	t_{BSYS}	30	—	—	ns	
$\overline{\text{BUSREQ}}$ delay time	t_{BRQD}	—	—	50	ns	Figure 10.13
$\overline{\text{BUSACK}}$ set-up time	t_{BAKS}	30	—	—	ns	

Table 10.12 MSCI Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –20 to +75°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
TXC cycle time (TXC input)	t_{TCYC}	1.4*[1]	—	— *[3]	t_{CYC}	Figure 10.14 to figure 10.22
TXC rise time (TXC input)	t_{TCr}	—	—	10	ns	
TXC fall time (TXC input)	t_{TCf}	—	—	10	ns	
TXC high-level pulse width (TXC input)	t_{TCHW}	0.55	—	—	t_{CYC}	
TXC low-level pulse width (TXC input)	t_{TCLW}	0.55	—	—	t_{CYC}	
TXD delay time (TXC input)	t_{TDD1}	—	—	95	ns	
TXD delay time (TXC output)	t_{TDD2}	—	—	50	ns	
RXC cycle time	t_{RCYC}	1.4*[1]	—	— *[3]	t_{CYC}	
RXC rise time	t_{RCr}	—	—	10	ns	
RXC fall time	t_{RCf}	—	—	10	ns	
RXC high-level pulse width	t_{RCHW}	0.55	—	—	t_{CYC}	
RXC low-level pulse width	t_{RCLW}	0.55	—	—	t_{CYC}	
RXD–RXC set-up time (RXC input)	t_{RDS1}	30	—	—	ns	
RXC–RXD hold time (RXC input)	t_{RDH1}	20	—	—	ns	
RXD–RXC set-up time (RXC output)	t_{RDS2}	80	—	—	ns	
RXC–RXD hold time (RXC output)	t_{RDH2}	20	—	—	ns	
ADPLL operating clock cycle time	t_{PLCY}	57	—	—	ns	
ADPLL operating clock rise time	t_{PLr}	—	—	8	ns	
ADPLL operating clock fall time	t_{PLf}	—	—	8	ns	

Table 10.12 MSCI Timing (cont)

Item	Symbol	Min	Typ	Max	Unit	Timing
ADPLL operating clock high-level pulse width	t_{PLHW}	10	—	—	ns	Figure 10.14, figure 10.22
ADPLL operating clock low-level pulse width	t_{PLLW}	10	—	—	ns	
CLK–BRG output delay time *[2]	t_{BGD}	—	—	95	ns	
TXC/RXC output rise time	t_{BGr}	—	—	30	ns	
TXC/RXC output fall time	t_{BGf}	—	—	30	ns	
RXC–SYNC set-up time	t_{SYSU}	2.5	—	—	t_{CYC}	
RXC–SYNC hold time	t_{SYHD}	2.5	—	—	t_{CYC}	
$\overline{CTS}$ high-level pulse width	t_{CTSHW}	2.0	—	—	t_{CYC}	
$\overline{CTS}$ low-level pulse width	t_{CTSLW}	2.0	—	—	t_{CYC}	
$\overline{DCD}$ high-level pulse width	t_{DCDHW}	2.0	—	—	t_{CYC}	
$\overline{DCD}$ low-level pulse width	t_{DCDLW}	2.0	—	—	t_{CYC}	
CLK–$\overline{RTS}$ delay time	t_{RTSD}	—	—	70	ns	

Notes: 1. In asynchronous mode and loop mode t_{TCYC} and t_{RCYC} = 2.5 t_{CYC} (min).
2. $f_{BRG} \neq f_{CLK}$ (f_{BRG} is the baud rate generator output frequency; f_{CLK} is the system clock (CLK) frequency.)
3. Maximum cycle time corresponds to 50 bits/s.

Table 10.13 Rise and Fall Times of Input Signals with No Characteristics Specified

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –20 to +75°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Input signal rise time	t_{Ir}	—	—	100	ns	Figure 10.23
Input signal fall time	t_{If}	—	—	100	ns	

10.2 Electrical Characteristics of HD64570CP16 and HD64570F16

10.2.1 Absolute Maximum Ratings

Table 10.14 Absolute Maximum Ratings

Item	Symbol	Rating	Unit
Supply voltage	V_{CC}	–0.3 to +7.0	V
Input voltage	V_{in}	–0.3 to V_{CC} +0.3	V
Operating temperature	T_{opr}	0 to +70	°C
Storage temperature	T_{stg}	–55 to +150	°C

Caution: The HD64570 may suffer permanent damage if it is subjected to conditions exceeding absolute maximum ratings. To assure normal operation, the following conditions should be satisfied: $V_{SS} \leq V_{in} \leq V_{CC}$

10.2.2 DC Characteristics

Table 10.15 DC Characteristics

(V_{CC} = 5 V ± 5%, V_{SS} = 0 V, Ta = 0 to +70°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Conditions
Input high level voltage for $\overline{\text{RESET}}$ and CLK	V_{IH1}	V_{CC} – 0.6	—	V_{CC} + 0.3	V	
Input high level voltage for pins other than $\overline{\text{RESET}}$ and CLK	V_{IH2}	2.0	—	V_{CC} + 0.3	V	
Input low level voltage for $\overline{\text{RESET}}$ and CLK	V_{IL1}	–0.3	—	0.6	V	
Input low level voltage for pins other than $\overline{\text{RESET}}$ and CLK	V_{IL2}	–0.3	—	0.8	V	
Output high level voltage for all output pins	V_{OH}	2.4	—	—	V	I_{OH} = –200 µA
		V_{CC} – 1.2	—	—		I_{OH} = –20 µA
Output low level voltage for all output pins	V_{OL}	—	—	0.45	V	I_{OL} = 2.2 mA
Input leakage current	I_{IL}	—	—	1.0	µA	V_{in} = 0.5 to V_{CC} – 0.5
Three-state leakage current	I_{TL}	—	—	1.0	µA	V_{in} = 0.5 to V_{CC} – 0.5
Current consumption* (normal operation)	I_{CC}	—	80	150	mA	f = 16.7 MHz
Current consumption* (system stop mode)		—	4	10	mA	f = 16.7 MHz
Pin capacitance	C_p	—	—	20	pF	V_{in} = 0 V, f = 1 MHz, Ta = 25°C

Note: V_{IH} min = V_{CC} – 1.0 V, V_{IL} max = 0.8 V (when no output pins are loaded)

10.2.3 AC Characteristics

Table 10.16 CPU Mode 0 Slave Mode Bus Timing

(V_{CC} = 5 V ± 5%, V_{SS} = 0 V, Ta = 0 to +70°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
$\overline{CS}$ set-up time	t_{CSS}	15	—	—	ns	Figure 10.1
$\overline{CS}$ hold time 1	t_{CSH1}	20	—	—	ns	
$\overline{CS}$ hold time 2	t_{CSH2}	0	—	—	ns	
Address set-up time	t_{ADS}	15	—	—	ns	
Address hold time	t_{ADH}	0	—	—	ns	
$\overline{RD}$ active set-up time	t_{RDS1}	15	—	—	ns	
$\overline{RD}$ inactive set-up time	t_{RDS2}	10	—	—	ns	
$\overline{RD}$ inactive hold time	t_{RDH1}	10	—	—	ns	
$\overline{RD}$ active hold time	t_{RDH2}	0	—	—	ns	
$\overline{WR}$ active set-up time	t_{WRS1}	15	—	—	ns	
$\overline{WR}$ inactive set-up time	t_{WRS2}	10	—	—	ns	
$\overline{WR}$ inactive hold time	t_{WRH1}	10	—	—	ns	
$\overline{WR}$ active hold time	t_{WRH2}	0	—	—	ns	
WAIT active delay time	t_{WTD1}	—	—	50	ns	
WAIT inactive delay time	t_{WTD2}	—	—	50	ns	
Read data active delay time	t_{DBD1}	—	—	60	ns	
Read data hold time	t_{DBD2}	10	—	—	ns	
Read data floating delay time	t_{DBZ}	—	—	60	ns	
Write data set-up time	t_{DBS}	20	—	—	ns	
Write data hold time	t_{DBH}	20	—	—	ns	

Note: The CLK timing is the same in this mode and DMA mode.

Table 10.17 CPU Mode 1 Slave Mode Bus Timing

(V_{CC} = 5 V ± 5%, V_{SS} = 0 V, Ta = 0 to +70°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Address set-up time	t_{ADS}	15	—	—	ns	Figure 10.2
Address hold time	t_{ADH}	0	—	—	ns	
$\overline{CS}$ set-up time	t_{CSS}	15	—	—	ns	
$\overline{CS}$ hold time	t_{CSH}	0	—	—	ns	
$\overline{RD}$ active set-up time	t_{RDS1}	15	—	—	ns	
$\overline{RD}$ inactive set-up time	t_{RDS2}	10	—	—	ns	
$\overline{RD}$ inactive hold time	t_{RDH1}	10	—	—	ns	
$\overline{RD}$ active hold time	t_{RDH2}	0	—	—	ns	
$\overline{WR}$ active set-up time	t_{WRS1}	15	—	—	ns	
$\overline{WR}$ inactive set-up time	t_{WRS2}	10	—	—	ns	
$\overline{WR}$ inactive hold time	t_{WRH1}	10	—	—	ns	
$\overline{WR}$ active hold time	t_{WRH2}	0	—	—	ns	
WAIT active delay time	t_{WTD1}	—	—	50	ns	
WAIT inactive delay time	t_{WTD2}	—	—	55	ns	
Read data active delay time	t_{DBD1}	—	—	60	ns	
Read data hold time	t_{DBD2}	6	—	—	ns	
Read data floating delay time	t_{DBZ}	—	—	60	ns	
Write data set-up time	t_{DBS}	15	—	—	ns	
Write data hold time	t_{DBH}	20	—	—	ns	

Note: The CLK timing is the same in this mode and DMA mode.

Table 10.18 CPU Mode 2 Slave Mode Bus Timing

(V_{CC} = 5 V ± 5%, V_{SS} = 0 V, Ta = 0 to +70°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Address set-up time	t_{ADS}	15	—	—	ns	Figure 10.3
Address hold time	t_{ADH}	0	—	—	ns	
$\overline{AS}$ set-up time	t_{ASS}	15	—	—	ns	
$\overline{AS}$ hold time 1	t_{ASH1}	0	—	—	ns	
$\overline{AS}$ hold time 2	t_{ASH2}	0	—	—	ns	
$\overline{CS}$ set-up time	t_{CSS}	15	—	—	ns	
$\overline{CS}$ hold time 1	t_{CSH1}	0	—	—	ns	
$\overline{CS}$ hold time 2	t_{CSH2}	0	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ active set-up time	t_{DSS1}	15	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ inactive set-up time	t_{DSS2}	10	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ inactive hold time	t_{DSH1}	10	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ active hold time	t_{DSH2}	0	—	—	ns	
R/$\overline{W}$ set-up time	t_{RWS}	15	—	—	ns	
R/$\overline{W}$ hold time 1	t_{RWH1}	0	—	—	ns	
R/$\overline{W}$ hold time 2	t_{RWH2}	0	—	—	ns	
WAIT inactive delay time	t_{WTD1}	—	—	50	ns	
WAIT active delay time	t_{WTD2}	—	—	55	ns	
Read data active delay time	t_{DBD1}	—	—	60	ns	
Read data hold time	t_{DBD2}	10	—	—	ns	
Read data floating delay time	t_{DBZ}	—	—	60	ns	
Write data set-up time	t_{DBS}	15	—	—	ns	
Write data hold time	t_{DBH}	20	—	—	ns	
Write data WAIT hold time	t_{DBWH}	0	—	—	ns	

Note: The CLK timing is the same in this mode and DMA mode.

Table 10.19 CPU Mode 3 Slave Mode Bus Timing

(V_{CC} = 5 V ± 5%, V_{SS} = 0 V, Ta = 0 to +70°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Address set-up time	t_{ADS}	15	—	—	ns	Figure 10.4
Address hold time	t_{ADH}	0	—	—	ns	
$\overline{AS}$ set-up time	t_{ASS}	15	—	—	ns	
$\overline{AS}$ hold time	t_{ASH}	0	—	—	ns	
$\overline{CS}$ set-up time	t_{CSS}	15	—	—	ns	
$\overline{CS}$ hold time	t_{CSH}	0	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ active set-up time	t_{DSS1}	15	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ inactive set-up time	t_{DSS2}	10	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ inactive hold time	t_{DSH1}	10	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ active hold time	t_{DSH2}	0	—	—	ns	
R/$\overline{W}$ set-up time	t_{RWS}	15	—	—	ns	
R/$\overline{W}$ hold time 1	t_{RWH1}	0	—	—	ns	
R/$\overline{W}$ hold time 2	t_{RWH2}	0	—	—	ns	
WAIT inactive delay time	t_{WTD1}	—	—	50	ns	
WAIT active delay time	t_{WTD2}	—	—	50	ns	
Read data active delay time	t_{DBD1}	—	—	60	ns	
Read data hold time	t_{DBD2}	10	—	—	ns	
Read data floating delay time	t_{DBZ}	—	—	60	ns	
Write data set-up time	t_{DBS}	15	—	—	ns	
Write data hold time	t_{DBH}	20	—	—	ns	
Write data WAIT hold time	t_{DBWH}	0	—	—	ns	

Note: The CLK timing is the same in this mode and DMA mode.

Table 10.20 CPU Mode 0 Master Mode Bus Timing

(V_{CC} = 5 V ± 5%, V_{SS} = 0 V, Ta = 0 to +70°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Clock cycle time	t_{CLCL}	60	—	500	ns	Figure 10.5, figure 10.6
Clock high-level pulse width	t_{CHCL}	25	—	—	ns	
Clock low-level pulse width	t_{CLCH}	25	—	—	ns	
Clock fall time	t_{CL2CL1}	—	—	5	ns	
Clock rise time	t_{CH1CH2}	—	—	5	ns	
Address delay time	t_{CLAV}	—	—	35	ns	
Address set-up time	t_{AVAL}	10	—	—	ns	
$\overline{AS}$ active delay time	t_{CHLL}	—	—	40	ns	
$\overline{RD}$ active delay time	t_{CLRL}	—	—	40	ns	
Address hold time	t_{LLAX}	10	—	—	ns	
$\overline{AS}$ inactive delay time	t_{CLLH}	—	—	40	ns	
$\overline{RD}$ inactive delay time	t_{CLRH}	—	—	40	ns	
Data read set-up time	t_{DVCL}	20	—	—	ns	
Data read hold time	t_{RDX}	0	—	—	ns	
WAIT set-up time	t_{RYLCL}	15	—	—	ns	
WAIT inactive set-up time	t_{RYHCH}	15	—	—	ns	
WAIT hold time	t_{CHRYX}	20	—	—	ns	
Write data floating delay time	t_{CHDX}	—	—	40	ns	
$\overline{WR}$ active delay time	t_{CVCTV}	—	—	40	ns	
Write data delay time	t_{CLDV}	—	—	60	ns	
Write data set-up time	t_{DVWL}	0	—	—	ns	
$\overline{WR}$ inactive delay time	t_{CVCTX}	—	—	45	ns	
$\overline{WR}$ pulse width	t_{WLWH}	40	—	—	ns	
Write data hold time	t_{WHDX}	10	—	—	ns	
$\overline{AS}$ high-level pulse width	t_{ASWH}	30	—	—	ns	
$\overline{AS}$ low-level pulse width	t_{ASWL}	50	—	—	ns	

Table 10.21 CPU Mode 1 Master Mode Bus Timing

(V_{CC} = 5 V ± 5%, V_{SS} = 0 V, Ta = 0 to +70°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Clock cycle time	t_{CYC}	60	—	500	ns	Figure 10.7
Clock high-level pulse width	t_{CHW}	25	—	—	ns	
Clock low-level pulse width	t_{CLW}	25	—	—	ns	
Clock fall time	t_{cf}	—	—	5	ns	
Clock rise time	t_{cr}	—	—	5	ns	
Address delay time	t_{AD}	—	—	45	ns	
Address set-up time	t_{AS}	5	—	—	ns	
$\overline{AS}$ delay time 1	t_{ASD1}	—	—	35	ns	
$\overline{RD}$ delay time 1	t_{RDD1}	—	—	35	ns	
Address hold time	t_{AH}	10	—	—	ns	
$\overline{AS}$ delay time 2	t_{ASD2}	—	—	35	ns	
$\overline{RD}$ delay time 2	t_{RDD2}	—	—	35	ns	
Data read set-up time	t_{DRS}	15	—	—	ns	
Data read hold time	t_{DRH}	5	—	—	ns	
WAIT set-up time	t_{WS}	15	—	—	ns	
WAIT hold time	t_{WH}	20	—	—	ns	
Write data floating delay time	t_{WDZ}	—	—	40	ns	
$\overline{WR}$ delay time 1	t_{WRD1}	—	—	45	ns	
Write data delay time	t_{WDD}	—	—	60	ns	
Write data set-up time	t_{WDS}	0	—	—	ns	
$\overline{WR}$ delay time 2	t_{WRD2}	—	—	35	ns	
$\overline{WR}$ pulse width	t_{WRP}	40	—	—	ns	
Write data hold time	t_{WDH}	10	—	—	ns	
$\overline{AS}$ high-level pulse width	t_{ASWH}	30	—	—	ns	
$\overline{AS}$ low-level pulse width	t_{ASWL}	50	—	—	ns	

Table 10.22 CPU Mode 2, 3 Master Mode Bus Timing

(V_{CC} = 5 V ± 5%, V_{SS} = 0 V, Ta = 0 to +70°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Clock cycle time	t_{CYC}	60	—	500	ns	Figure 10.8, figure 10.9
Clock high-level pulse width	t_{CH}	25	—	—	ns	
Clock low-level pulse width	t_{CL}	25	—	—	ns	
Clock fall time	t_{cf}	—	—	5	ns	
Clock rise time	t_{cr}	—	—	5	ns	
Address delay time 1	t_{AD1}	—	—	55	ns	
Set-up time from $\overline{AS}$	t_{ASS}	0	—	—	ns	
$\overline{AS}$ delay time	t_{ASD}	—	—	35	ns	
$\overline{HDS}$, $\overline{LDS}$ delay time 1	t_{DSD1}	—	—	45	ns	
Hold time from $\overline{AS}$ 1	t_{ASH1}	10	—	—	ns	
Hold time from $\overline{AS}$ 2	t_{ASH2}	10	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ delay time 3	t_{DSD3}	—	—	40	ns	
Read data set-up time	t_{RDS}	15	—	—	ns	
Read data hold time	t_{RDH}	0	—	—	ns	
WAIT set-up time	t_{WTS}	15	—	—	ns	
WAIT hold time	t_{WTH}	20	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ delay time 2	t_{DSD2}	—	—	45	ns	
$\overline{HDS}$, $\overline{LDS}$ low-level pulse width	t_{DSW}	40	—	—	ns	
Write data delay time	t_{WDD}	—	—	60	ns	
Write data set-up time	t_{WDS}	5	—	—	ns	
Write data hold time	t_{WDH}	10	—	—	ns	
Write data floating delay time	t_{WDZ}	—	—	60	ns	
$\overline{AS}$ high-level pulse width	t_{ASW1}	30	—	—	ns	
Read data strobe hold time	t_{RDHX}	0	—	—	ns	

Table 10.23 Interrupt Timing

(V_{CC} = 5 V ± 5%, V_{SS} = 0 V, Ta = 0 to +70°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
$\overline{\text{INT}}$ delay time	t_{IRD}	—	—	35	ns	Figure 10.10, figure 10.11
$\overline{\text{INTA}}$ active set-up time	t_{IAS1}	15	—	—	ns	
$\overline{\text{INTA}}$ inactive set-up time	t_{IAS2}	15	—	—	ns	
WAIT inactive delay time	t_{IWD1}	—	—	45	ns	
WAIT active delay time	t_{IWD2}	—	—	50	ns	
Vector data delay time	t_{IDBD1}	—	—	60	ns	
Vector data hold time	t_{IDBD2}	10	—	—	ns	
Vector data floating delay time	t_{IDBZ}	—	—	60	ns	

Table 10.24 Bus Arbitration Timing

(V_{CC} = 5 V ± 5%, V_{SS} = 0 V, Ta = 0 to +70°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
HOLD delay time	t_{HLDD}	—	—	55	ns	Figure 10.12
HOLDA set-up time	t_{HLAS}	15	—	—	ns	
$\overline{\text{BEO}}$ delay time	t_{BEOD}	—	—	50	ns	Figure 10.12, figure 10.13
$\overline{\text{BUSY}}$ delay time	t_{BSYD}	—	—	60	ns	
$\overline{\text{BUSY}}$ set-up time	t_{BSYS}	15	—	—	ns	
$\overline{\text{BUSREQ}}$ delay time	t_{BRQD}	—	—	50	ns	Figure 10.13
$\overline{\text{BUSACK}}$ set-up time	t_{BAKS}	15	—	—	ns	

Table 10.25 MSCI Timing

(V_{CC} = 5 V ± 5%, V_{SS} = 0 V, Ta = 0 to +70°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
TXC cycle time (TXC input)	t_{TCYC}	1.4*[1]	—	—	t_{CYC}	Figure 10.14 to figure 10.22
TXC rise time (TXC input)	t_{TCr}	—	—	10	ns	
TXC fall time (TXC input)	t_{TCf}	—	—	10	ns	
TXC high-level pulse width (TXC input)	t_{TCHW}	0.55	—	—	t_{CYC}	
TXC low-level pulse width (TXC input)	t_{TCLW}	0.55	—	—	t_{CYC}	
TXD delay time (TXC input)	t_{TDD1}	30	—	90	ns	
TXD delay time (TXC output)	t_{TDD2}	—	—	45	ns	
RXC cycle time	t_{RCYC}	1.4*[1]	—	—	t_{CYC}	
RXC rise time	t_{RCr}	—	—	10	ns	
RXC fall time	t_{RCf}	—	—	10	ns	
RXC high-level pulse width	t_{RCHW}	0.55	—	—	t_{CYC}	
RXC low-level pulse width	t_{RCLW}	0.55	—	—	t_{CYC}	
RXD–RXC set-up time (RXC input)	t_{RDS1}	15	—	—	ns	
RXC–RXD hold time (RXC input)	t_{RDH1}	10	—	—	ns	
RXD–RXC set-up time (RXC output)	t_{RDS2}	35	—	—	ns	
RXC–RXD hold time (RXC output)	t_{RDH2}	10	—	—	ns	
ADPLL operating clock cycle time	t_{PLCY}	57	—	—	ns	
ADPLL operating clock rise time	t_{PLr}	—	—	8	ns	
ADPLL operating clock fall time	t_{PLf}	—	—	8	ns	

Table 10.25 MSCI Timing (cont)

Item	Symbol	Min	Typ	Max	Unit	Timing
ADPLL operating clock high-level pulse width	t_{PLHW}	10	—	—	ns	Figure 10.14 to figure 10.22
ADPLL operating clock low-level pulse width	t_{PLLW}	10	—	—	ns	
CLK–BRG output delay time*[2]	t_{BGD}	—	—	90	ns	
TXC/RXC output rise time	t_{BGr}	—	—	30	ns	
TXC/RXC output fall time	t_{BGf}	—	—	30	ns	
RXC–SYNC set-up time	t_{SYSU}	2.5	—	—	t_{CYC}	
RXC–SYNC hold time	t_{SYHD}	2.5	—	—	t_{CYC}	
$\overline{CTS}$ high-level pulse width	t_{CTSHW}	2.0	—	—	t_{CYC}	
$\overline{CTS}$ low-level pulse width	t_{CTSLW}	2.0	—	—	t_{CYC}	
$\overline{DCD}$ high-level pulse width	t_{DCDHW}	2.0	—	—	t_{CYC}	
$\overline{DCD}$ low-level pulse width	t_{DCDLW}	2.0	—	—	t_{CYC}	
CLK–$\overline{RTS}$ delay time	t_{RTSD}	—	—	70	ns	

Notes: 1. In asynchronous mode and loop mode, t_{TCYC} and t_{RCYC} = 2.5 t_{CYC} (min).

2. $f_{BRG} \neq f_{CLK}$ (f_{BRG} is the baud rate generator output frequency; f_{CLK} is the system clock (CLK) frequency.)

Table 10.26 Rise and Fall Times of Input Signals with No Characteristics Specified

(V_{CC} = 5 V ± 5%, V_{SS} = 0 V, Ta = 0 to +70°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Input signal rise time	t_{Ir}	—	—	50	ns	Figure 10.23
Input signal fall time	t_{If}	—	—	50	ns	

10.3 Electrical Characteristics of HD64570CP8I and HD64570F8I

10.3.1 Absolute Maximum Ratings

Table 10.27 Absolute Maximum Ratings

Item	Symbol	Rating	Unit
Supply voltage	V_{CC}	–0.3 to +7.0	V
Input voltage	V_{in}	–0.3 to V_{CC} +0.3	V
Operating temperature	T_{opr}	–40 to +85	°C
Storage temperature	T_{stg}	–55 to +150	°C

Caution: Permanent damage to the HD64570 may result if it is subjected to conditions that exceed the absolute maximum ratings. To assure normal operation, the following conditions should be satisfied:

$$V_{SS} \leq V_{in} \leq V_{CC}$$

10.3.2 DC Characteristics

Table 10.28 DC Characteristics

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –40 to +85°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Conditions
Input high level voltage for $\overline{\text{RESET}}$ and CLK	V_{IH1}	V_{CC} – 0.6	—	V_{CC} + 0.3	V	
Input high level voltage for pins other than $\overline{\text{RESET}}$ and CLK	V_{IH2}	2.0	—	V_{CC} + 0.3	V	
Input low level voltage for $\overline{\text{RESET}}$ and CLK	V_{IL1}	–0.3	—	0.6	V	
Input low level voltage for pins other than $\overline{\text{RESET}}$ and CLK	V_{IL2}	–0.3	—	0.8	V	
Output high level voltage for all output pins	V_{OH}	2.4	—	—	V	I_{OH} = – 200 µA
		V_{CC} – 1.2	—	—		I_{OH} = – 20 µA
Output low level voltage for all output pins	V_{OL}	—	—	0.45	V	I_{OL} = 2.2 mA
Input leakage current	I_{IL}	—	—	1.0	µA	V_{in} = 0.5 to V_{CC} – 0.5
Three-state leakage current	I_{TL}	—	—	1.0	µA	V_{in} = 0.5 to V_{CC} – 0.5
Current consumption(Note) (normal operation)	I_{CC}	—	50	80	mA	f = 8 MHz
Current consumption(Note) (system stop mode)		—	2	5	mA	f = 8 MHz
Pin capacitance	C_p	—	—	20	pF	V_{in} = 0V, f = 1 MHz, Ta = 25°C

Note: V_{IH} min = V_{CC} – 1.0 V, V_{IL} max = 0.8 V (when no output pins are loaded)

10.3.3 AC Characteristics

Table 10.29 CPU Mode 0 Slave Mode Bus Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –40 to +85°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
$\overline{CS}$ set-up time	t_{CSS}	30	—	—	ns	Figure 10.1
$\overline{CS}$ hold time 1	t_{CSH1}	20	—	—	ns	
$\overline{CS}$ hold time 2	t_{CSH2}	0	—	—	ns	
Address set-up time	t_{ADS}	30	—	—	ns	
Address hold time	t_{ADH}	0	—	—	ns	
$\overline{RD}$ active set-up time	t_{RDS1}	30	—	—	ns	
$\overline{RD}$ inactive set-up time	t_{RDS2}	30	—	—	ns	
$\overline{RD}$ inactive hold time	t_{RDH1}	10	—	—	ns	
$\overline{RD}$ active hold time	t_{RDH2}	0	—	—	ns	
$\overline{WR}$ active set-up time	t_{WRS1}	30	—	—	ns	
$\overline{WR}$ inactive set-up time	t_{WRS2}	30	—	—	ns	
$\overline{WR}$ inactive hold time	t_{WRH1}	10	—	—	ns	
$\overline{WR}$ active hold time	t_{WRH2}	0	—	—	ns	
WAIT active delay time	t_{WTD1}	—	—	50	ns	
WAIT inactive delay time	t_{WTD2}	—	—	50	ns	
Read data active delay time	t_{DBD1}	—	—	65	ns	
Read data hold time	t_{DBD2}	10	—	—	ns	
Read data floating delay time	t_{DBZ}	—	—	60	ns	
Write data set-up time	t_{DBS}	25	—	—	ns	
Write data hold time	t_{DBH}	20	—	—	ns	

Notes: 1. The CLK timing is the same in this mode and DMA mode. See table 10.7.
2. For the measurement conditions of AC characteristics, see figure 9.25.

Table 10.30 CPU Mode 1 Slave Mode Bus Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –40 to +85°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Address set-up time	t_{ADS}	30	—	—	ns	Figure 10.2
Address hold time	t_{ADH}	0	—	—	ns	
$\overline{CS}$ set-up time	t_{CSS}	30	—	—	ns	
$\overline{CS}$ hold time	t_{CSH}	0	—	—	ns	
$\overline{RD}$ active set-up time	t_{RDS1}	30	—	—	ns	
$\overline{RD}$ inactive set-up time	t_{RDS2}	30	—	—	ns	
$\overline{RD}$ inactive hold time	t_{RDH1}	10	—	—	ns	
$\overline{RD}$ active hold time	t_{RDH2}	0	—	—	ns	
$\overline{WR}$ active set-up time	t_{WRS1}	30	—	—	ns	
$\overline{WR}$ inactive set-up time	t_{WRS2}	30	—	—	ns	
$\overline{WR}$ inactive hold time	t_{WRH1}	10	—	—	ns	
$\overline{WR}$ active hold time	t_{WRH2}	0	—	—	ns	
WAIT active delay time	t_{WTD1}	—	—	50	ns	
WAIT inactive delay time	t_{WTD2}	—	—	60	ns	
Read data active delay time	t_{DBD1}	—	—	60	ns	
Read data hold time	t_{DBD2}	6	—	—	ns	
Read data floating delay time	t_{DBZ}	—	—	60	ns	
Write data set-up time	t_{DBS}	25	—	—	ns	
Write data hold time	t_{DBH}	20	—	—	ns	

Note: The CLK timing is the same in this mode and DMA mode. See table 10.8.

Table 10.31 CPU Mode 2 Slave Mode Bus Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –40 to +85°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Address set-up time	t_{ADS}	30	—	—	ns	Figure 10.3
Address hold time	t_{ADH}	0	—	—	ns	
$\overline{AS}$ set-up time	t_{ASS}	30	—	—	ns	
$\overline{AS}$ hold time 1	t_{ASH1}	0	—	—	ns	
$\overline{AS}$ hold time 2	t_{ASH2}	0	—	—	ns	
$\overline{CS}$ set-up time	t_{CSS}	30	—	—	ns	
$\overline{CS}$ hold time 1	t_{CSH1}	0	—	—	ns	
$\overline{CS}$ hold time 2	t_{CSH2}	0	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ active set-up time	t_{DSS1}	30	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ inactive set-up time	t_{DSS2}	30	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ inactive hold time	t_{DSH1}	10	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ active hold time	t_{DSH2}	0	—	—	ns	
R/$\overline{W}$ set-up time	t_{RWS}	30	—	—	ns	
R/$\overline{W}$ hold time 1	t_{RWH1}	0	—	—	ns	
R/$\overline{W}$ hold time 2	t_{RWH2}	0	—	—	ns	
WAIT inactive delay time	t_{WTD1}	—	—	50	ns	
WAIT active delay time	t_{WTD2}	—	—	60	ns	
Read data active delay time	t_{DBD1}	—	—	60	ns	
Read data hold time	t_{DBD2}	10	—	—	ns	
Read data floating delay time	t_{DBZ}	—	—	60	ns	
Write data set-up time	t_{DBS}	25	—	—	ns	
Write data hold time	t_{DBH}	20	—	—	ns	
Write data WAIT hold time	t_{DBWH}	0	—	—	ns	

Note: The CLK timing is the same in this mode and DMA mode. See table 10.9.

Table 10.32 CPU Mode 3 Slave Mode Bus Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –40 to +85°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Address set-up time	t_{ADS}	30	—	—	ns	Figure 10.4
Address hold time	t_{ADH}	0	—	—	ns	
$\overline{AS}$ set-up time	t_{ASS}	30	—	—	ns	
$\overline{AS}$ hold time	t_{ASH}	0	—	—	ns	
$\overline{CS}$ set-up time	t_{CSS}	30	—	—	ns	
$\overline{CS}$ hold time	t_{CSH}	0	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ active set-up time	t_{DSS1}	30	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ inactive set-up time	t_{DSS2}	30	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ inactive hold time	t_{DSH1}	10	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ active hold time	t_{DSH2}	0	—	—	ns	
R/$\overline{W}$ set-up time	t_{RWS}	30	—	—	ns	
R/$\overline{W}$ hold time 1	t_{RWH1}	0	—	—	ns	
R/$\overline{W}$ hold time 2	t_{RWH2}	0	—	—	ns	
WAIT inactive delay time	t_{WTD1}	—	—	50	ns	
WAIT active delay time	t_{WTD2}	—	—	50	ns	
Read data active delay time	t_{DBD1}	—	—	60	ns	
Read data hold time	t_{DBD2}	10	—	—	ns	
Read data floating delay time	t_{DBZ}	—	—	60	ns	
Write data set-up time	t_{DBS}	25	—	—	ns	
Write data hold time	t_{DBH}	20	—	—	ns	
Write data WAIT hold time	t_{DBWH}	0	—	—	ns	

Note: The CLK timing is the same in this mode and DMA mode. See table 10.9.

Table 10.33 CPU Mode 0 Master Mode Bus Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –40 to +85°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Clock cycle time	t_{CLCL}	125	—	2000	ns	Figure 10.5, figure 10.6
Clock high-level pulse width	t_{CHCL}	50	—	—	ns	
Clock low-level pulse width	t_{CLCH}	50	—	—	ns	
Clock fall time	t_{CL2CL1}	—	—	10	ns	
Clock rise time	t_{CH1CH2}	—	—	10	ns	
Address delay time	t_{CLAV}	—	—	55	ns	
Address set-up time	t_{AVAL}	20	—	—	ns	
$\overline{AS}$ active delay time	t_{CHLL}	—	—	50	ns	
$\overline{RD}$ active delay time	t_{CLRL}	—	—	50	ns	
Address hold time	t_{LLAX}	10	—	—	ns	
$\overline{AS}$ inactive delay time	t_{CLLH}	—	—	50	ns	
$\overline{RD}$ inactive delay time	t_{CLRH}	—	—	50	ns	
Data read set-up time	t_{DVCL}	25	—	—	ns	
Data read hold time	t_{RDX}	0	—	—	ns	
WAIT set-up time	t_{RYLCL}	30	—	—	ns	
WAIT inactive set-up time	t_{RYHCH}	30	—	—	ns	
WAIT hold time	t_{CHRYX}	30	—	—	ns	
Write data floating delay time	t_{CHDX}	—	—	60	ns	
$\overline{WR}$ active delay time	t_{CVCTV}	—	—	50	ns	
Write data delay time	t_{CLDV}	—	—	60	ns	
Write data set-up time	t_{DVWL}	15	—	—	ns	
$\overline{WR}$ inactive delay time	t_{CVCTX}	—	—	55	ns	
$\overline{WR}$ pulse width	t_{WLWH}	110	—	—	ns	
Write data hold time	t_{WHDX}	10	—	—	ns	
$\overline{AS}$ high-level pulse width	t_{ASWH}	70	—	—	ns	
$\overline{AS}$ low-level pulse width	t_{ASWL}	80	—	—	ns	

Table 10.34 CPU Mode 1 Master Mode Bus Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –40 to +85°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Clock cycle time	t_{CYC}	125	—	2000	ns	Figure 10.7
Clock high-level pulse width	t_{CHW}	50	—	—	ns	
Clock low-level pulse width	t_{CLW}	50	—	—	ns	
Clock fall time	t_{cf}	—	—	10	ns	
Clock rise time	t_{cr}	—	—	10	ns	
Address delay time	t_{AD}	—	—	55	ns	
Address set-up time	t_{AS}	20	—	—	ns	
$\overline{AS}$ delay time 1	t_{ASD1}	—	—	50	ns	
$\overline{RD}$ delay time 1	t_{RDD1}	—	—	50	ns	
Address hold time	t_{AH}	10	—	—	ns	
$\overline{AS}$ delay time 2	t_{ASD2}	—	—	50	ns	
$\overline{RD}$ delay time 2	t_{RDD2}	—	—	50	ns	
Data read set-up time	t_{DRS}	25	—	—	ns	
Data read hold time	t_{DRH}	5	—	—	ns	
WAIT set-up time	t_{WS}	30	—	—	ns	
WAIT hold time	t_{WH}	30	—	—	ns	
Write data floating delay time	t_{WDZ}	—	—	60	ns	
$\overline{WR}$ delay time 1	t_{WRD1}	—	—	50	ns	
Write data delay time	t_{WDD}	—	—	60	ns	
Write data set-up time	t_{WDS}	15	—	—	ns	
$\overline{WR}$ delay time 2	t_{WRD2}	—	—	55	ns	
$\overline{WR}$ pulse width	t_{WRP}	110	—	—	ns	
Write data hold time	t_{WDH}	10	—	—	ns	
$\overline{AS}$ high-level pulse width	t_{ASWH}	70	—	—	ns	
$\overline{AS}$ low-level pulse width	t_{ASWL}	80	—	—	ns	

Table 10.35 CPU Mode 2, 3 Master Mode Bus Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –40 to +85°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Clock cycle time	t_{CYC}	125	—	2000	ns	Figure 10.8, figure 10.9
Clock high-level pulse width	t_{CH}	50	—	—	ns	
Clock low-level pulse width	t_{CL}	50	—	—	ns	
Clock fall time	t_{cf}	—	—	10	ns	
Clock rise time	t_{cr}	—	—	10	ns	
Address delay time 1	t_{AD1}	—	—	60	ns	
Set-up time from $\overline{AS}$	t_{ASS}	15	—	—	ns	
$\overline{AS}$ delay time	t_{ASD}	—	—	50	ns	
$\overline{HDS}$, $\overline{LDS}$ delay time 1	t_{DSD1}	—	—	50	ns	
Hold time from $\overline{AS}$ 1	t_{ASH1}	10	—	—	ns	
Hold time from $\overline{AS}$ 2	t_{ASH2}	10	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ delay time 3	t_{DSD3}	—	—	55	ns	
Read data set-up time	t_{RDS}	25	—	—	ns	
Read data hold time	t_{RDH}	20	—	—	ns	
WAIT set-up time	t_{WTS}	30	—	—	ns	
WAIT hold time	t_{WTH}	30	—	—	ns	
$\overline{HDS}$, $\overline{LDS}$ delay time 2	t_{DSD2}	—	—	50	ns	
$\overline{HDS}$, $\overline{LDS}$ low-level pulse width	t_{DSW}	110	—	—	ns	
Write data delay time	t_{WDD}	—	—	60	ns	
Write data set-up time	t_{WDS}	15	—	—	ns	
Write data hold time	t_{WDH}	10	—	—	ns	
Write data floating delay time	t_{WDZ}	—	—	60	ns	
$\overline{AS}$ high-level pulse width	t_{ASW1}	70	—	—	ns	
Read data strobe hold time	t_{RDHX}	0	—	—	ns	

Table 10.36 Interrupt Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –40 to +85°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
$\overline{\text{INT}}$ delay time	t_{IRD}	—	—	50	ns	Figure 10.10, figure 10.11
$\overline{\text{INTA}}$ active set-up time	t_{IAS1}	30	—	—	ns	
$\overline{\text{INTA}}$ inactive set-up time	t_{IAS2}	30	—	—	ns	
WAIT inactive delay time	t_{IWD1}	—	—	50	ns	
WAIT active delay time	t_{IWD2}	—	—	50	ns	
Vector data delay time	t_{IDBD1}	—	—	65	ns	
Vector data hold time	t_{IDBD2}	10	—	—	ns	
Vector data floating delay time	t_{IDBZ}	—	—	60	ns	

Table 10.37 Bus Arbitration Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –40 to +85°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
HOLD delay time	t_{HLDD}	—	—	55	ns	Figure 10.12
HOLDA set-up time	t_{HLAS}	30	—	—	ns	
$\overline{\text{BEO}}$ delay time	t_{BEOD}	—	—	50	ns	Figure 10.12, figure 10.13
$\overline{\text{BUSY}}$ delay time	t_{BSYD}	—	—	60	ns	
$\overline{\text{BUSY}}$ set-up time	t_{BSYS}	30	—	—	ns	
$\overline{\text{BUSREQ}}$ delay time	t_{BRQD}	—	—	50	ns	Figure 10.13
$\overline{\text{BUSACK}}$ set-up time	t_{BAKS}	30	—	—	ns	

Table 10.38 MSCI Timing

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –40 to +85°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
TXC cycle time (TXC input)	t_{TCYC}	1.4*[1]	—	— *[3]	t_{CYC}	Figure 10.14 to figure 10.22
TXC rise time (TXC input)	t_{TCr}	—	—	10	ns	
TXC fall time (TXC input)	t_{TCf}	—	—	10	ns	
TXC high-level pulse width (TXC input)	t_{TCHW}	0.55	—	—	t_{CYC}	
TXC low-level pulse width (TXC input)	t_{TCLW}	0.55	—	—	t_{CYC}	
TXD delay time (TXC input)	t_{TDD1}	—	—	95	ns	
TXD delay time (TXC output)	t_{TDD2}	—	—	50	ns	
RXC cycle time	t_{RCYC}	1.4*[1]	—	— *[3]	t_{CYC}	
RXC rise time	t_{RCr}	—	—	10	ns	
RXC fall time	t_{RCf}	—	—	10	ns	
RXC high-level pulse width	t_{RCHW}	0.55	—	—	t_{CYC}	
RXC low-level pulse width	t_{RCLW}	0.55	—	—	t_{CYC}	
RXD–RXC set-up time (RXC input)	t_{RDS1}	30	—	—	ns	
RXC–RXD hold time (RXC input)	t_{RDH1}	20	—	—	ns	
RXD–RXC set-up time (RXC output)	t_{RDS2}	80	—	—	ns	
RXC–RXD hold time (RXC output)	t_{RDH2}	20	—	—	ns	
ADPLL operating clock cycle time	t_{PLCY}	57	—	—	ns	
ADPLL operating clock rise time	t_{PLr}	—	—	8	ns	
ADPLL operating clock fall time	t_{PLf}	—	—	8	ns	

Table 10.38 MSCI Timing (cont)

Item	Symbol	Min	Typ	Max	Unit	Timing
ADPLL operating clock high-level pulse width	t_{PLHW}	10	—	—	ns	Figure 10.14, figure 10.22
ADPLL operating clock low-level pulse width	t_{PLLW}	10	—	—	ns	
CLK–BRG output delay time *[2]	t_{BGD}	—	—	95	ns	
TXC/RXC output rise time	t_{BGr}	—	—	30	ns	
TXC/RXC output fall time	t_{BGf}	—	—	30	ns	
RXC–SYNC set-up time	t_{SYSU}	2.5	—	—	t_{CYC}	
RXC–SYNC hold time	t_{SYHD}	2.5	—	—	t_{CYC}	
$\overline{CTS}$ high-level pulse width	t_{CTSHW}	2.0	—	—	t_{CYC}	
$\overline{CTS}$ low-level pulse width	t_{CTSLW}	2.0	—	—	t_{CYC}	
$\overline{DCD}$ high-level pulse width	t_{DCDHW}	2.0	—	—	t_{CYC}	
$\overline{DCD}$ low-level pulse width	t_{DCDLW}	2.0	—	—	t_{CYC}	
CLK–$\overline{RTS}$ delay time	t_{RTSD}	—	—	70	ns	

Notes: 1. In asynchronous mode and loop mode t_{TCYC} and t_{RCYC} = 2.5 t_{CYC} (min).
2. $f_{BRG} \neq f_{CLK}$ (f_{BRG} is the baud rate generator output frequency; f_{CLK} is the system clock (CLK) frequency.)
3. Maximum cycle time corresponds to 50 bits/s.

Table 10.39 Rise and Fall Times of Input Signals with No Characteristics Specified

(V_{CC} = 5 V ± 10%, V_{SS} = 0 V, Ta = –40 to +85°C unless otherwise specified)

Item	Symbol	Min	Typ	Max	Unit	Timing
Input signal rise time	t_{Ir}	—	—	100	ns	Figure 10.23
Input signal fall time	t_{If}	—	—	100	ns	

10.4 Timing Diagrams

10.4.1 Slave Mode Bus Timing

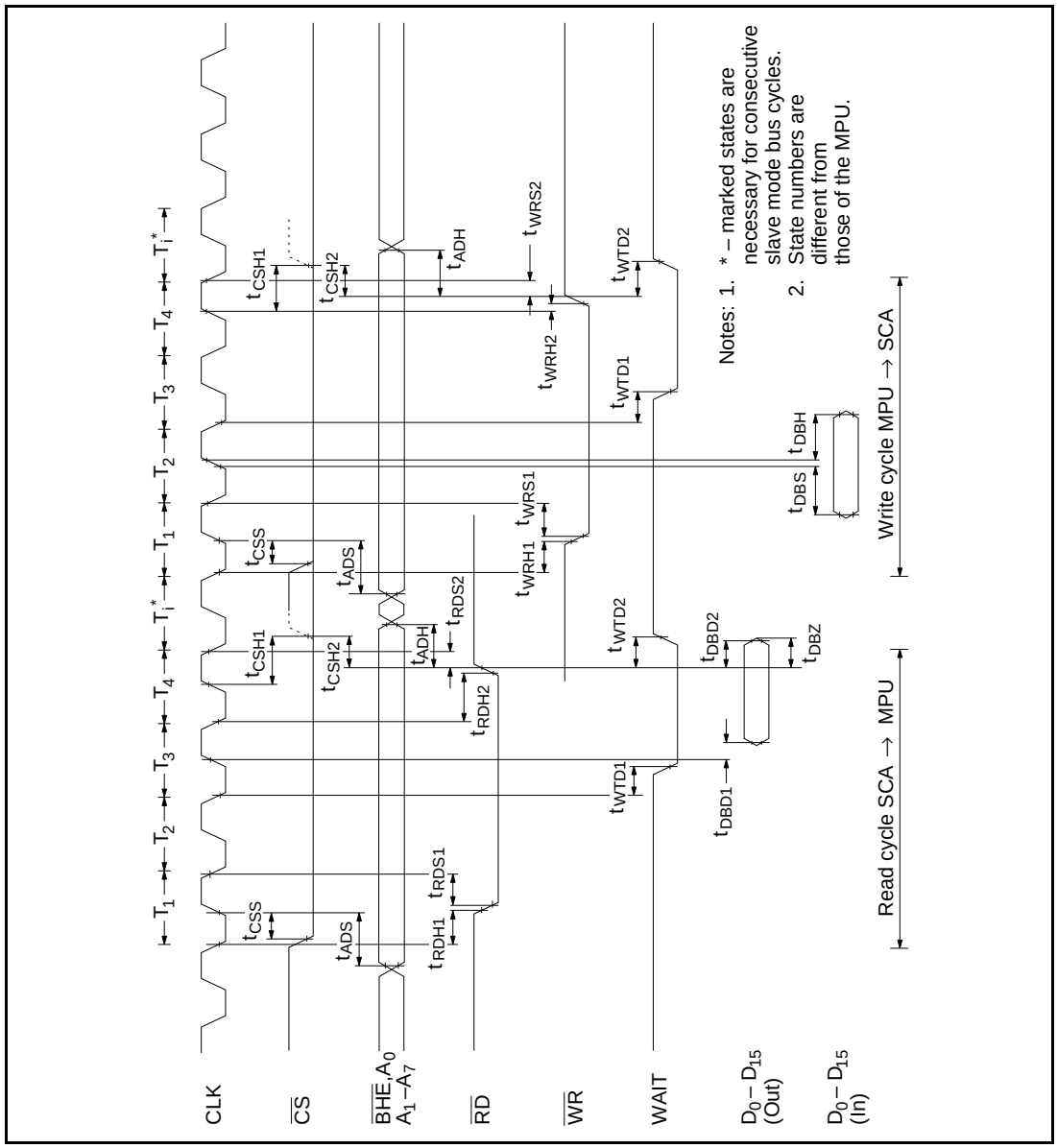


Figure 10.1 CPU Mode 0 Slave Mode Bus Timing

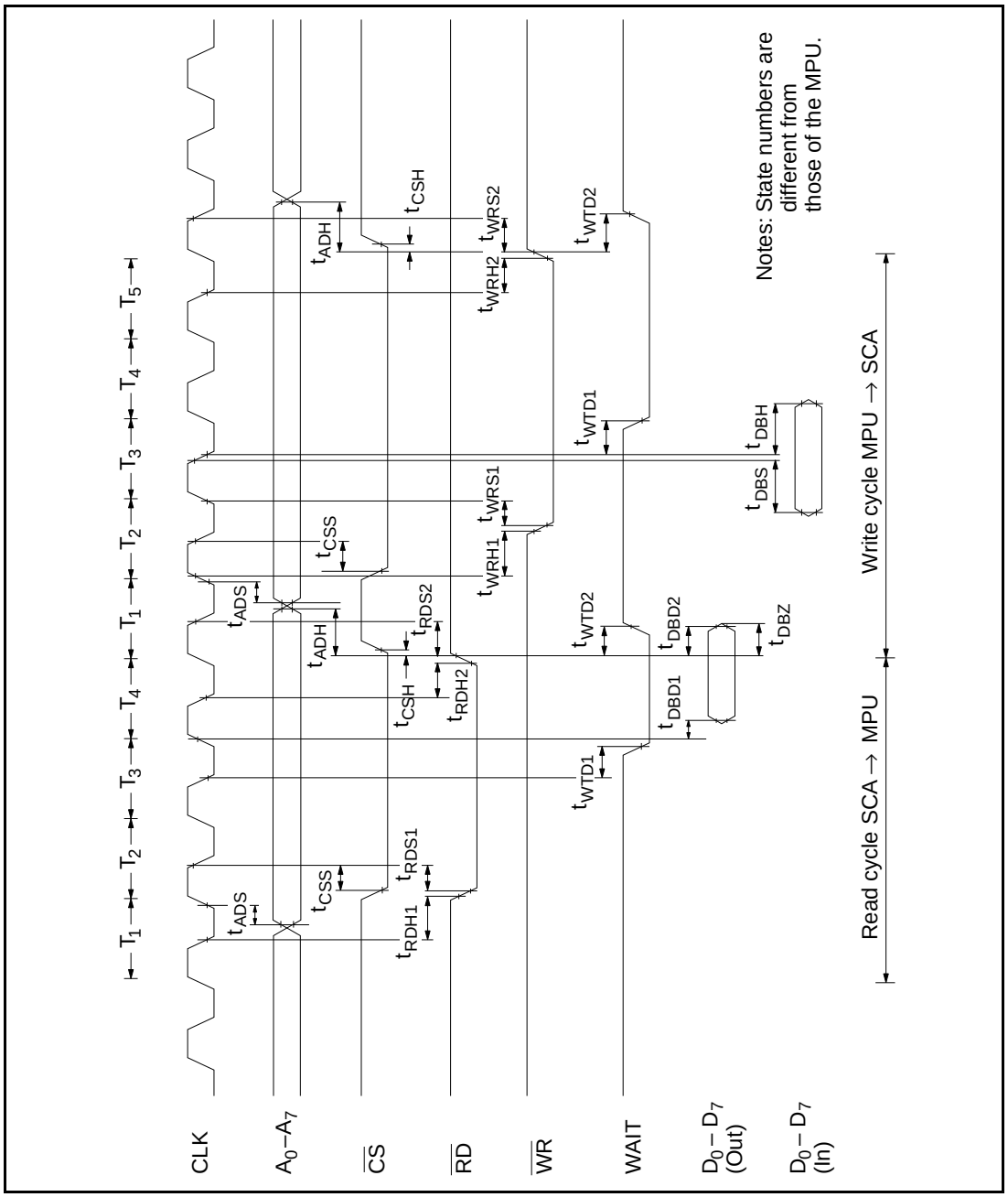


Figure 10.2 CPU Mode 1 Slave Mode Bus Timing

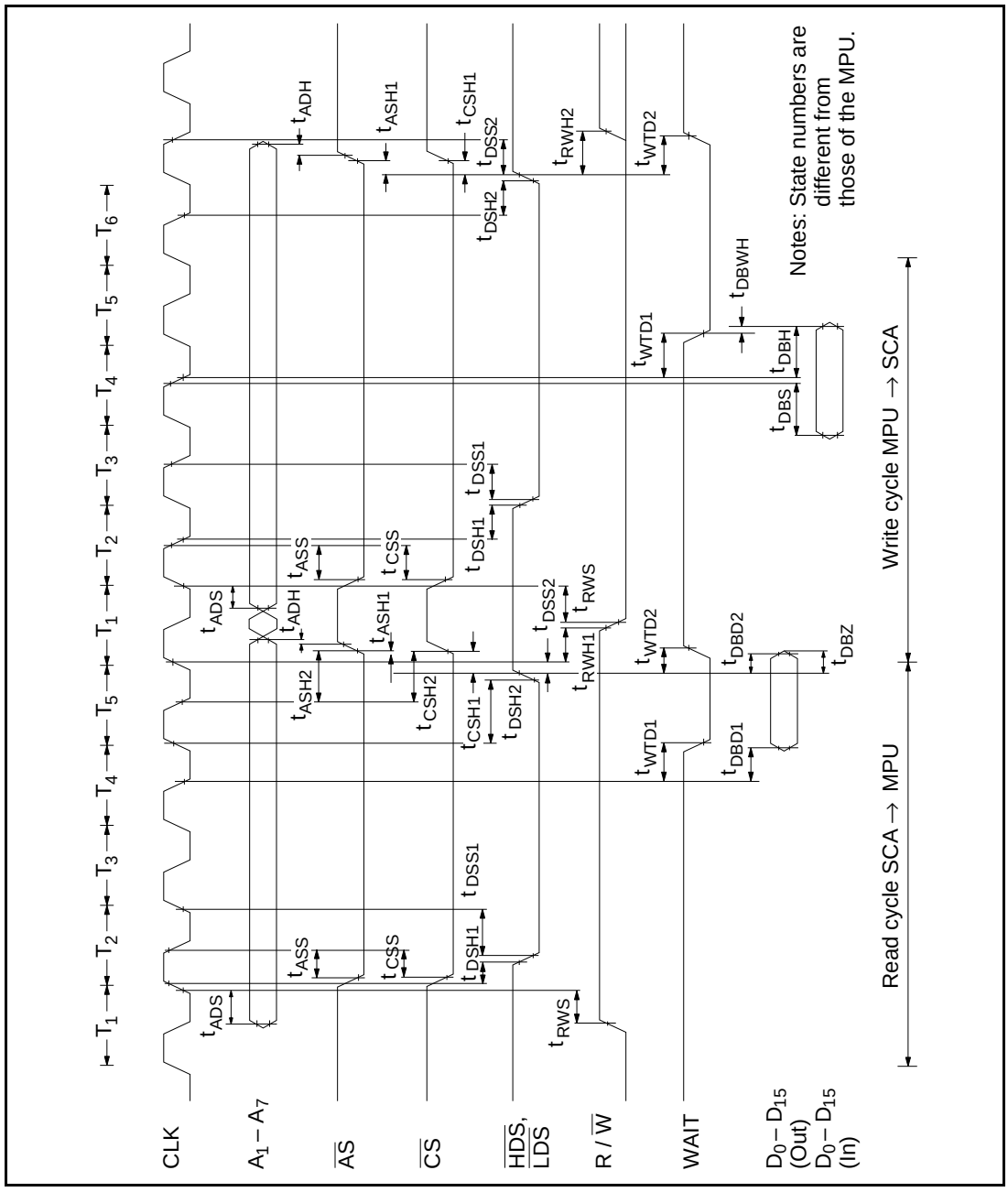


Figure 10.3 CPU Mode 2 Slave Mode Bus Timing

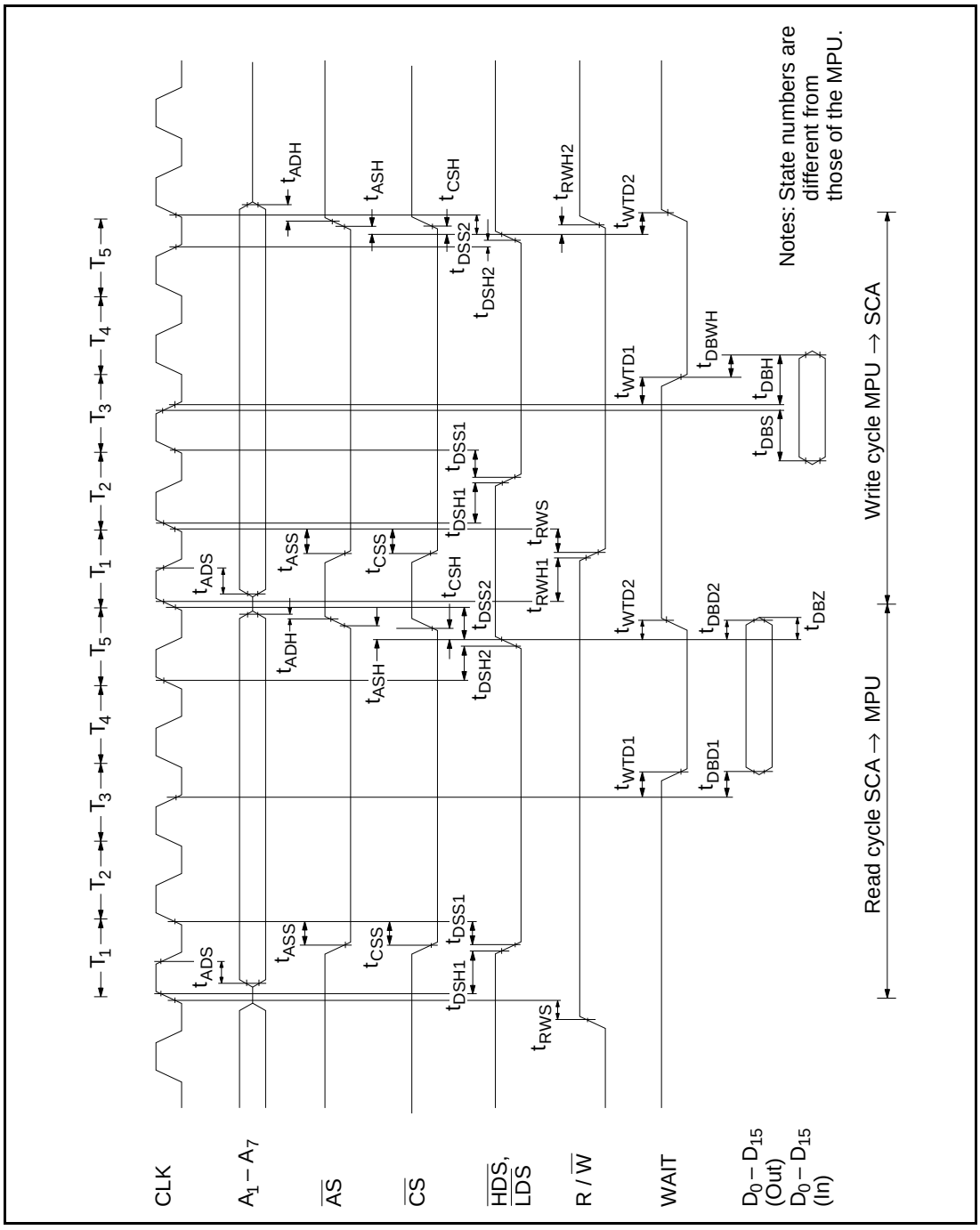


Figure 10.4 CPU Mode 3 Slave Mode Bus Timing

10.4.2 Master Mode Bus Timing

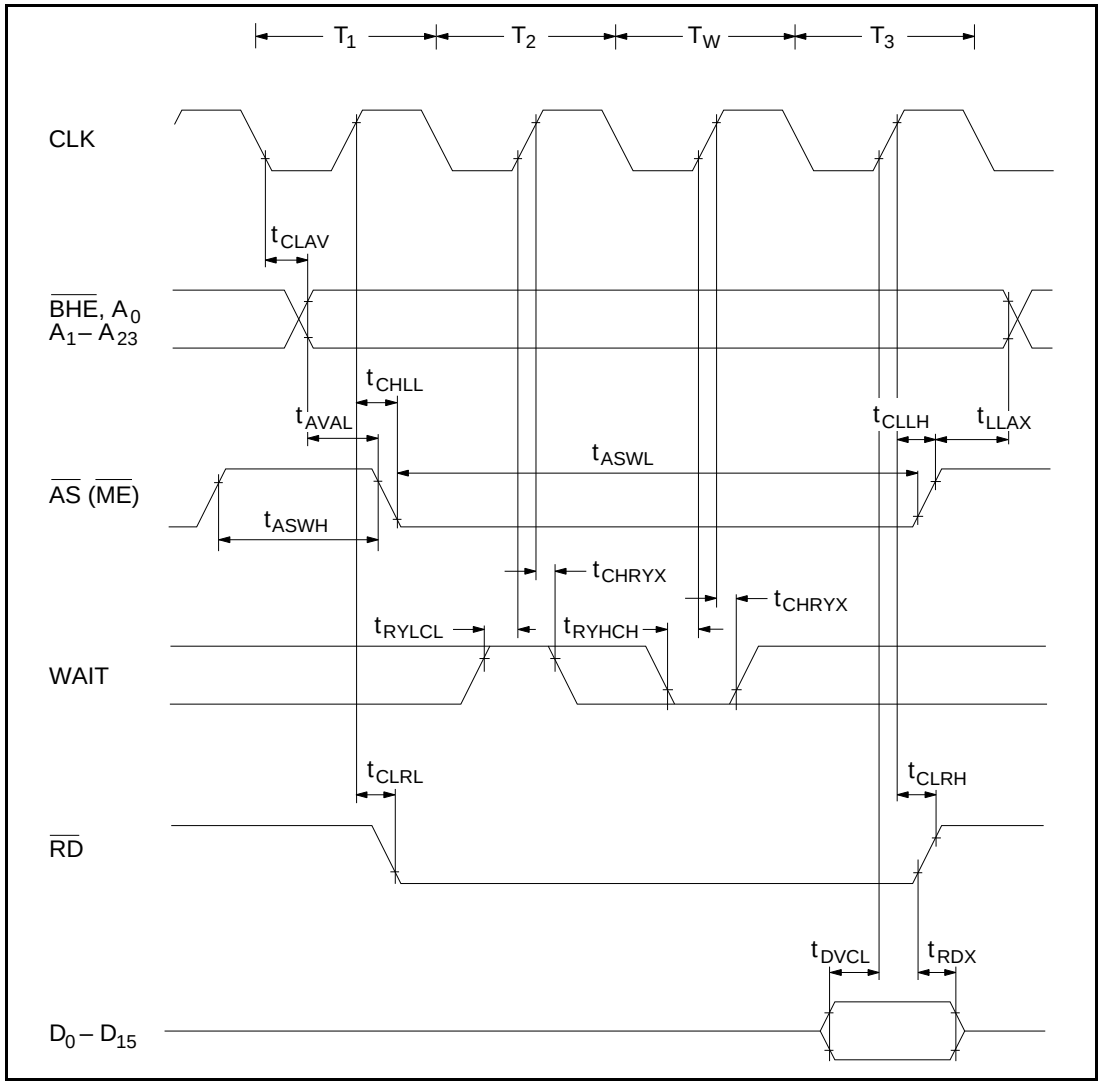


Figure 10.5 Master Mode Read Timing
(CPU Mode 0) (Memory ‡ SCA)

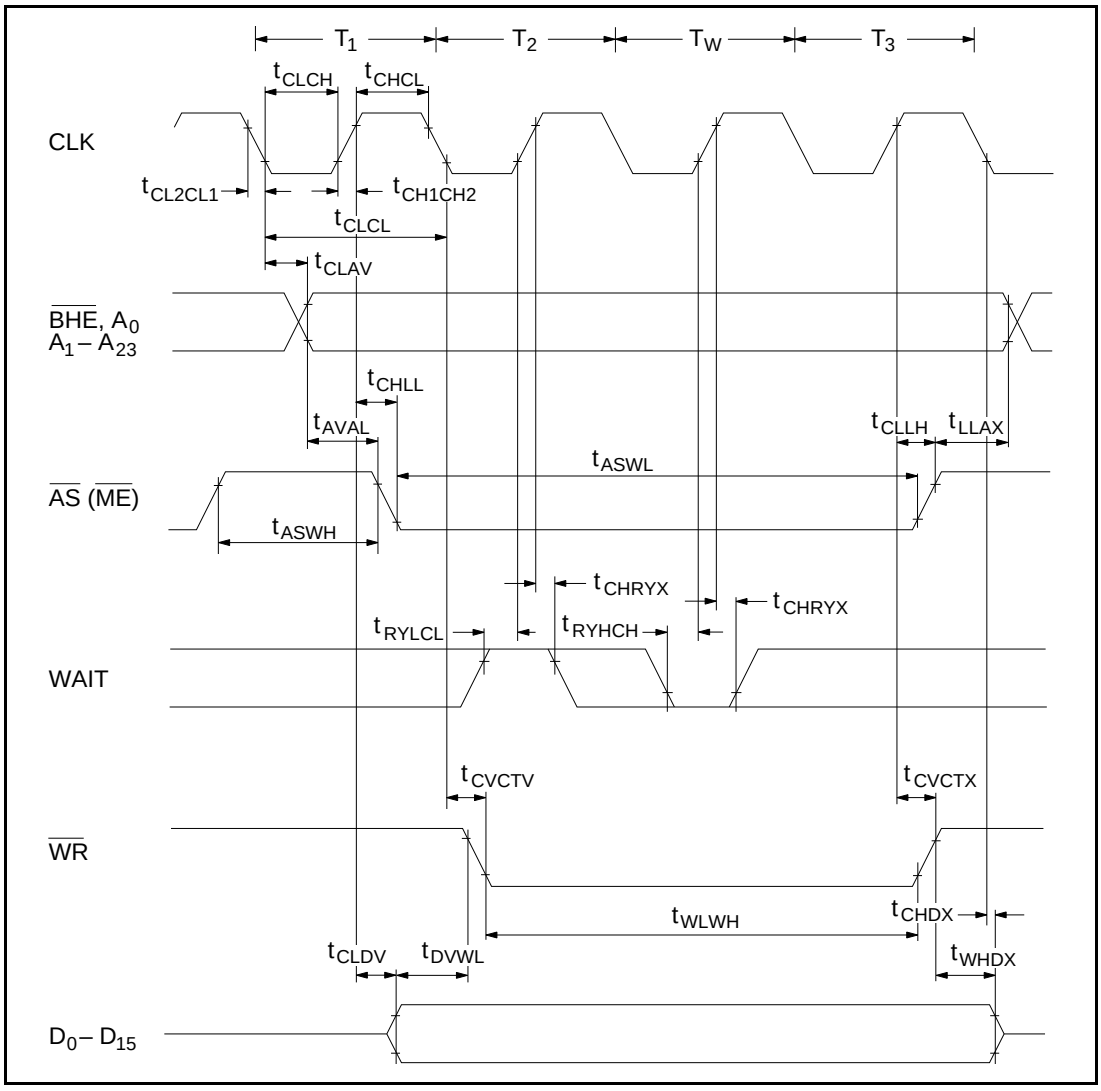


Figure 10.6 Master Mode Write Timing
(CPU Mode 0) (SCA ‡ Memory)

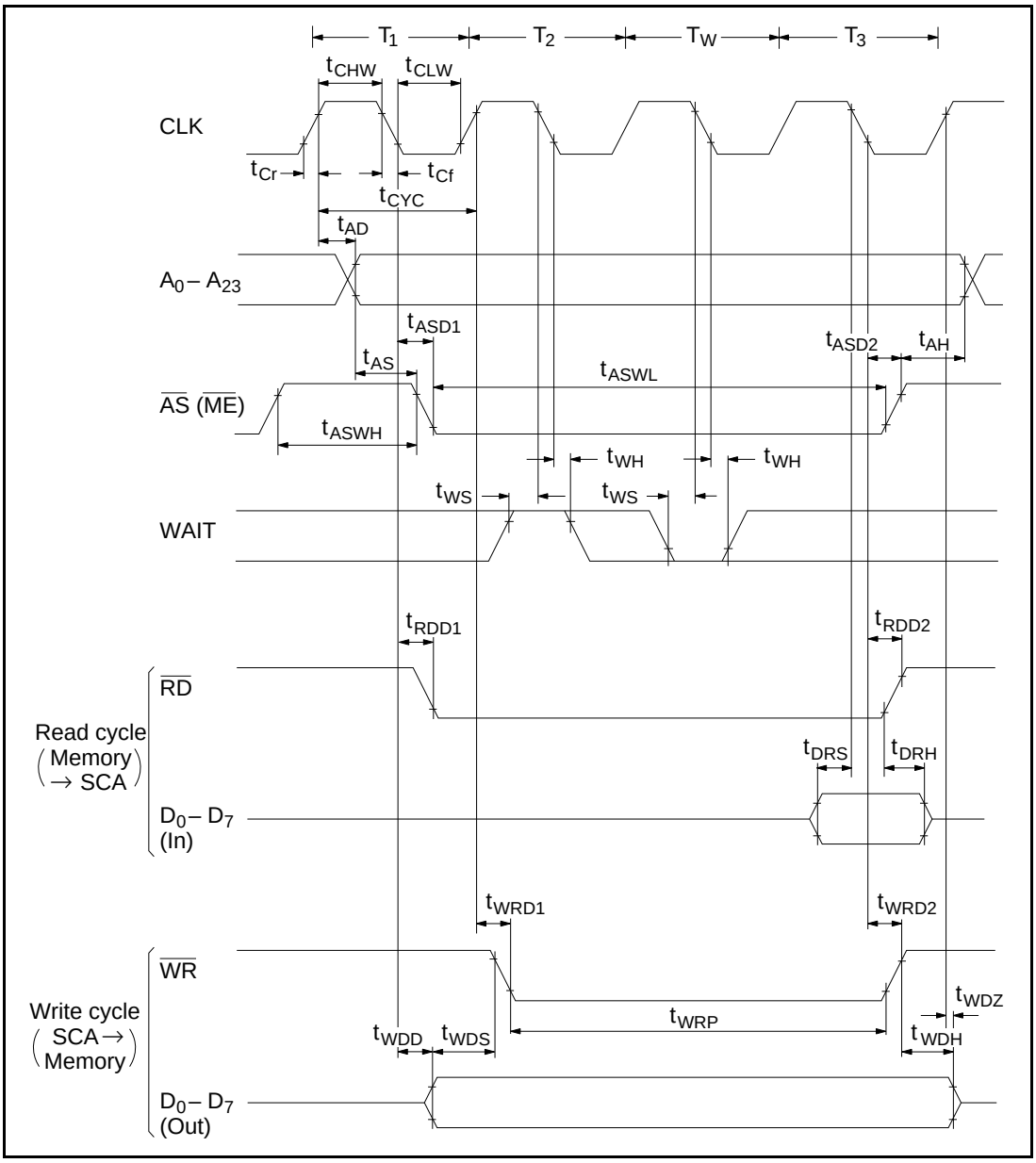


Figure 10.7 Master Mode Bus Timing
(CPU Mode 1)

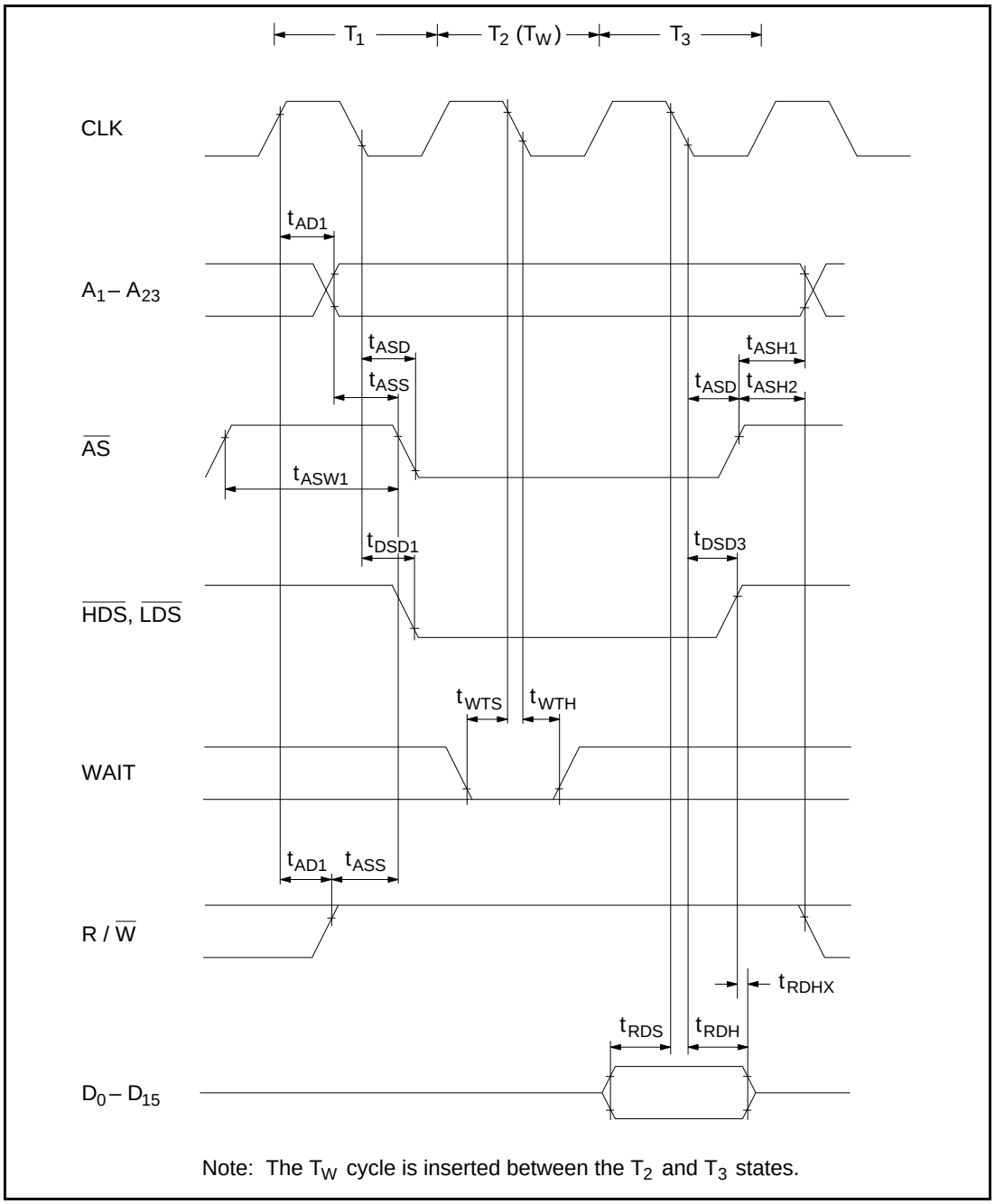


Figure 10.8 Master Mode Read Timing
(CPU Mode 2, 3) (Memory ‡ SCA)

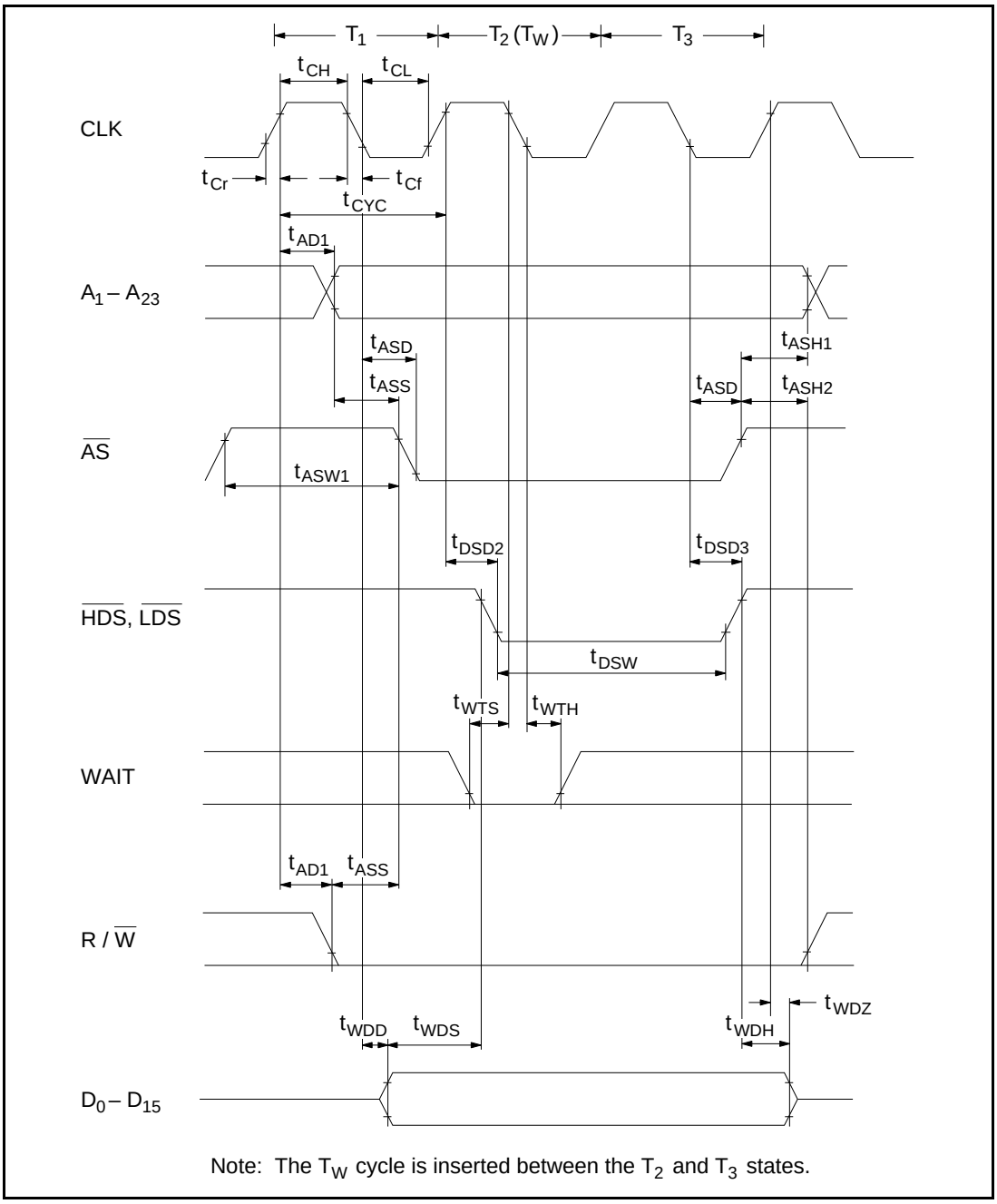


Figure 10.9 Master Mode Write Timing
(CPU Mode 2, 3) (SCA ‡ Memory)

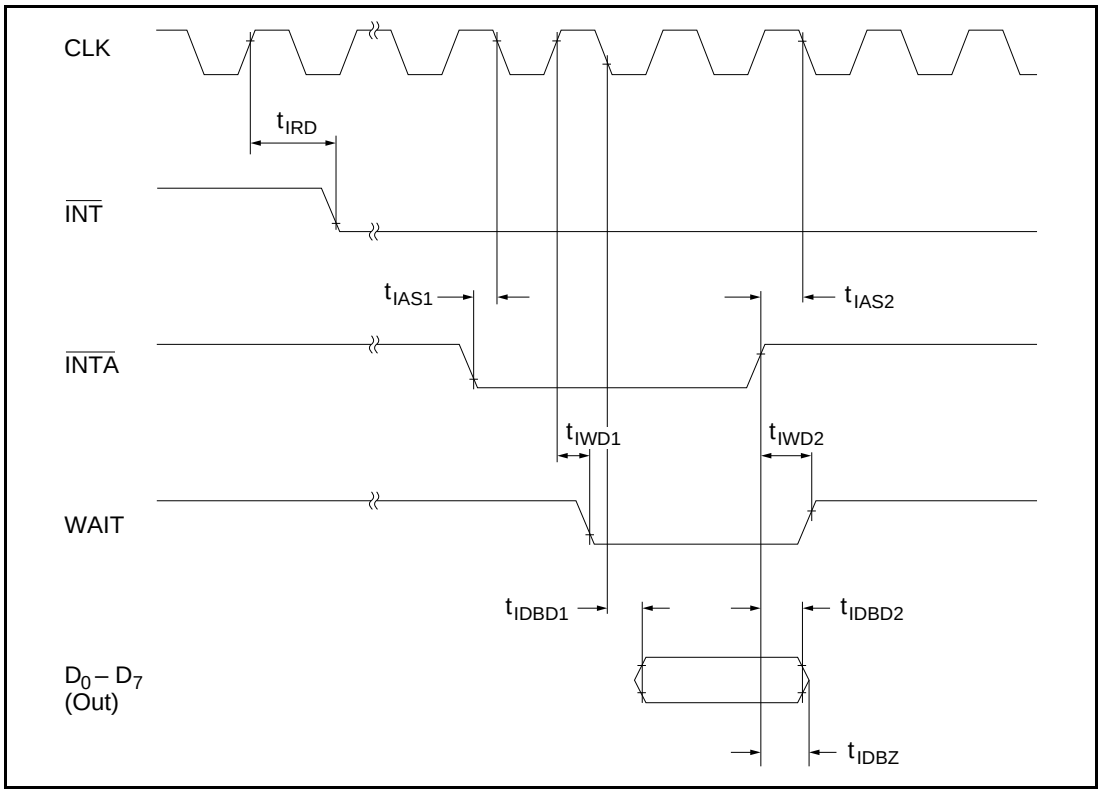


Figure 10.10 CPU Mode 0 Interrupt Timing

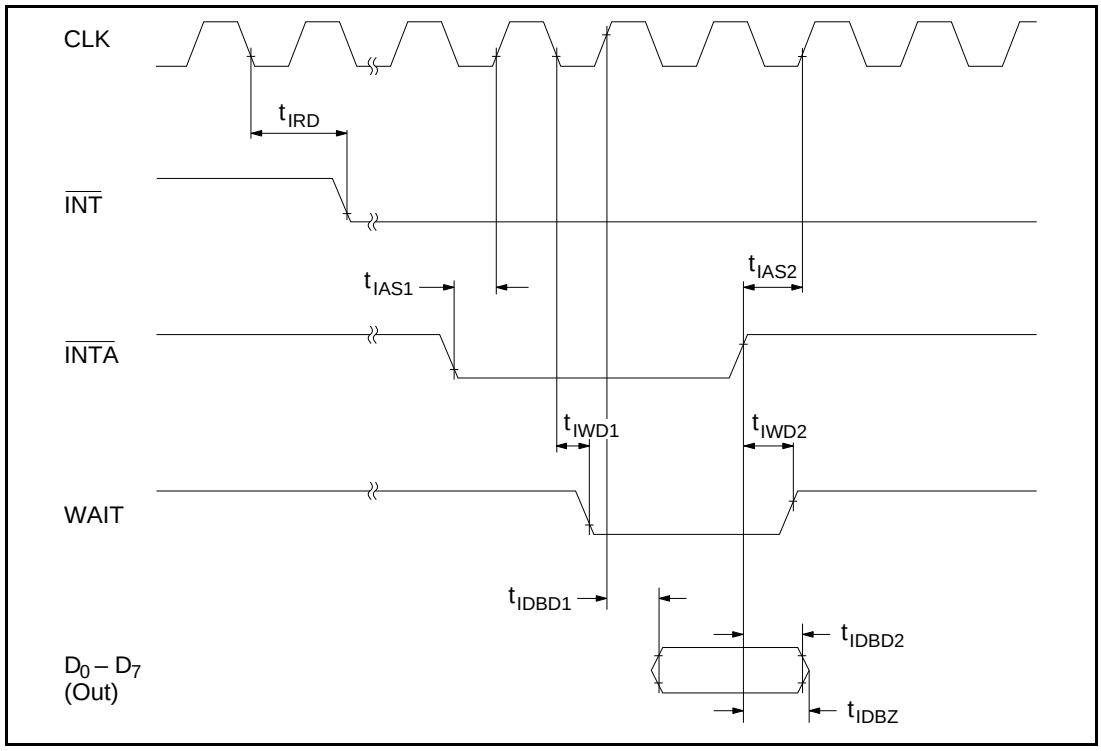
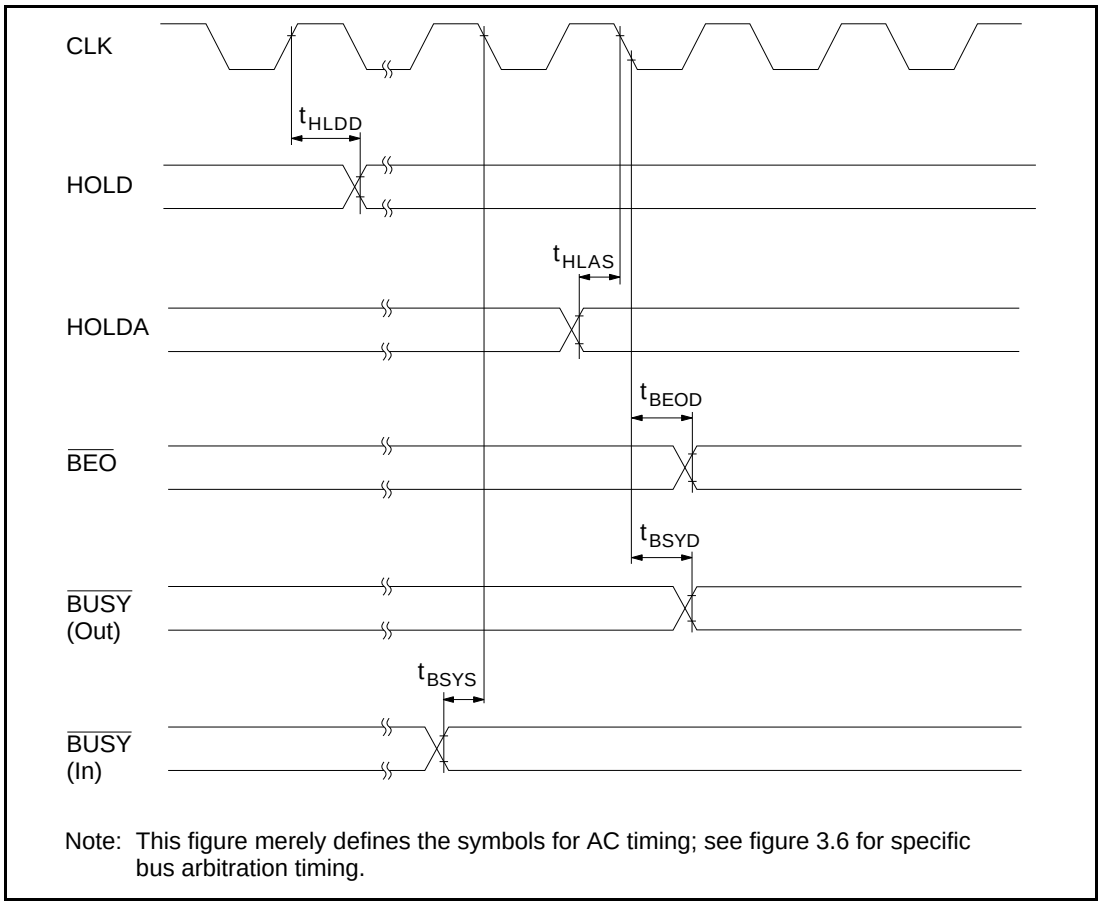
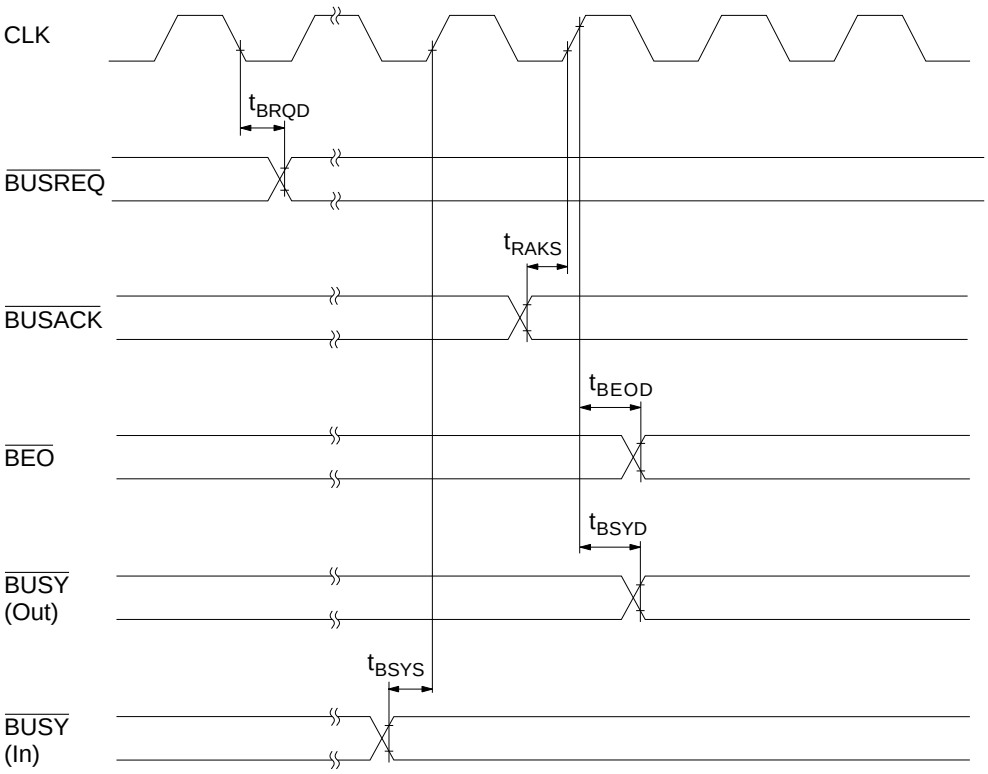


Figure 10.11 CPU Mode 1, 2, 3 Interrupt Timing



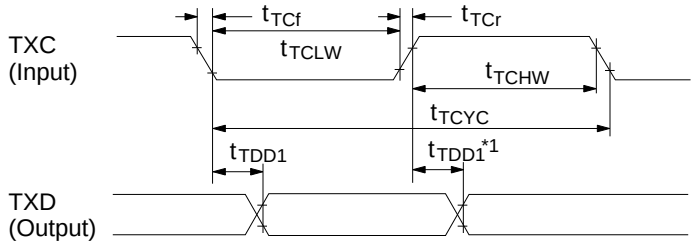
Note: This figure merely defines the symbols for AC timing; see figure 3.6 for specific bus arbitration timing.

Figure 10.12 CPU Mode 0 Bus Arbitration Timing



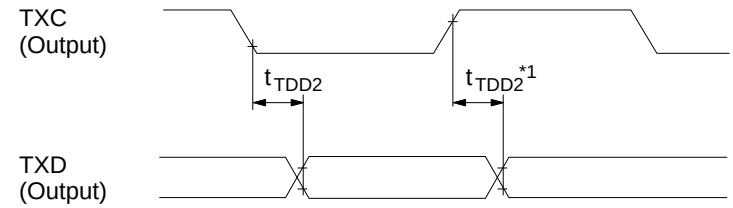
Note: This figure merely defines the symbols for AC timing; see figure 3.6 for specific bus arbitration timing.

Figure 10.13 CPU Mode 1, 2, 3 Bus Arbitration Timing



Note *1: There is no transition of the TXD at this point in NRZ mode.

Figure 10.14 Transmit Timing (TXC input)



For details of the TXC waveform, see figure 10.19, Baud Rate Generator Output Timing.

Note *1: There is no transition of the TXD at this point in NRZ mode.

Figure 10.15 Transmit Timing (TXC output)

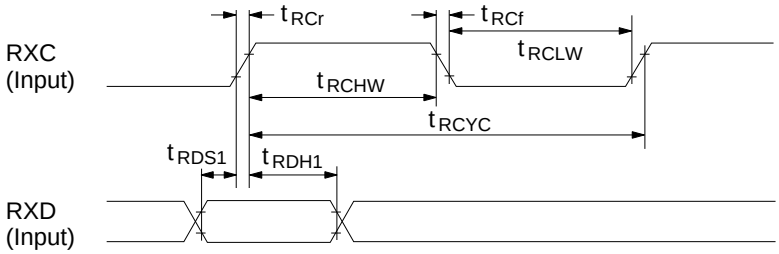
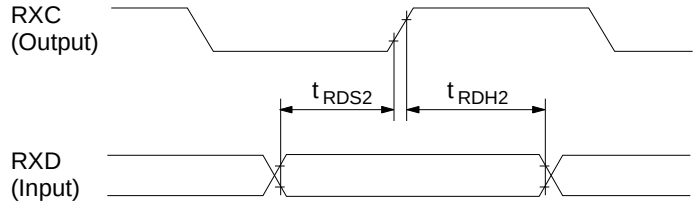


Figure 10.16 Receive Timing (RXC input)



For details of the RXC waveform, see figure 10.19, Baud Rate Generator Output Timing.

Figure 10.17 Receive Timing (RXC output)

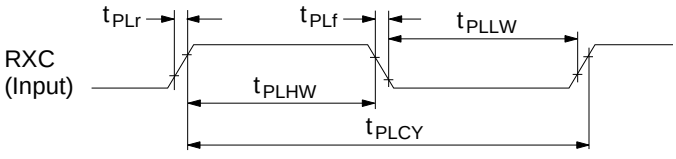


Figure 10.18 ADPLL Operating Clock Timing

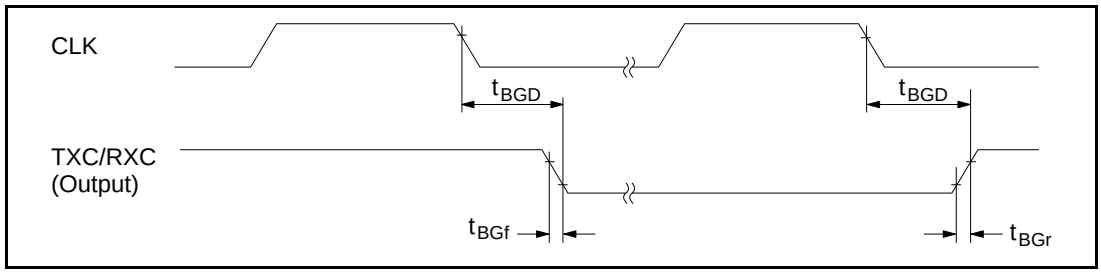


Figure 10.19 Baud Rate Generator Output Timing ($f_{BRG} \neq f_{CLK}$)

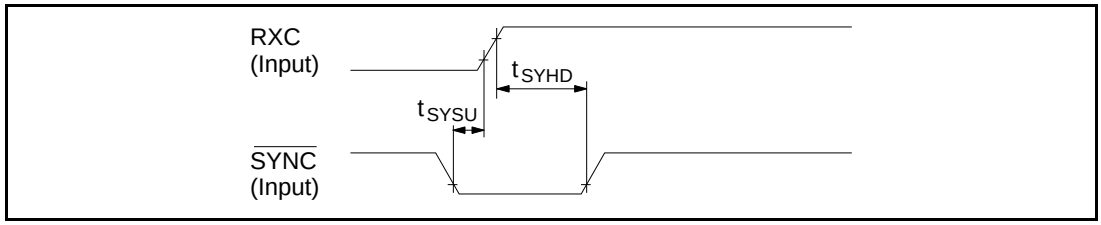


Figure 10.20 SYNC Timing

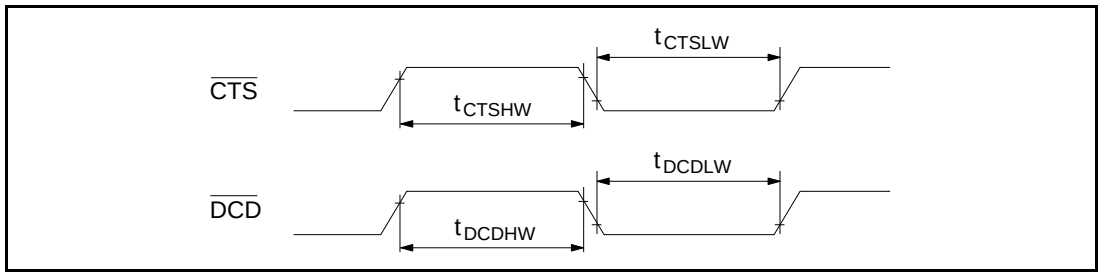


Figure 10.21 CTS and DCD Timing

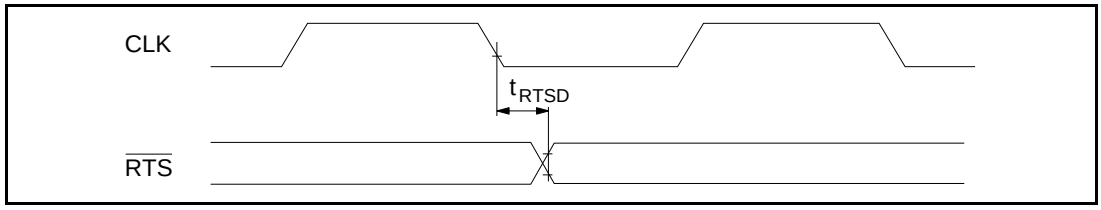


Figure 10.22 RTS Timing

Figure 10.23 Rise and Fall Times of Input Signals with No Characteristics Specified

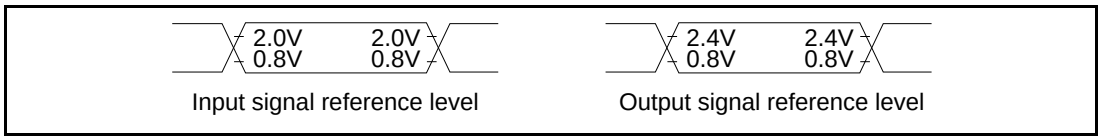


Figure 10.24 Reference Levels with No Characteristics Specified

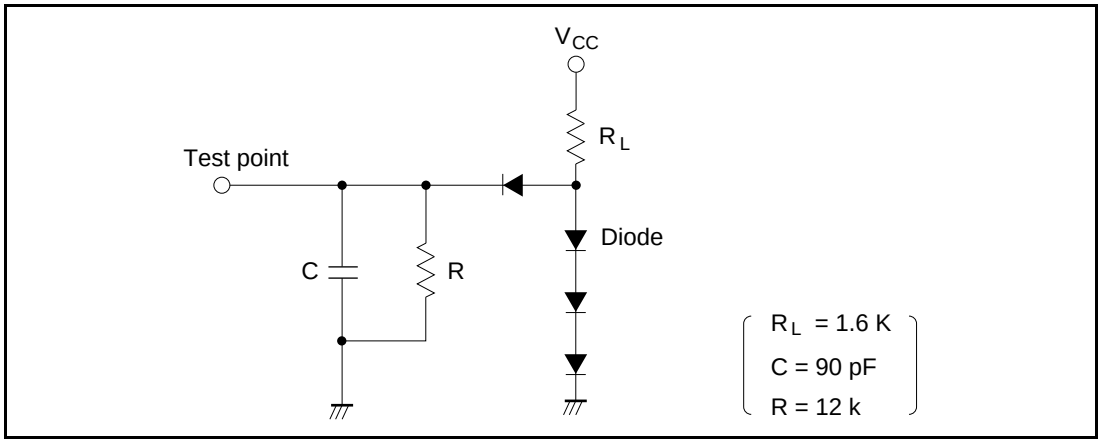


Figure 10.25 Bus Timing Load 1 (TTL load)

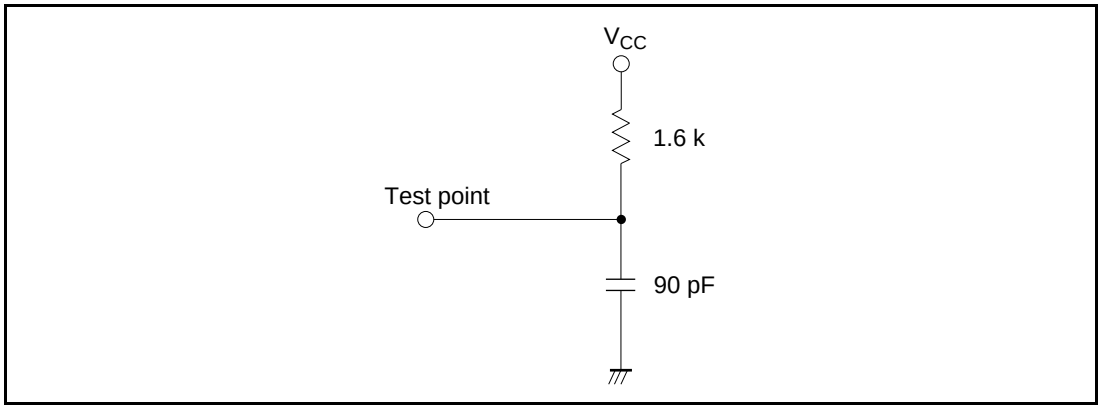


Figure 10.26 Bus Timing Load 2 (open-drain load)

Section 11 Package Dimensions

Figure 11.1 shows the package dimensions of the HD64570 (CP-84 and FP-88).

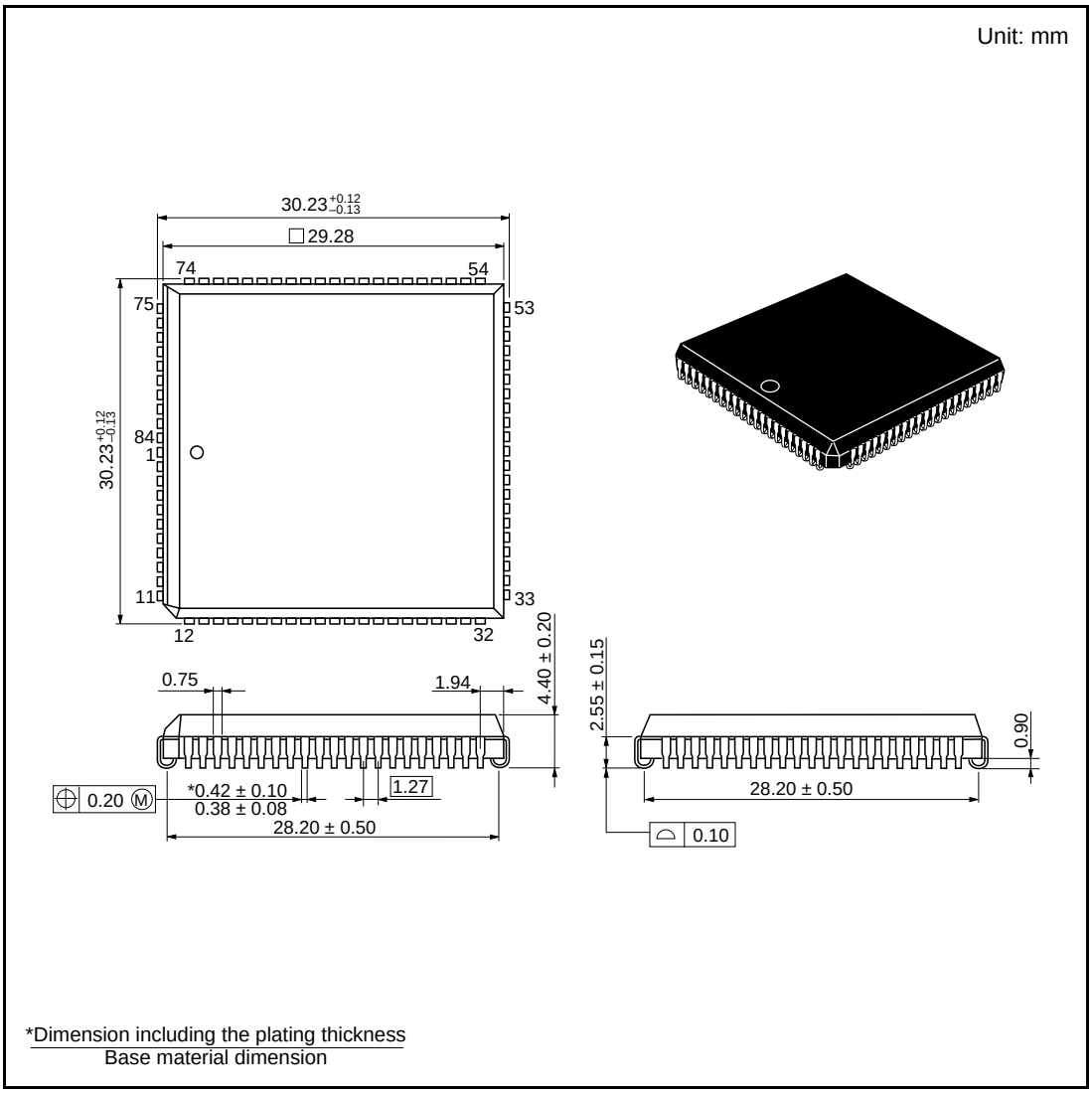


Figure 11.1 CP-84 Package Dimensions

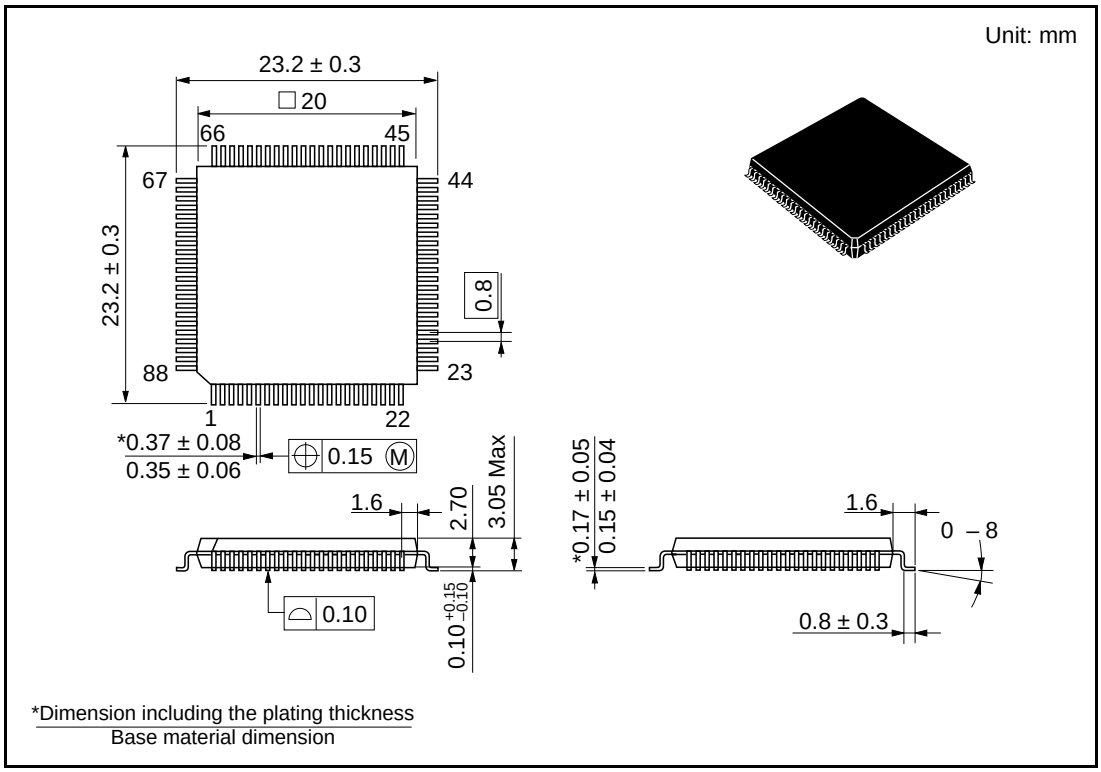
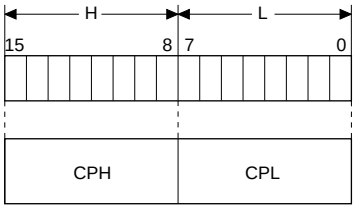
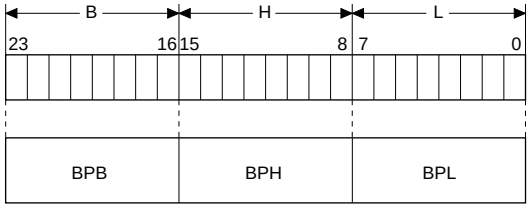


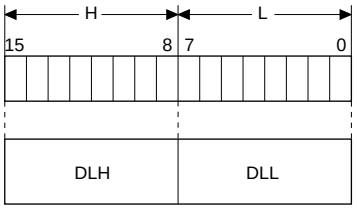
Figure 11.1 FP-88 Package Dimensions

Appendix A Descriptors

Descriptor	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
Chain Pointer L (CPL)	2 n	2 n + 1	
Chain Pointer H (CPH)	2 n + 1	2 n	
Buffer Pointer L (BPL)	2 n + 2	2 n + 3	
Buffer Pointer H (BPH)	2 n + 3	2 n + 2	
Buffer Pointer B (BPB)	2 n + 4	2 n + 5	
(Reserved)	2 n + 5	2 n + 4	
Data Length (DLL)	2 n + 6	2 n + 7	
Data Length (DLH)	2 n + 7	2 n + 6	
Status (ST)	2 n + 8	2 n + 9	
(Reserved)	2 n + 9	2 n + 8	







Status Configuration (transmission)

Bit	Function
7	EOM
6	Not used
5	Not used
4	Not used
3	Not used
2	Not used
1	Not used
0	EOT

Status Configuration (reception)

Bit	Function
7	EOM
6	Short frame
5	Abort
4	Residual bit
3	Overrun
2	CRC
1	Not used
0	Not used

Appendix B Registers

Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
System			
Low Power Register (LPR)	00H	01H	
Not used	01H	00H	

Bit name	7	6	5	4	3	2	1	0
	—	—	—	—	—	—	—	IOSTP
Read/Write	—	—	—	—	—	—	—	R/W
Initial value	0	0	0	0	0	0	0	0

I/O stop
0: No transition to system stop mode
1: Transition to system stop mode

Wait Control

Physical Address Boundary Registers 0 (PABR0) 02H 03H

Bit name	7	6	5	4	3	2	1	0
	PB07	PB06	PB05	PB04	PB03	PB02	PB01	PB00
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

PAL/PAM boundary address (high-order 8 bits)

Physical Address Boundary Registers 1 (PABR1) 03H 02H

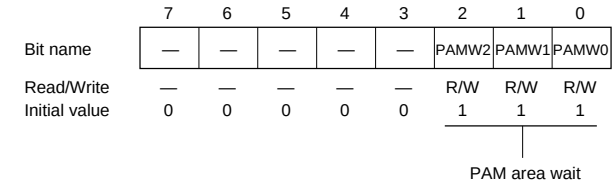
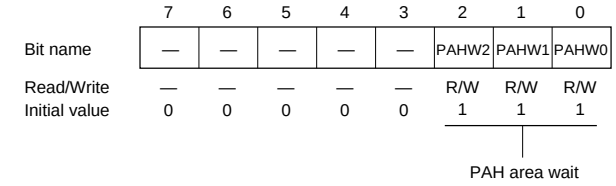
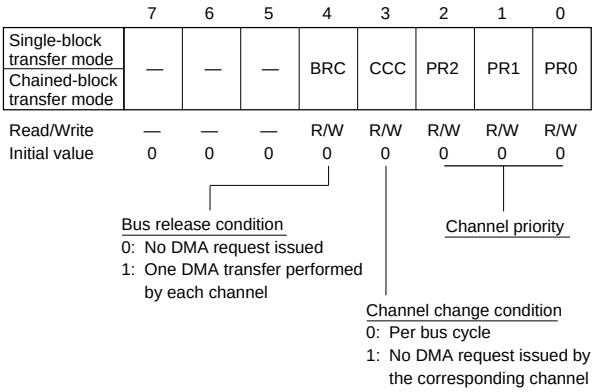
Bit name	7	6	5	4	3	2	1	0
	PB17	PB16	PB15	PB14	PB13	PB12	PB11	PB10
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

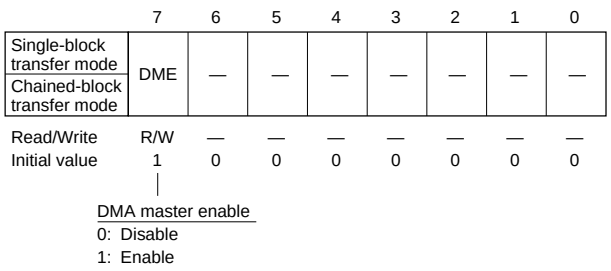
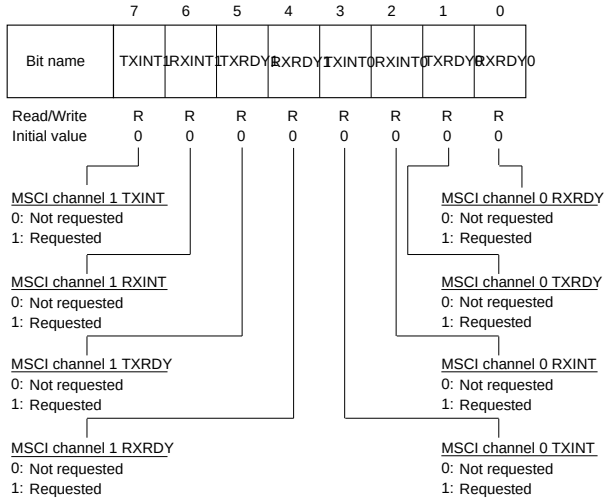
PAM/PAH boundary address (high-order 8 bits)

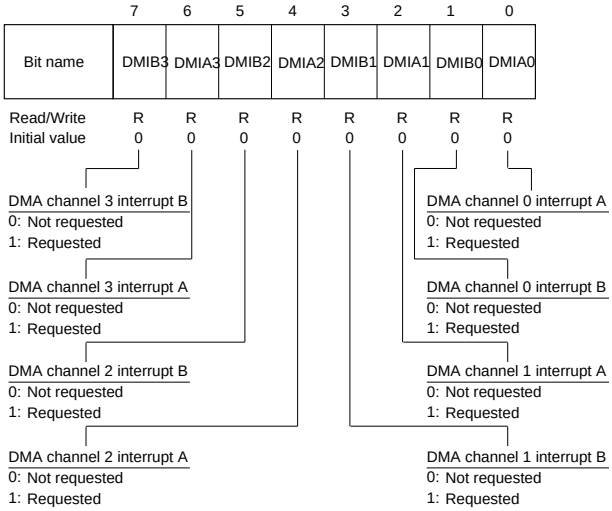
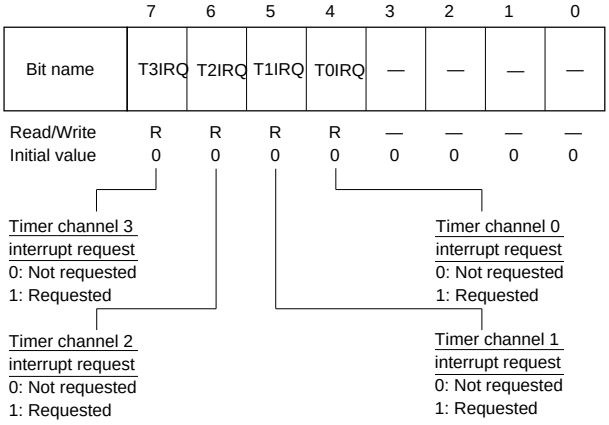
Wait Control Registers L (WCRL) 04H 05H

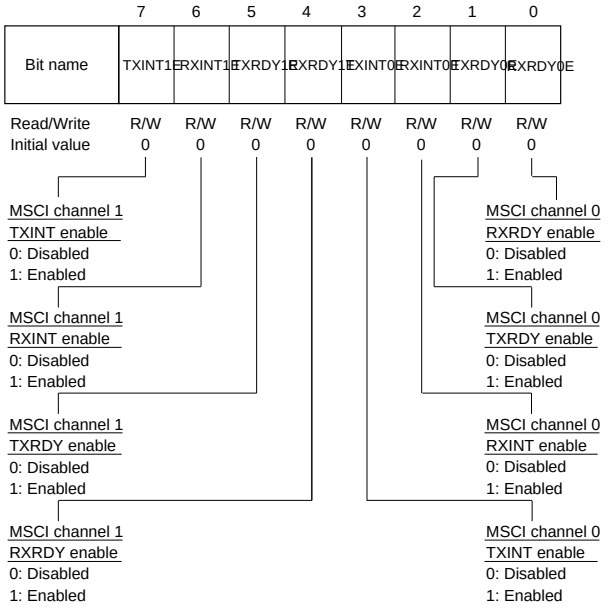
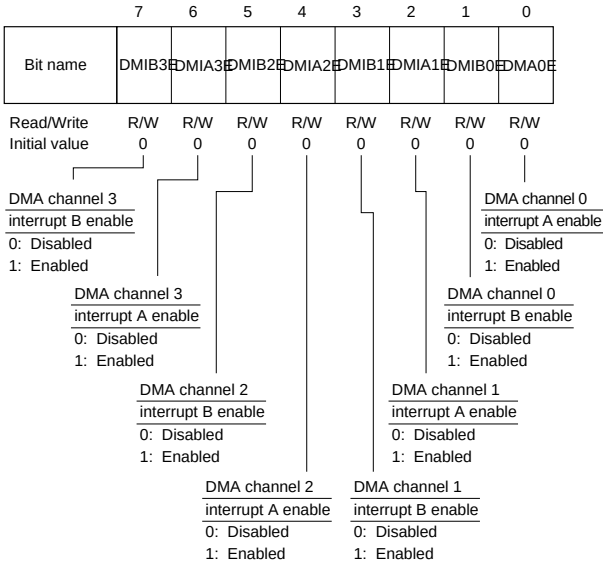
Bit name	7	6	5	4	3	2	1	0
	—	—	—	—	—	PALW2	PALW1	PALW0
Read/Write	—	—	—	—	—	R/W	R/W	R/W
Initial value	0	0	0	0	0	1	1	1

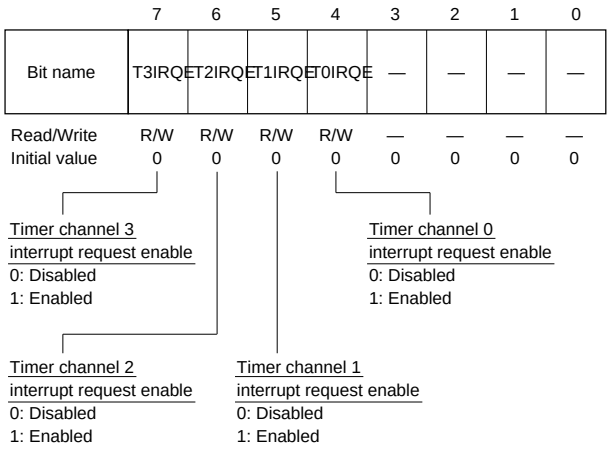
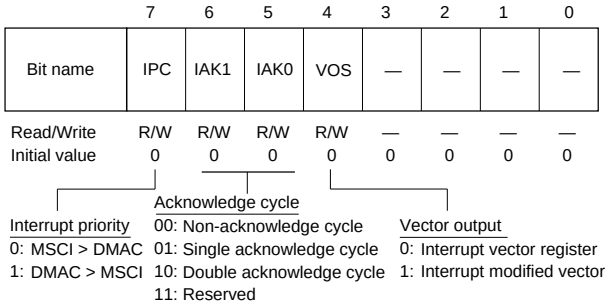
PAL area wait

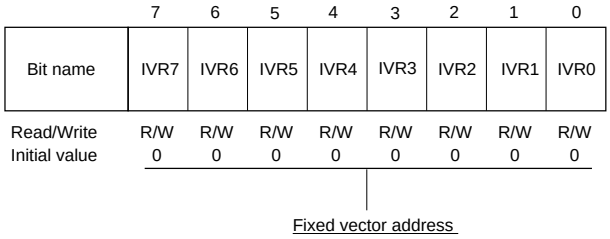
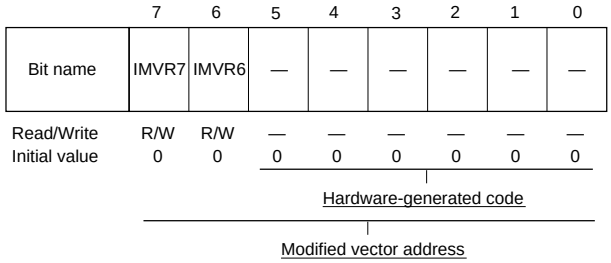
Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
Wait Control			
Wait Control Registers M (WCRM)	05H	04H	
Wait Control Registers H (WCRH)	06H	07H	
Not used	07H	06H	
DMAC (General)			
DMA Priority Control Register (PCR)	08H	09H	

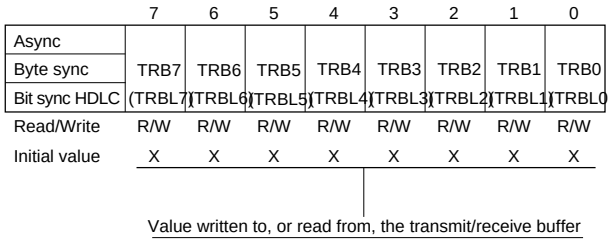
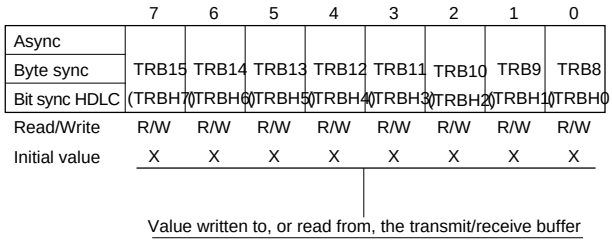
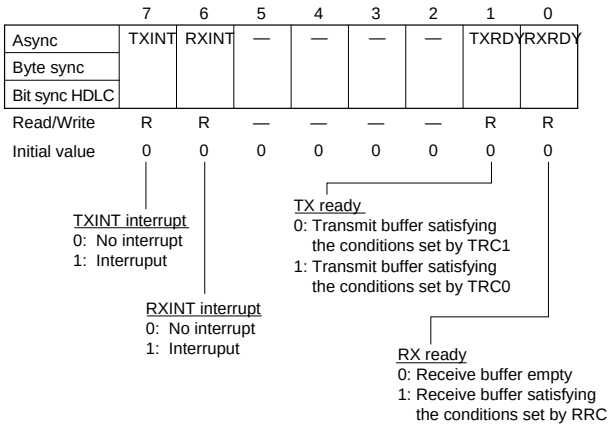
Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
DMAC (General)			
DMA Master Enable Register (DMER)	09H	08H	
Not used	0AH	0BH	
Not used	0BH	0AH	
Not used	0CH	0DH	
Not used	0DH	0CH	
Not used	0EH	0FH	
Not used	0FH	0EH	
Interrupt Control			
Interrupt Status Register 0 (ISR0)	10H	11H	

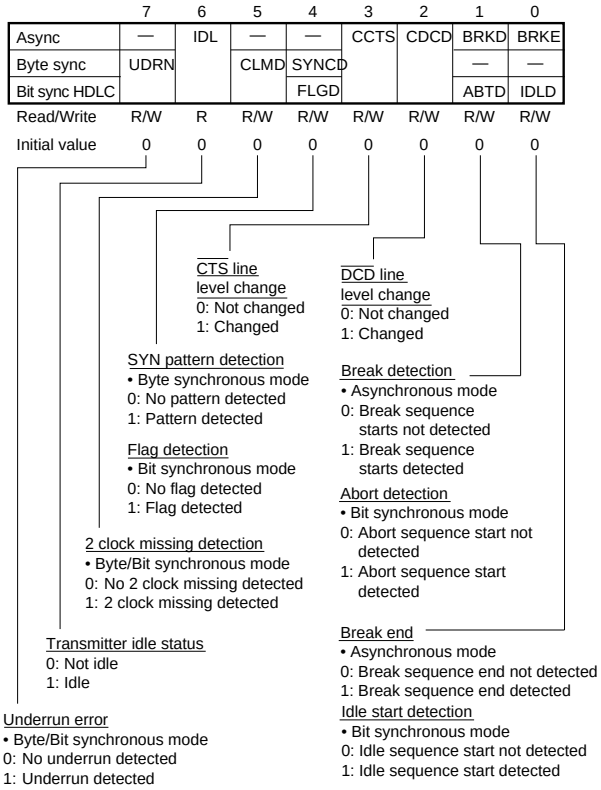
Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
Interrupt Control			
Interrupt Status Register 1 (ISR1)	11H	10H	
Interrupt Status Register 2 (ISR2)	12H	13H	
Not used	13H	12H	

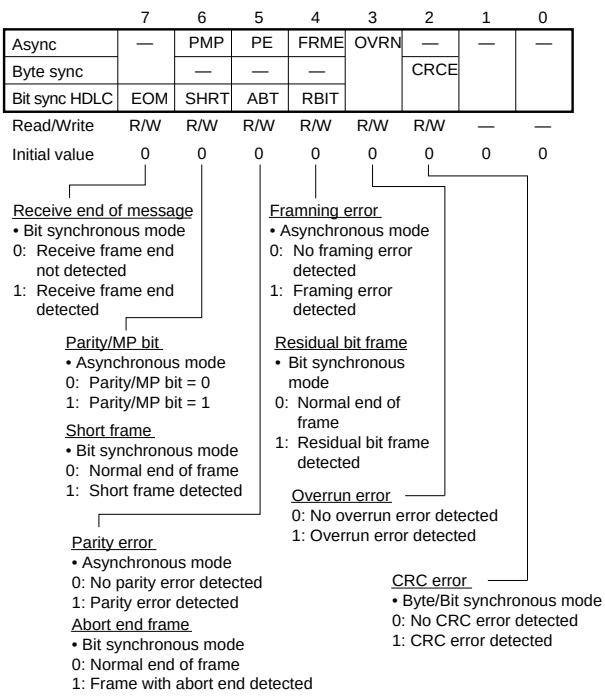
Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
Interrupt Control			
Interrupt Enable Register 0 (IER0)	14H	15H	
Interrupt Enable Register 1 (IER1)	15H	14H	

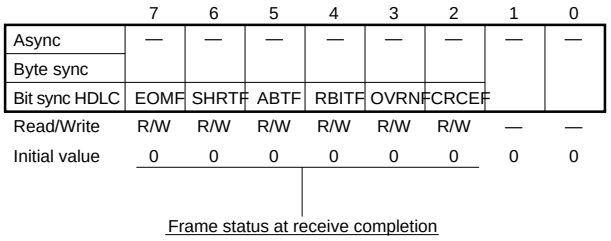
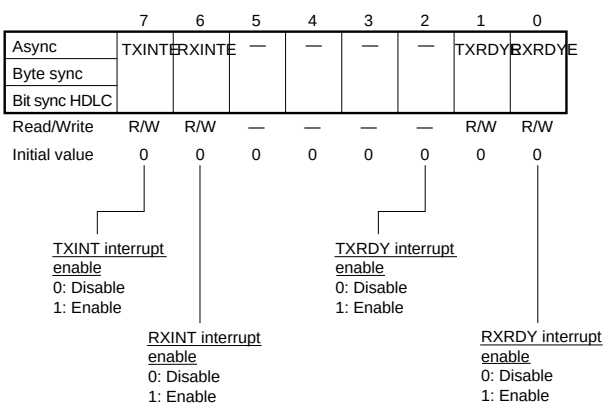
Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
Interrupt Control			
Interrupt Enable Register 2 (IER2)	16H	17H	
Not used	17H	16H	
Interrupt Control Register (ITCR)	18H	19H	
Not used	19H	18H	

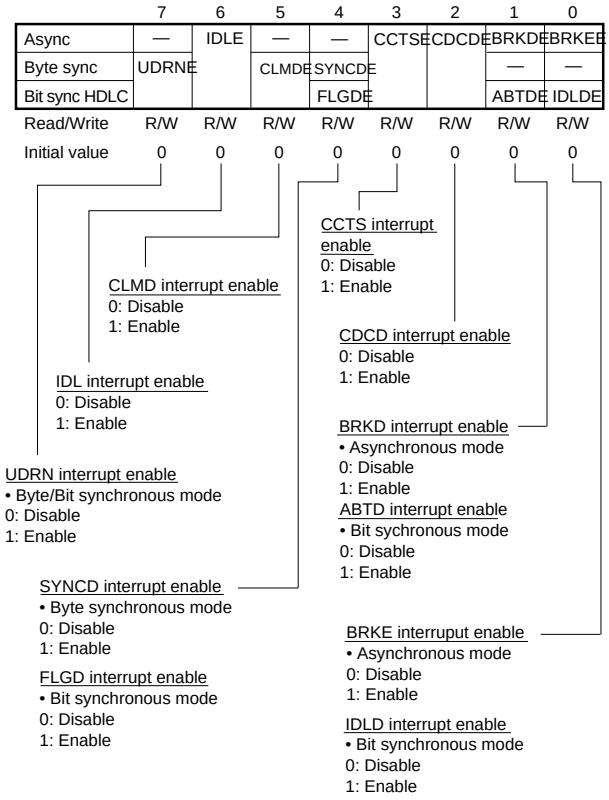
Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
Interrupt Control			
Interrupt Vector Register (IVR)	1AH	1BH	
Not used	1BH	1AH	
Interrupt Modified Vector Register (IMVR)	1CH	1DH	
Not used	1DH	1CH	
Not used	1EH	1FH	
Not used	1FH	1EH	

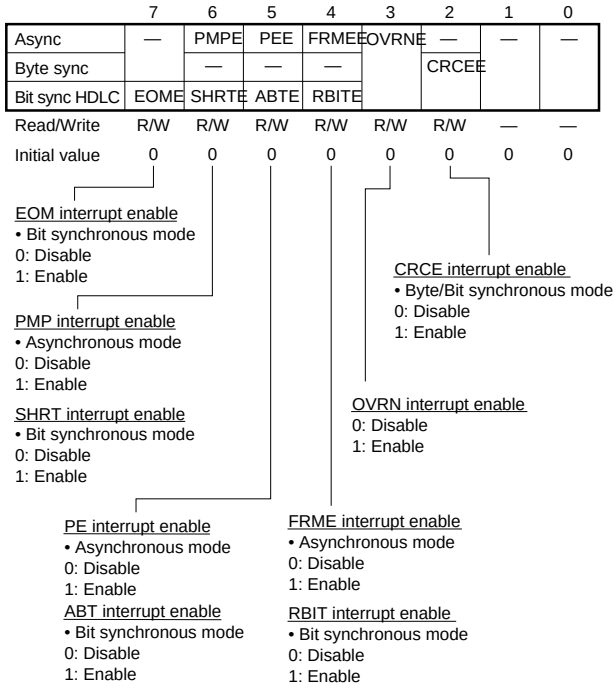
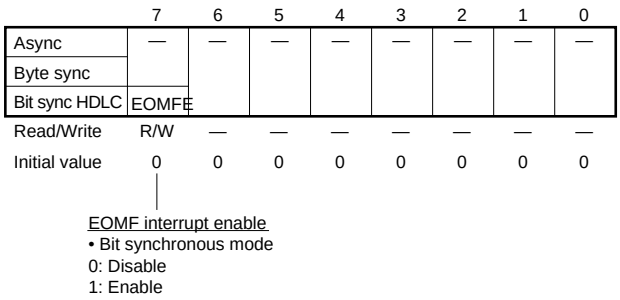
Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 0)			
MSCI TX/RX Buffer Register L Channel 0: TRBL Channel 0	20H	21H	
MSCI TX/RX Buffer Register H Channel 0: TRBH Channel 0	21H	20H	
MSCI Status Register 0 Channel 0: ST0 Channel 0	22H	23H	

Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
MSCI (Channel 0)			
MSCI Status Register 1 Channel 0: ST1 Channel 0	23H	22H	

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 0)			
MSCI Status Register 2 Channel 0: ST2 Channel 0	24H	25H	
MSCI Status Register 3 Channel 0: ST3 Channel 0	25H	24H	

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 0)			
MSCI Frame Status Register Channel 0: FST Channel 0	26H	27H	
Not used	27H	26H	
MSCI Interrupt Enable Register 0 Channel 0: IE0 Channel 0	28H	29H	

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 0)			
MSCI Interrupt Enable Register 1 Channel 0: IE1 Channel 0	29H	28H	

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 0)			
MSCI Interrupt Enable Register 2 Channel 0: IE2 Channel 0	2AH	2BH	
MSCI Frame Interrupt Enable Register Channel 0: FIE Channel 0	2BH	2AH	

Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
MSCI (Channel 0)			
MSCI Command Register Channel 0: CMD Channel 0	2CH	2DH	
Not used	2DH	2CH	
MSCI Mode Register 0 Channel 0: MD0 Channel 0	2EH	2FH	

	7	6	5	4	3	2	1	0
Async	—*1	—*1	CMD5	CMD4	CMD3	CMD2	CMD1	CMD0
Byte sync								
Bit sync HDLC								
Read/Write	—	—	W	W	W	W	W	W
Initial value	—	—	—	—	—	—	—	—

Command

• Transmit commands
000001: TX reset
000010: TX enable
000011: TX disable
000100: TX CRC initialization
000101: TX CRC calculation exclusion
000110: End-of-message
000111: Abort transmission
001000: MP bit on
001001: TX buffer clear
Others: Reserved

• Receive commands
010001: RX reset
010010: RX enable
010011: RX disable
010100: RX CRC initialization
010101: Message reject
010110: Search MP bit
010111: RX CRC calculation exclusion
011000: Forcing RX CRC calculation

• Other commands
100001: Channel reset
110001: Enter search mode
000000: No operation

	7	6	5	4	3	2	1	0
Async	PRTCL2	PRTCL1	PRTCL0	AUTO	—	—	STOP1	STOP0
Byte sync						CRCC	CRC1	CRC0
Bit sync HDLC								
Read/Write	R/W	R/W	R/W	R/W	—	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

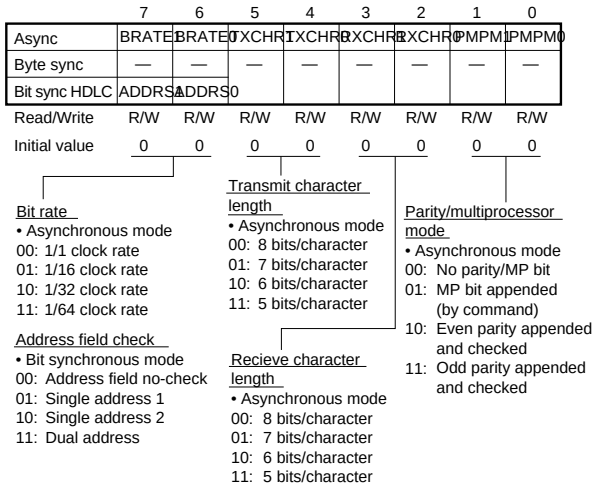
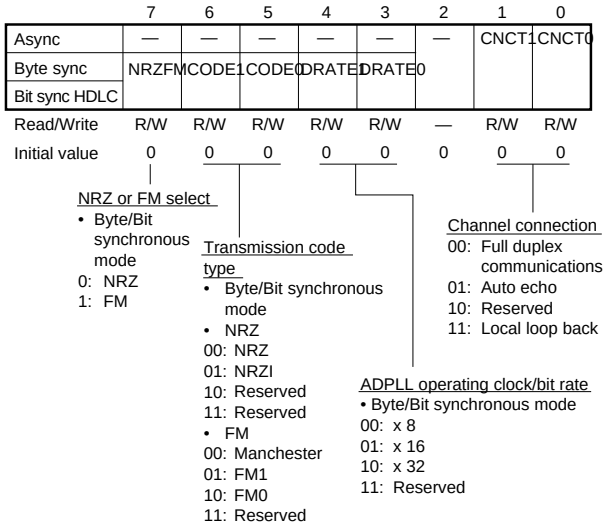
Protocol mode
000: Asynchronous mode
001: Byte-sync mono-sync mode
010: Byte-sync Bi-sync mode
011: Byte-sync external synchronous mode
100: Bit-sync HDLC mode
101: Reserved
110: Reserved
111: Reserved

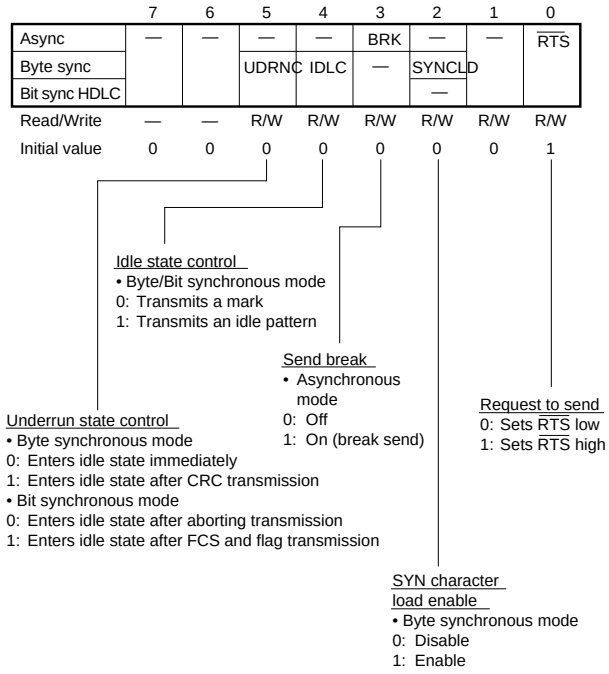
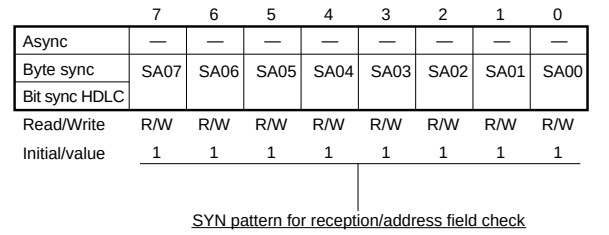
Auto-enable
0: Auto-enable reset
1: Auto-enable set

CRC code caluculation
• Byte/Bit synchronous mode
0: Disable
1: Enable

Stop bit length
• Asynchronous mode
00: 1 bit
01: 1.5 bits
10: 2 bits
11: Reseved

CRC calcuration expression and initial value
• Byte/Bit synchronous mode
0X: CRC-16
1X: CRC-CCITT
X0: Initial value = all 0s
X1: Initial value = all 1s

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 0)			
MSCI Mode Register 1 Channel 0: MD1 Channel 0	2FH	2EH	
MSCI Mode Register 2 Channel 0: MD2 Channel 0	30H	31H	

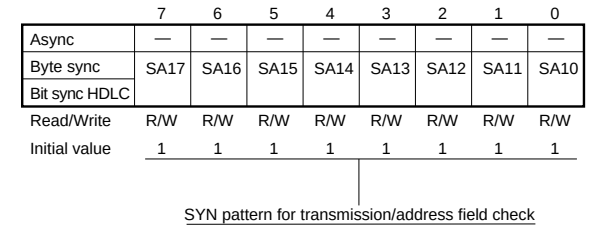
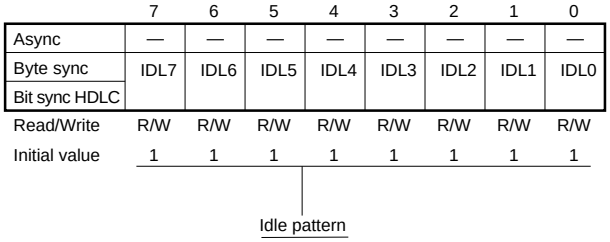
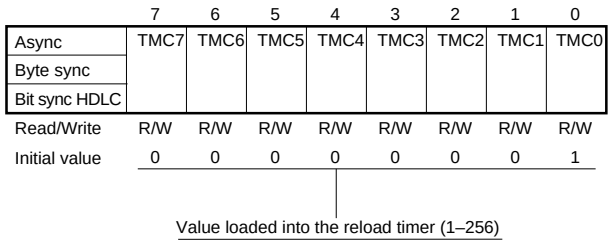
Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 0)			
MSCI Control Register Channel 0: CTL Channel 0	31H	30H	
MSCI Synchronous/ Address Register 0 Channel 0: SA0 Channel 0	32H	33H	

• Byte synchronous mode

Mono-sync	SYN pattern for reception
Bi-sync	SYN pattern for transmission and reception (bit7–bit0)
External-sync	Not used

• Bit synchronous mode

HDLC mode	No address field checked	Not used
	Single address 1	Bit7–bit0 of the secondary station address
	Single address 2	Not used
	Dual address	Bit7–bit0 of the secondary station address

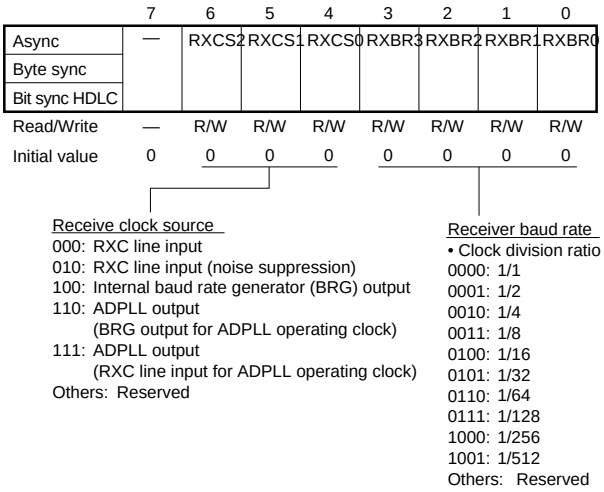
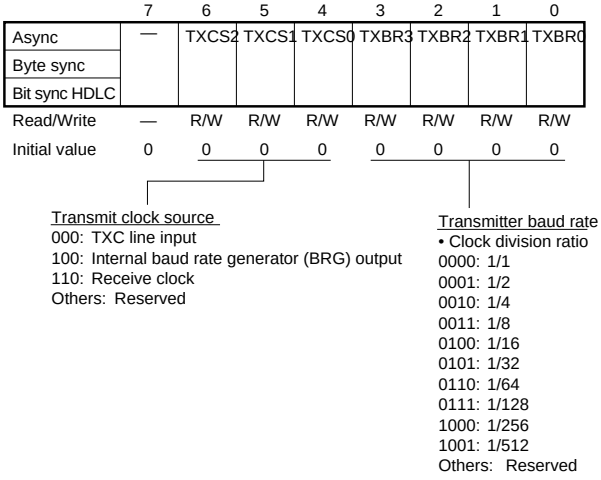
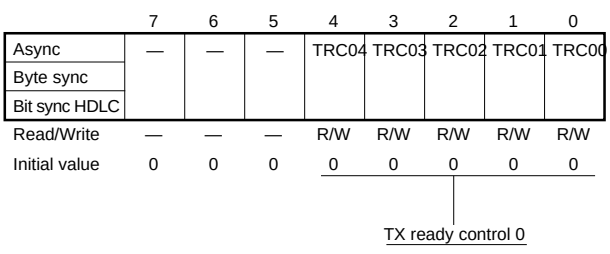
Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 0)			
MSCI Synchronous/ Address Register 0 Channel 0: SA1 Channel 0	33H	32H	
MSCI Idle Pattern Register Channel 0: IDL Channel 0	34H	35H	
MSCI Time Constant Register Channel 0: TMC Channel 0	35H	34H	

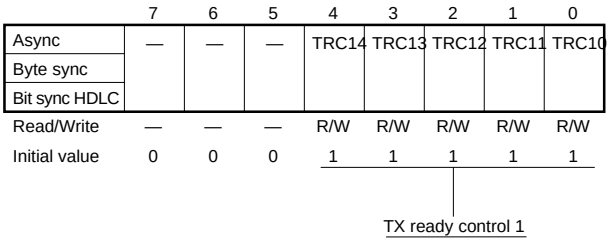
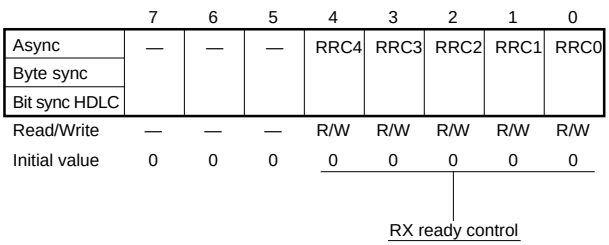
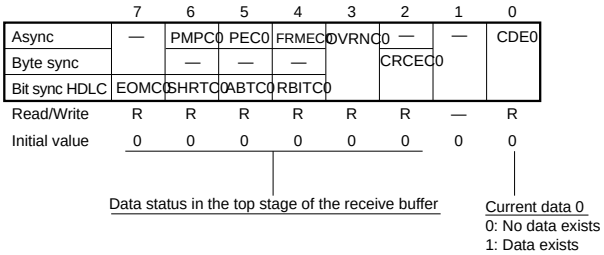
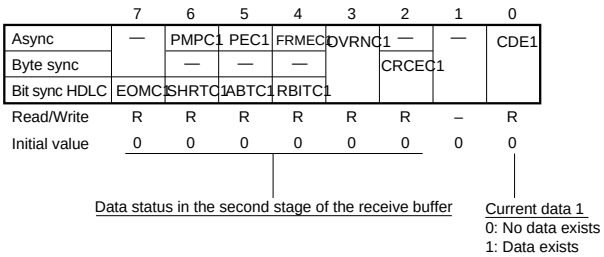
• Byte synchronous mode

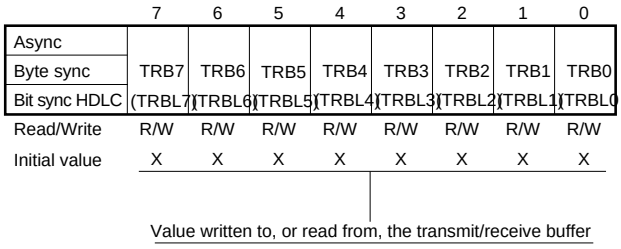
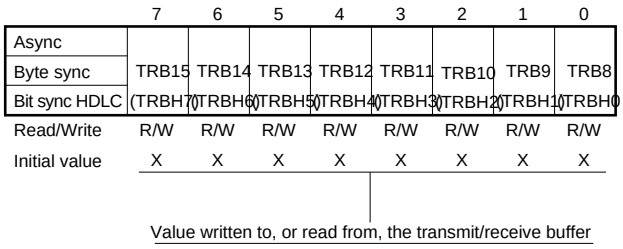
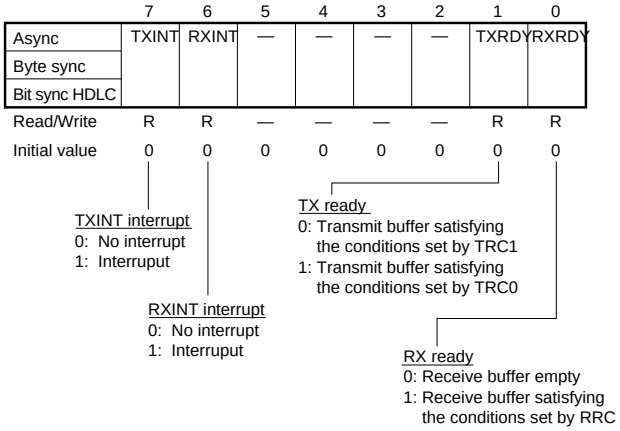
Mono-sync	SYN pattern for transmission
Bi-sync	SYN pattern for transmission and reception (bit15–bit8)
External-sync	SYN pattern for transmission

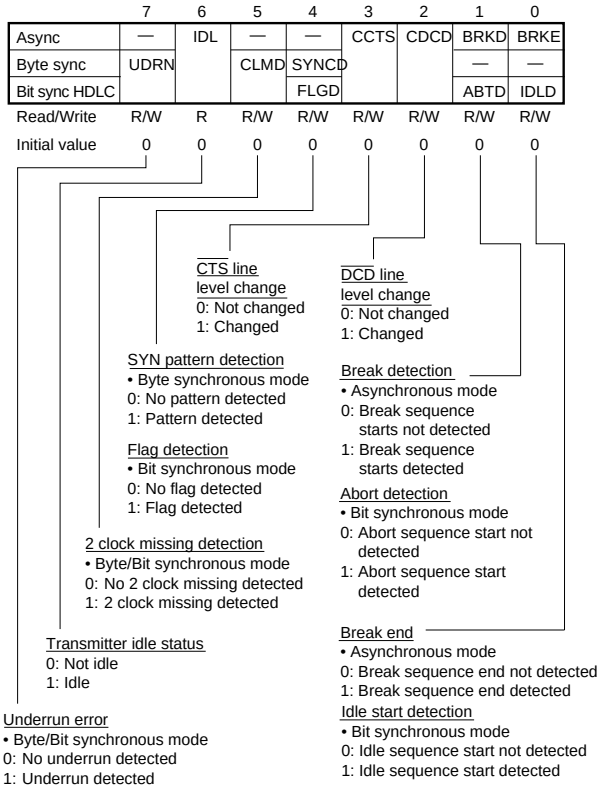
• Bit synchronous mode

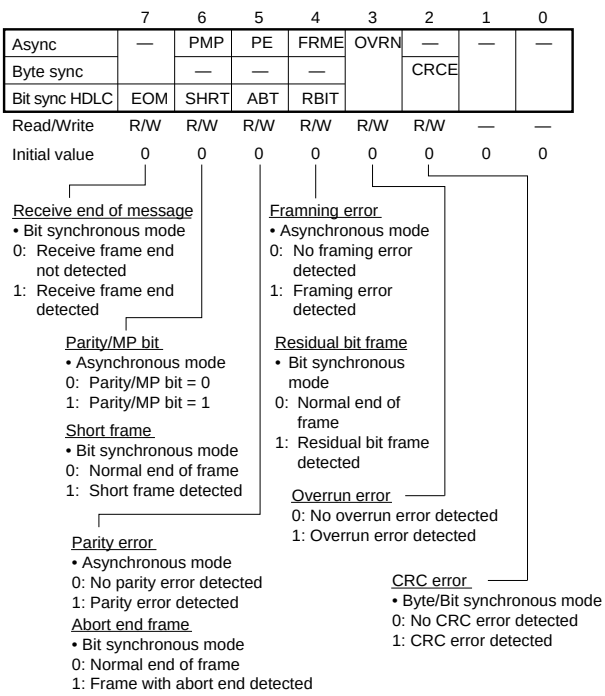
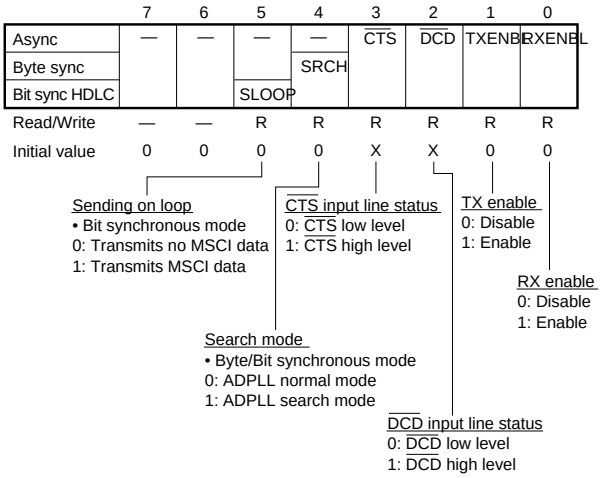
HDLC mode	No address field checked	Not used
	Single address 1	Not used
	Single address 2	Bit15–bit8 of the secondary station address
	Dual address	Bit15–bit8 of the secondary station address

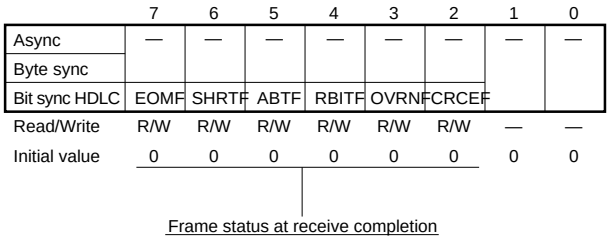
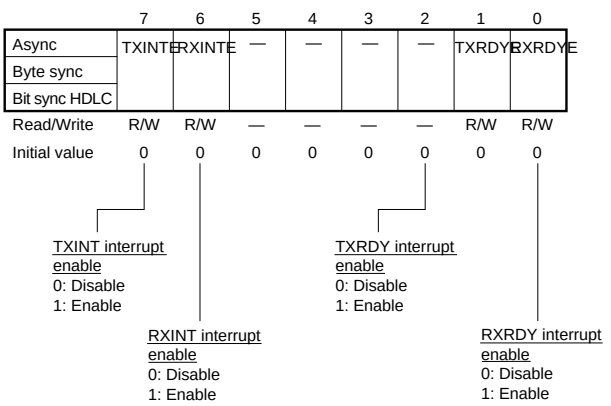
Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 0)			
MSCI RX Clock Source Register Channel 0: RXS Channel 0	36H	37H	
MSCI TX Clock Source Register Channel 0: TXS Channel 0	37H	36H	
MSCI TX Ready Control Register 0 Channel 0: TRC0 Channel 0	38H	39H	

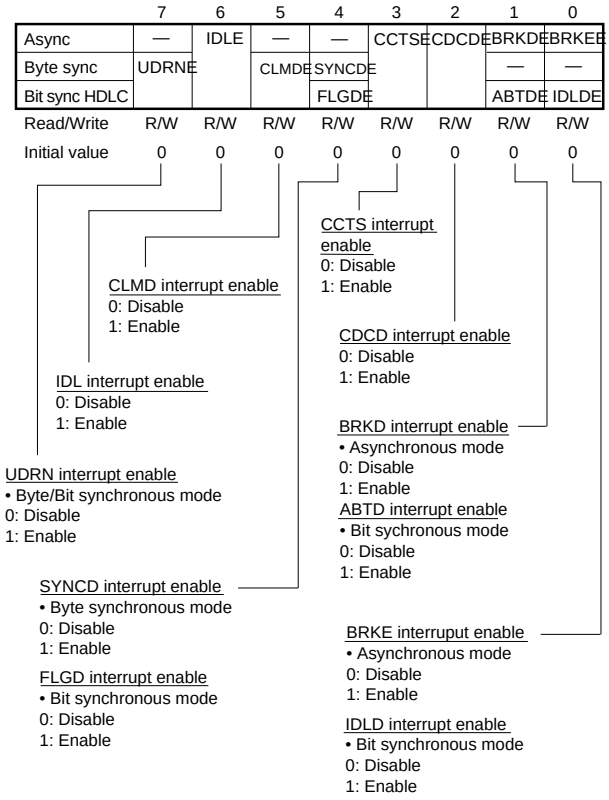
Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 0)			
MSCI TX Ready Control Register 1 Channel 0: TRC1 Channel 0	39H	38H	
MSCI RX Ready Control Register Channel 0: RRC Channel 0	3AH	3BH	
Not used	3BH	3AH	
MSCI RX Ready Control Register Channel 0: RRC Channel 0	3CH	3DH	
MSCI Current Status Register 1 Channel 0: CST1 Channel 0	3DH	3CH	
Not used	3EH	3FH	
Not used	3FH	3EH	

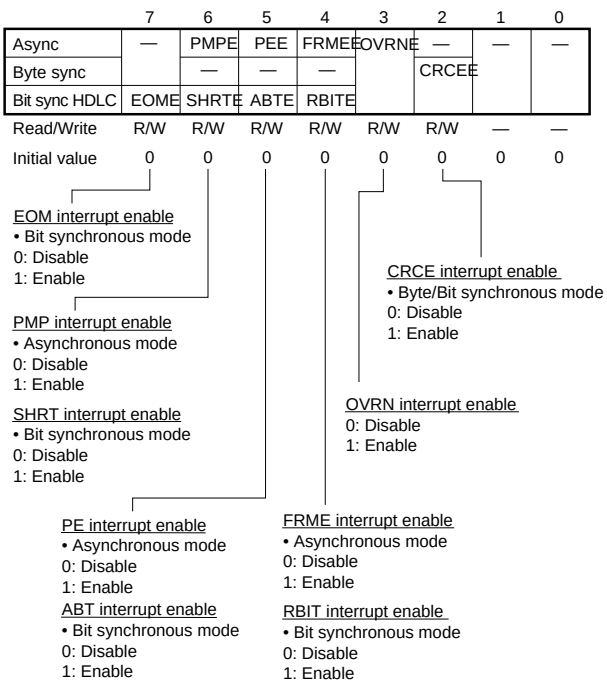
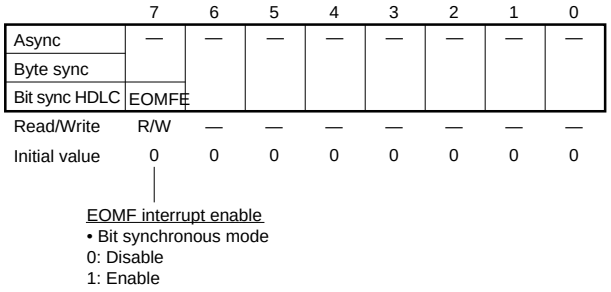
Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 1)			
MSCI TX/RX Buffer Register L Channel 1: TRBL Channel 1	40H	41H	
MSCI TX/RX Buffer Register H Channel 1: TRBH Channel 1	41H	40H	
MSCI Status Register 0 Channel 1: ST0 Channel 1	42H	43H	

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 1)			
MSCI Status Register 1 Channel 1: ST1 Channel 1	43H	42H	

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 1)			
MSCI Status Register 2 Channel 1: ST2 Channel 1	44H	45H	
SCI Status Register 3 Channel 1: ST3 Channel 1	45H	44H	

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 1)			
MSCI Frame Status Register Channel 1: FST Channel 1	46H	47H	
Not used	47H	46H	
MSCI Interrupt Enable Register 0 Channel 1: IE0 Channel 1	48H	49H	

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 1)			
MSCI Interrupt Enable Register 1 Channel 1: IE1 Channel 1	49H	48H	

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 1)			
MSCI Interrupt Enable Register 2 Channel 1: IE2 Channel 1	4AH	4BH	
MSCI Frame Interrupt Enable Register Channel 1: FIE Channel 1	4BH	4AH	

Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
MSCI (Channel 1)			
MSCI Command Register Channel 1: CMD Channel 1	4CH	4DH	
Not used	4DH	4CH	
MSCI Mode Register 0 Channel 1: MD0 Channel 1	4EH	4FH	

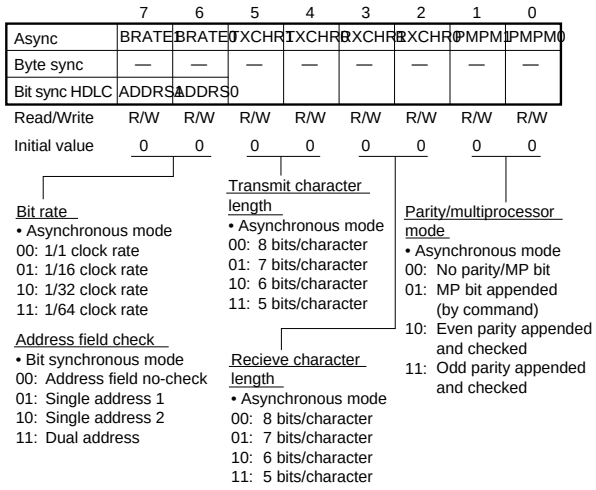
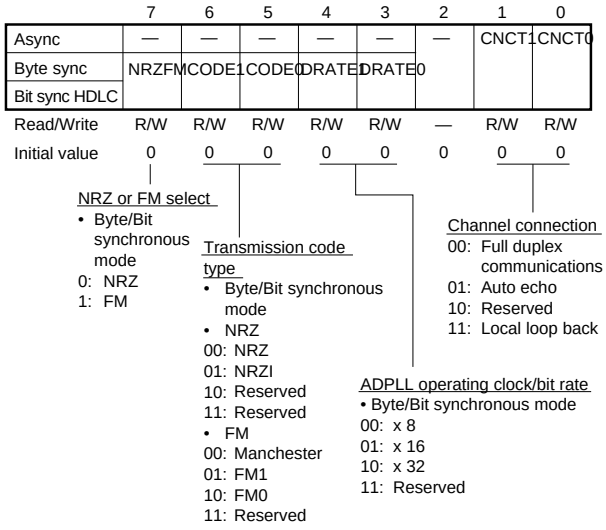
	7	6	5	4	3	2	1	0
Async / Byte sync / Bit sync HDLC	—*1	—*1	CMD5	CMD4	CMD3	CMD2	CMD1	CMD0
Read/Write	—	—	W	W	W	W	W	W
Initial value	—	—	—	—	—	—	—	—

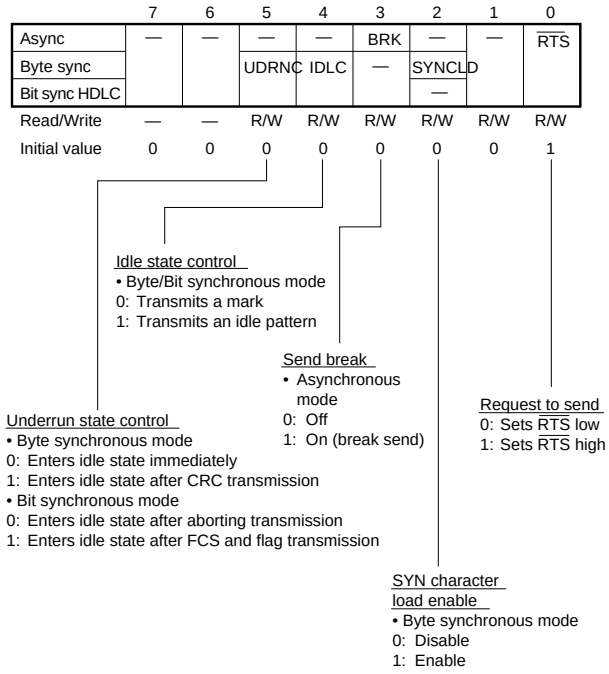
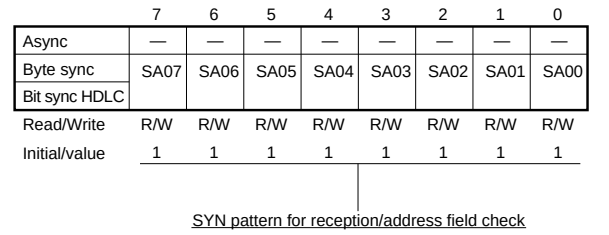
Command

• Transmit commands
000001: TX reset
000010: TX enable
000011: TX disable
000100: TX CRC initialization
000101: TX CRC calculation exclusion
000110: End-of-message
000111: Abort transmission
001000: MP bit on
001001: TX buffer clear
Others: Reserved

• Receive commands
010001: RX reset
010010: RX enable
010011: RX disable
010100: RX CRC initialization
010101: Message reject
010110: Search MP bit
010111: RX CRC calculation exclusion
011000: Forcing RX CRC calculation

• Other commands
100001: Channel reset
110001: Enter search mode
000000: No operation

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 1)			
MSCI Mode Register 1 Channel 1: MD1 Channel 1	4FH	4EH	
MSCI Mode Register 2 Channel 1: MD2 Channel 1	50H	51H	

Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
MSCI (Channel 1)			
MSCI Control Register Channel 1: CTL Channel 1	51H	50H	
MSCI Synchronous/ Address Register 0 Channel 1: SA0 Channel 1	52H	53H	

• Byte synchronous mode

Mono-sync	SYN pattern for reception
Bi-sync	SYN pattern for transmission and reception (bit7–bit0)
External-sync	Not used

• Bit synchronous mode

HDLC mode	No address field checked	Not used
	Single address 1	Bit7–bit0 of the secondary station address
	Single address 2	Not used
	Dual address	Bit7–bit0 of the secondary station address

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 1)			
MSCI Synchronous/ Address Register 0 Channel 1: SA1 Channel 1	53H	52H	

	7	6	5	4	3	2	1	0
Async	—	—	—	—	—	—	—	—
Byte sync	SA17	SA16	SA15	SA14	SA13	SA12	SA11	SA10
Bit sync HDLC								
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	1	1	1	1	1	1	1	1

SYN pattern for transmission/address field check

• Byte synchronous mode

Mono-sync	SYN pattern for transmission
Bi-sync	SYN pattern for transmission and reception (bit15–bit8)
External-sync	SYN pattern for transmission

• Bit synchronous mode

HDLC mode	No address field checked	Not used
	Single address 1	Not used
	Single address 2	Bit15–bit8 of the secondary station address
	Dual address	Bit15–bit8 of the secondary station address

MSCI Idle Pattern Register Channel 1: IDL Channel 1 — 54H / 55H

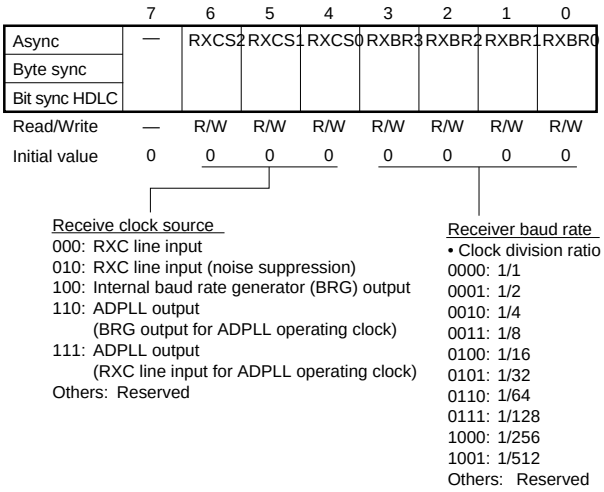
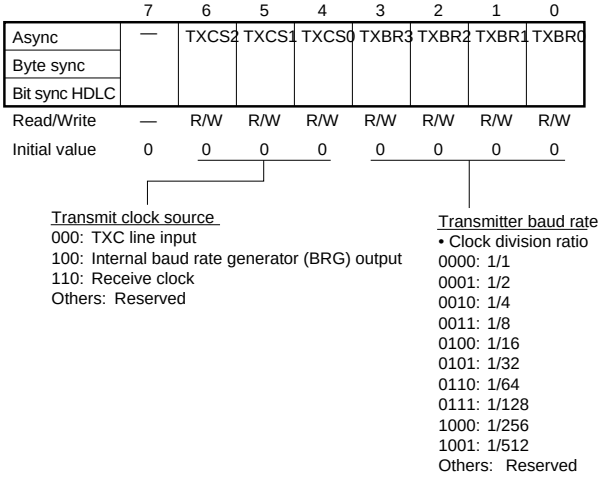
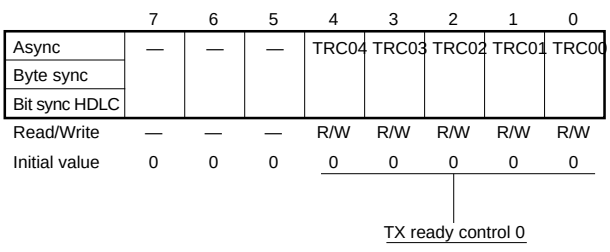
	7	6	5	4	3	2	1	0
Async	—	—	—	—	—	—	—	—
Byte sync	IDL7	IDL6	IDL5	IDL4	IDL3	IDL2	IDL1	IDL0
Bit sync HDLC								
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	1	1	1	1	1	1	1	1

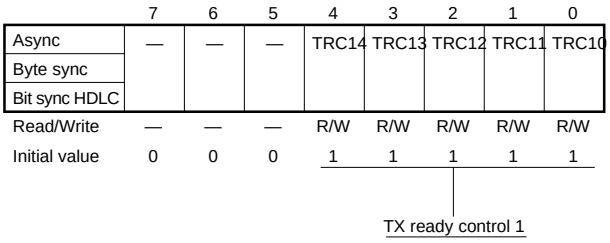
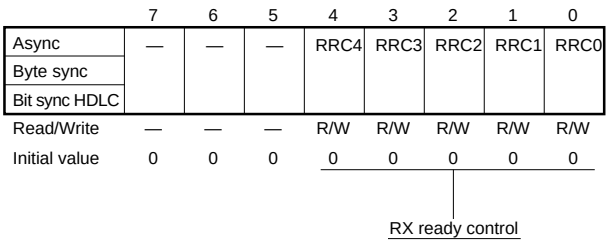
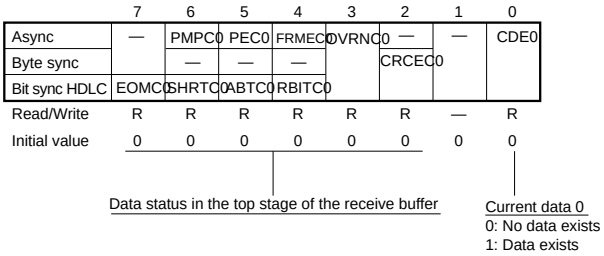
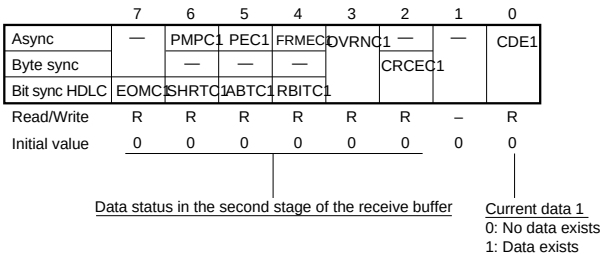
Idle pattern

MSCI Time Constant Register Channel 1: TMC Channel 1 — 55H / 54H

	7	6	5	4	3	2	1	0
Async	TMC7	TMC6	TMC5	TMC4	TMC3	TMC2	TMC1	TMC0
Byte sync								
Bit sync HDLC								
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	1

Value loaded into the reload timer (1–256)

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 1)			
MSCI RX Clock Source Register Channel 1: RXS Channel 1	56H	57H	
MSCI TX Clock Source Register Channel 1: TXS Channel 1	57H	56H	
MSCI TX Ready Control Register 0 Channel 1: TRC0 Channel 1	58H	59H	

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
MSCI (Channel 1)			
MSCI TX Ready Control Register 1 Channel 1: TRC1 Channel 1	59H	58H	
MSCI RX Ready Control Register Channel 1: RRC Channel 1	5AH	5BH	
Not used	5BH	5AH	
MSCI Current Status Register 0 Channel 1: CST0 Channel 1	5CH	5DH	
MSCI Current Status Register 1 Channel 1: CST1 Channel 1	5DH	5CH	
Not used	5EH	5FH	
Not used	5FH	5EH	

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
Timer (Channel 0)			
Timer up-counter Channel 0: TCNTL Channel 0	60H	61H	
Timer up-counter Channel 0: TCNTH Channel 0	61H	60H	
Timer Constant Register Channel 0: TCONRL Channel 0	62H	63H	
Timer Constant Register Channel 0: TCONRH Channel 0	63H	62H	
Timer Control/Status Register Channel 0: TCSR Channel 0	64H	65H	

TCNTL Channel 0

	7	6	5	4	3	2	1	0
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

TCNTH Channel 0

	15	14	13	12	11	10	9	8
Read/Write	R/W	R/W	R/W	R/W	R/W	R/W	R/W	R/W
Initial value	0	0	0	0	0	0	0	0

TCONRH Channel 0

	15	14	13	12	11	10	9	8
Read/Write	W	W	W	W	W	W	W	W
Initial value	1	1	1	1	1	1	1	1

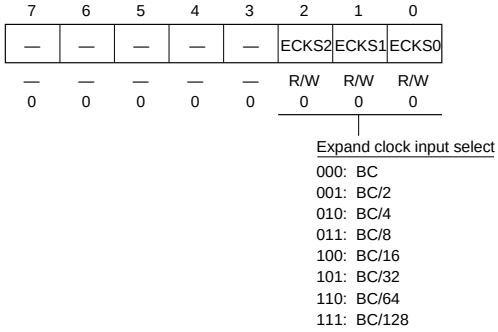
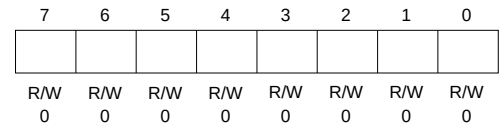
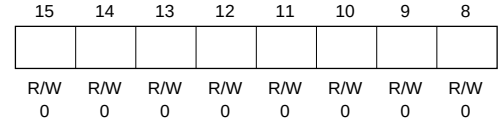
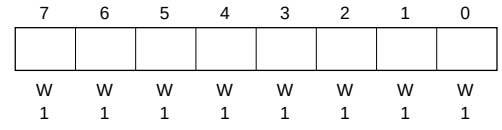
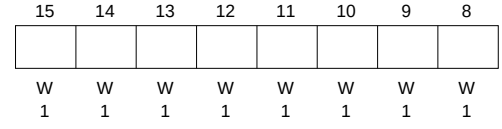
TCSR Channel 0

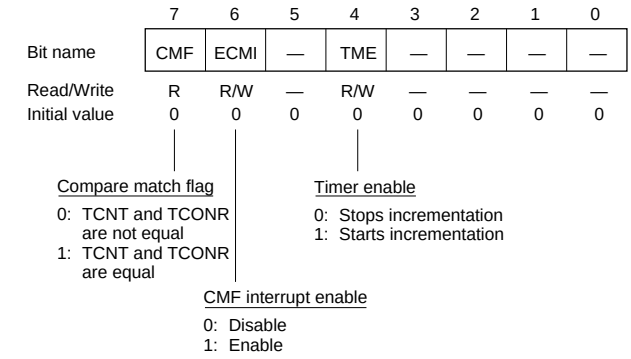
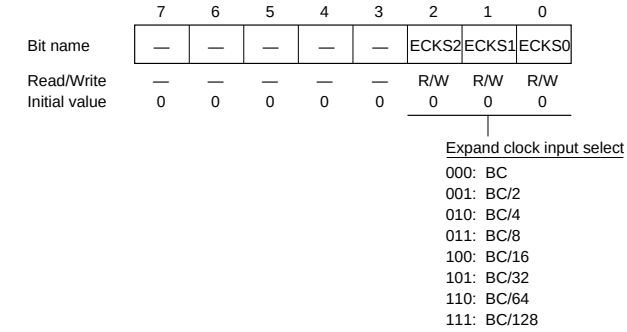
	7	6	5	4	3	2	1	0
Bit name	CMF	ECMI	—	TME	—	—	—	—
Read/Write	R	R/W	—	R/W	—	—	—	—
Initial value	0	0	0	0	0	0	0	0

Compare match flag
0: TCNT and TCONR are not equal
1: TCNT and TCONR are equal

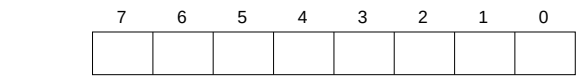
CMF interrupt enable
0: Disable
1: Enable

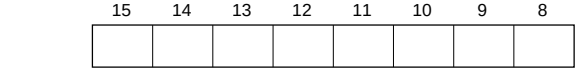
Timer enable
0: Stops incrementation
1: Starts incrementation

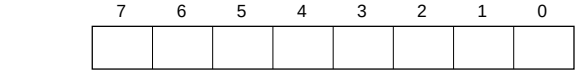
Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
Timer (Channel 0)			
Timer Expand Prescale Register Channel 0: TEPR Channel 0	65H	64H	
Not used	66H	67H	
Not used	67H	66H	
Timer (Channel 1)			
Timer up-counter Channel 1: TCNTL Channel 1	68H	69H	
Timer up-counter Channel 1: TCNTH Channel 1	69H	68H	
Timer Constant Register Channel 1: TCONRL Channel 1	6AH	6BH	
Timer Constant Register Channel 1: TCONRH Channel 1	6BH	6AH	

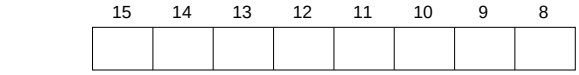
Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
Timer (Channel 1)			
Timer Control/Status Register Channel 1: TCSR Channel 1	6CH	6DH	
Timer Expand Prescale Register Channel 1: TEPR Channel 1	6DH	6CH	
Not used	6EH	6FH	
Not used	6FH	6EH	

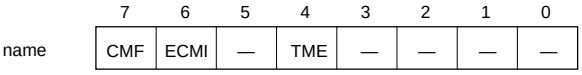
Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
Timer (Channel 2)			
Timer up-counter Channel 2: TCNTL Channel 2	70H	71H	Read/write: R/W; Initial value: 0
Timer up-counter Channel 2: TCNTH Channel 2	71H	70H	Read/Write: R/W; Initial value: 0
Timer Constant Register Channel 2: TCONRL Channel 2	72H	73H	Read/Write: W; Initial value: 1
Timer Constant Register Channel 2: TCONRH Channel 2	73H	72H	Read/write: W; Initial value: 1
Timer Control/Status Register Channel 2: TCSR Channel 2	74H	75H	Bit name: CMF, ECMI, —, TME, —, —, —, — Read/Write: R, R/W, —, R/W, —, —, —, — Initial value: 0, 0, 0, 0, 0, 0, 0, 0 Compare match flag 0: TCNT and TCONR are not equal 1: TCNT and TCONR are equal CMF interrupt enable 0: Disable 1: Enable Timer enable 0: Stops incrementation 1: Starts incrementation

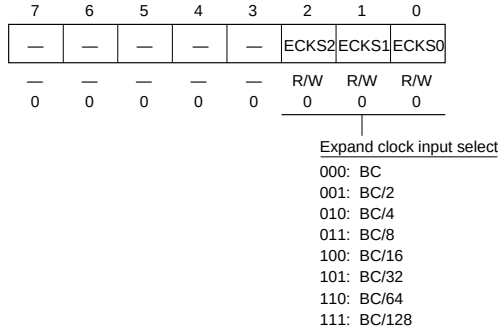
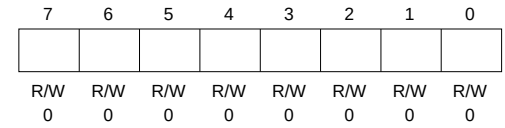
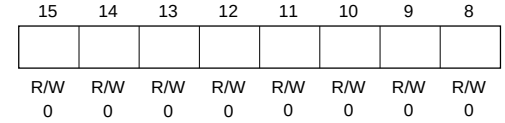
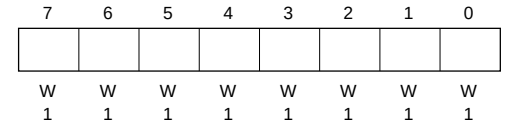
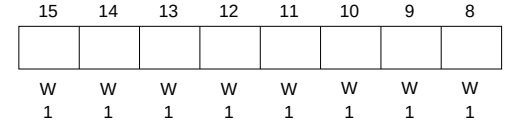


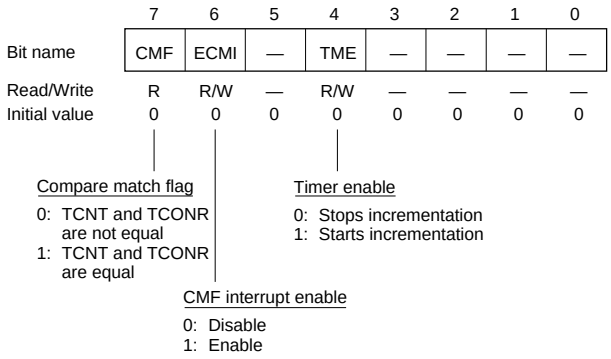
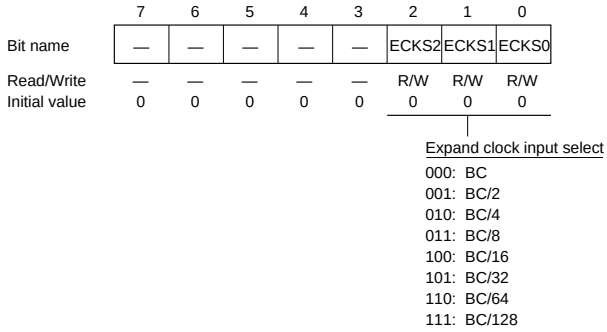


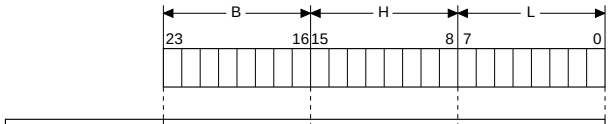
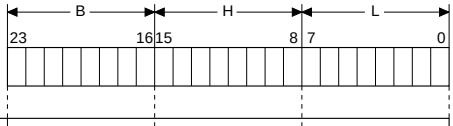


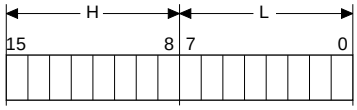
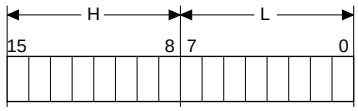
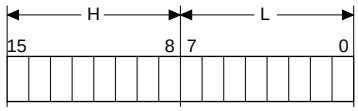


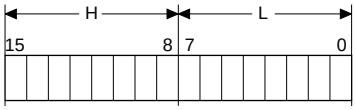
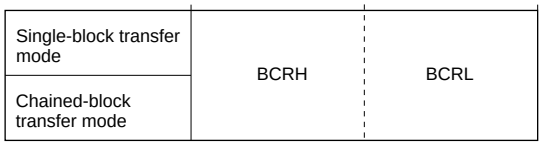
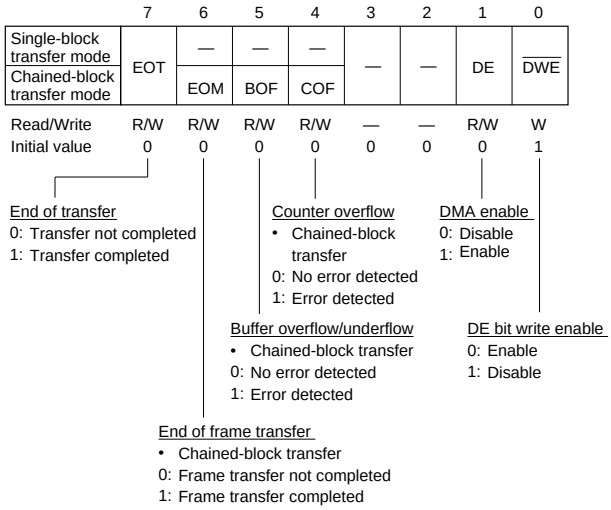


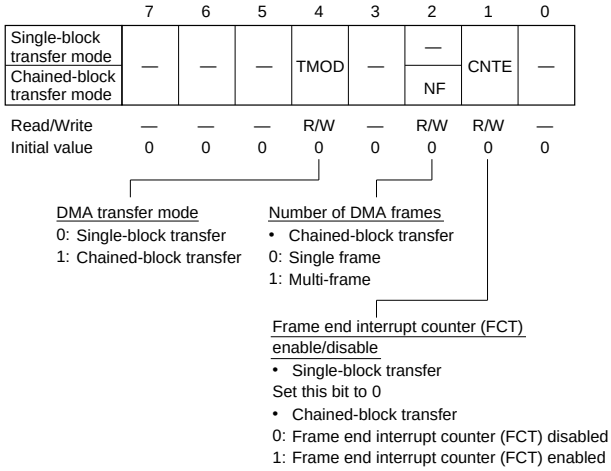
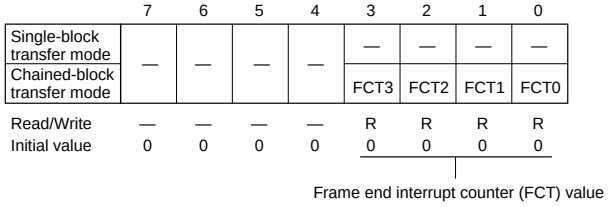
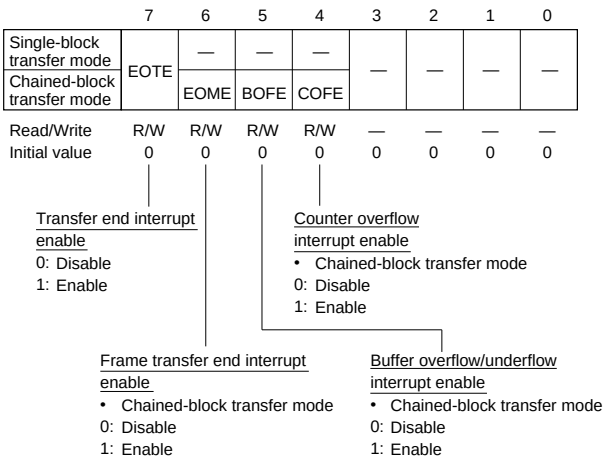
Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
Timer (Channel 2)			
Timer Expand Prescale Register Channel 2: TEPR Channel 2	75H	74H	
Not used	76H	77H	
Not used	77H	76H	
Timer (Channel 3)			
Timer up-counter Channel 3: TCNTL Channel 3	78H	79H	
Timer up-counter Channel 3: TCNTH Channel 3	79H	78H	
Timer Constant Register Channel 3: TCONRL Channel 3	7AH	7BH	
Timer Constant Register Channel 3: TCONRH Channel 3	7BH	7AH	

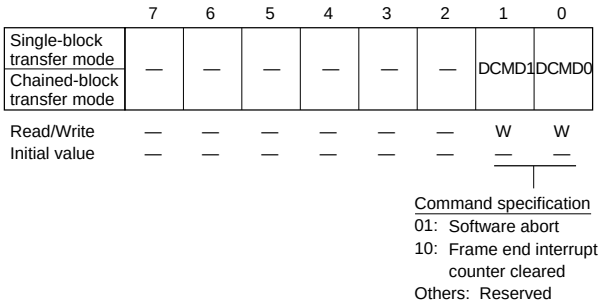
Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
Timer (Channel 3)			
Timer Control/Status Register Channel 3: TCSR Channel 3	7CH	7DH	
Timer Expand Prescale Register Channel 3: TEPR Channel 3	7DH	7CH	
Not used	7EH	7FH	
Not used	7FH	7EH	

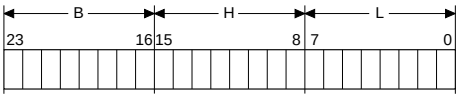
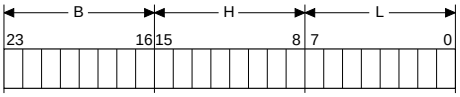
Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
DMAC (Channel 0)			
Destination Address Register Channel 0: DARL Channel 0 Buffer Address Register Channel 0: BARL Channel 0	80H	81H	 Single-block transfer mode: DARB / DARH / DARL Chained-block transfer mode: BARB / BARH / BARL
Destination Address Register Channel 0: DARH Channel 0 Buffer Address Register Channel 0: BARH Channel 0	81H	80H	
Destination Address Register Channel 0: DARB Channel 0 Buffer Address Register Channel 0: BARB Channel 0	82H	83H	
Not used	83H	82H	
Not used	84H	85H	
Not used	85H	84H	
Chain Pointer Base Channel 0: CPB Channel 0	86H	87H	Single-block transfer mode: Not used / Not used / Not used Chained-block transfer mode: CPB / Not used / Not used
Not used	87H	86H	

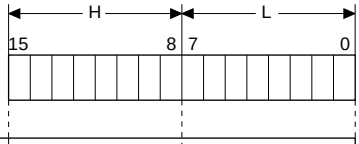
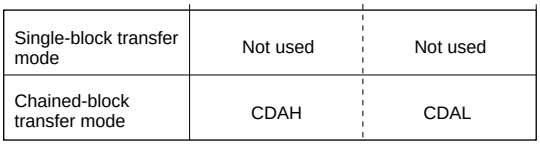
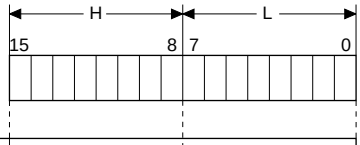
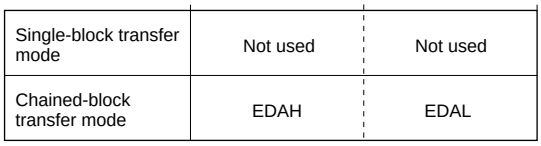
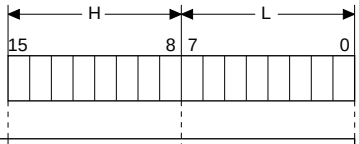
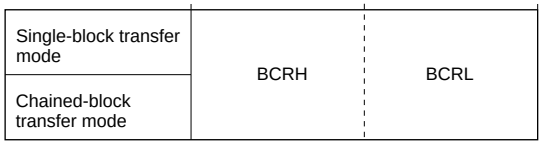
Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
DMAC (Channel 0)			
Current Descriptor Address Register Channel 0: CDAL Channel 0	88H	89H	
Current Descriptor Address Register Channel 0: CDAH Channel 0	89H	88H	Single-block transfer mode: Not used / Not used Chained-block transfer mode: CDAH / CDAL
Error Descriptor Address Register Channel 0: EDAL Channel 0	8AH	8BH	
Error Descriptor Address Register Channel 0: EDAH Channel 0	8BH	8AH	Single-block transfer mode: Not used / Not used Chained-block transfer mode: EDAH / EDAL
Receive Buffer Length Register Channel 0: BFLL Channel 0	8CH	8DH	
Receive Buffer Length Register Channel 0: BFLH Channel 0	8DH	8CH	Single-block transfer mode: Not used / Not used Chained-block transfer mode, Memory to MSCI: Not used / Not used Chained-block transfer mode, MSCI to memory: BFLH / BFLL

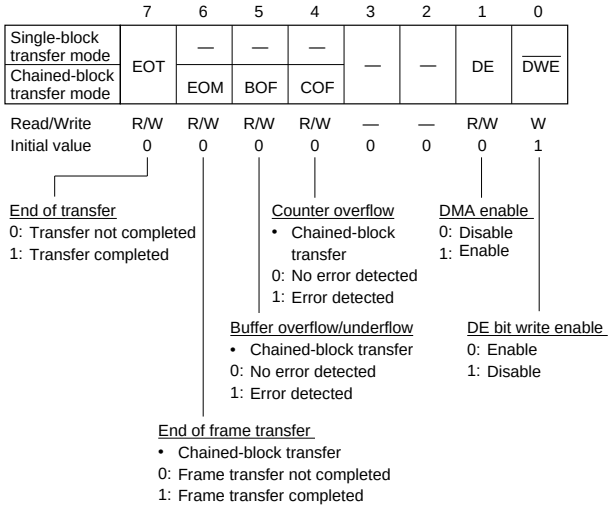
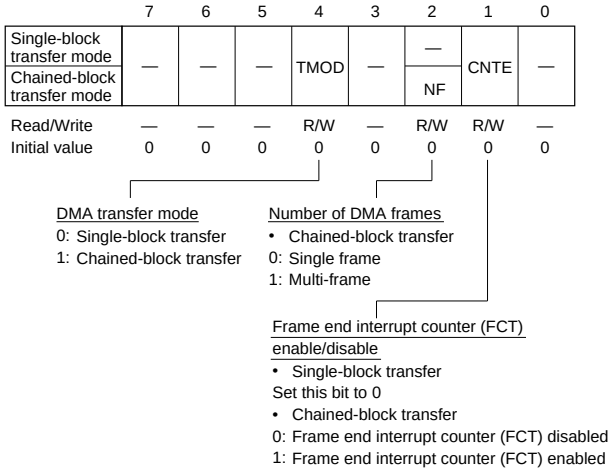
Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
DMAC (Channel 0)			
Byte Count Register Channel 0: BCRL Channel 0	8EH	8FH	
Byte Count Register Channel 0: BCRH Channel 0	8FH	8EH	
DMA Status Register Channel 0: DSR Channel 0	90H	91H	

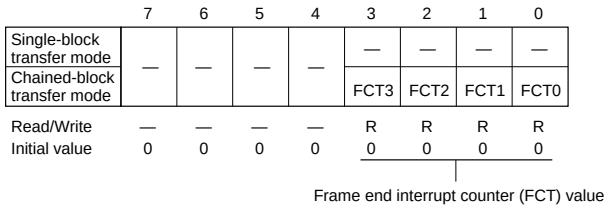
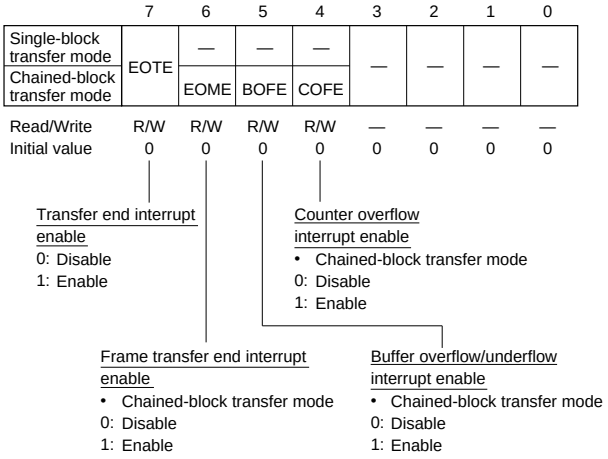
Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
DMAC (Channel 0)			
DMA Mode Register Channel 0: DMR Channel 0	91H	90H	
Not used	92H	93H	
Frame End Interrupt Counter Channel 0: FCT Channel 0	93H	92H	
DMA Interrupt Enable Register Channel 0: DIR Channel 0	94H	95H	

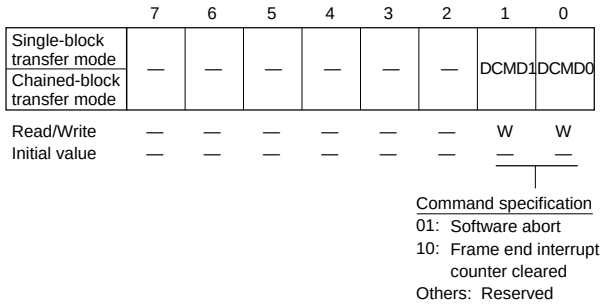
Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
DMAC (Channel 0)			
DMA Command Register Channel 0: DCR Channel 0	95H	94H	
Not used	96H	97H	
Not used	97H	96H	
Not used	98H	99H	
Not used	99H	98H	
Not used	9AH	9BH	
Not used	9BH	9AH	
Not used	9CH	9DH	
Not used	9DH	9CH	
Not used	9EH	9FH	
Not used	9FH	9EH	

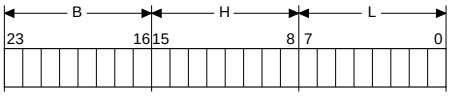
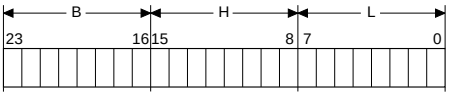
Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
DMAC (Channel 1)			
Buffer Address Register Channel 1: BARL Channel 1	A0H	A1H	
Buffer Address Register Channel 1: BARH Channel 1	A1H	A0H	Single-block transfer mode: Not used / Not used / Not used Chained-block transfer mode: BARB / BARH / BARL
Buffer Address Register Channel 1: BARB Channel 1	A2H	A3H	
Not used	A3H	A2H	
Source Address Register Channel 1: SARL Channel 1 (Chain Pointer Base Channel 1: CPB Channel 1)	A4H	A5H	
Source Address Register Channel 1: SARH Channel 1 (Chain Pointer Base Channel 1: CPB Channel 1)	A5H	A4H	Single-block transfer mode: SARB / SARH / SARL Chained-block transfer mode: CPB / Not used / Not used
Source Address Register Channel 1: SARB Channel 1 (Chain Pointer Base Channel 1: CPB Channel 1)	A6H	A7H	
Not used	A7H	A6H	

Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
DMAC (Channel 1)			
Current Descriptor Address Register Channel 1: CDAL Channel 1	A8H	A9H	
Current Descriptor Address Register Channel 1: CDAH Channel 1	A9H	A8H	
Error Descriptor Address Register Channel 1: EDAL Channel 1	AAH	ABH	
Error Descriptor Address Register Channel 1: EDAH Channel 1	ABH	AAH	
Not used	ACH	ADH	
Not used	ADH	ACH	
Byte Count Register Channel 1: BCRL Channel 1	AEH	AFH	
Byte Count Register Channel 1: BCRH Channel 1	AFH	AEH	

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
DMAC (Channel 1)			
DMA Status Register Channel 1: DSR Channel 1	B0H	B1H	
DMA Mode Register Channel 1: DMR Channel 1	B1H	B0H	
Not used	B2H	B3H	

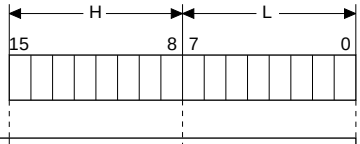
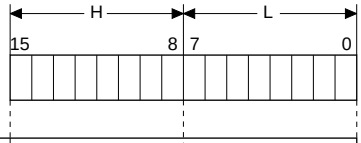
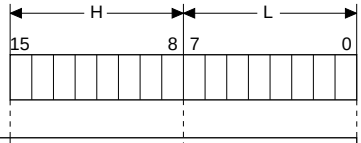
Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
DMAC (Channel 1)			
Frame End Interrupt Counter Channel 1: FCT Channel 1	B3H	B2H	
DMA Interrupt Enable Register Channel 1: DIR Channel 1	B4H	B5H	

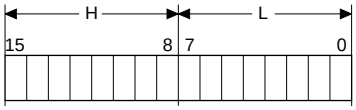
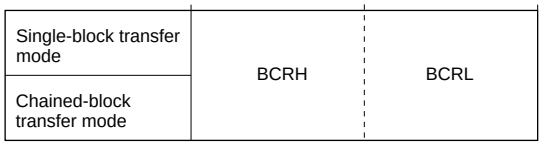
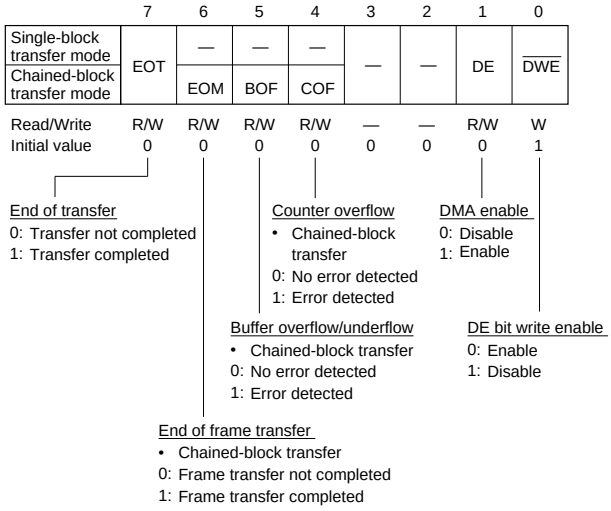
Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
DMAC (Channel 1)			
DMA Command Register Channel 1: DCR Channel 1	B5H	B4H	
Not used	B6H	B7H	
Not used	B7H	B6H	
Not used	B8H	B9H	
Not used	B9H	B8H	
Not used	BAH	BBH	
Not used	BBH	BAH	
Not used	BCH	BDH	
Not used	BDH	BCH	
Not used	BEH	BFH	
Not used	BFH	BEH	

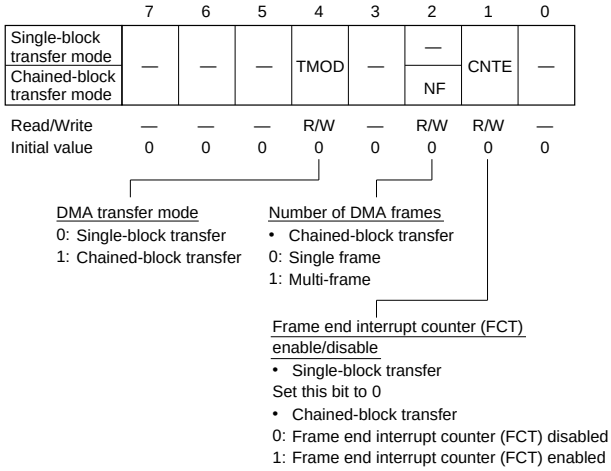
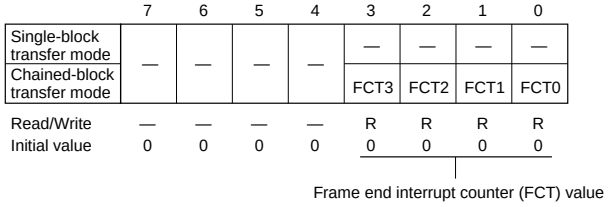
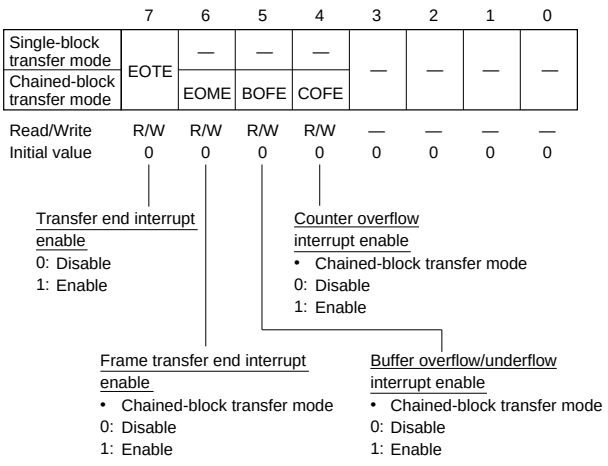
Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
DMAC (Channel 2)			
Destination Address Register Channel 2: DARL Channel 2 Buffer Address Register Channel 2: BARL Channel 2	C0H	C1H	
Destination Address Register Channel 2: DARH Channel 2 Buffer Address Register Channel 2: BARH Channel 2	C1H	C0H	
Destination Address Register Channel 2: DARB Channel 2 Buffer Address Register Channel 2: BARB Channel 2	C2H	C3H	
Not used	C3H	C2H	
Not used	C4H	C5H	
Not used	C5H	C4H	
Chain Pointer Base Channel 2: CPB Channel 2	C6H	C7H	
Not used	C7H	C6H	

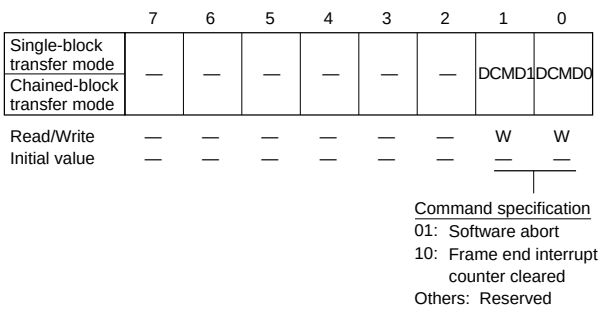
Single-block transfer mode	DARB	DARH	DARL
Chained-block transfer mode	BARB	BARH	BARL

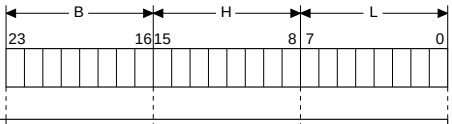
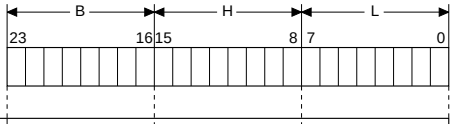
Single-block transfer mode	Not used	Not used	Not used
Chained-block transfer mode	CPB	Not used	Not used

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
DMAC (Channel 2)			
Current Descriptor Address Register Channel 2: CDAL Channel 2	C8H	C9H	
Current Descriptor Address Register Channel 2: CDAH Channel 2	C9H	C8H	Single-block transfer mode: Not used / Not used Chained-block transfer mode: CDAH / CDAL
Error Descriptor Address Register Channel 2: EDAL Channel 2	CAH	CBH	
Error Descriptor Address Register Channel 2: EDAH Channel 2	CBH	CAH	Single-block transfer mode: Not used / Not used Chained-block transfer mode: EDAH / EDAL
Receive Buffer Length Register Channel 2: BFLL Channel 2	CCH	CDH	
Receive Buffer Length Register Channel 2: BFLH Channel 2	CDH	CCH	Single-block transfer mode: Not used / Not used Chained-block transfer mode, Memory to MSCI: Not used / Not used Chained-block transfer mode, MSCI to memory: BFLH / BFLL

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
DMAC (Channel 2)			
Byte Count Register Channel 2: BCRL Channel 2	CEH	CFH	
Byte Count Register Channel 2: BCRH Channel 2	CFH	CEH	
DMA Status Register Channel 2: DSR Channel 2	D0H	D1H	

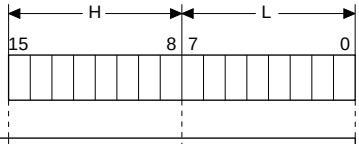
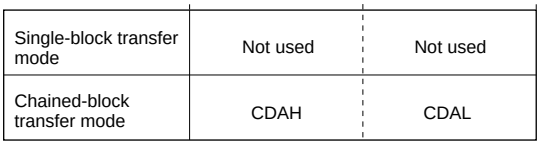
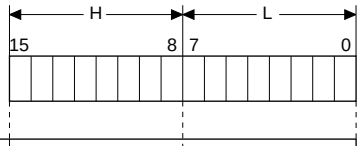
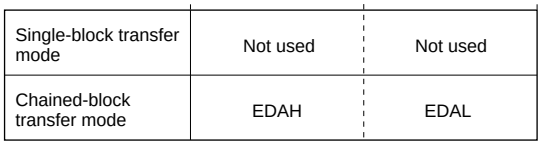
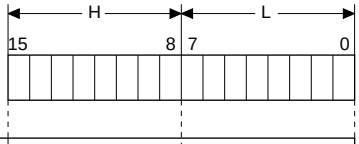
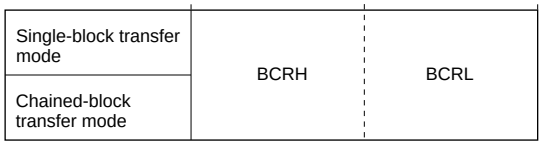
Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
DMAC (Channel 2)			
DMA Mode Register Channel 2: DMR Channel 2	D1H	D0H	
Not used	D2H	D3H	
Frame End Interrupt Counter Channel 2: FCT Channel 2	D3H	D2H	
DMA Interrupt Enable Register Channel 2: DIR Channel 2	D4H	D5H	

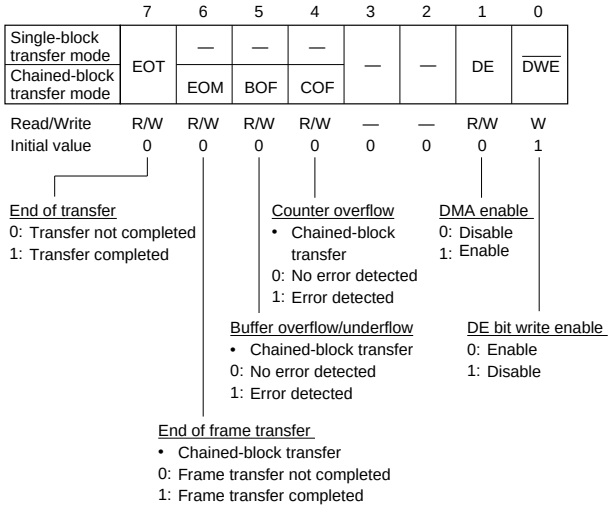
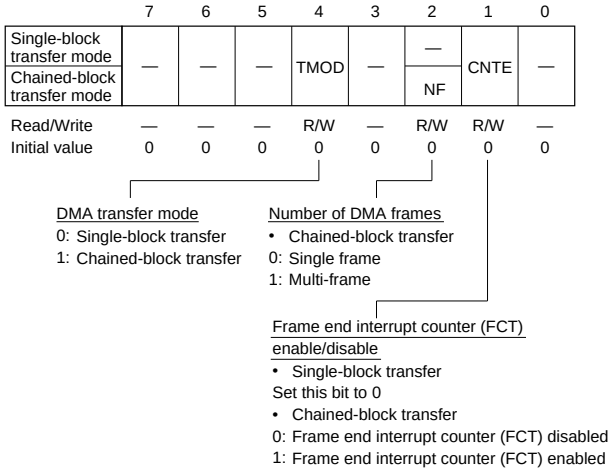
Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
DMAC (Channel 2)			
DMA Command Register Channel 2: DCR Channel 2	D5H	D4H	
Not used	D6H	D7H	
Not used	D7H	D6H	
Not used	D8H	D9H	
Not used	D9H	D8H	
Not used	DAH	DBH	
Not used	DBH	DAH	
Not used	DCH	DDH	
Not used	DDH	DCH	
Not used	DEH	DFH	
Not used	DFH	DEH	

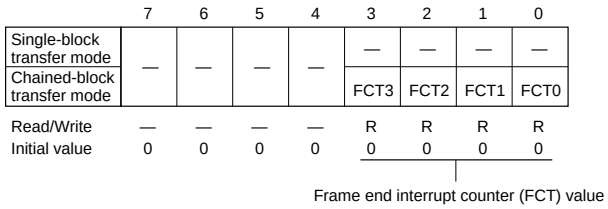
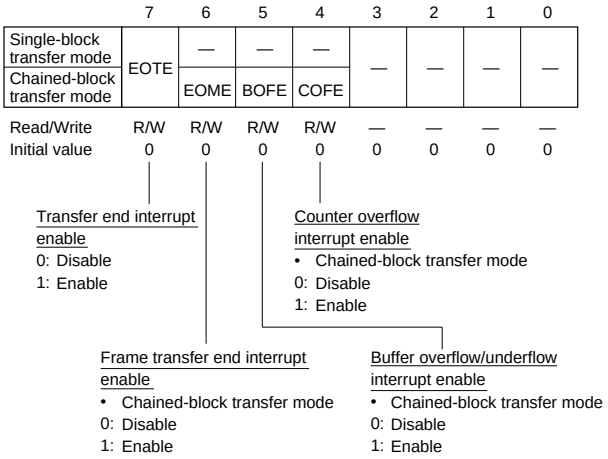
Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
DMAC (Channel 3)			
Buffer Address Register Channel 3: BARL Channel 3	E0H	E1H	
Buffer Address Register Channel 3: BARH Channel 3	E1H	E0H	
Buffer Address Register Channel 3: BARB Channel 3	E2H	E3H	
Not used	E3H	E2H	
Source Address Register Channel 3: SARL Channel 3 (Chain Pointer Base Channel 3: CPB Channel 3)	E4H	E5H	
Source Address Register Channel 3: SARH Channel 3 (Chain Pointer Base Channel 3: CPB Channel 3)	E5H	E4H	
Source Address Register Channel 3: SARB Channel 3 (Chain Pointer Base Channel 3: CPB Channel 3)	E6H	E7H	
Not used	E7H	E6H	

Single-block transfer mode	Not used	Not used	Not used
Chained-block transfer mode	BARB	BARH	BARL

Single-block transfer mode	SARB	SARH	SARL
Chained-block transfer mode	CPB	Not used	Not used

Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
DMAC (Channel 3)			
Current Descriptor Address Register Channel 3: CDAL Channel 3	E8H	E9H	
Current Descriptor Address Register Channel 3: CDAH Channel 3	E9H	E8H	
Error Descriptor Address Register Channel 3: EDAL Channel 3	EAH	EBH	
Error Descriptor Address Register Channel 3: EDAH Channel 3	EBH	EAH	
Not used	ECH	EDH	
Not used	EDH	ECH	
Byte Count Register Channel 3: BCRL Channel 3:	EEH	EFH	
Byte Count Register Channel 3: BCRH Channel 3	EFH	EEH	

Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
DMAC (Channel 3)			
DMA Status Register Channel 3: DSR Channel 3	F0H	F1H	
DMA Mode Register Channel 3: DMR Channel 3	F1H	F0H	
Not used	F2H	F3H	

Register	Address: CPU Mode 0, 1	Address: CPU Mode 2, 3	Remarks
DMAC (Channel 3)			
Frame End Interrupt Counter Channel 3: FCT Channel 3	F3H	F2H	
DMA Interrupt Enable Register Channel 3: DIR Channel 3	F4H	F5H	

Register	Address CPU Mode 0, 1	Address CPU Mode 2, 3	Remarks
DMAC (Channel 3)			
DMA Command Register Channel 3: DCR Channel 3	F5H	F4H	See bit table below
Not used	F6H	F7H	
Not used	F7H	F6H	
Not used	F8H	F9H	
Not used	F9H	F8H	
Not used	FAH	FBH	
Not used	FBH	FAH	
Not used	FCH	FDH	
Not used	FDH	FCH	
Not used	FEH	FFH	
Not used	FFH	FEH	

	7	6	5	4	3	2	1	0
Single-block transfer mode / Chained-block transfer mode	—	—	—	—	—	—	DCMD1	DCMD0
Read/Write	—	—	—	—	—	—	W	W
Initial value	—	—	—	—	—	—	—	—

Command specification
01: Software abort
10: Frame end interrupt counter cleared
Others: Reserved

HD64570 SCA User's Manual

Publication Date: 1st Edition, September 1990
3rd Edition, August 1998
Published by: Electronic Devices Sales & Marketing Group
Semiconductor & Integrated Circuits Group
Hitachi, Ltd.
Edited by: Technical Documentation Group
UL Media Co., Ltd.