To all our customers

Regarding the change of names mentioned in the document, such as Mitsubishi Electric and Mitsubishi XX, to Renesas Technology Corp.

The semiconductor operations of Hitachi and Mitsubishi Electric were transferred to Renesas Technology Corporation on April 1st 2003. These operations include microcomputer, logic, analog and discrete devices, and memory chips other than DRAMs (flash memory, SRAMs etc.) Accordingly, although Mitsubishi Electric, Mitsubishi Electric Corporation, Mitsubishi Semiconductors, and other Mitsubishi brand names are mentioned in the document, these names have in fact all been changed to Renesas Technology Corp. Thank you for your understanding. Except for our corporate trademark, logo and corporate statement, no changes whatsoever have been made to the contents of the document, and these changes do not constitute any alteration to the contents of the document itself.

Note : Mitsubishi Electric will continue the business operations of high frequency & optical devices and power devices.

EOL announced

Renesas Technology Corp.
Customer Support Dept.
April 1, 2003

MITSUBISHI 16-BIT SINGLE-CHIP MICROCOMPUTER
7700 FAMILY / 7700 SERIES

7721 Group

User's Manual

EOL announced

RENESAS
Renesas Technology Corp.

keep safety first in your circuit designs !

- Mitsubishi Electric Corporation puts the maximum effort into making semiconductor products better and more reliable, but there is always the possibility that trouble may occur with them. Trouble with semiconductors may lead to personal injury, fire or property damage. Remember to give due consideration to safety when making your circuit designs, with appropriate measures such as (i) placement of substitutive, auxiliary circuits, (ii) use of non-flammable material or (iii) prevention against any malfunction or mishap.

Notes regarding these materials

- These materials are intended as a reference to assist our customers in the selection of the Mitsubishi semiconductor product best suited to the customer's application; they do not convey any license under any intellectual property rights, or any other rights, belonging to Mitsubishi Electric Corporation or a third party.

- Mitsubishi Electric Corporation assumes no responsibility for any damage, or infringement of any third-party's rights, originating in the use of any product data, diagrams, charts or circuit application examples contained in these materials.

- All information contained in these materials, including product data, diagrams and charts, represent information on products at the time of publication of these materials, and are subject to change by Mitsubishi Electric Corporation without notice due to product improvements or other reasons. It is therefore recommended that customers contact Mitsubishi Electric Corporation or an authorized Mitsubishi Semiconductor product distributor for the latest product information before purchasing a product listed herein.

- Mitsubishi Electric Corporation semiconductors are not designed or manufactured for use in a device or system that is used under circumstances in which human life is potentially at stake. Please contact Mitsubishi Electric Corporation or an authorized Mitsubishi Semiconductor product distributor when considering the use of a product contained herein for any specific purposes, such as apparatus or systems for transportation, vehicular, medical, aerospace, nuclear, or undersea repeater use.

- The prior written approval of Mitsubishi Electric Corporation is necessary to reprint or reproduce in whole or in part these materials.

- If these products or technologies are subject to the Japanese export control restrictions, they must be exported under a license from the Japanese government and cannot be imported into a country other than the approved destination.
  Any diversion or reexport contrary to the export control laws and regulations of JAPAN and/or the country of destination is prohibited.

- Please contact Mitsubishi Electric Corporation or an authorized Mitsubishi Semiconductor product distributor for further details on these materials or the products contained therein.

EOL announced

REVISION DESCRIPTION LIST	7721 Group User's Manual

Rev. No.	Revision Description	Rev. date
1.0	First Edition	970926

EOL announced

Preface

This manual describes the hardware of the Mitsubishi CMOS 16-bit microcomputers 7721 Group. After reading this manual, the user will be able to understand the functions, so that their capabilities can fully be utilized.

EOL announced

BEFORE USING THIS MANUAL

1. Constitution

This user's manual consists of the following chapters. Refer to the chapters relevant to the products.

- **Chapter 1. DESCRIPTION through Chapter 16. APPLICATION**
  Functions which are common to the M37721S1BFP and the M37721S2BFP are explained, using the M37721S2BFP as an example.
  Differences between the M37721S1BFP and the M37721S2BFP are described as notes.

- **Appendix**
  Practical information for using the 7721 Group is described.

2. Remark

- **Product expansion**
  Refer to the latest catalog and data book, or contact the appropriate office, as listed in "CONTACT ADDRESSES FOR FURTHER INFORMATION" on the last page.

- **Electrical characteristics**
  Refer to the latest data book.

- **Software**
  Refer to **"7700 Family Software Manual."**

- **Development support tools**
  Refer to the latest data book of the development support tools.

3. Signal levels in Figure

As a rule, signal levels in each operation example and timing diagram are as follows.

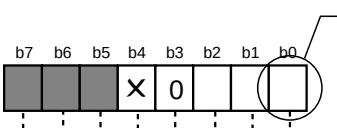
• Signal levels
  The upper line indicates "1," and the lower line indicates "0."
• Input/output levels of pin
  The upper line indicates "H," and the lower line indicates "L."

For the exception, the level is shown on the left side of a signal.

EOL announced

4. Register structure

Below is the structure diagram for all registers:



XXX register (Address XX_{16})

✻1 ✻2 ✻3

Bit	Bit name	Functions	At reset	RW
0	... select bit	0 : ... 1 : ...	0	RW
1	... select bit	0 : ... 1 : ... The value is “0” at reading.	Undefined	WO
2	... flag	0 : ... 1 : ...	0	RO
3	Fix this bit to “0.”		0	RW
4	This bit is invalid in ... mode.		0	RW
7 to 5	Nothing is assigned.		Undefined	–

✻4

✻1

Blank : Set to “0” or “1” according to the usage.
0 : Set to “0” at writing.
1 : Set to “1” at writing.
× : Invalid depending on the mode or state. It may be “0” or “1.”
■ : Nothing is assigned.

✻2

0 : “0” immediately after reset.
1 : “1” immediately after reset.
Undefined : Undefined immediately after reset.

✻3

RW : It is possible to read the bit state at reading. The written value becomes valid.
RO : It is possible to read the bit state at reading. The written value becomes invalid. Accordingly, the written value may be “0” or “1.”
WO : The written value becomes valid. It is impossible to read the bit state. The value is undefined at reading. However, when [“0” at reading”] is indicated in the “Function” or “Note” column, the bit is always “0” at reading. (See ✻4 above.)
— : It is impossible to read the bit state. The value is undefined at reading. However, when [“0” at reading”] is indicated in the “Function” or “Note” column, the bit is always “0” at reading. (See ✻4 above.)
The written value becomes invalid. Accordingly, the written value may be “0” or “1.”

EOL announced

Table of contents

CHAPTER 1 DESCRIPTION

1.1 Performance overview 1-2
1.2 Pin configuration 1-3
1.3 Pin description 1-4
1.4 Block diagram 1-7

CHAPTER 2 CENTRAL PROCESSING UNIT (CPU)

2.1 Central processing unit 2-2
2.1.1 Accumulator (Acc) 2-3
2.1.2 Index register X (X) 2-3
2.1.3 Index register Y (Y) 2-3
2.1.4 Stack pointer (S) 2-4
2.1.5 Program counter (PC) 2-5
2.1.6 Program bank register (PG) 2-5
2.1.7 Data bank register (DT) 2-5
2.1.8 Direct page register (DPR) 2-6
2.1.9 Processor status register (PS) 2-7
2.2 Bus interface unit 2-9
2.2.1 Overview 2-9
2.2.2 Functions of bus interface unit (BIU) 2-11
2.2.3 Operation of bus interface unit (BIU) 2-13
2.3 Access space 2-15
2.3.1 Banks 2-16
2.3.2 Direct page 2-16
2.4 Memory assignment 2-17
2.4.1 Memory assignment in internal area 2-17
2.4.2 External area 2-18
2.5 Bus access right 2-23

CHAPTER 3 CONNECTION WITH EXTERNAL DEVICES

3.1 Signals required for accessing external devices 3-2
3.1.1 Descriptions of signals 3-2
3.1.2 Operation of bus interface unit (BIU) 3-5
3.2 Software Wait 3-8
3.3 Ready function 3-10
3.3.1 Operation description 3-11
3.4 Hold function 3-12
3.4.1 Operation description 3-12
[Precautions for Hold function] 3-16

EOL announced

CHAPTER 4 RESET

4.1 Hardware reset 4-2
4.1.1 Pin state 4-3
4.1.2 State of CPU, SFR area, and internal RAM area 4-4
4.1.3 Internal processing sequence after reset 4-11
4.1.4 Time supplying “L” level to $\overline{\text{RESET}}$ pin 4-12
4.2 Software reset 4-13

CHAPTER 5 CLOCK GENERATING CIRCUIT

5.1 Oscillation circuit examples 5-2
5.1.1 Connection example using resonator/oscillator 5-2
5.1.2 Externally generated clock input example 5-2
5.2 Clocks 5-3
5.2.1 Clocks generated in clock generating circuit 5-4
5.3 Stop mode 5-5
5.3.1 Stop mode 5-5
[Precautions for Stop mode] 5-8
5.4 Wait mode 5-9
5.4.1 Wait mode 5-9
[Precautions for Wait mode] 5-11

CHAPTER 6 INPUT/OUTPUT PINS

6.1 Overview 6-2
6.2 Programmable I/O ports 6-2
6.2.1 Direction register 6-3
6.2.2 Port register 6-4
6.3 Examples of handling unused pins 6-7

CHAPTER 7 INTERRUPTS

7.1 Overview 7-2
7.2 Interrupt sources 7-4
7.3 Interrupt control 7-5
7.3.1 Interrupt disable flag (I) 7-7
7.3.2 Interrupt request bit 7-7
7.3.3 Interrupt priority level select bits and processor interrupt priority level (IPL) 7-7
7.4 Interrupt priority level 7-9
7.5 Interrupt priority level detection circuit 7-10
7.6 Interrupt priority level detection time 7-12
7.7 Sequence from acceptance of interrupt request until execution of interrupt routine 7-13
7.7.1 Change in IPL at acceptance of interrupt request 7-14
7.7.2 Push operation for registers 7-15
7.8 Return from interrupt routine 7-16
7.9 Multiple interrupts 7-16
7.10 External interrupts ($\overline{\text{INTi}}$ interrupt) 7-18
7.10.1 Functions of $\overline{\text{INTi}}$ interrupt request bit 7-20
7.10.2 Switching of $\overline{\text{INTi}}$ interrupt request occurrence factor 7-21
7.11 Precautions for interrupts 7-22

EOL announced

CHAPTER 8 TIMER A

8.1 Overview 8-2
8.2 Block description 8-3
8.2.1 Counter and reload register (timer Ai register) 8-4
8.2.2 Count start register 8-5
8.2.3 Timer Ai mode register 8-6
8.2.4 Timer Ai interrupt control register 8-7
8.2.5 Port P5 direction register 8-8
8.3 Timer mode 8-9
8.3.1 Setting for timer mode 8-11
8.3.2 Count source 8-13
8.3.3 Operation in timer mode 8-14
8.3.4 Selectable functions 8-15
[Precautions for timer mode] 8-17
8.4 Event counter mode 8-18
8.4.1 Setting for event counter mode 8-21
8.4.2 Operation in event counter mode 8-23
8.4.3 Switching between countup and countdown 8-24
8.4.4 Selectable functions 8-25
[Precautions for event counter mode] 8-28
8.5 One-shot pulse mode 8-29
8.5.1 Setting for one-shot pulse mode 8-31
8.5.2 Count source 8-33
8.5.3 Trigger 8-34
8.5.4 Operation in one-shot pulse mode 8-35
[Precautions for one-shot pulse mode] 8-37
8.6 Pulse width modulation (PWM) mode 8-38
8.6.1 Setting for PWM mode 8-40
8.6.2 Count source 8-42
8.6.3 Trigger 8-42
8.6.4 Operation in PWM mode 8-43
[Precautions for PWM mode] 8-47

CHAPTER 9 TIMER B

9.1 Overview 9-2
9.2 Block description 9-2
9.2.1 Counter and reload register (timer Bi register) 9-3
9.2.2 Count start register 9-4
9.2.3 Timer Bi mode register 9-5
9.2.4 Timer Bi interrupt control register 9-6
9.2.5 Port P5 direction register 9-7
9.3 Timer mode 9-8
9.3.1 Setting for timer mode 9-10
9.3.2 Count source 9-11
9.3.3 Operation in timer mode 9-12
[Precautions for timer mode] 9-13
9.4 Event counter mode 9-14
9.4.1 Setting for event counter mode 9-16
9.4.2 Operation in event counter mode 9-17
[Precautions for event coutner mode] 9-18

EOL announced

9.5 Pulse period/Pulse width measurement mode 9-19
9.5.1 Setting for pulse period/pulse width measurement mode 9-21
9.5.2 Count source 9-22
9.5.3 Operation in pulse period/pulse width measurement mode 9-23
[Precautions for pulse period/pulse width measurement mode] 9-25

CHAPTER 10 REAL-TIME OUTPUT

10.1 Overview 10-2
10.2 Block description 10-4
10.2.1 Real-time output control register 10-4
10.2.2 Pulse output data registers 0 and 1 10-5
10.2.3 Port P6 direction register 10-6
10.2.4 Timers A0 and A1 10-6
10.3 Setting of real-time output 10-7
10.4 Real-time output operation 10-10

CHAPTER 11 SERIAL I/O

11.1 Overview 11-2
11.2 Block description 11-3
11.2.1 UARTi transmit/receive mode register 11-4
11.2.2 UARTi transmit/receive control register 0 11-6
11.2.3 UARTi transmit/receive control register 1 11-7
11.2.4 UARTi transmit register and UARTi transmit buffer register 11-9
11.2.5 UARTi receive register and UARTi receive buffer register 11-11
11.2.6 UARTi baud rate register (BRGi) 11-13
11.2.7 UARTi transmit interrupt control and UARTi receive interrupt control registers 11-14
11.2.8 Port P8 direction register 11-15
11.3 Clock synchronous serial I/O mode 11-16
11.3.1 Transfer clock (Synchronizing clock) 11-17
11.3.2 Method of transmission 11-18
11.3.3 Transmit operation 11-21
11.3.4 Method of reception 11-23
11.3.5 Receive operation 11-27
11.3.6 Processing on detecting overrun error 11-29
[Precautions for clock synchronous serial I/O mode] 11-30
11.4 Clock asynchronous serial I/O (UART) mode 11-31
11.4.1 Transfer rate (Frequency of transfer clock) 11-32
11.4.2 Transfer data format 11-33
11.4.3 Method of transmission 11-34
11.4.4 Transmit operation 11-38
11.4.5 Method of reception 11-41
11.4.6 Receive operation 11-44
11.4.7 Processing on detecting error 11-46
11.4.8 Sleep mode 11-47

Not announced

CHAPTER 12 A-D CONVERTER

12.1 Overview 12-2
12.2 Block description 12-3
12.2.1 A-D control register 12-4
12.2.2 A-D sweep pin select register 12-6
12.2.3 A-D register i (i = 0 to 7) 12-7
12.2.4 A-D conversion interrupt control register 12-8
12.2.5 Port P7 direction register 12-9
12.3 A-D conversion method 12-10
12.4 Absolute accuracy and differential non-linearity error 12-12
12.4.1 Absolute accuracy 12-12
12.4.2 Differential non-linearity error 12-13
12.5 One-shot mode 12-14
12.5.1 Settings for one-shot mode 12-14
12.5.2 One-shot mode operation description 12-16
12.6 Repeat mode 12-17
12.6.1 Settings for repeat mode 12-17
12.6.2 Repeat mode operation description 12-19
12.7 Single sweep mode 12-20
12.7.1 Settings for single sweep mode 12-20
12.7.2 Single sweep mode operation description 12-22
12.8 Repeat sweep mode 12-24
12.8.1 Settings for repeat sweep mode 12-24
12.8.2 Repeat sweep mode operation description 12-26
12.9 Precautions for A-D converter 12-28

CHAPTER 13 DMA CONTROLLER

13.1 Overview 13-2
13.1.1 Performance overview 13-2
13.1.2 Bus use priority levels 13-3
13.1.3 Modes 13-3
13.2 Block description 13-6
13.2.1 Bus access control circuit 13-7
13.2.2 DMAC control register L 13-10
13.2.3 DMAC control register H 13-11
13.2.4 Source address register i (SARi) 13-12
13.2.5 Destination address register i (DARi) 13-12
13.2.6 Transfer counter register i (TCRi) 13-12
13.2.7 Incrementer/Decrementer 13-13
13.2.8 Decrementer 13-13
13.2.9 DMA latch 13-13
13.2.10 DMAi mode register L 13-14
13.2.11 DMAi mode register H 13-15
13.2.12 DMAi control register 13-16
13.2.13 DMAi interrupt control register 13-17
13.2.14 Port P9 direction register 13-18
[Precautions for DMAC] 13-18

Announced

13.3 Control 13-19
13.3.1 DMA enabling 13-19
13.3.2 DMA requests 13-20
13.3.3 Channel priority levels 13-21
13.3.4 Processing from DMA request until DMA transfer execution 13-23
13.3.5 Termination of DMA transfer 13-25
13.3.6 DMA transfer restart after termination 13-28
13.4 Operation 13-30
13.4.1 2-bus cycle transfer 13-30
[Precautions for 2-bus cycle transfer] 13-37
13.4.2 1-bus cycle transfer 13-38
[Precautions for 1-bus cycle transfer] 13-47
13.4.3 Burst transfer mode 13-48
[Precautions for burst transfer mode] 13-50
13.4.4 Cycle-steal transfer mode 13-51
[Precautions for cycle-steal transfer mode] 13-52
13.5 Single transfer mode 13-54
13.5.1 Setting of single transfer mode 13-56
13.5.2 Operation in single transfer mode 13-59
13.6 Repeat transfer mode 13-61
13.6.1 Setting of repeat transfer mode 13-63
13.6.2 Operation in repeat transfer mode 13-66
13.7 Array chain transfer mode 13-68
13.7.1 Transfer parameter memory in array chain transfer mode 13-70
13.7.2 Setting of array chain transfer mode 13-72
13.7.3 Operation in array chain transfer mode 13-75
[Precautions for array chain transfer mode] 13-79
13.8 Link array chain transfer mode 13-80
13.8.1 Transfer parameter memory in link array chain transfer mode 13-82
13.8.2 Setting of link array chain transfer mode 13-84
13.8.3 Operation in link array chain transfer mode 13-87
[Precautions for link array chain transfer mode] 13-97
13.9 DMA transfer time 13-98
13.9.1 Cycle-steal transfer mode 13-98
13.9.2 Burst transfer mode 13-101

CHAPTER 14 DRAM CONTROLLER

14.1 Overview 14-2
14.2 Block description 14-2
14.2.1 DRAM control register 14-3
14.2.2 Refresh timer 14-5
14.2.3 Address comparator 14-6
14.2.4 $\overline{RAS}$ and $\overline{CAS}$ generating circuit 14-6
14.2.5 Address multiplexer 14-6
14.3 Setting for DRAMC 14-7
14.4 DRAMC operation 14-8
14.4.1 Waveform example of DRAM control signals 14-8
14.4.2 Refresh request 14-10
14.5 Precautions for DRAMC 14-12

EOL announced

CHAPTER 15 WATCHDOG TIMER

15.1 Block description 15-2
15.1.1 Watchdog timer 15-3
15.1.2 Watchdog timer frequency select register 15-3
15.2 Operation description 15-4
15.2.1 Basic operation 15-4
15.2.2 Stop period 15-6
15.2.3 Operation in Stop mode 15-6
15.3 Precautions for Watchdog timer 15-7

CHAPTER 16 APPLICATION

16.1 Memory connection 16-2
16.1.1 Memory connection model 16-2
16.1.2 How to calculate timing 16-4
16.1.3 Example of memory connection 16-20
16.1.4 Example of I/O expansion 16-40
16.2 Examples of using DMA controller 16-43
16.2.1 Example of Centronics interface configuration 16-43
16.2.2 Example of stepping motor control 16-48
16.2.3 Example of dynamic lighting for LED 16-53
16.3 Comparison of sample program execution rate 16-56
16.3.1 Differences depending on data bus width and software Wait 16-56
16.3.2 Comparison between software Wait ($f(X_{IN})$ = 20 MHz) and software Wait + Ready ($f(X_{IN})$ = 25 MHz) 16-58

APPENDIX

Appendix 1. Memory assignment of 7721 Group 17-2
Appendix 2. Memory assignment in SFR area 17-3
Appendix 3. Control registers 17-9
Appendix 4. Package outline 17-40
Appendix 5. Examples of handling unused pins 17-41
Appendix 6. Machine instructions 17-42
Appendix 7. Hexadecimal instruction code table 17-56
Appendix 8. Countermeasure against noise 17-59
Appendix 9. 7721 Group Q & A 17-65
Appendix 10. Differences between 7721 Group and 7720 Group 17-79
Appendix 11. Electrical characteristics 17-80
Appendix 12. Standard characteristics 17-107

GLOSSARY

Not announced

MEMORANDUM

EOL announced

CHAPTER 1
DESCRIPTION

1.1 Performance overview
1.2 Pin configuration
1.3 Pin description
1.4 Block diagram

EOL announced

1.1 Performance overview

Table 1.1.1 lists the performance overview of the M37721.

Table 1.1.1 M37721 performance overview

Parameters			Functions
Number of basic instructions			103
Instruction execution time			160 ns (the minimum instruction at f(X_{IN}) = 25 MHz)
External clock input frequency f(X_{IN})			25 MHz (maximum)
Memory sizes	ROM		External
	RAM	M37721S2BFP	1024 bytes
		M37721S1BFP	512 bytes
Programmable Input/Output ports	P5–P10		8 bits × 6
	P4		5 bits × 1
Multifunctional timers	TA0–TA4		16 bits × 5
	TB0–TB2		16 bits × 3
Serial I/O	UART0, UART1		(UART or clock synchronous serial I/O) × 2
A-D converter			8-bit successive approximation method × 1 (8 channels)
Watchdog timer			12 bits × 1
DMA controller			4 channels Maximum transfer rate : 12.5 Mbytes/sec. (at f(X_{IN}) = 25 MHz, 1-bus cycle transfer) Maximum transfer rate : 6.25 Mbytes/sec. (at f(X_{IN}) = 25 MHz, 2-bus cycle transfer)
DRAM controller			$\overline{CAS}$ before $\overline{RAS}$ refreshing method
Real-time output			4 bits × 2 channels or 6 bits × 1 channel + 2 bits × 1 channel
Interrupts			3 external, 20 internal (priority levels 0 to 7 can be set for each interrupt with software)
Clock generating circuit			Built-in (externally connected to a ceramic resonator or a quartz-crystal oscillator)
Supply voltage			5 V ±10 %
Power dissipation			135 mW (at f(X_{IN}) = 25 MHz, typ.)
Port Input/Output characteristics	Input/Output withstand voltage		5 V
	Output current		5 mA
Memory expansion			Maximum 16 Mbytes
Operating temperature range			–20°C to 85°C
Device structure			CMOS high-performance silicon gate process
Package			100-pin plastic molded QFP

EOL announced

1.2 Pin configuration

Figure 1.2.1 shows the M37721S2BFP pin configuration.

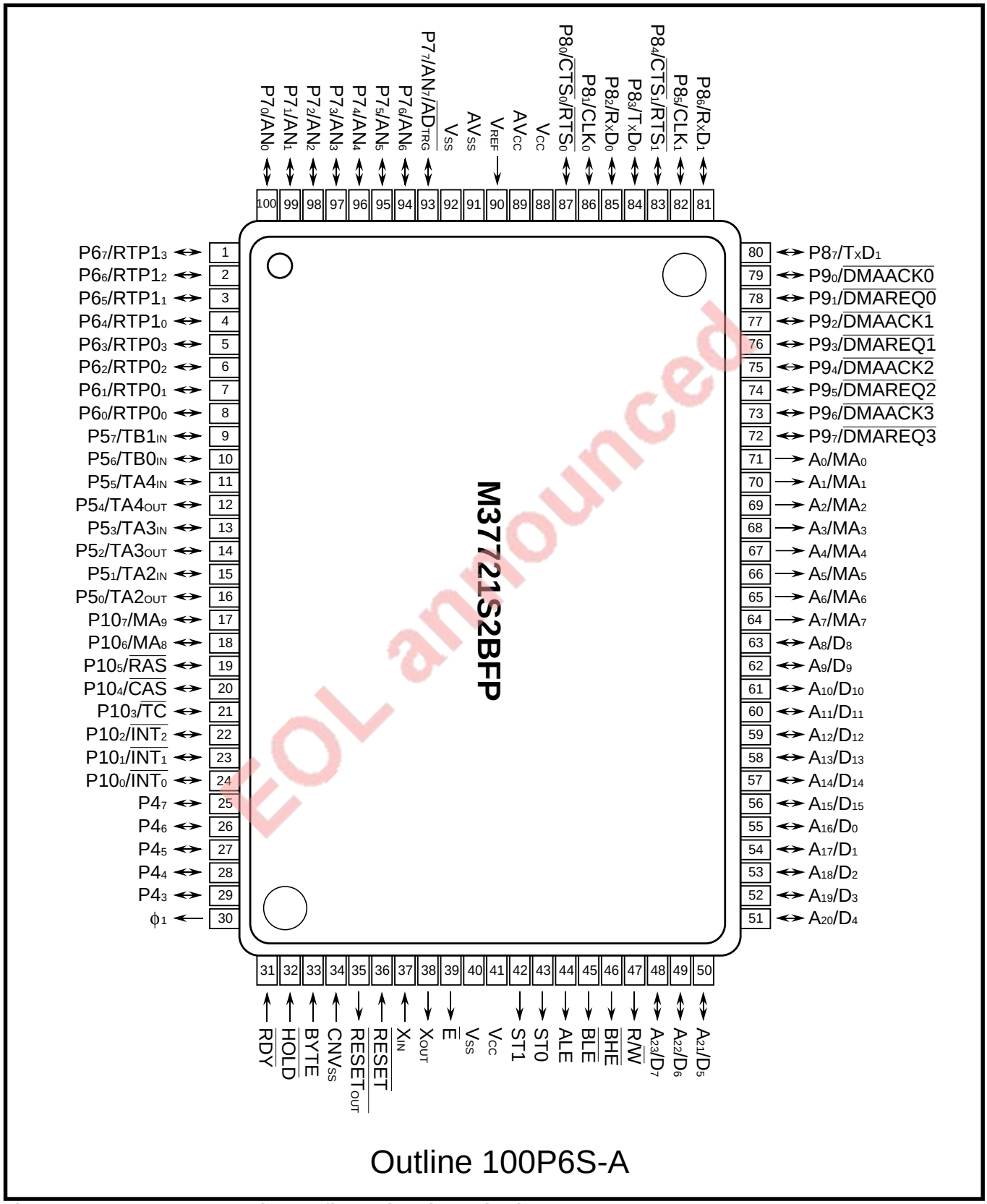


Fig. 1.2.1 M37721S2BFP pin configuration (top view)

1.3 Pin description

Tables 1.3.1 to 1.3.3 list the pin description.

Table 1.3.1 Pin description (1)

Pin	Name	Input/Output	Functions
Vcc, Vss	Power supply		Supply 5 V ±10 % to Vcc pin and 0 V to Vss pin.
CNVss	CNVss	Input	Connect to Vss or Vcc pin.
$\overline{\text{RESET}}$	Reset input	Input	The microcomputer is reset when supplying "L" level to this pin.
$\overline{\text{RESET}}_{\text{OUT}}$	Reset output	Output	When input to $\overline{\text{RESET}}$ pin is "L," this pin outputs "L." Output from this pin returns "H" after the release of reset. When writing "1" to the software reset bit, this pin outputs "L."
X_{IN}	Clock input	Input	These are I/O pins of the internal clock generating circuit. Connect a ceramic resonator or quartz-crystal oscillator between pins X_{IN} and X_{OUT}. When using an external clock, the clock source should be input to X_{IN} pin and X_{OUT} pin should be left open.
X_{OUT}	Clock output	Output	
$\overline{E}$	Enable output	Output	Data/instruction code read or data write is performed when output from this pin is "L" level.
BYTE	Exernal data bus width selection input	Input	Input level to this pin determines whether the external data bus has a 16-bit width or an 8-bit width. The width is 16 bits when the level is "L", and 8 bits when the level is "H".
ST0 ST1	Status signal output	Output	The bus use state is output in 2-bit code. ST1 / ST0 / Bus use state 0 / 0 / DRAM refresh 0 / 1 / Hold 1 / 0 / DMA 1 / 1 / CPU
AVcc AVss	Analog supply input		The power supply pin for the A-D converter. Connect AVcc to Vcc pin. Connect AVss to Vss pin.
V_{REF}	Reference voltage input	Input	This is a reference voltage input pin for the A-D converter.

EOL announced

Table 1.3.2 Pin description (2)

Pin	Name	Input/Output	Functions
A_0/MA_0–A_7/MA_7	Address low-order/ DRAM address	Output	Low-order 8 bits (A_0–A_7) of the address are output. When the DRAM is accessed, the row and column addresses are output with the time-sharing.
A_8/D_8–A_{15}/D_{15}	Address middle-order/ data high-order	I/O	●External data bus width = 8 bits (When the BYTE pin is "H" level) Middle-order 8 bits (A_8–A_{15}) of the address are output. ●External data bus width = 16 bits (When the BYTE pin is "L" level) Data (D_8–D_{15}) input/output and output of the middle-order 8 bits (A_8–A_{15}) of the address are performed with the time-sharing.
A_{16}/D_0–A_{23}/D_7	Address high-order/ data low-order	I/O	Data (D_0–D_7) input/output and output of the high-order 8 bits (A_{16}–A_{23}) of the address are performed with the time-sharing.
R/$\overline{W}$, $\overline{BHE}$, $\overline{BLE}$, ALE	Memory control signal output	Output	●R/$\overline{W}$ The Read/Write signal indicates the data bus state. The state is read while this signal is "H" level, and write while this signal is "L" level. ●$\overline{BHE}$ "L" level is output when an odd-numbered address is accessed. ●$\overline{BLE}$ "L" level is output when an even-numbered address is accessed. ●ALE This is used to obtain only the address from address and data multiplex signals.
$\overline{HOLD}$	Hold input	Input	The microcomputer is in Hold state while "L" level is input to the $\overline{HOLD}$ pin.
$\overline{RDY}$	Ready input	Input	The microcomputer is in Ready state while "L" level is input to the $\overline{RDY}$ pin.
ϕ_1	Clock ϕ_1 output	Output	This is the ϕ_1 output pin.

EOL announced

1.3 Pin description

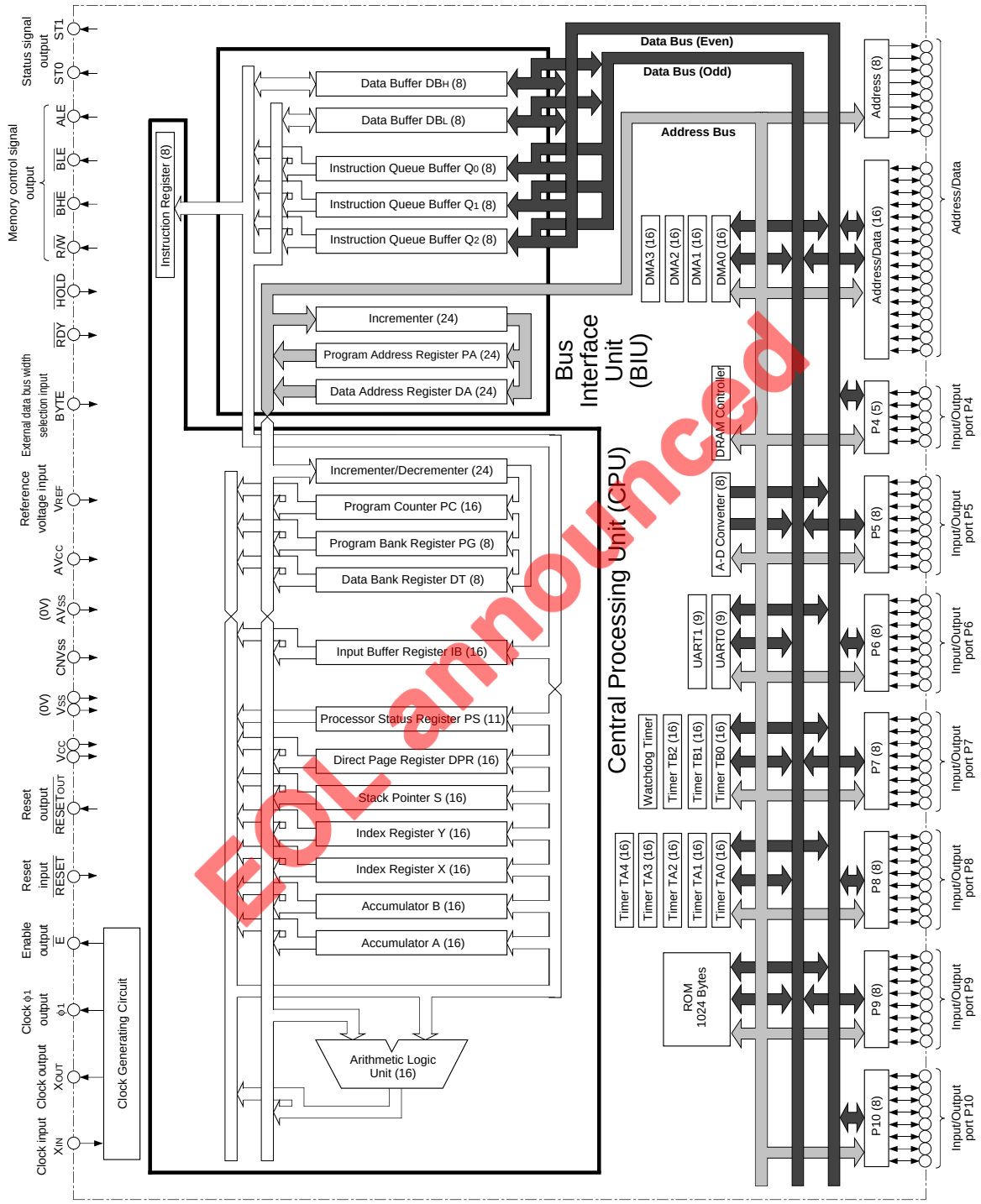
Table 1.3.3 Pin description (3)

Pin	Name	Input/Output	Functions
$P4_3$–$P4_7$	I/O port P4	I/O	Port P4 is a 5-bit CMOS I/O port. This port has an I/O direction register and each pin can be programmed for input or output.
$P5_0$–$P5_7$	I/O port P5	I/O	Port P5 is an 8-bit I/O port with the same function as P4. These pins can be programmed as I/O pins for Timers A2–A4 and I/O pins for Timers B0, B1.
$P6_0$–$P6_7$	I/O port P6	I/O	Port P6 is an 8-bit I/O port with the same function as P4. These pins can be programmed as output pins for the real-time output.
$P7_0$–$P7_7$	I/O port P7	I/O	Port P7 is an 8-bit I/O port with the same function as P4. These pins can be programmed as input pins for A-D converter.
$P8_0$–$P8_7$	I/O port P8	I/O	Port P8 is an 8-bit I/O port with the same function as P4. These pins can be programmed as I/O pins for Serial I/O.
$P9_0$–$P9_7$	I/O port P9	I/O	Port P9 is an 8-bit I/O port with the same function as P4. These pins can be programmed as I/O pins for DMA controller.
$P10_0$–$P10_7$	I/O port P10	I/O	Port P10 is an 8-bit I/O port with the same function as P4. These pins can be programmed as I/O pin for $\overline{TC}$ and output pins for DRAM controller. $P10_0$–$P10_2$ also function as input pins for $\overline{INT_0}$–$\overline{INT_2}$.

EOL announced

1.4 Block diagram

Figure 1.4.1 shows the M37721 block diagram.



Note: For the M37721S1BFP, the RAM size is 512 bytes.

Fig. 1.4.1 M37721 block diagram

MEMORANDUM

EOL announced

CHAPTER 2

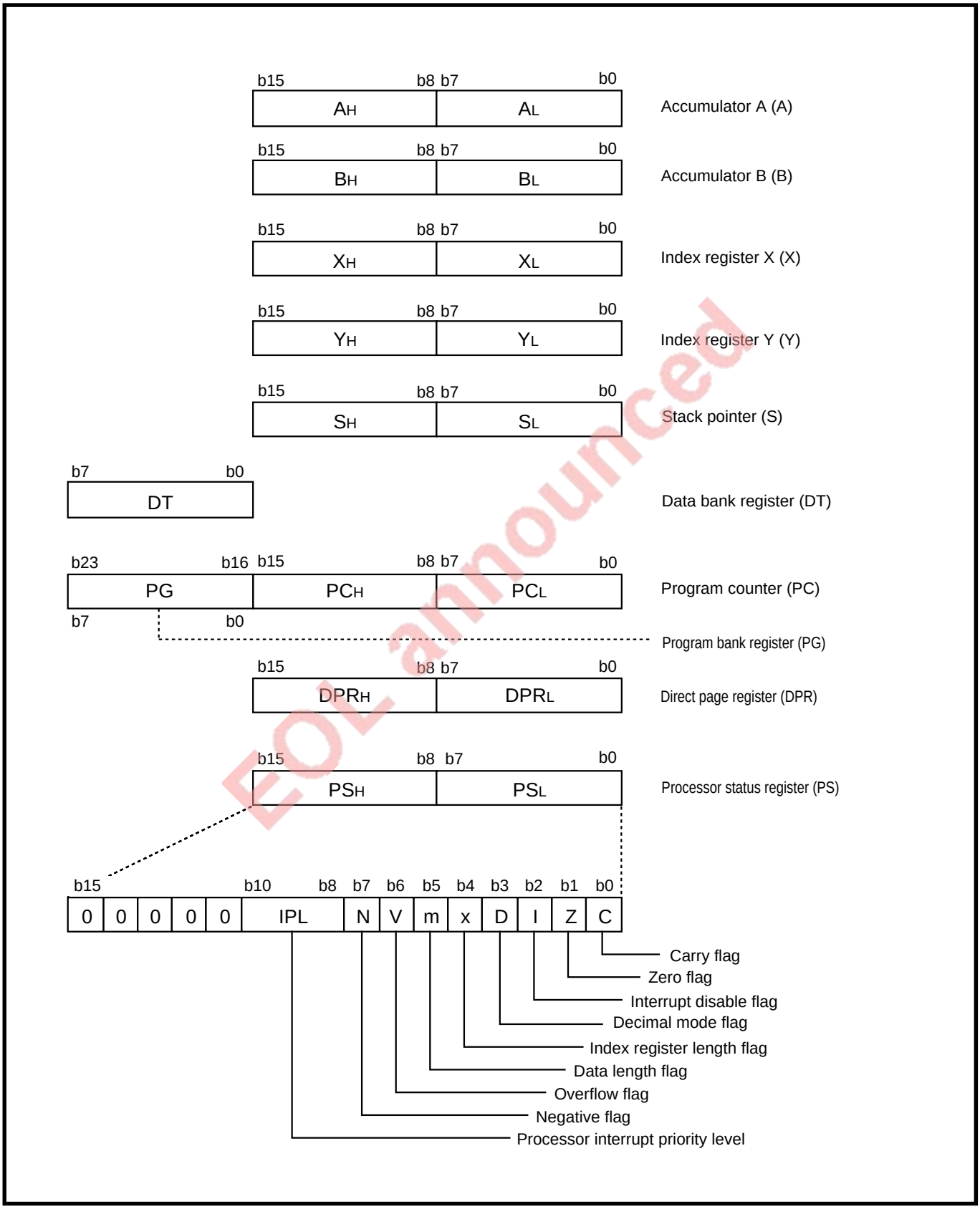
CENTRAL PROCESSING UNIT (CPU)

EOL announced

2.1 Central processing unit
2.2 Bus interface unit
2.3 Access space
2.4 Memory assignment
2.5 Bus access right

2.1 Central processing unit

The CPU (Central Processing Unit) has the ten registers as shown in Figure 2.1.1.



EOL announced

Fig. 2.1.1 CPU registers structure

2.1.1 Accumulator (Acc)

Accumulators A and B are available.

(1) Accumulator A (A)

Accumulator A is the main register of the microcomputer. The transaction of data such as calculation, data transfer, and input/output are performed mainly through accumulator A. It consists of 16 bits, and the low-order 8 bits can also be used separately. The data length flag (m) determines whether the register is used as a 16-bit register or as an 8-bit register. When an 8-bit register is selected, only the low-order 8 bits of accumulator A are used and the contents of the high-order 8 bits is unchanged.

(2) Accumulator B (B)

Accumulator B is a 16-bit register with the same function as accumulator A. Accumulator B can be used instead of accumulator A. The use of accumulator B, however except for some instructions, requires more instruction bytes and execution cycles than that of accumulator A. Accumulator B is also controlled by the data length flag (m) just as in accumulator A.

2.1.2 Index register X (X)

Index register X consists of 16 bits and the low-order 8 bits can also be used separately. The index register length flag (x) determines whether the register is used as a 16-bit register or as an 8-bit register. When an 8-bit register is selected, only the low-order 8 bits of index register X are used and the contents of the high-order 8 bits is unchanged.
In an addressing mode in which index register X is used as an index register, the address obtained by adding the contents of this register to the operand's contents is accessed.

In the **MVP** or **MVN** instruction, a block transfer instruction, the contents of index register X indicate the low-order 16 bits of the source address. The third byte of the instruction is the high-order 8 bits of the source address.

Note: Refer to **"7700 Family Software Manual"** for addressing modes.

2.1.3 Index register Y (Y)

Index register Y is a 16-bit register with the same function as index register X. Just as in index register X, the index register length flag (x) determines whether this register is used as a 16-bit register or as an 8-bit register.
In the **MVP** or **MVN** instruction, a block transfer instruction, the contents of index register Y indicate the low-order 16 bits of the destination address. The second byte of the instruction is the high-order 8 bits of the destination address.

EOL announced

2.1.4 Stack pointer (S)

The stack pointer (S) is a 16-bit register. It is used for a subroutine call or an interrupt. It is also used when addressing modes using the stack are executed. The contents of S indicate an address (stack area) for storing registers during subroutine calls and interrupts. Stack area is selected by the stack bank select bit described later (bit 7 at address $5E_{16}$). The stack area is specified to bank 0_{16} when the stack bank select bit is "0," and the stack area is specified to bank FF_{16} when it is "1."

When an interrupt request is accepted, the microcomputer stores the contents of the program bank register (PG) at the address indicated by the contents of S and decrements the contents of S by 1. Then the contents of the program counter (PC) and the processor status register (PS) are stored. The contents of S after accepting an interrupt request is equal to the contents of S decremented by 5 before the accepting of the interrupt request. (Refer to **"Figure 2.1.2."**)

When completing the process in the interrupt routine and returning to the original routine, the contents of registers stored in the stack area are restored into the original registers in the reverse sequence (PS→PC→PG) by executing the **RTI** instruction. The contents of S is returned to the state before accepting an interrupt request.

The same operation is performed during a subroutine call, however, the contents of PS is not automatically stored. (The contents of PG may not be stored. This depends on the addressing mode.)

The user should store registers other than those described above with software when the user needs them during interrupts or subroutine calls.

Additionally, initialize S at the beginning of the program because its contents are undefined at reset. The stack area changes when subroutines are nested or when multiple interrupt requests are accepted. Therefore, make sure of the subroutine's nesting depth not to destroy the necessary data.

Note: Refer to **"7700 Family Software Manual"** for addressing modes.

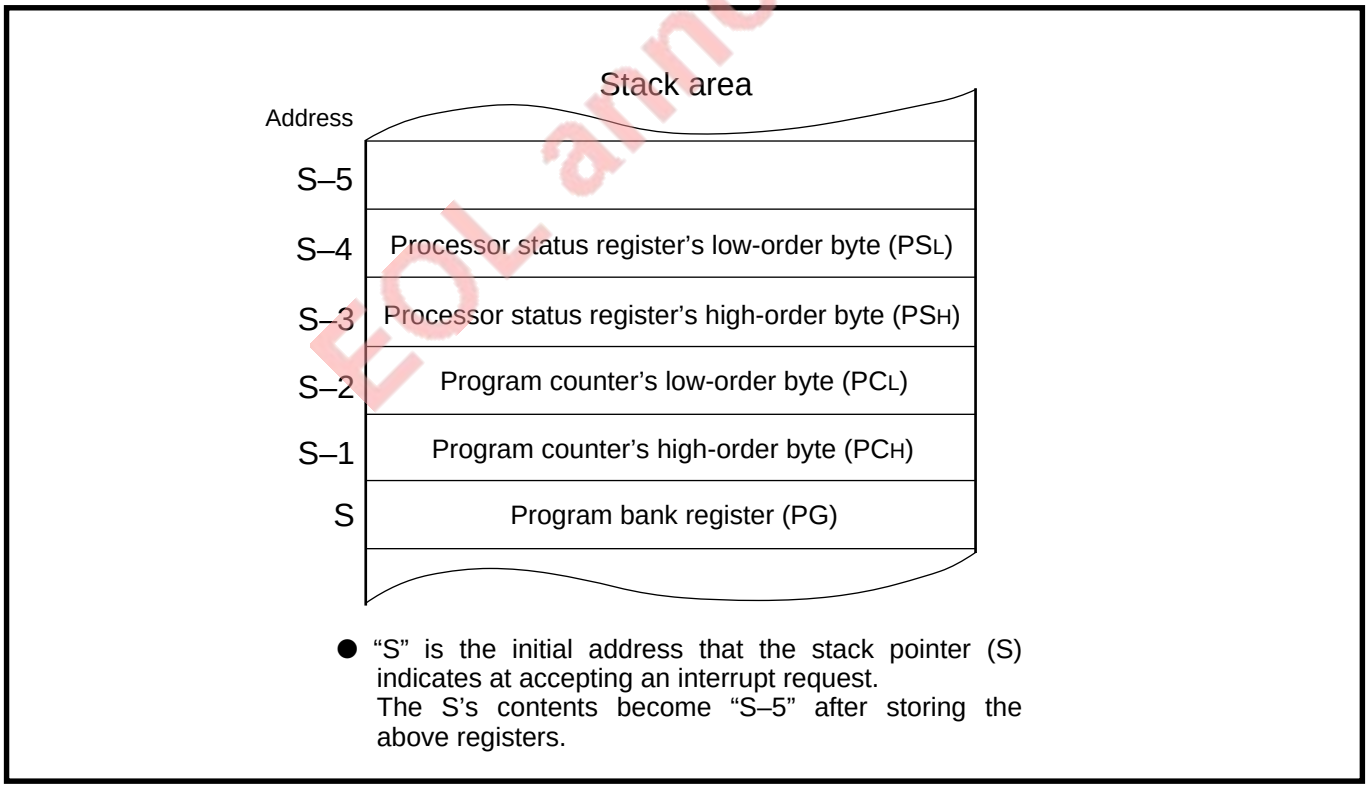


Fig. 2.1.2 Stored registers of the stack area

EOL announced

2.1.5 Program counter (PC)

The program counter is a 16-bit counter that indicates the low-order 16 bits of the address (24 bits) at which an instruction to be executed next (in other words, an instruction to be read out from an instruction queue buffer next) is stored. The contents of the high-order program counter (PC_H) become "FF_{16}," and the low-order program counter (PC_L) becomes "FE_{16}" at reset. The contents of the program counter becomes the contents of the reset's vector address (addresses $FFFE_{16}$, $FFFF_{16}$) immediately after reset.
Figure 2.1.3 shows the program counter and the program bank register.

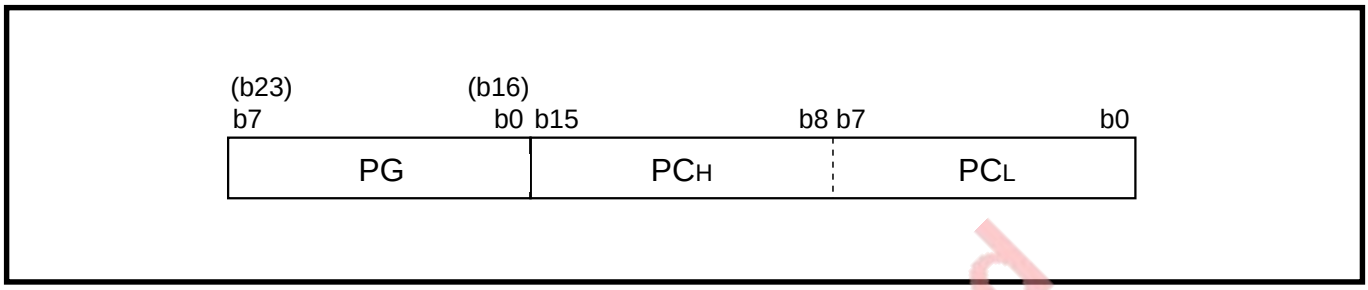


Fig. 2.1.3 Program counter and program bank register

2.1.6 Program bank register (PG)

The access space is divided in units of 64 Kbytes. This unit is called "bank." (Refer to section **"2.3 Access space."**)
The program bank register is an 8-bit register. This register indicates the high-order 8 bits (bank) of the address (24 bits) at which an instruction to be executed next (in other words, an instruction to be read out from an instruction queue buffer next) is stored. These 8 bits are called bank.
When a carry occurs after adding the contents of the program counter or adding the offset value to the contents of the program counter in the branch instruction and others, the contents of the program bank register is automatically incremented by 1. When a borrow occurs after subtracting the contents of the program counter, the contents of the program bank register is automatically decremented by 1. Accordingly, there is no need to consider bank boundaries in programming, usually.

This register is cleared to "00_{16}" at reset.

2.1.7 Data bank register (DT)

The data bank register is an 8-bit register. In the following addressing modes using the data bank register, the contents of this register is used as the high-order 8 bits (bank) of a 24-bit address to be accessed. Use the **LDT** instruction to set a value to this register.

This register is cleared to "00_{16}" at reset.

●Addressing modes using data bank register
- •Direct indirect
- •Direct indexed X indirect
- •Direct indirect indexed Y
- •Absolute
- •Absolute bit
- •Absolute indexed X
- •Absolute indexed Y
- •Absolute bit relative
- •Stack pointer relative indirect indexed Y

EOL announced

2.1.8 Direct page register (DPR)

The direct page register is a 16-bit register. The contents of this register indicate the direct page area which is allocated in bank 0_{16} or in the space across banks 0_{16} and 1_{16}. The following addressing modes use the direct page register.

The contents of the direct page register indicate the base address (the lowest address) of the direct page area. The space which extends to 256 bytes above that address is specified as a direct page.

The direct page register can contain a value from "0000_{16}" to "$FFFF_{16}$." When it contains a value equal to or more than "$FF01_{16}$," the direct page area spans the space across banks 0_{16} and 1_{16}.

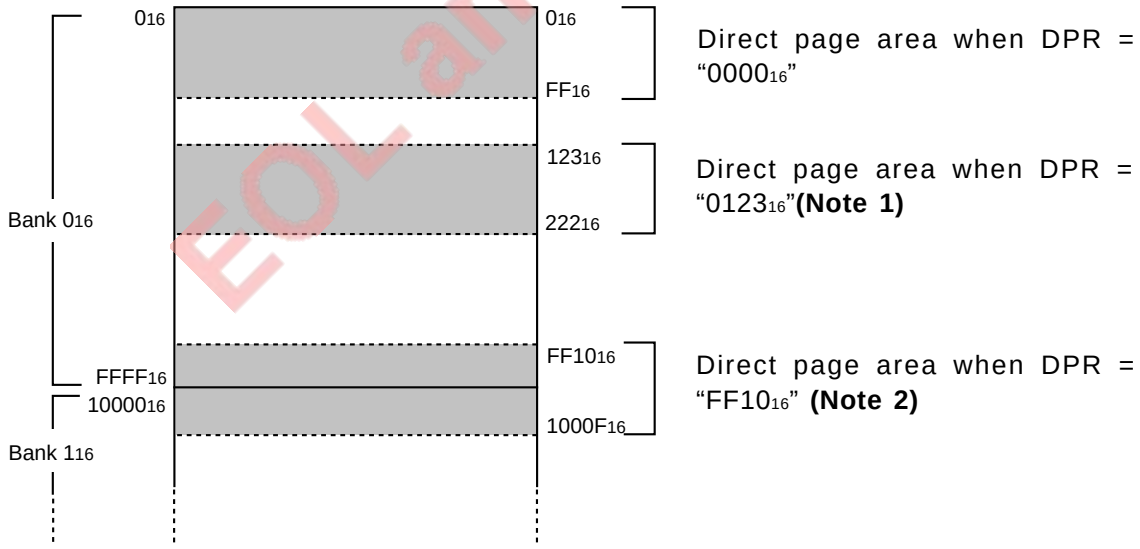
When the contents of low-order 8 bits of the direct page register is "00_{16}," the number of cycles required to generate an address is 1 cycle smaller than the number when its contents are not "00_{16}." Accordingly, the access efficiency can be enhanced in this case.

This register is cleared to "0000_{16}" at reset.

Figure 2.1.4 shows a setting example of the direct page area.

●Addressing modes using direct page register
- •Direct
- •Direct bit
- •Direct indexed X
- •Direct indexed Y
- •Direct indirect
- •Direct indexed X indirect
- •Direct indirect indexed Y
- •Direct indirect long
- •Direct indirect long indexed Y
- •Direct bit relative

EOL announced



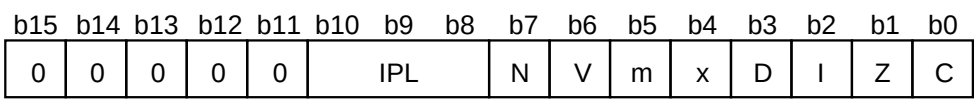
Notes 1: The number of cycles required to generate an address is 1 cycle smaller when the low-order 8 bits of the DPR are "00_{16}."

2: The direct page area spans the space across banks 0_{16} and 1_{16} when the DPR is "$FF01_{16}$" or more.

Fig. 2.1.4 Setting example of direct page area

2.1.9 Processor status register (PS)

The processor status register is an 11-bit register.
Figure 2.1.5 shows the structure of the processor status register.



Note: Bits 11–15 is always “0” at reading.

Fig. 2.1.5 Processor status register structure

(1) Bit 0: Carry flag (C)

It retains a carry or a borrow generated in the arithmetic and logic unit (ALU) during an arithmetic operation. This flag is also affected by shift and rotate instructions. When the **BCC** or **BCS** instruction is executed, this flag’s contents determine whether the program causes a branch or not.
Use the **SEC** or **SEP** instruction to set this flag to “1,” and use the **CLC** or **CLP** instruction to clear it to “0.”

(2) Bit 1: Zero flag (Z)

It is set to “1” when a result of an arithmetic operation or data transfer is “0,” and cleared to “0” when otherwise. When the **BNE** or **BEQ** instruction is executed, this flag’s contents determine whether the program causes a branch or not.
Use the **SEP** instruction to set this flag to “1,” and use the **CLP** instruction to clear it to “0.”

Note: This flag is invalid in the decimal mode addition (the **ADC** instruction).

(3) Bit 2: Interrupt disable flag (I)

It disables all maskable interrupts (interrupts other than watchdog timer, the **BRK** instruction, and zero division). Interrupts are disabled when this flag is “1.” When an interrupt request is accepted, this flag is automatically set to “1” to avoid multiple interrupts. Use the **SEI** or **SEP** instruction to set this flag to “1,” and use the **CLI** or **CLP** instruction to clear it to “0.” This flag is set to “1” at reset.

(4) Bit 3: Decimal mode flag (D)

It determines whether addition and subtraction are performed in binary or decimal. Binary arithmetic is performed when this flag is “0.” When it is “1,” decimal arithmetic is performed with 8 bits treated as two digits decimal (the data length flag (m) = “1”) or 16 bits treated as four digits decimal (the data length flag (m) = “0”). Decimal adjust is automatically performed. Decimal operation is possible only with the **ADC** and **SBC** instructions. Use the **SEP** instruction to set this flag to “1,” and use the **CLP** instruction to clear it to “0.” This flag is cleared to “0” at reset.

(5) Bit 4: Index register length flag (x)

It determines whether each of index register X and index register Y is used as a 16-bit register or an 8-bit register. That register is used as a 16-bit register when this flag is “0,” and as an 8-bit register when it is “1.” Use the **SEP** instruction to set this flag to “1,” and use the **CLP** instruction to clear it to “0.” This flag is cleared to “0” at reset.

Note: When transferring data between registers which are different in bit length, the data is transferred with the length of the destination register, but except for the **TXA**, **TYA**, **TXB**, **TYB**, and **TXS** instructions. Refer to **“7700 Family Software Manual”** for details.

ECL announced

(6) Bit 5: Data length flag (m)

It determines whether to use a data as a 16-bit unit or as an 8-bit unit. A data is treated as a 16-bit unit when this flag is "0," and as an 8-bit unit when it is "1."
Use the **SEM** or **SEP** instruction to set this flag to "1," and use the **CLM** or **CLP** instruction to clear it to "0." This flag is cleared to "0" at reset.

Note: When transferring data between registers which are different in bit length, the data is transferred with the length of the destination register, but except for the **TXA**, **TYA**, **TXB**, **TYB**, and **TXS** instructions. Refer to **"7700 Family Software Manual"** for details.

(7) Bit 6: Overflow flag (V)

It is used when adding or subtracting with a word regarded as signed binary. When the data length flag (m) is "0," the overflow flag is set to "1" when the result of addition or subtraction exceeds the range between –32768 and +32767, and cleared to "0" in all other cases. When the data length flag (m) is "1," the overflow flag is set to "1" when the result of addition or subtraction exceeds the range between –128 and +127, and cleared to "0" in all other cases.
The overflow flag is also set to "1" when a result of division exceeds the register length to be stored in a division instruction **DIV**.
When the **BVC** or **BVS** instruction is executed, this flag's contents determine whether the program causes a branch or not.
Use the **SEP** instruction to set this flag to "1," and use the **CLV** or **CLP** instruction to clear it to "0."

Note: This flag is invalid in the decimal mode.

(8) Bit 7: Negative flag (N)

It is set to "1" when a result of arithmetic operation or data transfer is negative. (Bit 15 of the result is "1" when the data length flag (m) is "0," or bit 7 of the result is "1" when the data length flag (m) is "1.") It is cleared to "0" in all other cases. When the **BPL** or **BMI** instruction is executed, this flag determines whether the program causes a branch or not. Use the **SEP** instruction to set this flag to "1," and use the **CLP** instruction to clear it to "0."

Note: This flag is invalid in the decimal mode.

(9) Bits 10 to 8: Processor interrupt priority level (IPL)

These three bits can determine the processor interrupt priority level to one of levels 0 to 7. The interrupt is enabled when the interrupt priority level of a required interrupt, which is set in each interrupt control register, is higher than IPL. When an interrupt request is accepted, IPL is stored in the stack area, and IPL is replaced by the interrupt priority level of the accepted interrupt request. There are no instruction to directly set or clear the bits of IPL. IPL can be changed by storing the new IPL into the stack area and updating the processor status register with the **PUL** or **PLP** instruction. The contents of IPL is cleared to "000_2" at reset.

EOL announced

2.2 Bus interface unit

A bus interface unit (BIU) is built-in between the central processing unit (CPU) and memory•I/O devices. BIU's function and operation are described below.
When externally connecting devices, refer to **"CHAPTER 3. CONNECTION WITH EXTERNAL DEVICES."**

2.2.1 Overview

Transfer operation between the CPU and memory•I/O devices is always performed via the BIU.

① The BIU reads an instruction from the memory before the CPU executes it.

② When the CPU reads data from the memory•I/O device, the CPU first specifies the address from which data is read to the BIU. The BIU reads data from the specified address and passes it to the CPU.

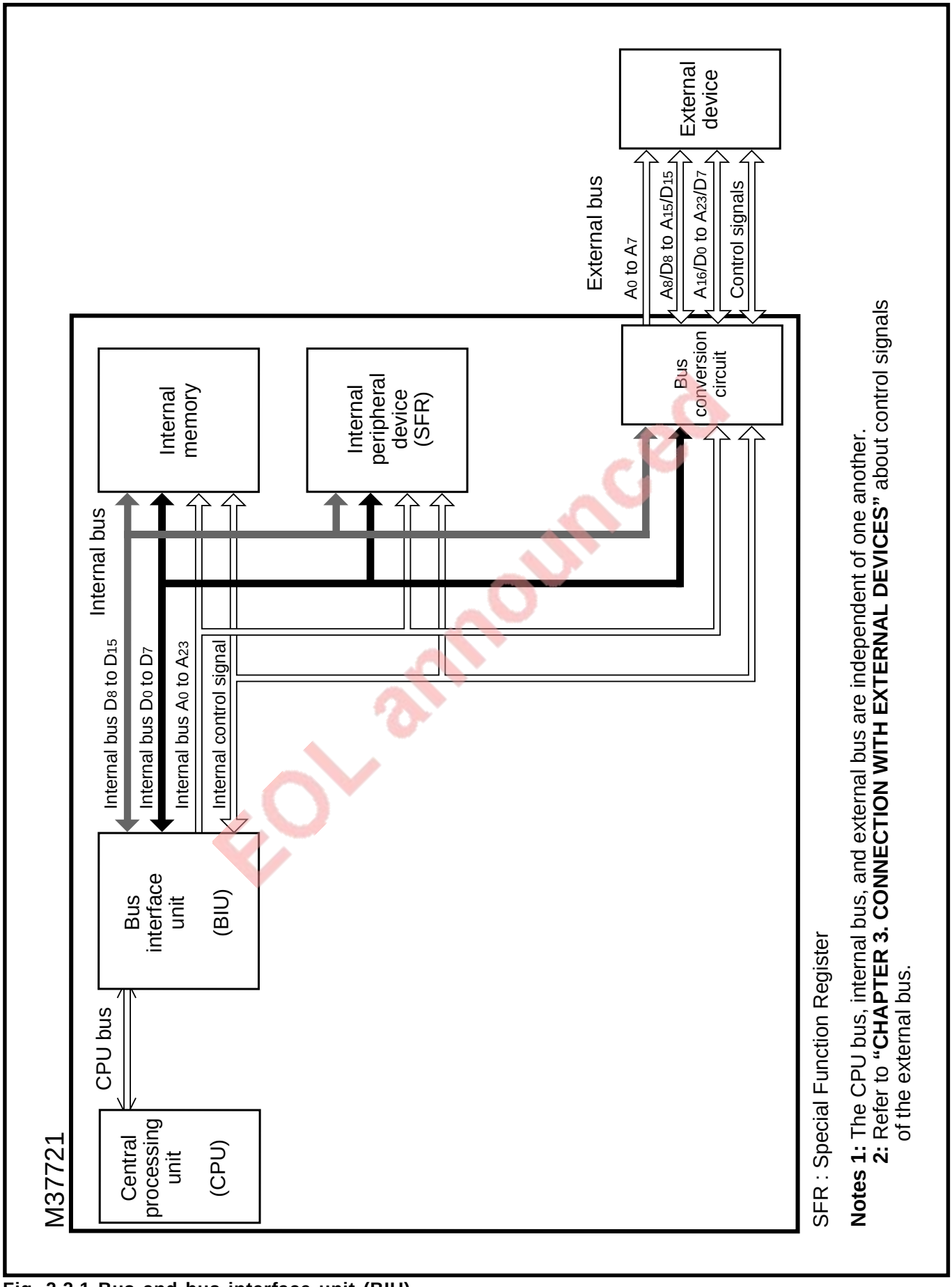
③ When the CPU writes data to the memory•I/O device, the CPU first specifies the address to which data is written to the BIU and write data. The BIU writes the data to the specified address.

④ To perform the above operations ① to ③, the BIU inputs and outputs the control signals, and control the bus.

Figure 2.2.1 shows the bus and bus interface unit (BIU).

EOL announced

2.2 Bus interface unit



EOL announced

Fig. 2.2.1 Bus and bus interface unit (BIU)

2.2.2 Functions of bus interface unit (BIU)

The bus interface unit (BIU) consists of four registers shown in Figure 2.2.2. Table 2.2.1 lists the functions of each register.

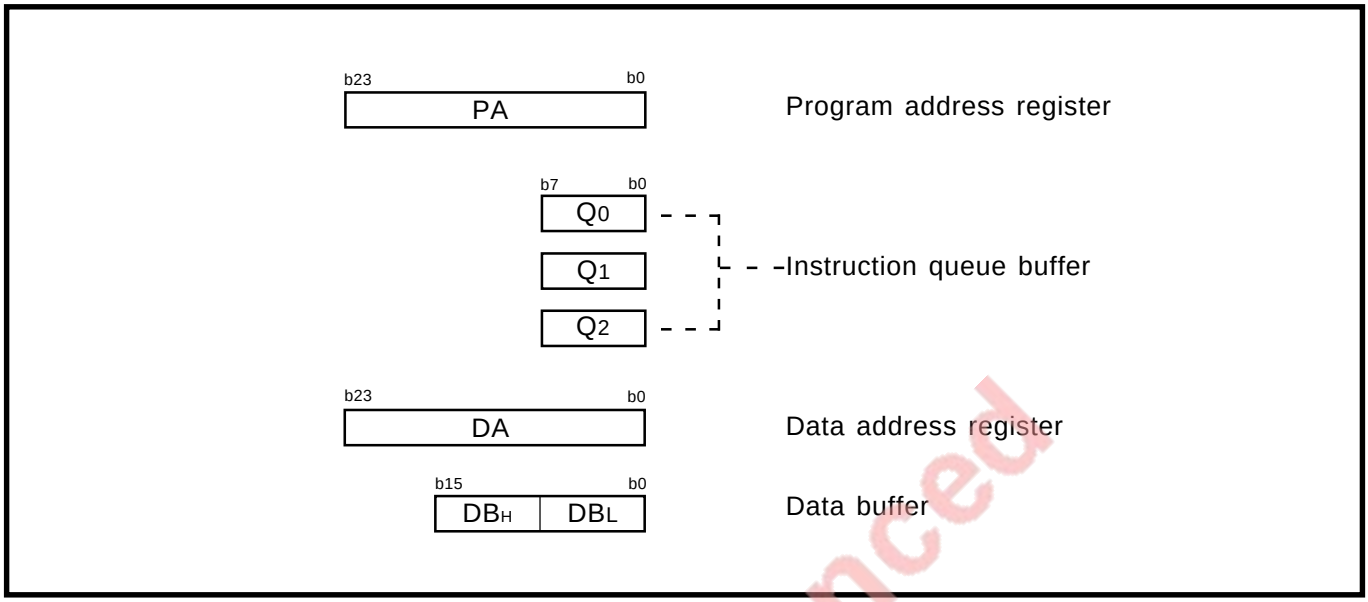


Fig. 2.2.2 Register structure of bus interface unit (BIU)

Table 2.2.1 Functions of each register

Name	Functions
Program address register	Indicates the storage address for the instruction which is next taken into the instruction queue buffer.
Instruction queue buffer	Temporarily stores the instruction which has been taken in.
Data address register	Indicates the address for the data which is next read from or written to.
Data buffer	Temporarily stores the data which is read from the memory•I/O device by the BIU or which is written to the memory•I/O device by the CPU.

EOL announced

2.2 Bus interface unit

The CPU and the bus send or receive data via BIU because each operates based on different clocks **(Note)**. The BIU allows the CPU to operate at high speed without waiting for access to the memory · I/O devices that require a long access time.
The BIU's functions are described bellow.

Note: The CPU operates based on ϕCPU. The period of ϕCPU is normally the same as that of ϕ. The internal bus operates based on the $\overline{E}$ signal. The period of the $\overline{E}$ signal is twice that of ϕ at a minimum.

(1) Reading out instruction (Instruction prefetch)

When the CPU does not require to read or write data, that is, when the bus is not in use, the BIU reads instructions from the memory and stores them in the instruction queue buffer. This is called instruction prefetch.
The CPU reads instructions from the instruction queue buffer and executes them, so that the CPU can operate at high speed without waiting for access to the memory which requires a long access time.
When the instruction queue buffer becomes empty or contains only 1 byte of an instruction, the BIU performs instruction prefetch. The instruction queue buffer can store instructions up to 3 bytes.
The contents of the instruction queue buffer is initialized when a branch or jump instruction is executed, and the BIU reads a new instruction from the destination address.
When instructions in the instruction queue buffer are insufficient for the CPU's needs, the BIU extends the pulse duration of clock ϕCPU in order to keep the CPU waiting until the BIU fetches the required number of instructions or more.

(2) Reading data from memory•I/O device

The CPU specifies the storage address of data to be read to the BIU's data address register, and requires data. The CPU waits until data is ready in the BIU.
The BIU outputs the address received from the CPU onto the address bus, reads contents at the specified address, and takes it into the data buffer.
The CPU continues processing, using data in the data buffer.
However, if the BIU uses the bus for instruction prefetch when the CPU requires to read data, the BIU keeps the CPU waiting.

(3) Writing data to memory•I/O device

The CPU specifies the address of data to be written to the BIU's data address register. Then, the CPU writes data into the data buffer. The BIU outputs the address received from the CPU onto the address bus and writes data in the data buffer into the specified address.
The CPU advances to the next processing without waiting for completion of BIU's write operation. However, if the BIU uses the bus for instruction prefetch when the CPU requires to write data, the BIU keeps the CPU waiting.

(4) Bus control

To perform the above operations (1) to (3), the BIU inputs and outputs the control signals, and controls the address bus and the data bus. The cycle in which the BIU controls the bus and accesses the memory•I/O device is called the bus cycle.
Refer to "**CHAPTER 3. CONNECTION WITH EXTERNAL DEVICES**" about the bus cycle at accessing the external devices.

EOL announced

2.2.3 Operation of bus interface unit (BIU)

Figure 2.2.3 shows the basic operating waveforms of the bus interface unit (BIU).
About signals which are input/output externally when accessing external devices, refer to **"CHAPTER 3. CONNECTION WITH EXTERNAL DEVICES."**

(1) When fetching instructions into the instruction queue buffer

① When the instruction which is next fetched is located at an even address, the BIU fetches 2 bytes at a time with the timing of waveform (a).
However, when accessing an external device which is connected with the 8-bit external data bus width (BYTE = "H"), only 1 byte of the instruction is fetched.

② When the instruction which is next fetched is located at an odd address, the BIU fetches only 1 byte with the timing of waveform (a). The contents at the even address are not taken into the instruction queue buffer.

(2) When reading or writing data to and from the memory•I/O device

① When accessing a 16-bit data which begins at an even address, waveform (a) is applied. The 16 bits of data are accessed at a time.

② When accessing a 16-bit data which begins at an odd address, waveform (b) is applied. The 16 bits of data are accessed separately in 2 operations, 8 bits at a time. Invalid data is not fetched into the data buffer.

③ When accessing an 8-bit data at an even address, waveform (a) is applied. The data at the odd address is not fetched into the data buffer.

④ When accessing an 8-bit data at an odd address, waveform (a) is applied. The data at the even address is not fetched into the data buffer.

For instructions that are affected by the data length flag (m) and the index register length flag (x), operation ① or ② is applied when flag m or x = "0"; operation ③ or ④ is applied when flag m or x = "1."

EOL announced

2.2 Bus interface unit

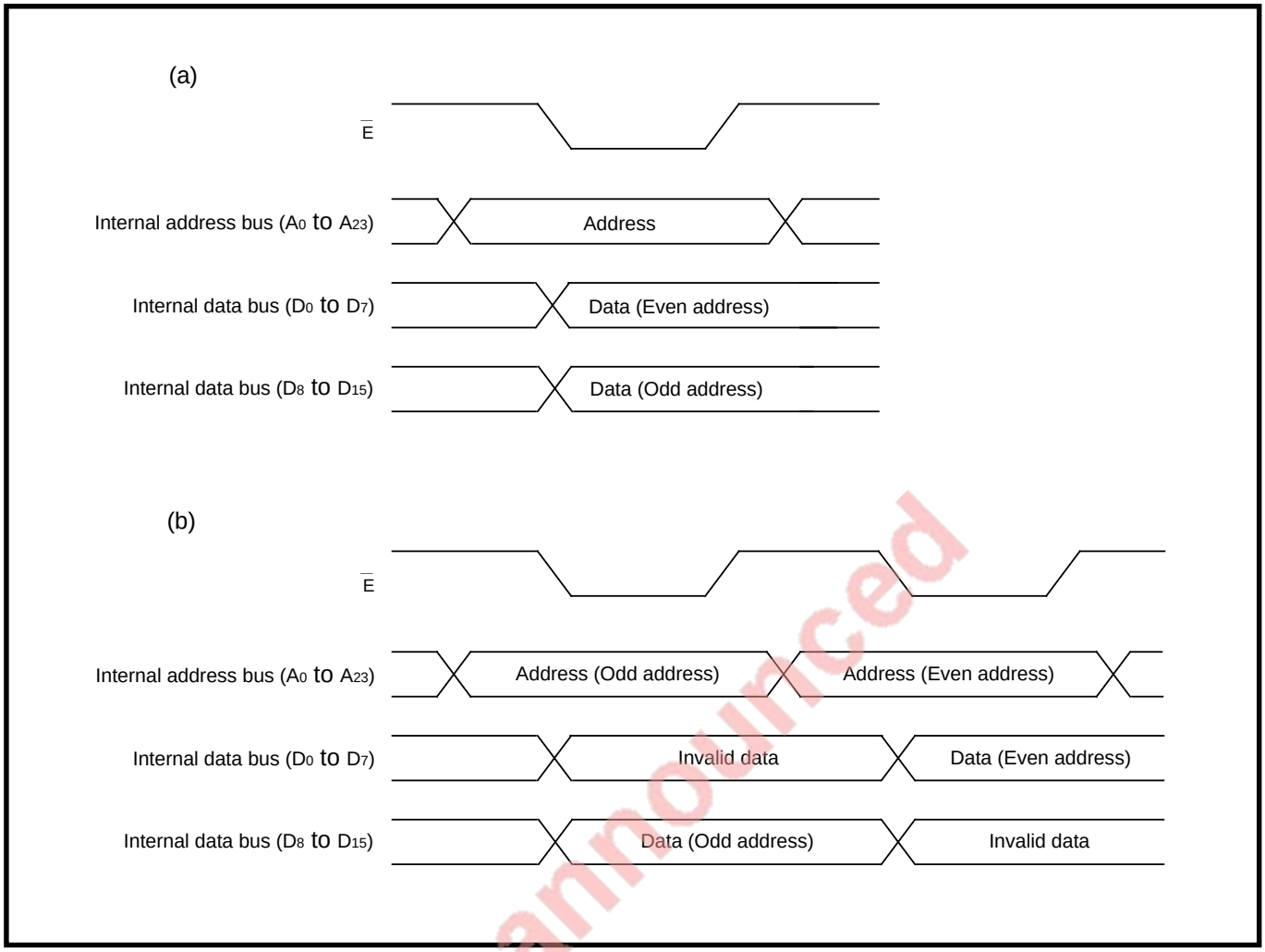


Fig. 2.2.3 Basic operating waveforms of bus interface unit (BIU)

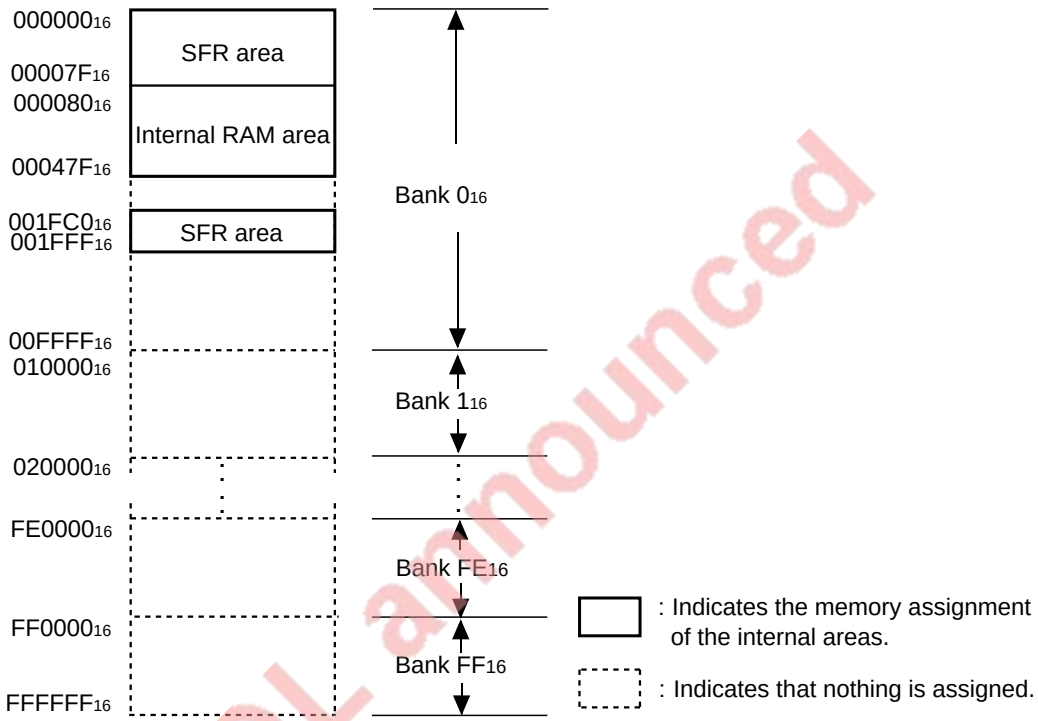
EOL announced

2.3 Access space

Figure 2.3.1 shows the M37721's access space.

By combination of the program counter (PC), which is 16 bits of structure, and the program bank register (PG), a 16-Mbyte space from addresses 0_{16} to $FFFFFF_{16}$ can be accessed. For details about access of an external area, refer to **"CHAPTER 3. CONNECTION WITH EXTERNAL DEVICES."**

The memory and I/O devices are assigned in the same access space. Accordingly, it is possible to perform transfer and arithmetic operations using the same instructions without discrimination of the memory from I/O devices.



Note : Memory assignment of internal RAM area varies according to the type of microcomputer. This figure shows the case of the M37721S2BFP. Refer to **"Figure 2.4.1"** for the M37721S1BFP.

SFR **:** Special Function Register

EOL announced

Fig. 2.3.1 M37721's access space

2.3 Access space

2.3.1 Banks

The access space is divided in units of 64 Kbytes. This unit is called “bank.” The high-order 8 bits of address (24 bits) indicate a bank, which is specified by the program bank register (PG) or data bank register (DT). Each bank can be accessed efficiently by using an addressing mode that uses the data bank register (DT).

If the program counter (PC) overflows at a bank boundary, the contents of the program bank register (PG) is incremented by 1. If a borrow occurs in the program counter (PC) as a result of subtraction, the contents of the program bank register (PG) is decremented by 1. Normally, accordingly, the user can program without concern for bank boundaries.

SFR (Special Function Register) and internal RAM are assigned in bank 0_{16}. For details, refer to section **“2.4 Memory assignment.”**

2.3.2 Direct page

A 256-byte space specified by the direct page register (DPR) is called “direct page.” A direct page is specified by setting the base address (the lowest address) of the area to be specified as a direct page into the direct page register (DPR).

By using a direct page addressing mode, a direct page can be accessed with less instruction cycles than otherwise.

Note: Refer also to section **“2.1 Central processing unit.”**

EOL announced

2.4 Memory assignment

This section describes the internal area's memory assignment. For more information about the external area, refer also to **"CHAPTER 3. CONNECTION WITH EXTERNAL DEVICES."**
Figure 2.4.1 shows the memory assignment.

2.4.1 Memory assignment in internal area

SFR (Special Function Register) and internal RAM are assigned in the internal area.

(1) SFR area

The registers for setting internal peripheral devices are assigned at addresses 0_{16} to $7F_{16}$ and $1FC0_{16}$ to $1FFF_{16}$. This area is called SFR. Figures 2.4.2 and 2.4.3 show the SFR area's memory assignment. For each register in the SFR area, refer to each functional description in this manual.
For the state of the SFR area immediately after reset, refer to section **"4.1.2 State of CPU, SFR area, and internal RAM area."**

(2) Internal RAM area

The M37721S2BFP (**Note 1**) assigns the 1024-byte static RAM at addresses 80_{16} to $47F_{16}$. 512 bytes of that can be selected either it is used as the internal RAM or it is used as the external area. **(Note 2)**
The internal RAM area is used as a stack area **(Note 3)**, as well as an area to store data. Accordingly, note that set the nesting depth of a subroutine and multiple interrupts' level not to destroy the necessary data.

Notes 1: The M37721S1BFP assigns the 512-byte static RAM at addresses 80_{16} to $27F_{16}$.
2: The internal RAM area becomes 512 bytes after reset because the internal RAM area select bit is "0."
3: Either bank 0_{16} or bank FF_{16} can be selected as the stack area by the stack bank select bit (bit 7 at address $5E_{16}$).

Figure 2.4.4 shows the structure of the processor mode registers 0, 1.

EOL announced

2.4 Memory assignment

2.4.2 External area

Table 2.4.1 lists the external area. When connecting the external device, follow the procedure described bellow:

- Connect the ROM to addresses $FFCE_{16}$ to $FFFF_{16}$ because they are interrupt vector table.
- Stack area can be assigned to bank 0_{16} or bank FF_{16}. Select the stack area by the stack bank select bit (bit 7 at address $5E_{16}$). (Refer to **"Figure 2.4.4."**)
- When using the DRAM controller, DRAM area can be selected from address $FFFFFF_{16}$ toward the low-order address in a unit of 1 Mbytes. (Refer to **"CHAPTER 14. DRAM CONTROLLER."**)

In the case connecting an external device to the area where overlaps the internal area, when reading out the overlapping area, the central processing unit (CPU) take in data of the internal area and do not take in data of the external area.

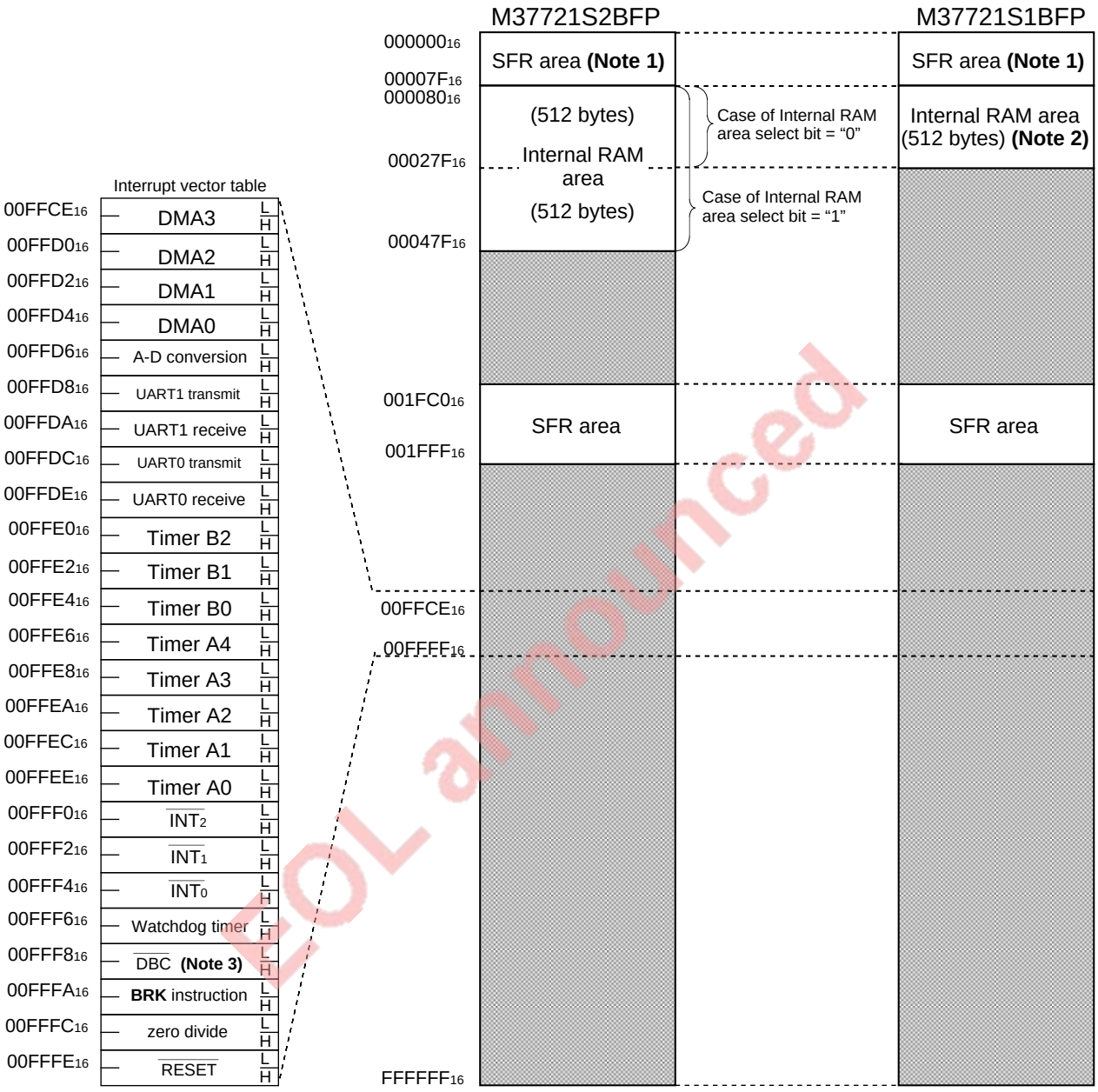
When writing to the overlapping area, data is written to the internal area. The signal is output to the external at the same timing when data is written to the internal area.

Table 2.4.1 External area

Type name	M37721S2BFP		M37721S1BFP
Internal RAM area select bit	1	0	"0" (Fix this bit to "0.")
External area	2_{16}–9_{16} 480_{16}–$1FBF_{16}$ 2000_{16}–$FFFFFF_{16}$		2_{16}–9_{16} 280_{16}–$1FBF_{16}$ 2000_{16}–$FFFFFF_{16}$

Internal RAM area select bit : bit 1 at address $5F_{16}$

EOL announced



EOL announced

: The internal memory is not assigned.

SFR area: Refer to **"Figure 2.4.2"** and **"Figure 2.4.3."**

Notes 1: Addresses 2_{16} to 9_{16} are the external memory area.
2: For the M37721S1BFP, fix the internal RAM area select bit to "0."
3: DBC is an interrupt only for debugging; do not use this interrupt.

Fig. 2.4.1 Memory assignment

2.4 Memory assignment

Address	
00000016	
00000116	
00000216	
00000316	
00000416	
00000516	
00000616	
00000716	
00000816	
00000916	
00000A16	Port P4 register
00000B16	Port P5 register
00000C16	Port P4 direction register
00000D16	Port P5 direction register
00000E16	Port P6 register
00000F16	Port P7 register
00001016	Port P6 direction register
00001116	Port P7 direction register
00001216	Port P8 register
00001316	Port P9 register
00001416	Port P8 direction register
00001516	Port P9 direction register
00001616	Port P10 register
00001716	
00001816	Port P10 direction register
00001916	
00001A16	Pulse output data register 0
00001B16	
00001C16	Pulse output data register 1
00001D16	
00001E16	A-D control register
00001F16	A-D sweep pin select register
00002016	A-D register 0
00002116	
00002216	A-D register 1
00002316	
00002416	A-D register 2
00002516	
00002616	A-D register 3
00002716	
00002816	A-D register 4
00002916	
00002A16	A-D register 5
00002B16	
00002C16	A-D register 6
00002D16	
00002E16	A-D register 7
00002F16	
00003016	UART0 transmit/receive mode register
00003116	UART0 baud rate register (BRG0)
00003216 00003316	UART0 transmit buffer register
00003416	UART0 transmit/receive control register 0
00003516	UART0 transmit/receive control register 1
00003616 00003716	UART0 receive buffer register
00003816	UART1 transmit/receive mode register
00003916	UART1 baud rate register (BRG1)
00003A16 00003B16	UART1 transmit buffer register
00003C16	UART1 transmit/receive control register 0
00003D16	UART1 transmit/receive control register 1
00003E16 00003F16	UART1 receive buffer register

Address	
00004016	Count start register
00004116	
00004216	One-shot start register
00004316	
00004416	Up-down register
00004516	
00004616 00004716	Timer A0 register
00004816 00004916	Timer A1 register
00004A16 00004B16	Timer A2 register
00004C16 00004D16	Timer A3 register
00004E16 00004F16	Timer A4 register
00005016 00005116	Timer B0 register
00005216 00005316	Timer B1 register
00005416 00005516	Timer B2 register
00005616	Timer A0 mode register
00005716	Timer A1 mode register
00005816	Timer A2 mode register
00005916	Timer A3 mode register
00005A16	Timer A4 mode register
00005B16	Timer B0 mode register
00005C16	Timer B1 mode register
00005D16	Timer B2 mode register
00005E16	Processor mode register 0
00005F16	Processor mode register 1
00006016	Watchdog timer register
00006116	Watchdog timer frequency select register
00006216	Real-time output control register
00006316	
00006416	DRAM control register
00006516	
00006616	Refresh timer
00006716	
00006816	DMAC control register L
00006916	DMAC control register H
00006A16	
00006B16	
00006C16	DMA0 interrupt control register
00006D16	DMA1 interrupt control register
00006E16	DMA2 interrupt control register
00006F16	DMA3 interrupt control register
00007016	A-D conversion interrupt control register
00007116	UART0 transmit interrupt control register
00007216	UART0 receive interrupt control register
00007316	UART1 transmit interrupt control register
00007416	UART1 receive interrupt control register
00007516	Timer A0 interrupt control register
00007616	Timer A1 interrupt control register
00007716	Timer A2 interrupt control register
00007816	Timer A3 interrupt control register
00007916	Timer A4 interrupt control register
00007A16	Timer B0 interrupt control register
00007B16	Timer B1 interrupt control register
00007C16	Timer B2 interrupt control register
00007D16	$\overline{INT_0}$ interrupt control register
00007E16	$\overline{INT_1}$ interrupt control register
00007F16	$\overline{INT_2}$ interrupt control register

EOL announced

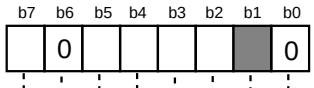
Fig. 2.4.2 SFR area's memory map (1)

Address		
001FC0₁₆	Source address register 0	L
001FC1₁₆		M
001FC2₁₆		H
001FC3₁₆		
001FC4₁₆	Destination address register 0	L
001FC5₁₆		M
001FC6₁₆		H
001FC7₁₆		
001FC8₁₆	Transfer counter register 0	L
001FC9₁₆		M
001FCA₁₆		H
001FCB₁₆		
001FCC₁₆	DMA0 mode register L	
001FCD₁₆	DMA0 mode register H	
001FCE₁₆	DMA0 control register	
001FCF₁₆		
001FD0₁₆	Source address register 1	L
001FD1₁₆		M
001FD2₁₆		H
001FD3₁₆		
001FD4₁₆	Destination address register 1	L
001FD5₁₆		M
001FD6₁₆		H
001FD7₁₆		
001FD8₁₆	Transfer counter register 1	L
001FD9₁₆		M
001FDA₁₆		H
001FDB₁₆		
001FDC₁₆	DMA1 mode register L	
001FDD₁₆	DMA1 mode register H	
001FDE₁₆	DMA1 control register	
001FDF₁₆		
001FE0₁₆	Source address register 2	L
001FE1₁₆		M
001FE2₁₆		H
001FE3₁₆		
001FE4₁₆	Destination address register 2	L
001FE5₁₆		M
001FE6₁₆		H
001FE7₁₆		
001FE8₁₆	Transfer counter register 2	L
001FE9₁₆		M
001FEA₁₆		H
001FEB₁₆		
001FEC₁₆	DMA2 mode register L	
001FED₁₆	DMA2 mode register H	
001FEE₁₆	DMA2 control register	
001FEF₁₆		
001FF0₁₆	Source address register 3	L
001FF1₁₆		M
001FF2₁₆		H
001FF3₁₆		
001FF4₁₆	Destination address register 3	L
001FF5₁₆		M
001FF6₁₆		H
001FF7₁₆		
001FF8₁₆	Transfer counter register 3	L
001FF9₁₆		M
001FFA₁₆		H
001FFB₁₆		
001FFC₁₆	DMA3 mode register L	
001FFD₁₆	DMA3 mode register H	
001FFE₁₆	DMA3 control register	
001FFF₁₆		

EQ announced

Fig. 2.4.3 SFR area's memory map (2)

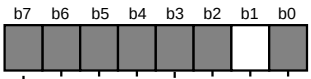
2.4 Memory assignment



Processor mode register 0 (Address $5E_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Fix this bit to "0."		0	RW
1	Nothing is assigned. The value is "1" at reading.		1	–
2	Wait bit	0 : Software Wait is inserted when accessing external area. 1 : No software Wait is inserted when accessing external area.	0	RW
3	Software reset bit	The microcomputer is reset by writing "1" to this bit. The value is "0" at reading.	0	WO
4	Interrupt priority detection time select bits	b5 b4 0 0 : 7 cycles of φ 0 1 : 4 cycles of φ 1 0 : 2 cycles of φ 1 1 : Do not select.	0	RW
5			0	RW
6	Fix this bit to "0."		0	RW
7	Stack bank select bit	0 : Bank 0_{16} 1 : Bank FF_{16}	0	RW

: Bits 0 to 6 are not used for the memory assignment.



Processor mode register 1 (Address $5F_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Nothing is assigned.		Undefined	–
1	Internal RAM area select bit **(Notes 1, 2)**	0 : 512 bytes (addresses 80_{16} to $27F_{16}$) 1 : 1024 bytes (addresses 80_{16} to $47F_{16}$)	0	RW
7 to 2	Nothing is assigned.		Undefined	–

Notes 1: For the M37721S1BFP, fix bit 1 to "0."
2: For the M37721S2BFP, set bit 1 before setting the stack pointer.

EOL announced

Fig. 2.4.4 Structure of processor mode registers 0, 1

2.5 Bus access right

The M37721's bus is used for DRAMC, Hold function, and DMAC besides CPU. When the bus requests of two or more source are detected at the same timing, the highest bus access priority levels get the access right.

The bus priority levels are fixed by hardware. Additionally the bus use state is output from the status signal output pins ST0 and ST1. Table 2.5.1 lists the bus use priority levels and the status signals depending on the bus use state.

Table 2.5.1 Bus use priority levels and status signals depending on bus use state

Bus use priority levels	Bus use state	Status signal	
		ST1	ST0
1 (Highest)	DRAM refresh	0	0
2	Hold	0	1
3	DMAC	1	0
4 (Lowest)	CPU (Including the term that CPU does not use the bus during calculation etc.)	1	1

For details, refer to section **"13.2.1 Bus access control circuit"** and chapter for each peripheral devices.

EOL announced

MEMORANDUM

EOL announced

CHAPTER 3
CONNECTION WITH EXTERNAL DEVICES

3.1 Signals required for accessing external devices
3.2 Software Wait
3.3 Ready function
3.4 Hold function
[Precautions for Hold function]

EOL announced

3.1 Signals required for accessing external devices

The functions and operations of the signals which are required for accessing the external devices are described below.

When connecting an external device that requires long access time, refer to sections **"3.2 Software Wait," "3.3 Ready function,"** and **"3.4 Hold function,"** as well as this section. When the external DRAM is controlled by using DRAM controller, refer to **"CHAPTER 14. DRAM CONTROLLER."**

3.1.1 Descriptions of signals

Figure 3.1.1 shows the pin configurations when the external data bus width is 16 bits and 8 bits.

(1) External buses (A_0–A_7, A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7)

The external area is specified by the address (A_0–A_{23}) output.

The A_8–A_{23} pins of the external address bus and the D_0–D_{15} pins of the external data bus are assigned to the same pins.

When the BYTE pin level, described later, is "L" (external data bus width is 16 bits), the A_8/D_8–A_{15}/D_{15} and A_{16}/D_0–A_{23}/D_7 pins perform address output and data input/output with time-sharing.

When the BYTE pin level is "H" (external data bus width is 8 bits), the A_{16}/D_0–A_{23}/D_7 pins perform address output and data input/output with time-sharing, and the A_8–A_{15} pins output the address.

(2) External data bus width switching signal (BYTE pin level)

This signal is used to select the external data bus width from 8 bits and 16 bits. The width is 16 bits when the level is "L," and 8 bits when the level is "H." Fix this signal to either "H" or "L" level. This signal is valid only for the external area. (When accessing the internal area, the data bus width is always 16 bits.)

(3) Enable signal ($\overline{E}$)

This signal becomes "L" level while reading or writing data from and to the data bus. (Refer to **"Table 3.1.1."**)

(4) Read/Write signal (R/$\overline{W}$)

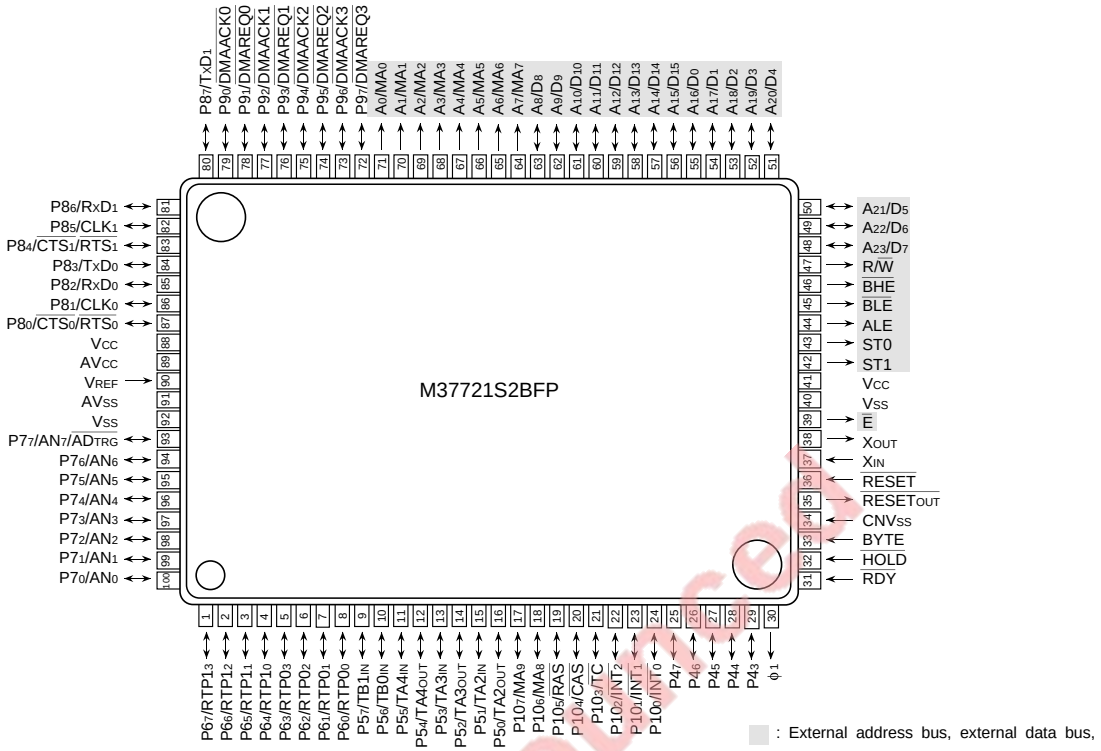
This signal indicates the state of the data bus. This signal becomes "L" level while writing data to the data bus. Table 3.1.1 lists the state of the data bus indicated with the $\overline{E}$ and R/$\overline{W}$ signals.

Table 3.1.1 State of data bus indicated with $\overline{E}$ and R/$\overline{W}$ signals

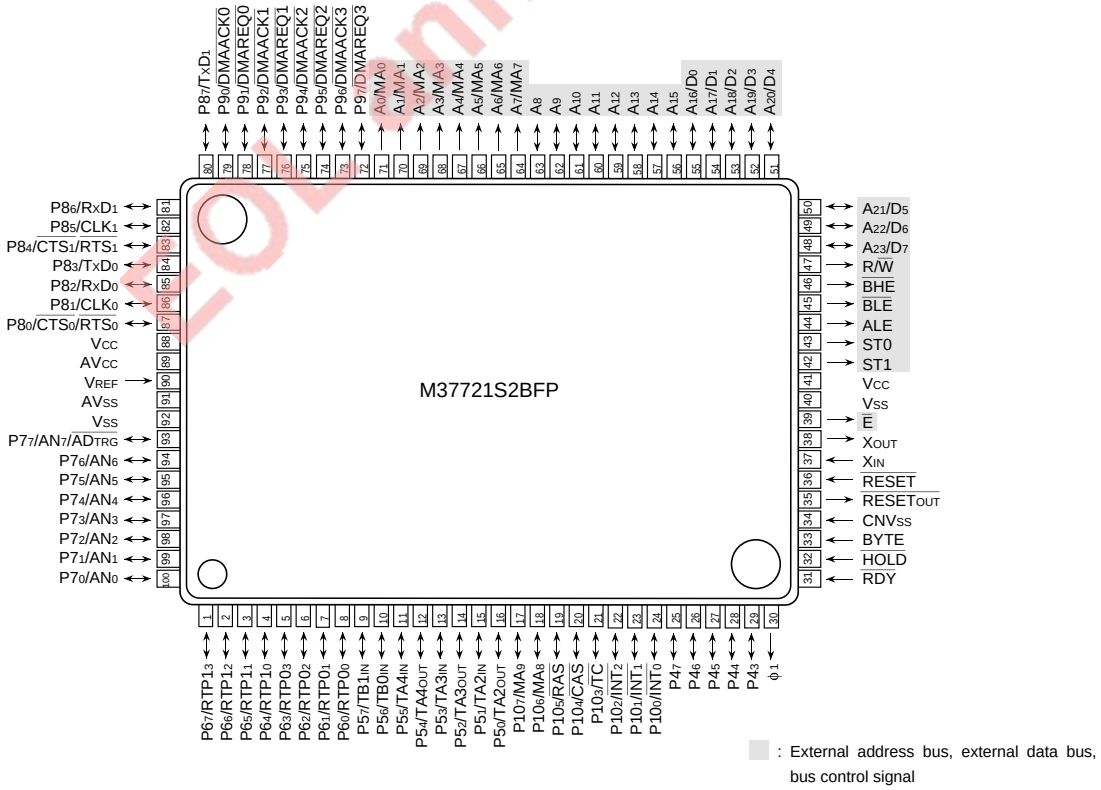
$\overline{E}$	R/$\overline{W}$	State of data bus
H	H	Not used
	L	
L	H	Read data
	L	Write data

EOL announced

●External data bus width = 16 bits (BYTE = “L”)



●External data bus width = 8 bits (BYTE = “H”)



Note: For the DRAM control signals, refer to **“CHAPTER 14. DRAM CONTROLLER.”**

Fig. 3.1.1 Pin configurations when external data bus width is 16 bits and 8 bits (top view)

3.1 Signals required for accessing external devices

(5) Byte low enable signal ($\overline{BLE}$), Byte high enable signal ($\overline{BHE}$)

The $\overline{BLE}$ signal indicates the access to an even address. This signal becomes "L" level when accessing only an even address or when simultaneously accessing both an even and an odd address. The $\overline{BHE}$ signal indicates the access to an odd address. This signal becomes "L" level when accessing only an odd address or when simultaneously accessing both an odd and an even address.

These signals are used to connect memories or I/O devices of which data bus width is 8 bits when the external data bus width is 16 bits. Table 3.1.2 lists levels of the $\overline{BLE}$ and $\overline{BHE}$ signals and access addresses.

Table 3.1.2 Levels of $\overline{BLE}$ and $\overline{BHE}$ signals and access addresses

Access address	Even and odd addresses (Simultaneous 2-byte access)	Even address (1-byte access)	Odd address (1-byte access)
$\overline{BLE}$	L	L	H
$\overline{BHE}$	L	H	L

(6) Address latch enable signal (ALE)

This signal is used to latch the address from the multiplexed signal, which consists of the address and data. (This multiplexed signal is input to or output from the A_8/D_8–A_{15}/D_{15} and A_{16}/D_0–A_{23}/D_7 pins.) When the ALE signal is "H," latch the address and simultaneously output the addresses. When this signal is "L," retain the latched address.

(7) Ready function-related signal ($\overline{RDY}$)

This is the signal to use Ready function (Refer to section **"3.3 Ready function."**)

(8) Hold function-related signal ($\overline{HOLD}$)

This is the signal to use Hold function. (Refer to section **"3.4 Hold function."**)

(9) Status signals (ST0, ST1)

These signals indicate the bus use status. Table 3.1.3 lists the bus use status indicated by the ST0 and ST1 signals.

(10) Clock ϕ_1

This signal has the same period as ϕ.

EOL announced

Table 3.1.3 Bus use status indicated by ST0 and ST1 signals

ST1	ST0	Bus use status
L	L	DRAM refresh
L	H	Hold
H	L	DMA
H	H	CPU

3.1.2 Operation of bus interface unit (BIU)

Figures 3.1.2 and 3.1.3 show the examples of operating waveforms of the signals input from or output to the external when accessing external devices. The following explains these waveforms, being compared with the basic operating waveform. (Refer to section **"2.2.3 Operation of bus interface unit (BIU)."**)

(1) When fetching instructions into instruction queue buffer

① When the instruction which is next fetched is located at an even address

When the external data bus width is 16 bits, the BIU fetches 2 bytes of the instruction at a time with waveform (a). When the external data bus width is 8 bits, the BIU fetches only 1 byte of the instruction with the first half of waveform (e).

② When the instruction which is next fetched is located at an odd address

When the external data bus width is 16 bits, the BIU fetches only 1 byte of the instruction with waveform (d). When the external data bus width is 8 bits, the BIU fetches only 1 byte of the instruction with the first half of waveform (f).

When a branch to an odd address is caused by a branch instruction etc. with the 16-bit external data bus width, the BIU first fetches 1 byte of the instruction with waveform (d), and after that, fetches instructions in a unit of 2 bytes with waveform (a).

(2) When reading or writing data from and to memories or I/O devices

① When accessing 16-bit data which begins at an even address, waveform (a) or (e) is applied.
② When accessing 16-bit data which begins at an odd address, waveform (b) or (f) is applied.
③ When accessing 8-bit data at an even address, waveform (c) or the first half of (e) is applied.
④ When accessing 8-bit data at an odd address, waveform (d) or the first half of (f) is applied.

For instructions that are affected by the data length flag (m) and the index register length flag (x), operation ① or ② is applied when flag m or x = "0"; operation ③ or ④ is applied when flag m or x = "1."

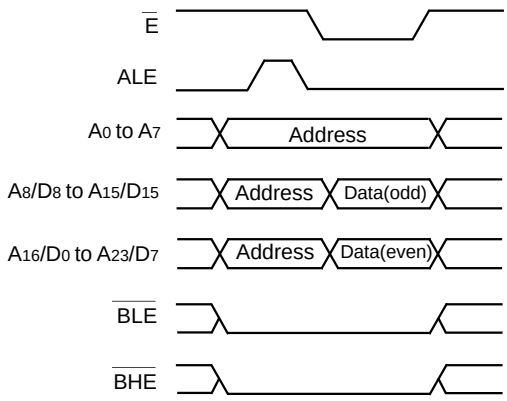
The setup of flags m and x and the selection of the external data bus width do not affect each other.

EOL announced

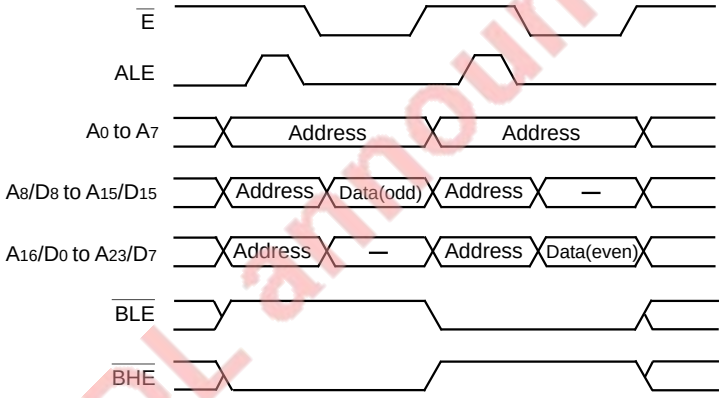
● External data bus width = 16 bits (BYTE = “L”)

<16-bit data access>

(a) Access beginning at even address



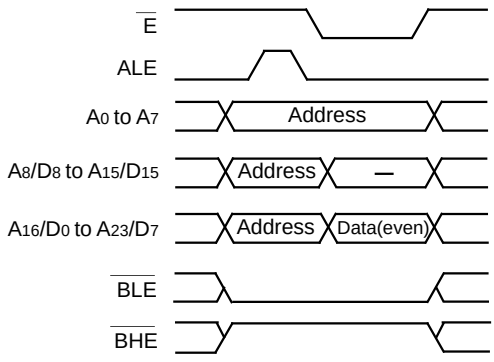
(b) Access beginning at odd address



EOL announced

<8-bit data access>

(c) Access to even address



(d) Access to odd address

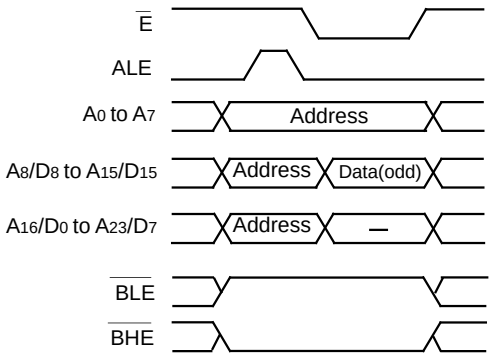
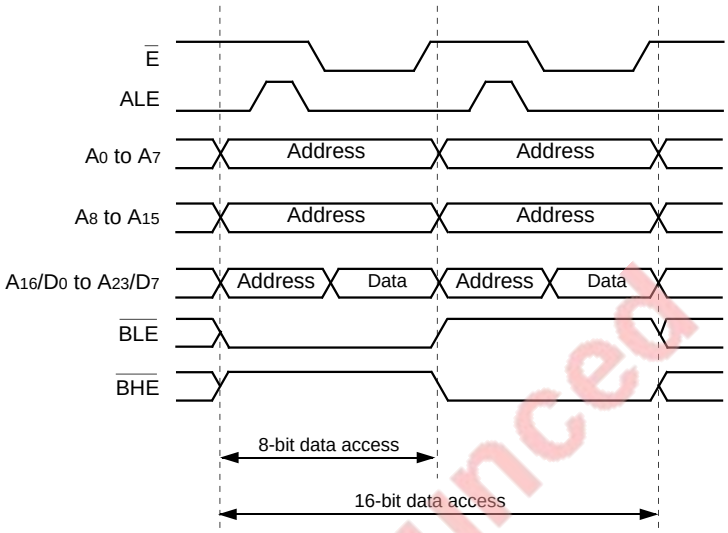


Fig. 3.1.2 Examples of operating waveforms of signals input from or output to the external (1)

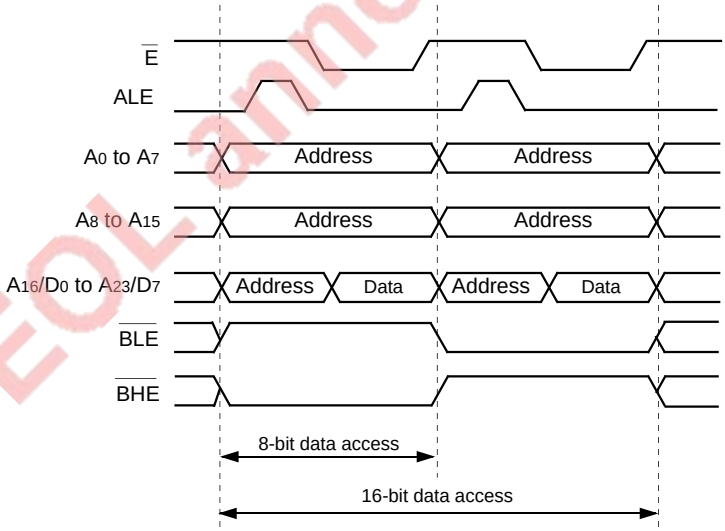
● External data bus width = 8 bits (BYTE = “H”)

<8/16-bit data access>

(e) Access beginning at even address



(f) Access beginning at odd address



EOL announced

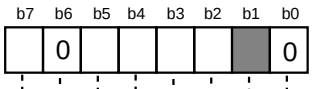
Note: When accessing 16-bit data, 2 times of access are performed; the low-order 8 bits are accessed first, and after that, the high-order 8 bits are accessed.

Fig. 3.1.3 Examples of operating waveforms of signals input from or output to the external (2)

3.2 Software Wait

Software Wait provides a function to facilitate access to external devices that require long access time. To select the software Wait, use the wait bit (bit 2 at address $5E_{16}$). Figure 3.2.1 shows the structure of processor mode register 0 (address $5E_{16}$). Figure 3.2.2 shows examples of bus timing when software Wait is used.

Software Wait is valid only for the eternal area. The internal area is always accessed with no Wait.



Processor mode register 0 (Address $5E_{16}$)

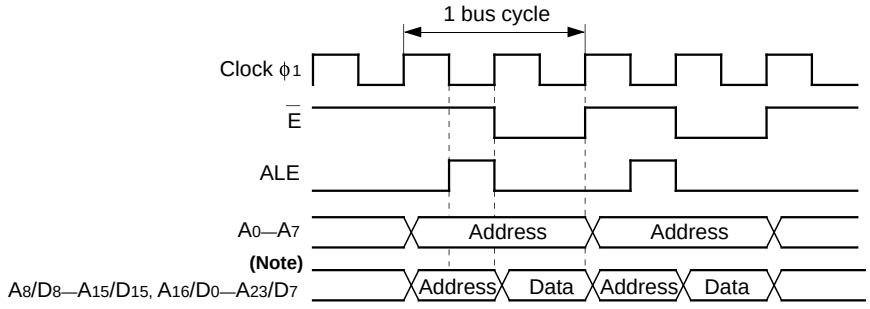
Bit	Bit name	Functions	At reset	RW
0	Fix this bit to "0."		0	RW
1	Nothing is assigned. The value is "1" at reading.		1	–
2	Wait bit	0 : Software Wait is inserted when accessing external area. 1 : No software Wait is inserted when accessing external area.	0	RW
3	Software reset bit	The microcomputer is reset by writing "1" to this bit. The value is "0" at reading.	0	WO
4	Interrupt priority detection time select bits	b5 b4 0 0 : 7 cycles of ϕ 0 1 : 4 cycles of ϕ 1 0 : 2 cycles of ϕ 1 1 : Do not select.	0	RW
5			0	RW
6	Fix this bit to "0."		0	RW
7	Stack bank select bit	0 : Bank 0_{16} 1 : Bank FF_{16}	0	RW

: Bits 0, 1, and 3 to 6 are not used for accessing external area.

EOL announced

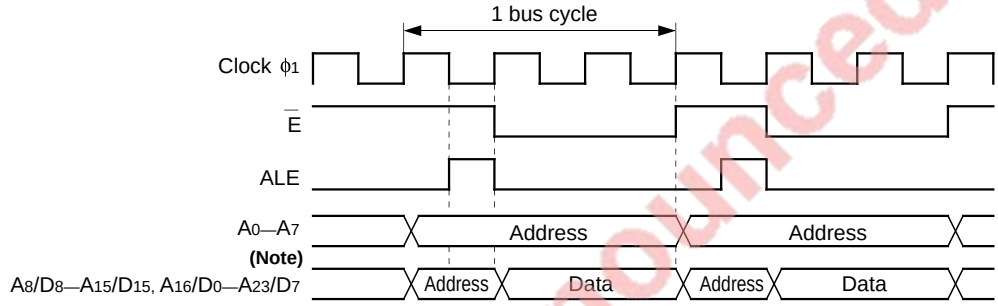
Fig. 3.2.1 Structure of processor mode register 0

<No Wait>



●Internal areas are always accessed with this waveform.

<Wait>



Note: When the external data bus is 8 bits wide (BYTE = "H"), A_8/D_8 to A_{15}/D_{15} operate with the same bus timing as A_0 to A_7.

Fig. 3.2.2 Examples of bus timing when software Wait is used (BYTE = "L")

EOL announced

3.3 Ready function

Ready function provides the function to facilitate access to external devices that require long access time. The microcomputer enters Ready state by input of "L" level to the $\overline{RDY}$ pin and retains this state while the level of the $\overline{RDY}$ pin is at "L." Table 3.3.1 lists the microcomputer's state in Ready state.
In Ready state, the oscillator's oscillation does not stop. Accordingly, the internal peripheral devices can operate. Ready function is valid for the internal and external areas.

Table 3.3.1 Microcomputer's state in Ready state

Item	State
Oscillation	Operating
ϕ_{CPU}, ϕ	Stopped at "L"
Pins A_0 to A_7, A_8/D_8 to A_{15}/D_{15}, A_{16}/D_0 to A_{23}/D_7, $\overline{E}$, $R/\overline{W}$, $\overline{BHE}$, $\overline{BLE}$, ST0, ST1, ALE	Retain the state when Ready request was accepted.
Pins $P4_3$ to $P4_7$, P5 to P10 **(Note)**	
Pin ϕ_1	Outputs clock ϕ_1.
Watchdog timer	Operating

Note: This applies when this functions as a programmable I/O port.

EOL announced

3.3.1 Operation description

The input level of the $\overline{RDY}$ pin is judged at the falling edge of clock ϕ_1. When "L" level is detected at this point, the microcomputer enters Ready state. (This is called "Acceptance of Ready request.")
In Ready state, the input level of the $\overline{RDY}$ pin is judged at every falling edge of clock ϕ_1. When "H" level is detected at this point, the microcomputer terminates Ready state at the next rising edge of clock ϕ_1.
Figure 3.3.1 shows timing of acceptance of Ready request and termination of Ready state. Refer also to section "**16.1 Memory connection**" for usage of Ready function.

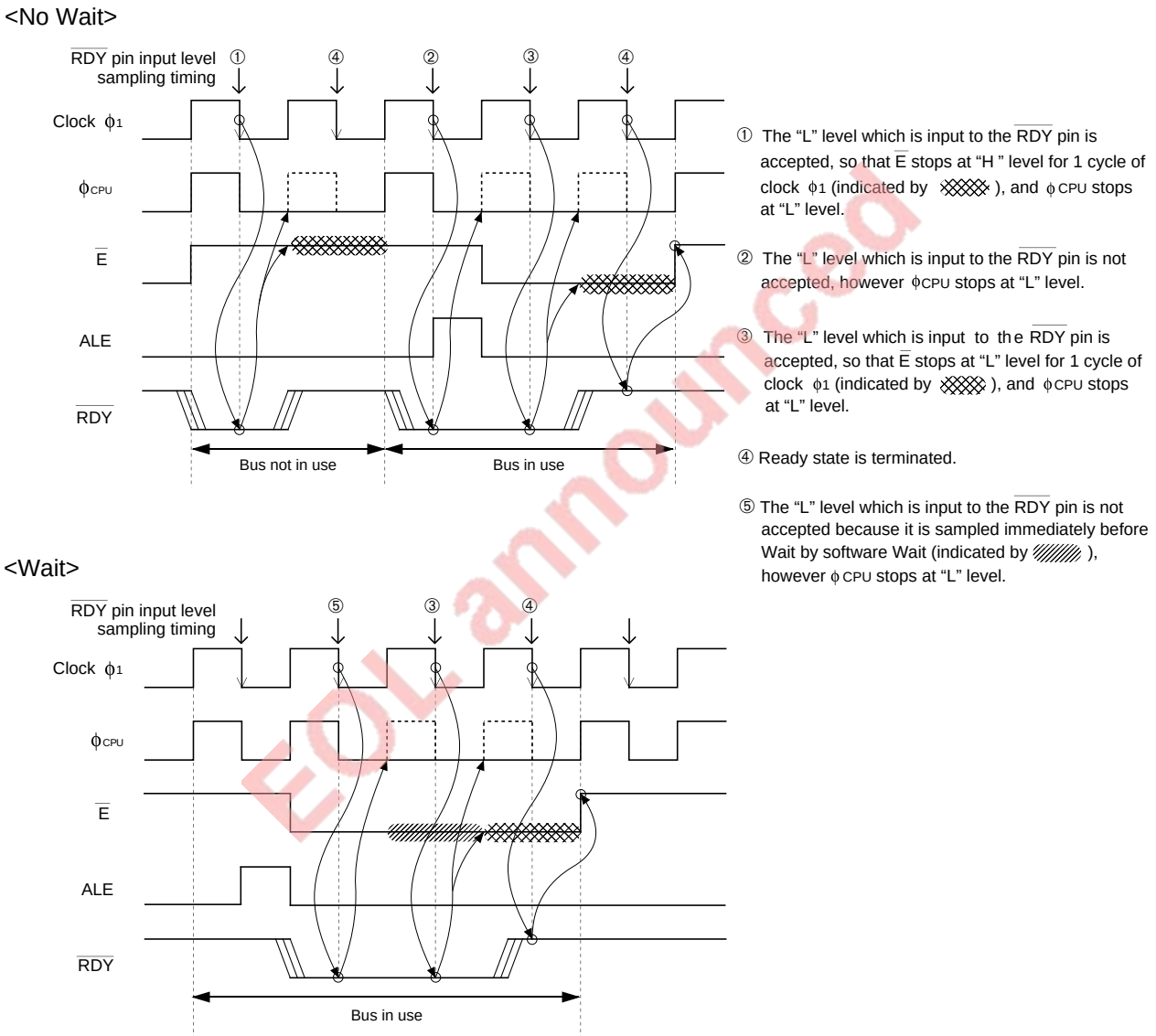


Fig. 3.3.1 Timing of acceptance of Ready request and termination of Ready state

EOL announced

3.4 Hold function

When composing the external circuit which accesses the bus without using the central processing unit (CPU), Hold function is used to generate a timing for transferring the right to use the bus from the CPU to the external circuit.

The microcomputer enters Hold state by input of "L" level to the $\overline{\text{HOLD}}$ pin and retains this state while the level of the $\overline{\text{HOLD}}$ pin is at "L." Table 3.4.1 lists the microcomputer's state in Hold state.

In Hold state, the oscillation of the oscillator does not stop. Accordingly, the internal peripheral devices can operate. However, Watchdog timer stops operating.

Table 3.4.1 Microcomputer's state in Hold state

Item	State
Oscillation	Operating
ϕ	Operating
ϕ_{CPU}	Stopped at "L"
$\overline{E}$	Stopped at "H"
Pins A_0 to A_7, A_8/D_8 to A_{15}/D_{15}, A_{16}/D_0 to A_{23}/D_7, R/$\overline{W}$, $\overline{BHE}$, $\overline{BLE}$	Floating
Pins ALE, ST1	Output "L" level.
Pin ST0	Outputs "H" level.
Pin ϕ_1	Outputs clock ϕ_1.
Pins $P4_3$ to $P4_7$, P5 to P10 **(Note)**	Retain the state when Hold request was accepted.
Watchdog timer	Stopped

Note: This applies when this functions as a programmable I/O port.

3.4.1 Operation description

Judgment of the $\overline{\text{HOLD}}$ pin input level is performed at every falling edge of ϕ_1. When "L" level is detected at judgment of the input level, bus request (Hold) becomes "1," when "H" level is detected, bus request (Hold) becomes "0."

Bus request (Hold) is sampled within a period when the bus request sampling signal is "1" and bus request is accepted when there is no bus request (DRAMC). (This is called "Acceptance of Hold request.") For bus request, refer to section **"13.2.1 Bus access control circuit."**

When Hold request is accepted, ϕ_{CPU} stops at "L" level at the next rising edge of ϕ and the ST0 pin's level becomes "H," the ST1 pin's level becomes "L." When 1 cycle of ϕ has passed after the levels of the ST0 and ST1 pins are changed, the R/$\overline{W}$, $\overline{BHE}$, $\overline{BLE}$ pins and the external bus enter the floating state.

In Hold state, when the $\overline{\text{HOLD}}$ pin's input level becomes "H," the ST0 and ST1 pins' levels are changed at the next rising edge of ϕ. When 1 cycle of ϕ has passed after the levels of the ST0 and ST1 pins are changed, the microcomputer terminates Hold state.

Figures 3.4.1 to 3.4.3 show timing of acceptance of Hold request and termination of Hold state.

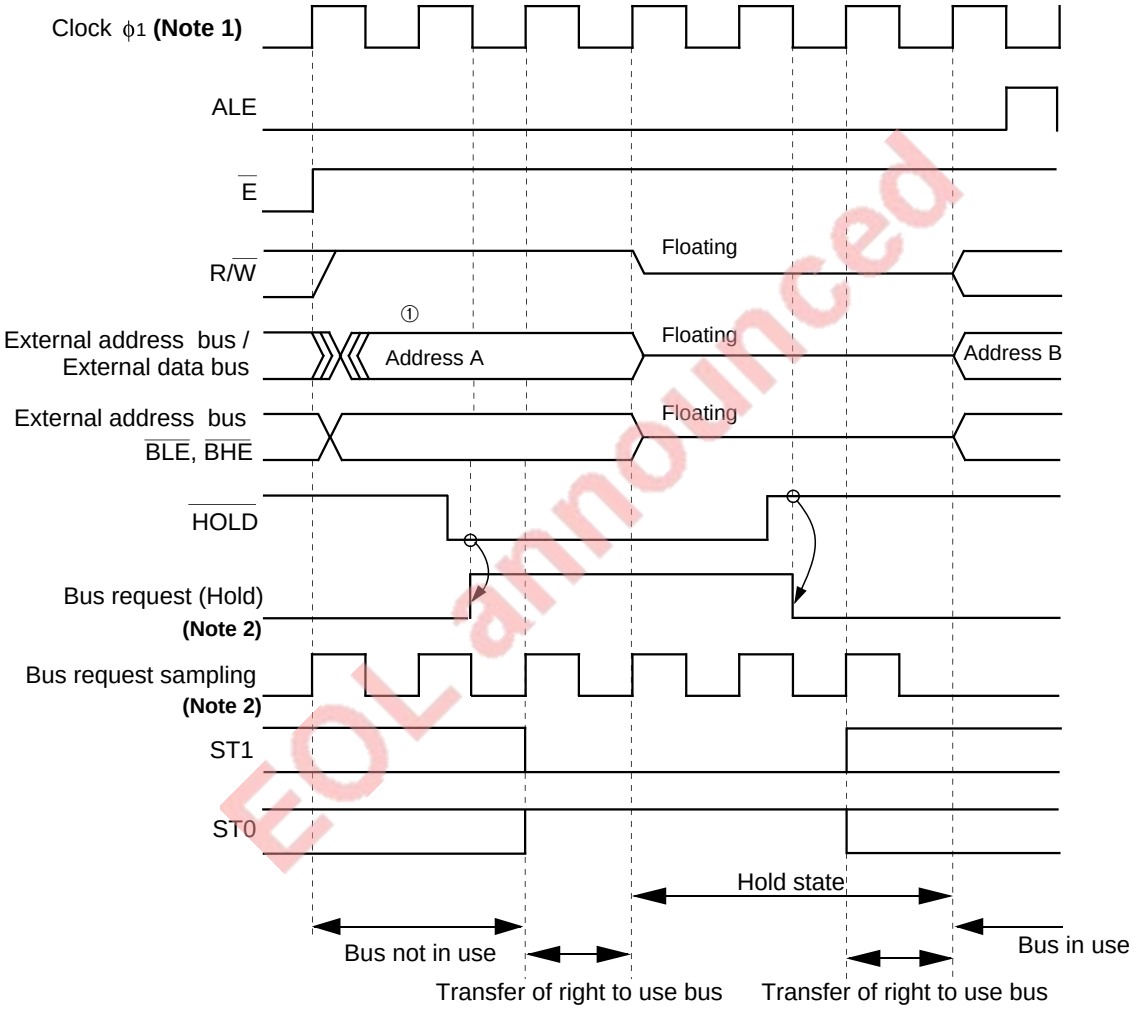
Note: ϕ has the same polarity and the same frequency as clock ϕ_1.
However, ϕ stops by acceptance of Ready request, or executing the **STP** or **WIT** instruction. Accordingly, judgment of the input level of the $\overline{\text{HOLD}}$ pin is not performed during Ready state.

FOR annotated

<When inputting "L" level to $\overline{\text{HOLD}}$ pin while bus is unused>

● State when inputting "L" level to $\overline{\text{HOLD}}$ pin

External data bus	Data length	External data bus width
Unused	8	8, 16
	16	8, 16



EOL announced

① This is the period in which the bus is not used, so that not a new address but the address which was output immediately before is output again.

Notes 1: Clock φ1 has the same polarity and the same frequency as φ.
Timing of signals to be input from or output to the external is ordained on the basis of clock φ1.
2: Bus request (Hold) and bus request sampling are internal signals.

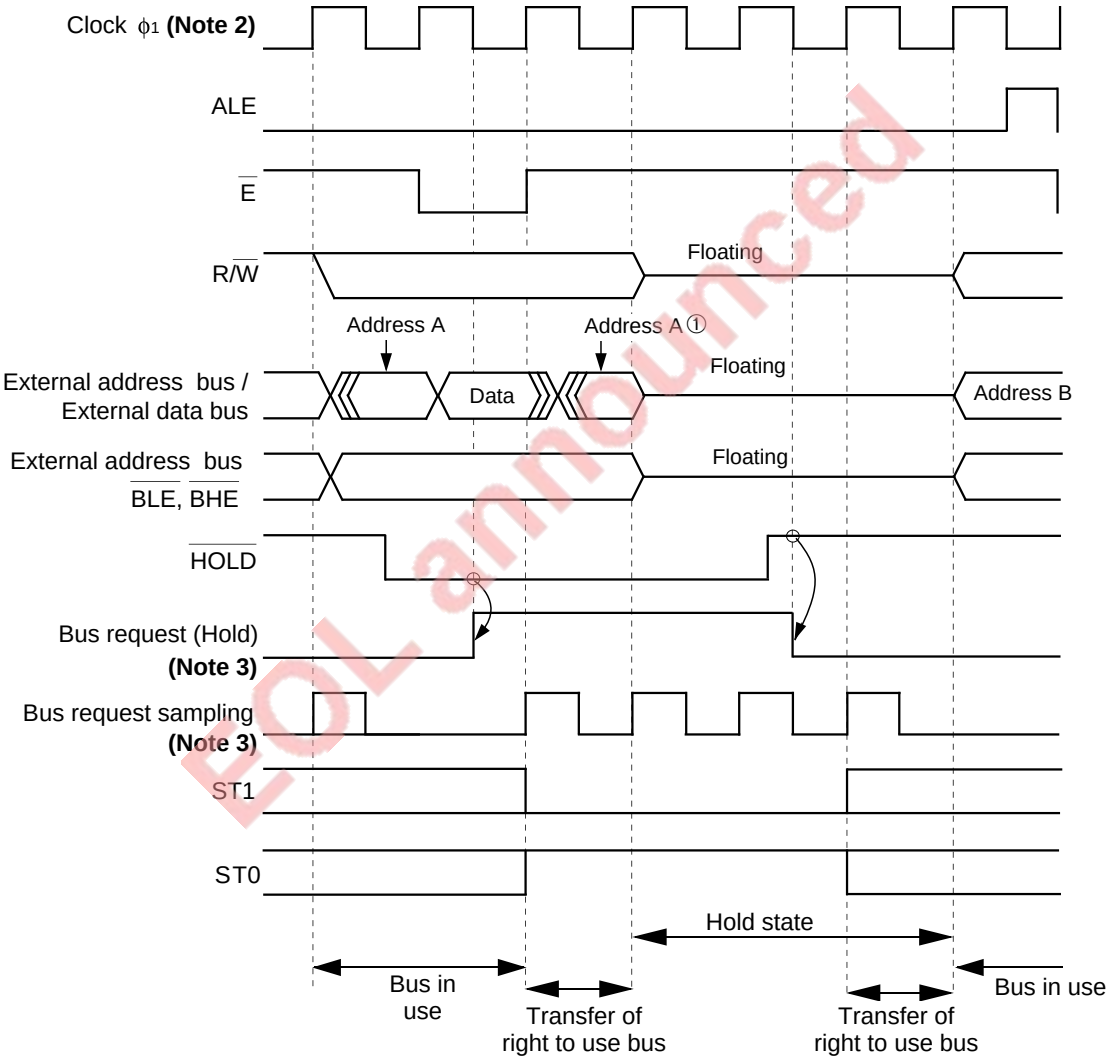
Fig. 3.4.1 Timing of acceptance of Hold request and termination of Hold state (1)

3.4 Hold function

<When inputting "L" level to $\overline{\text{HOLD}}$ pin while bus is used; when data access is completed with 1-bus cycle>

● State when inputting "L" level to $\overline{\text{HOLD}}$ pin

External data bus	Data length	External data bus width
Used	8	8, 16
	16	16 (Access beginning at even address)



EOL announced

① When a Hold request is accepted, not a new address but the address which was output immediately before is output again.

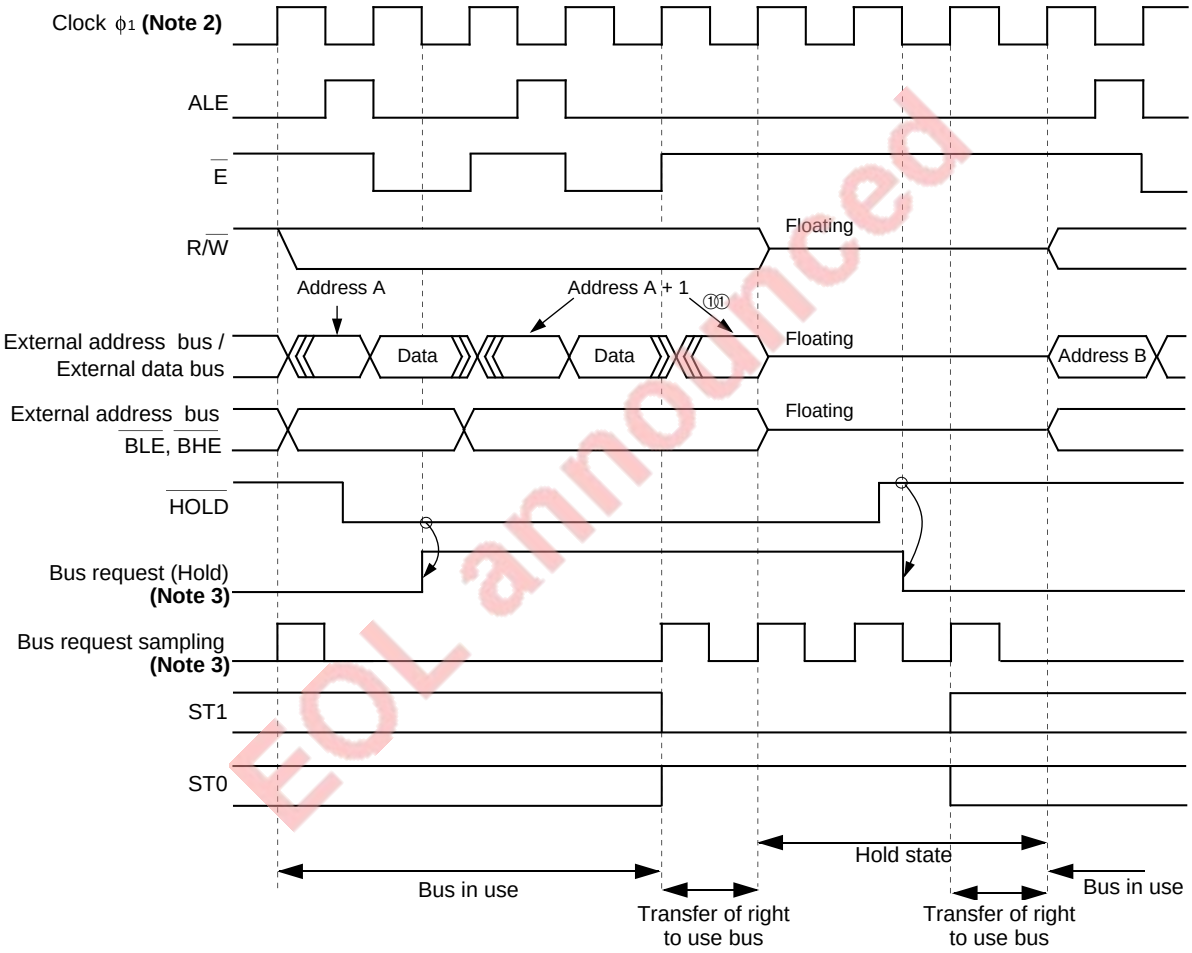
Notes 1: The above diagram shows the case of no Wait.
2: Clock φ1 has the same polarity and the same frequency as φ. Timing of signals to be input from or output to the external is ordained on the basis of clock φ1.
3: Bus request (Hold) and bus request sampling are internal signals.

Fig. 3.4.2 Timing of acceptance of Hold request and termination of Hold state (2)

<When inputting "L" level to $\overline{\text{HOLD}}$ pin while bus is used; when data access is completed with continuous 2-bus cycle>

● State when inputting "L" level to $\overline{\text{HOLD}}$ pin

External data bus	Data length	External data bus width
Used	16	8
		16 (Access beginning at odd address)



① When a Hold request is accepted, not a new address but the address which was output immediately before is output again.

Notes 1: The above diagram shows the case of 2-φ access in low-speed running.
2: Clock φ1 has the same polarity and the same frequency as φ.
Timing of signals to be input from or output to the external is ordained on the basis of clock φ1.
3: Bus request (Hold) and bus request sampling are internal signals.

Fig. 3.4.3 Timing of acceptance of Hold request and termination of Hold state (3)

EOL announced

3.4 Hold function

[Precautions for Hold function]

When a DRAM refresh request occurs in Hold state, DRAM refresh is performed immediately because the bus use priority level of DRAM refresh is higher than that of Hold function.

EOL announced

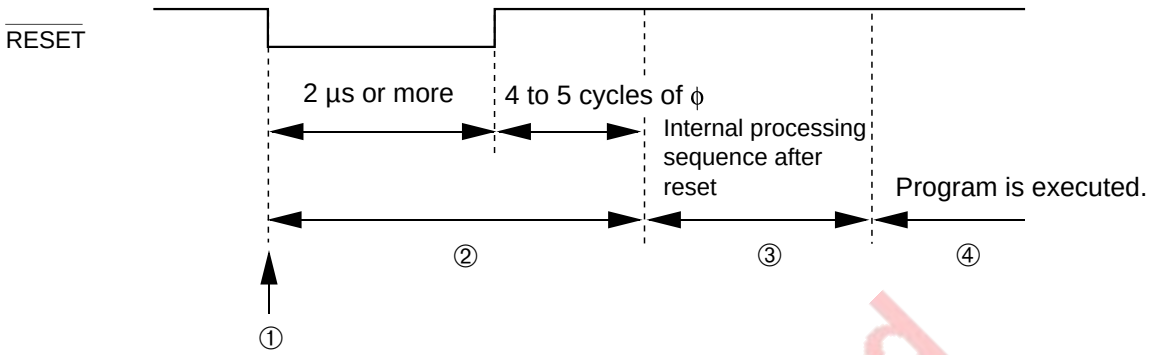
CHAPTER 4
RESET

4.1 Hardware reset
4.2 Software reset

EOL announced

4.1 Hardware reset

When the power source voltage satisfies the microcomputer's recommended operating conditions, the microcomputer is reset by supplying "L" level to the $\overline{\text{RESET}}$ pin. This is called a hardware reset. Figure 4.1.1 shows an example of hardware reset timing.



Note: When the clock is stably supplied. (Refer to section **"4.1.4 Time supplying "L" level to RESET pin."**)

Fig. 4.1.1 Example of hardware reset timing

The following explains how the microcomputer operates in periods ① to ④ above.

① After supplying "L" level to the $\overline{\text{RESET}}$ pin, the microcomputer initializes pins within a period of several ten ns. (Refer to **"Table 4.1.1."**)
② While the $\overline{\text{RESET}}$ pin is "L" level and within a period of 4 to 5 cycles of ϕ after the $\overline{\text{RESET}}$ pin goes from "L" to "H," the microcomputer initializes the central processing unit (CPU) and SFR area. At this time, the contents of the internal RAM area become undefined (except when Stop or Wait mode is terminated). (Refer to **"Figures 4.1.3 to 4.1.9."**)
③ After ②, the microcomputer performs "Internal processing sequence after reset." (Refer to **"Figure 4.1.10."**)
④ The microcomputer executes a program beginning with the address set into the reset vector addresses ($FFFE_{16}$ and $FFFF_{16}$).

EOL announced

4.1.1 Pin state

Table 4.1.1 lists the microcomputer's pin state while $\overline{\text{RESET}}$ pin is at "L" level. Figure 4.1.2 shows the $\overline{\text{RESET}}_{\text{OUT}}$ output retaining timing.

Table 4.1.1 Pin state while $\overline{\text{RESET}}$ pin is at "L" level

Pin (Bus, Port) name	Pin state
A_0/MA_0–A_7/MA_7, A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7, $\overline{\text{BHE}}$, $\overline{\text{BLE}}$	Outputs "H" or "L" level.
R/$\overline{\text{W}}$, $\overline{\text{E}}$, ST0, ST1	Outputs "H" level.
ALE, $\overline{\text{RESET}}_{\text{OUT}}$	Outputs "L" level.
ϕ_1	Outputs ϕ_1.
$\overline{\text{HOLD}}$, $\overline{\text{RDY}}$, $P4_3$–$P4_7$, P5–P10	Floating.

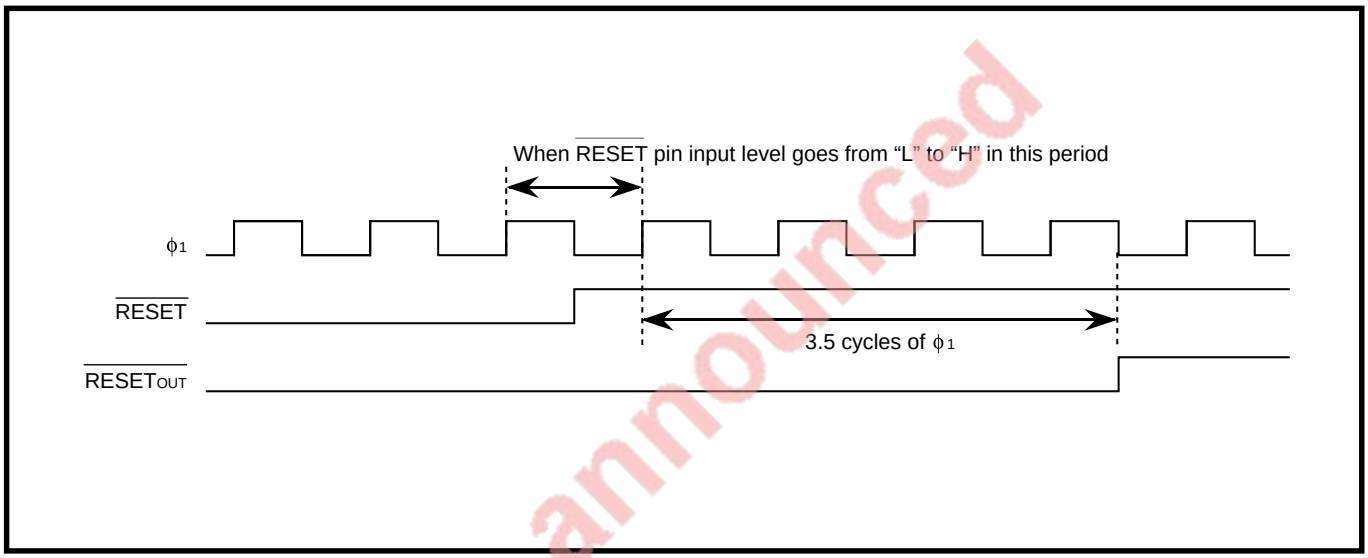


Fig. 4.1.2 $\overline{\text{RESET}}_{\text{OUT}}$ output retaining timing

EOL announced

4.1 Hardware reset

4.1.2 State of CPU, SFR area, and internal RAM area

Figure 4.1.3 shows the state of the CPU registers immediately after reset. Figures 4.1.4 to 4.1.9 show the state of the SFR and internal RAM areas immediately after reset.

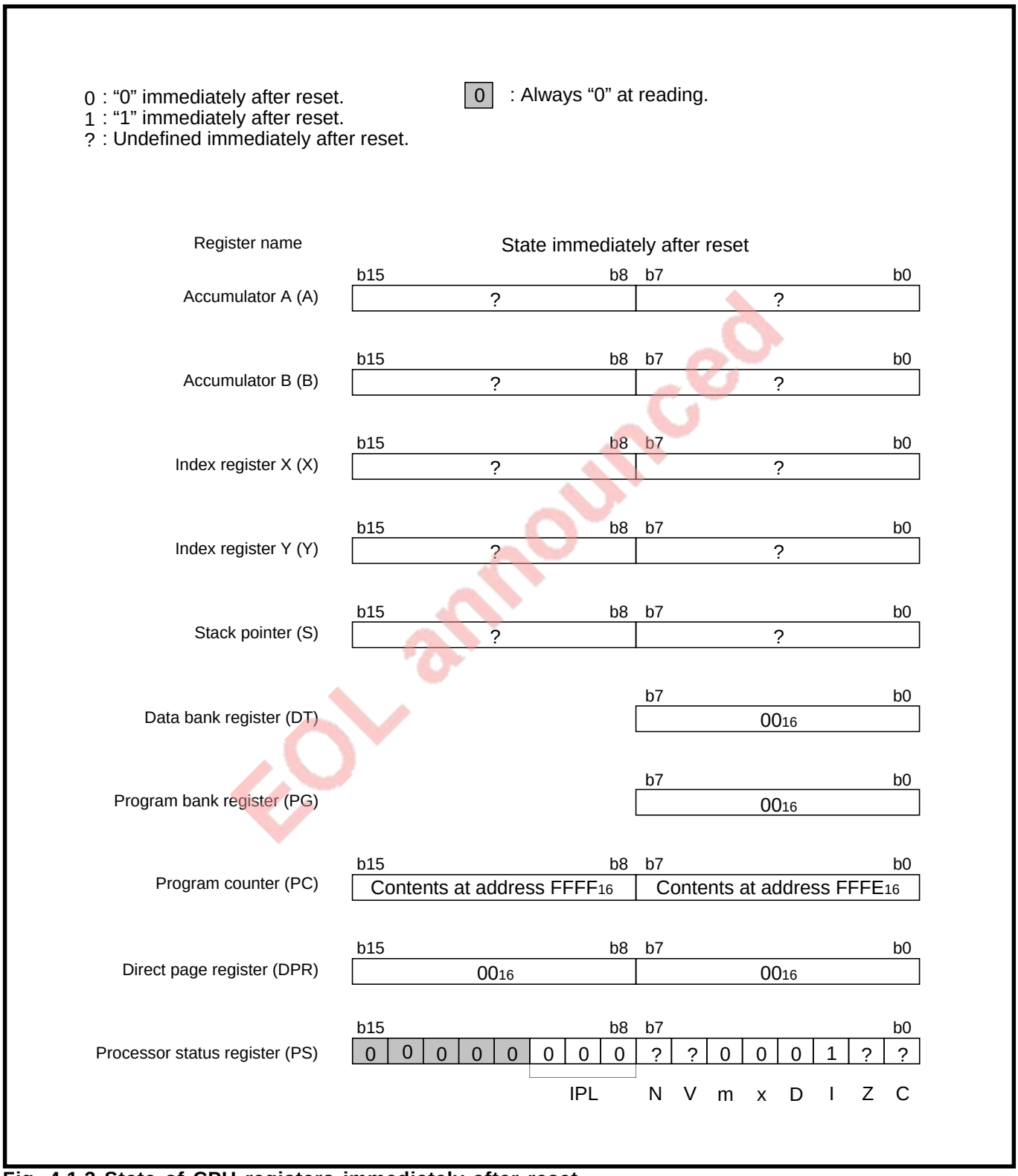


Fig. 4.1.3 State of CPU registers immediately after reset

EOL announced

●SFR area (0_{16} to $7F_{16}$, $1FC0_{16}$ to $1FFF_{16}$)

Access characteristics

RW: It is possible to read the bit state at reading. The written value becomes valid.
RO : It is possible to read the bit state at reading. The written value becomes invalid.
WO : The written value becomes valid. It is impossible to read the bit state.
: Nothing is assigned. It is impossible to read the bit state. The written value becomes invalid.

State immediately after reset

0 : "0" immediately after reset.
1 : "1" immediately after reset.
? : Underfined immediately after reset.

0 : Always "0" at reading.
1 : Always "1" at reading.
? : Always undefined at reading.
0 : "0" immediately after reset. Fix this bit to "0."

Address	Register name	Access characteristics b7 b0	State immediately after reset b7 b0
0_{16}			?
1_{16}			?
2_{16}			?
3_{16}			?
4_{16}			?
5_{16}			?
6_{16}			?
7_{16}			?
8_{16}			?
9_{16}			?
A_{16}	Port P4 register	RW	? 0 0 0
B_{16}	Port P5 register	RW	?
C_{16}	Port P4 direction register	RW	0 0 0 0 0 0 0 0
D_{16}	Port P5 direction register	RW	00_{16}
E_{16}	Port P6 register	RW	?
F_{16}	Port P7 register	RW	?
10_{16}	Port P6 direction register	RW	00_{16}
11_{16}	Port P7 direction register	RW	00_{16}
12_{16}	Port P8 register	RW	?
13_{16}	Port P9 register	RW	?
14_{16}	Port P8 direction register	RW	00_{16}
15_{16}	Port P9 direction register	RW	00_{16}
16_{16}	Port P10 register	RW	?
17_{16}			?
18_{16}	Port P10 direction register	RW	00_{16}
19_{16}			?
$1A_{16}$	Pulse output data register 0	WO	?
$1B_{16}$			?
$1C_{16}$	Pulse output data register 1	WO	?
$1D_{16}$			?
$1E_{16}$	A-D control register	RW	0 0 0 0 0 ? ? ?
$1F_{16}$	A-D sweep pin select register	RW	? ? ? ? ? ? 1 1

EOL announced

Fig. 4.1.4 State of SFR and internal RAM areas immediately after reset (1)

4.1 Hardware reset

Address	Register name	Access characteristics b7 b0	State immediately after reset b7 b0
2016	A-D register 0	RO	?
2116			?
2216	A-D register 1	RO	?
2316			?
2416	A-D register 2	RO	?
2516			?
2616	A-D register 3	RO	?
2716			?
2816	A-D register 4	RO	?
2916			?
2A16	A-D register 5	RO	?
2B16			?
2C16	A-D register 6	RO	?
2D16			?
2E16	A-D register 7	RO	?
2F16			?
3016	UART0 transmit/receive mode register	RW	0016
3116	UART0 baud rate register	WO	?
3216	UART0 transmit buffer register	WO	?
3316		WO	?
3416	UART0 transmit/receive control register 0	RO RW	? ? ? ? 1 0 0 0
3516	UART0 transmit/receive control register 1	RO RW RO RW	0 0 0 0 0 0 1 0
3616	UART0 receive buffer register	RO	?
3716		RO	0 0 0 0 0 0 0 ?
3816	UART1 transmit/receive mode register	RW	0016
3916	UART1 baud rate register	WO	?
3A16	UART1 transmit buffer register	WO	?
3B16		WO	?
3C16	UART1 transmit/receive control register 0	RO RW	? ? ? ? 1 0 0 0
3D16	UART1 transmit/receive control register 1	RO RW RO RW	0 0 0 0 0 0 1 0
3E16	UART1 receive buffer register	RO	?
3F16		RO	0 0 0 0 0 0 0 ?

EOL announced

Fig. 4.1.5 State of SFR and internal RAM areas immediately after reset (2)

Address	Register name	Access characteristics (b7 … b0)	State immediately after reset (b7 … b0)
4016	Count start register	RW	0016
4116			?
4216	One-shot start register	WO	? 0 0 0 0 0
4316			?
4416	Up-down register	WO RW	0 0 0 0 0 0 0 0
4516			?
4616	Timer A0 register	RW	?
4716		RW	?
4816	Timer A1 register	RW	?
4916		RW	?
4A16	Timer A2 register	**(Note 1)**	?
4B16		**(Note 1)**	?
4C16	Timer A3 register	**(Note 1)**	?
4D16		**(Note 1)**	?
4E16	Timer A4 register	**(Note 1)**	?
4F16		**(Note 1)**	?
5016	Timer B0 register	**(Note 2)**	?
5116		**(Note 2)**	?
5216	Timer B1 register	**(Note 2)**	?
5316		**(Note 2)**	?
5416	Timer B2 register	RW	?
5516		RW	?
5616	Timer A0 mode register	RW	0 0 0 0 0 0 0 0
5716	Timer A1 mode register	RW	0 0 0 0 0 0 0 0
5816	Timer A2 mode register	RW	0016
5916	Timer A3 mode register	RW	0016
5A16	Timer A4 mode register	RW	0016
5B16	Timer B0 mode register	RW (Note 3) RW	0 0 ? ? 0 0 0 0
5C16	Timer B1 mode register	RW (Note 3) RW	0 0 ? ? 0 0 0 0
5D16	Timer B2 mode register	RW (Note 3) RW	0 0 ? ? 0 0 0 0
5E16	Processor mode register 0	RW WO RW RW	0 0 0 0 0 0 1 0
5F16	Processor mode register 1	RW	? (Note 4) ?

EOL announced

Notes 1: The access characteristics at addresses 4A16 to 4F16 vary according to Timer A's operating mode. (Refer to **"CHAPTER 8. TIMER A."**)

2: The access characteristics at addresses 5016 to 5316 vary according to Timer B's operating mode. (Refer to **"CHAPTER 9. TIMER B."**)

3: The access characteristics for bit 5 at addresses 5B16 and 5C16 vary according to Timer B's operating mode. Bit 5 at address 5D16 is invalid. (Refer to **"CHAPTER 9. TIMER B."**)

4: Bit 1 at address 5F16 becomes "0" immediately after reset. For the M37721S1BFP, fix this bit to "0."

Fig. 4.1.6 State of SFR and internal RAM areas immediately after reset (3)

4.1 Hardware reset

Address	Register name	Access characteristics (b7 … b0)	State immediately after reset (b7 … b0)
6016	Watchdog timer register	(Note 5)	?(Note 6)
6116	Watchdog timer frequency select register	RW	? 0
6216	Real-time output control register	RW	0 0 0 0 0 0 0 0
6316			?
6416	DRAM control register	RW RW	0 0 0 0 0 0 0 0
6516			?
6616	Refresh timer	WO	?
6716			?
6816	DMAC control register L	(Note 7) RW	0 0 0 0 ? ? 0 0
6916	DMAC control register H	RW WO	0 0 0 0 0 0 0 0
6A16			?
6B16			?
6C16	DMA0 interrupt control register	RW	? 0 0 0 0
6D16	DMA1 interrupt control register	RW	? 0 0 0 0
6E16	DMA2 interrupt control register	RW	? 0 0 0 0
6F16	DMA3 interrupt control register	RW	? 0 0 0 0
7016	A-D conversion interrupt control register	RW	? 0 0 0 0
7116	UART0 transmit interrupt control register	RW	? 0 0 0 0
7216	UART0 receive interrupt control register	RW	? 0 0 0 0
7316	UART1 transmit interrupt control register	RW	? 0 0 0 0
7416	UART1 receive interrupt control register	RW	? 0 0 0 0
7516	Timer A0 interrupt control register	RW	? 0 0 0 0
7616	Timer A1 interrupt control register	RW	? 0 0 0 0
7716	Timer A2 interrupt control register	RW	? 0 0 0 0
7816	Timer A3 interrupt control register	RW	? 0 0 0 0
7916	Timer A4 interrupt control register	RW	? 0 0 0 0
7A16	Timer B0 interrupt control register	RW	? 0 0 0 0
7B16	Timer B1 interrupt control register	RW	? 0 0 0 0
7C16	Timer B2 interrupt control register	RW	? 0 0 0 0
7D16	$\overline{INT_0}$ interrupt control register	RW	? 0 0 0 0 0 0
7E16	$\overline{INT_1}$ interrupt control register	RW	? 0 0 0 0 0 0
7F16	$\overline{INT_2}$ interrupt control register	RW	? 0 0 0 0 0 0

Notes 5: By writing dummy data to address 6016, the value "FFF16" is set to the watchdog timer. The dummy data is not retained anywhere.
6: The value "FFF16" is set to the watchdog timer. (Refer to "**CHAPTER 15. WATCHDOG TIMER**.")
7: It is possible to read the bit state at reading. When writing "0" to this bit, this bit becomes "0." But when writing "1" to this bit, this bit does not change.

EOL announced

Fig. 4.1.7 State of SFR and internal RAM areas immediately after reset (4)

Address	Register name	Access characteristics (b7 … b0)	State immediately after reset (b7 … b0)
1FC016	Source address register 0	RW	?
1FC116		RW	?
1FC216		RW	?
1FC316			?
1FC416	Destination address register 0	RW	?
1FC516		RW	?
1FC616		RW	?
1FC716			?
1FC816	Transfer counter register 0	RW	?
1FC916		RW	?
1FCA16		RW	?
1FCB16			?
1FCC16	DMA0 mode register L	RW	0 0 0 0 0 0 0 0
1FCD16	DMA0 mode register H	RW	0 0 0 0 0 0 0 0
1FCE16	DMA0 control register	RW	? ? 0 0 0 0 0 0
1FCF16			?
1FD016	Source address register 1	RW	?
1FD116		RW	?
1FD216		RW	?
1FD316			?
1FD416	Destination address register 1	RW	?
1FD516		RW	?
1FD616		RW	?
1FD716			?
1FD816	Transfer counter register 1	RW	?
1FD916		RW	?
1FDA16		RW	?
1FDB16			?
1FDC16	DMA1 mode register L	RW	0 0 0 0 0 0 0 0
1FDD16	DMA1 mode register H	RW	0 0 0 0 0 0 0 0
1FDE16	DMA1 control register	RW	? ? 0 0 0 0 0 0
1FDF16			?

EOL announced

Fig. 4.1.8 State of SFR and internal RAM areas immediately after reset (5)

4.1 Hardware reset

Address	Register name	Access characteristics b7 b0	State immediately after reset b7 b0
1FE016		RW	?
1FE116	Source address register 2	RW	?
1FE216		RW	?
1FE316			?
1FE416		RW	?
1FE516	Destination address register 2	RW	?
1FE616		RW	?
1FE716			?
1FE816		RW	?
1FE916	Transfer counter register 2	RW	?
1FEA16		RW	?
1FEB16			?
1FEC16	DMA2 mode register L	RW	0 0 0 0 0 0 0 0
1FED16	DMA2 mode register H	RW	0 0 0 0 0 0 0 0
1FEE16	DMA2 control register	RW	? ? 0 0 0 0 0 0
1FEF16			?
1FF016		RW	?
1FF116	Source address register 3	RW	?
1FF216		RW	?
1FF316			?
1FF416		RW	?
1FF516	Destination address register 3	RW	?
1FF616		RW	?
1FF716			?
1FF816		RW	?
1FF916	Transfer counter register 3	RW	?
1FFA16		RW	?
1FFB16			?
1FFC16	DMA3 mode register L	RW	0 0 0 0 0 0 0 0
1FFD16	DMA3 mode register H	RW	0 0 0 0 0 0 0 0
1FFE16	DMA3 control register	RW	? ? 0 0 0 0 0 0
1FFF16			?

EOL announced

●Internal RAM area (addresses 8016 to 27F16)*

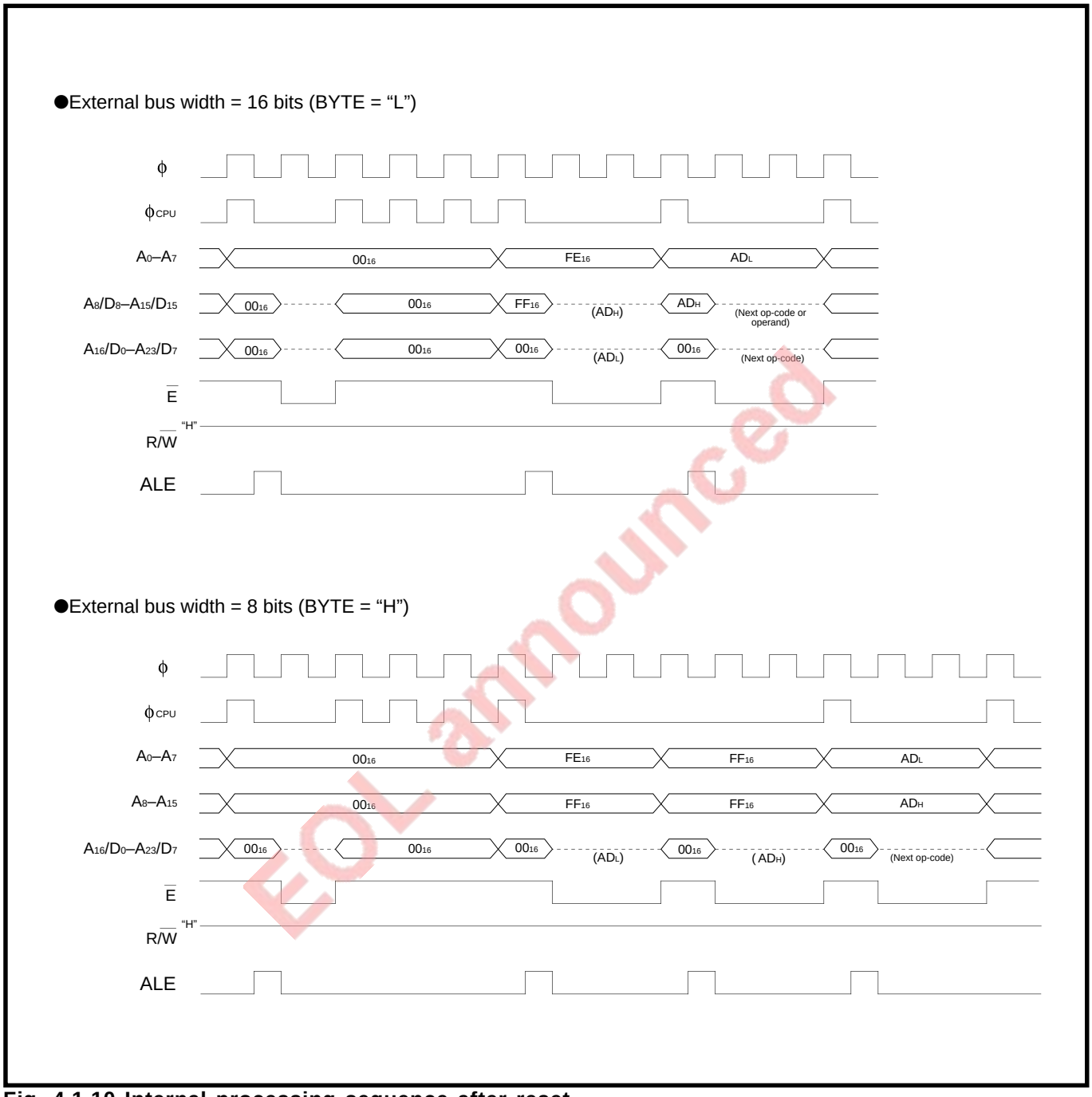
- •At hardware reset
  (Except the case where Stop or Wait mode is terminated)..... Undefined.
- •At software reset..... Retains the state immediately before reset
- •At termination of Stop or Wait mode
  (Hardware reset is used to terminate it.)..... Retains the state immediately before the **STP** or **WIT** instruction is executed

✻ For the M37721S2BFP, the internal RAM area can be assigned to addresses 8016 to 47F16 by setting the internal RAM area select bit (bit 1 at address 5F16). (Refer to section **"2.4 Memory assignment."**)

Fig. 4.1.9 State of SFR and internal RAM areas immediately after reset (6)

4.1.3 Internal processing sequence after reset

Figure 4.1.10 shows the internal processing sequence after reset.



EOL announced

Fig. 4.1.10 Internal processing sequence after reset

4.1 Hardware reset

4.1.4 Time supplying "L" level to $\overline{\text{RESET}}$ pin

Time supplying "L" level to the $\overline{\text{RESET}}$ pin varies according to the state of the clock oscillation circuit.

- When the oscillator is stably oscillating or a stable clock is input from the X_{IN} pin, supply "L" level for 2 μs or more.

- When the oscillator is not stably oscillating (including the case at power-on reset or in Stop mode), supply "L" level until the oscillation is stabilized.
  The time required for stabilizing oscillation varies according to the oscillator. For details, contact the oscillator manufacturer.
  Figure 4.1.11 shows the power-on reset conditions. Figure 4.1.12 shows an example of a power-on reset circuit.

✻ For details about Stop mode, refer to section **"5.3 Stop mode."** For details about clocks, refer to **"CHAPTER 5. CLOCK GENERATING CIRCUIT."**

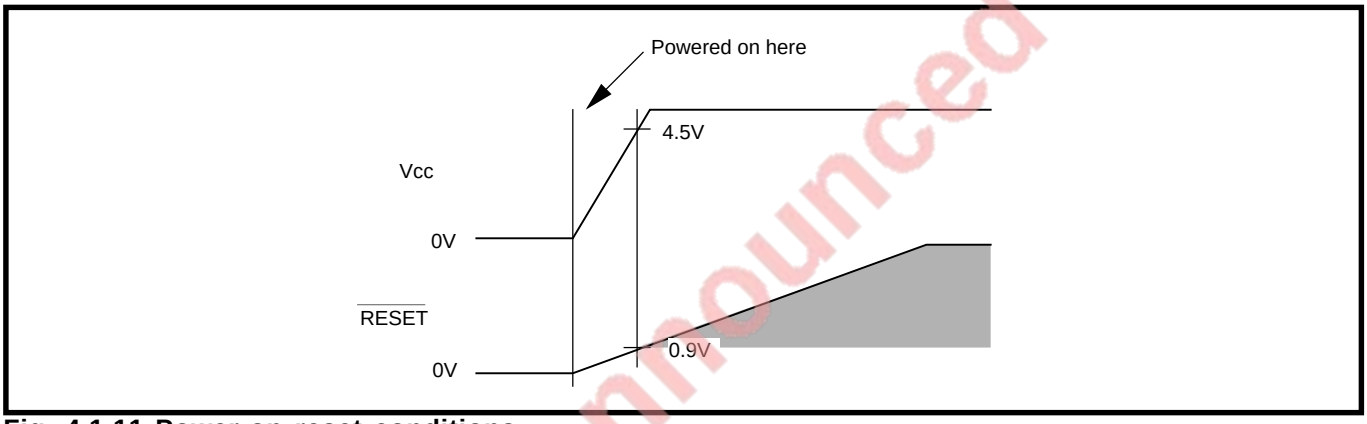


Fig. 4.1.11 Power-on reset conditions

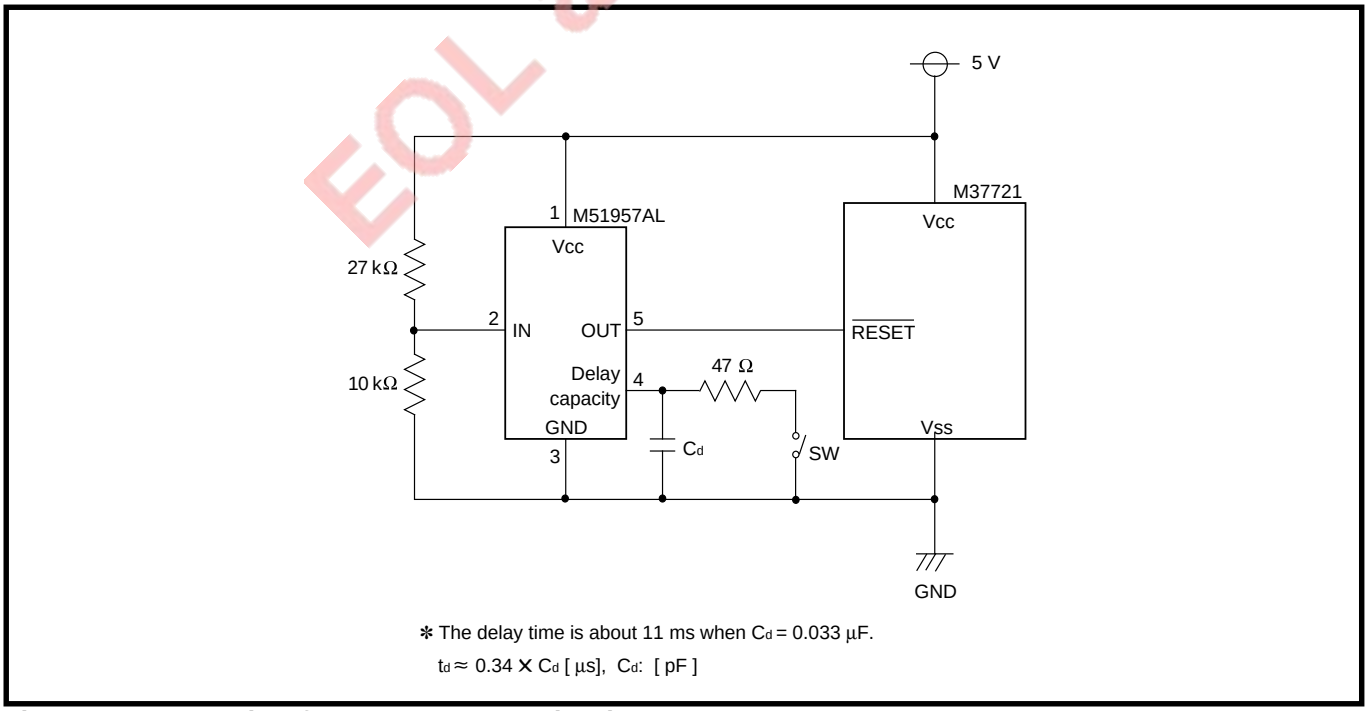


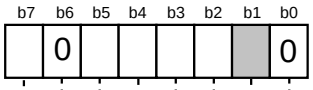
Fig. 4.1.12 Example of power-on reset circuit

EOL announced

4.2 Software reset

When the power source voltage satisfies the microcomputer's recommended operating conditions, the microcomputer is reset by writing "1" to the software reset bit (bit 3 at address $5E_{16}$). (This is called " Software reset.") In this case, the microcomputer initializes pins, CPU, and SFR area just as in the case of a hardware reset. However, the microcomputer retains the contents of the internal RAM area. (Refer to **"Table 4.1.1"** and **"Figures 4.1.3 to 4.1.9."**) Figure 4.2.1 shows the structure of the processor mode register 0 (address $5E_{16}$).

After completing initialization, the microcomputer performs "internal processing sequence after reset." (Refer to **"Figure 4.1.10."**) After that, it executes a program beginning from the address set into the reset vector addresses ($FFFE_{16}$ and $FFFF_{16}$).



Processor mode register 0 (Address $5E_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Fix this bit to "0"		0	RW
1	Nothing is assigned. This bit is "1" at reading.		1	—
2	Wait bit	0 : Software Wait is inserted when accessing external area. 1 : No software Wait is inserted when accessing external area.	0	RW
3	Software reset bit	The microcomputer is reset by writing "1" to this bit. The value of this bit is "0" at reading.	0	WO
4	Interrupt priority detection time select bits	b5 b4 0 0 : 7 cycles of ϕ 0 1 : 4 cycles of ϕ 1 0 : 2 cycles of ϕ 1 1 : Do not select.	0	RW
5			0	RW
6	Fix this bit to " 0"		0	RW
7	Stack bank select bit	0 : Bank 0_{16} 1 : Bank FF_{16}	0	RW

: Bits 0 to 2 and bits 4 to 7 are not used for software reset.

EOL announced

Fig. 4.2.1 Structure of processor mode register 0

4.2 Software reset

When the software reset bit is set to “1,” the $\overline{\text{RESET}}_{\text{OUT}}$ pin’s output level becomes “L.” In a period of 4.5 cycles of clock ϕ_1 after the software reset bit is set to “1,” the $\overline{\text{RESET}}_{\text{OUT}}$ pin’s output level is “L.” Figure 4.2.2 shows the $\overline{\text{RESET}}_{\text{OUT}}$ output timing at software reset.

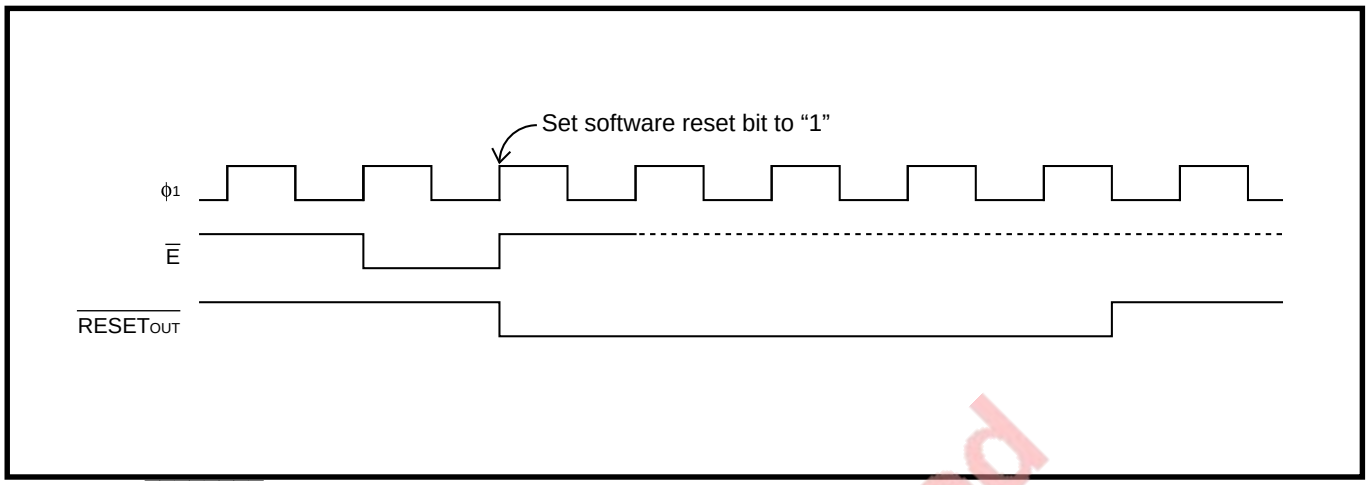


Fig. 4.2.2 $\overline{\text{RESET}}_{\text{OUT}}$ output timing

EOL announced

CHAPTER 5

CLOCK GENERATING CIRCUIT

EOL announced

5.1 Oscillation circuit examples
5.2 Clocks
5.3 Stop mode
[Precautions for Stop mode]
5.4 Wait mode
[Precautions for Wait mode]

5.1 Oscillation circuit examples

To the oscillation circuit, a ceramic resonator or a quartz-crystal oscillator can be connected, or the clock which is externally generated can be input. Oscillation circuit examples are shown below.

5.1.1 Connection example using resonator/oscillator

Figure 5.1.1 shows an example when connecting a ceramic resonator/quartz-crystal oscillator between pins X_{IN} and X_{OUT}.

The circuit constants such as R_f, R_d, C_{IN}, and C_{OUT} (shown in **"Figure 5.1.1"**) depend on the resonator/oscillator. These values shall be set to the values recommended by the resonator/oscillator manufacturer.

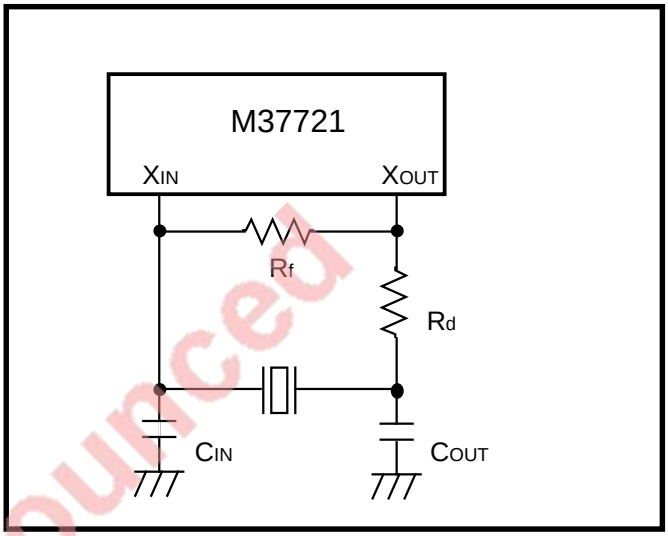


Fig. 5.1.1 Connection example using resonator/oscillator

5.1.2 Externally generated clock input example

Figure 5.1.2 shows an input example of the clock which is externally generated. The external clock must be input from the X_{IN} pin, and the X_{OUT} pin must be left open.

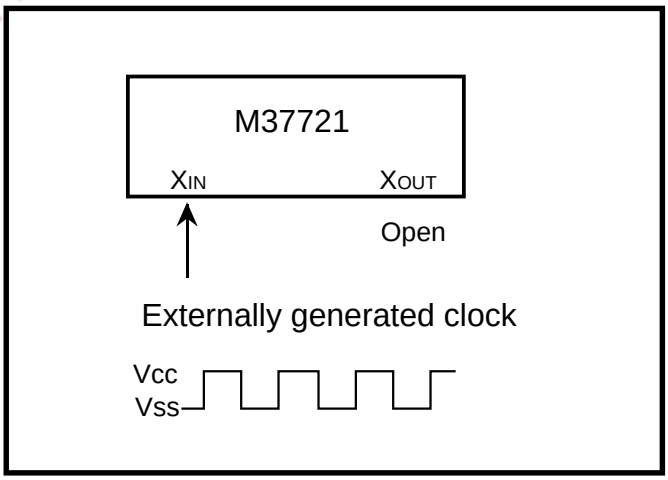
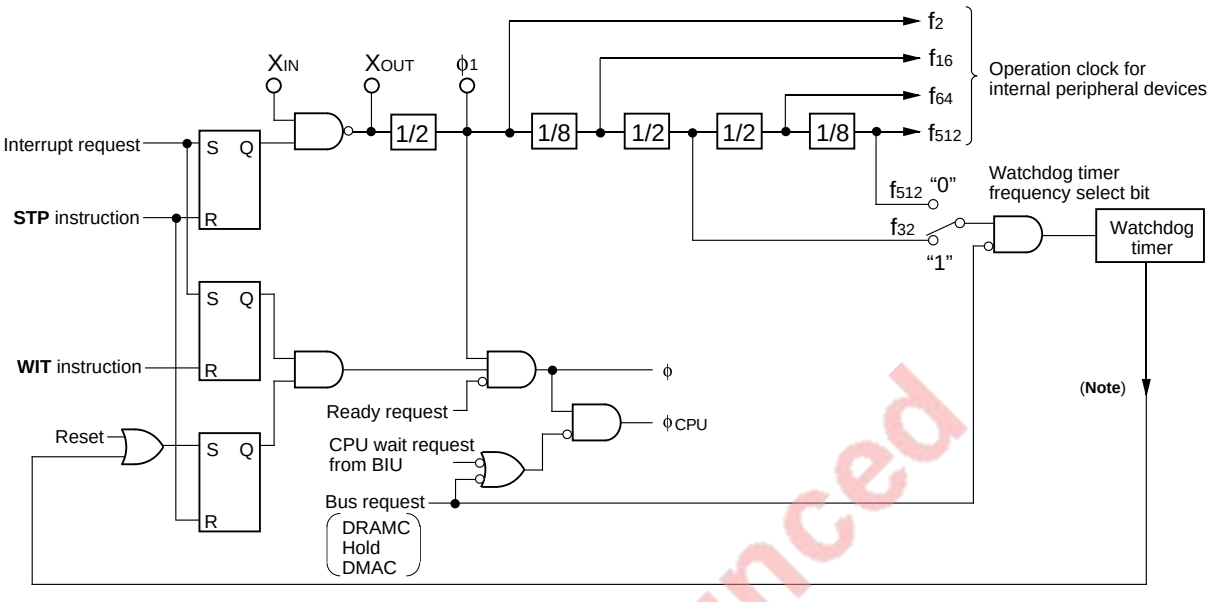


Fig. 5.1.2 Externally generated clock input example

EOL announced

5.2 Clocks

Figure 5.2.1 shows the clock generating circuit block diagram.



CPU : Central Processing Unit
BIU : Bus Interface Unit
Watchdog timer frequency select bit : Bit 0 at address 61_{16}

Note: This signal is generated when the watchdog timer's most significant bit becomes "0."

Fig. 5.2.1 Clock generating circuit block diagram

EOL announced

5.2 Clocks

5.2.1 Clocks generated in clock generating circuit

(1) ϕ

It is the operation clock of BIU. It is also the clock source of ϕ_{CPU}.

ϕ stops by acceptance of Ready request or execution of the **STP** or **WIT** instruction. It is not stopped by acceptance of bus request.

(2) ϕ_{CPU}

It is the operation clock of CPU. ϕ_{CPU} stops by the following:

- Execution of the **STP** or **WIT** instruction,
- Acceptance of Ready request; "L" level input to the $\overline{RDY}$ pin
- CPU wait request from BIU; Acceptance of bus request is included.

(3) **Clock** ϕ_1

It has the same period as ϕ and is output to the external from the ϕ_1 pin. Clock ϕ_1 stops by execution of the **STP** instruction.

It is not stopped by acceptance of Ready or bus request, or execution of the **WIT** instruction.

(4) **f_2 to f_{512}**

Each of them is the internal peripheral devices' operation clock.

Note: Refer to each functional description for details:

- Execution of **STP** instruction ..... "**5.3 Stop mode**"
- Execution of **WIT** instruction ..... "**5.4 Wait mode**"
- Ready ..... "**3.3 Ready function**"
- Bus request ..... "**13.2.1 Bus access control circuit**"

EOL announced

5.3 Stop mode

Stop mode is used to stop oscillation when there is no need to operate the central processing unit (CPU). The microcomputer enters Stop mode when the **STP** instruction is executed.
Stop mode can be terminated by an interrupt request occurrence or the hardware reset.

5.3.1 Stop mode

When the **STP** instruction is executed, the oscillator stops oscillating. This state is called "Stop mode." In Stop mode, the contents of the internal RAM can be retained intact when Vcc (power source voltage) is 2 V or more. Additionally, the microcomputer's power consumption is lowered. It is because the CPU and all internal peripheral devices using clocks f_2 to f_{512} stop the operation.
Table 5.3.1 lists the microcomputer's state and operation in and after Stop mode.

Table 5.3.1 Microcomputer's state and operation in and after Stop mode

Item			State and Operation
State in Stop mode	Oscillation		Stopped
	ϕ_{CPU}, ϕ		
	Clock ϕ_1, f_2 to f_{512}		
	Internal peripheral devices	Timers A, B	Can operate only in event counter mode
		Serial I/O	Can operate only when an external clock is selected
		A-D converter	Stopped
		DMA controller	
		DRAM controller	Stopped **(Note)**
		Watchdog timer	Stopped
		Pins	Retains the same state in which the **STP** instruction was executed
Operation after terminating Stop mode	By interrupt request occurrence		Supply of ϕ_{CPU} and ϕ starts after a certain time measured by Watchdog timer has passed.
	By hardware reset		Operates in the same way as hardware reset

Note: DRAM refresh is not performed because the refresh timer also stops.

EOL announced

5.3 Stop mode

(1) Termination by interrupt request occurrence

When terminating Stop mode by interrupt request occurrence, instructions are executed after a certain time measured by the watchdog timer has passed.

① When an interrupt request occurs, the oscillator starts oscillating. Simultaneously, supply of clock ϕ_1, f_2 to f_{512} starts.
② The watchdog timer starts counting owing to the oscillation start. The watchdog timer counts f_{32} regardless of the watchdog timer frequency select bit's (bit 0 at address 61_{16}) contents.
③ When the watchdog timer's MSB becomes "0," supply of ϕ_{CPU} and ϕ starts. At the same time, the watchdog timer's count source returns to f_{32} or f_{512} that is selected by the watchdog timer frequency select bit.
④ The interrupt request which occurred in ① is accepted.

Table 5.3.2 lists the interrupts used to terminate Stop mode.

Table 5.3.2 Interrupts used to terminate Stop mode

Interrupt	Conditions for using each function to generate interrupt request
$\overline{INT}$i interrupt (i = 0 to 2)	———
Timer Ai interrupt (i = 2 to 4)	In event counter mode
Timer Bi interrupt (i = 0, 1)	
UARTi transmit interrupt (i = 0, 1)	When external clock is selected
UARTi receive interrupt (i = 0, 1)	

Notes 1: Since the oscillator has stopped oscillating, interrupts not listed above cannot be used. Also, even the interrupts listed above cannot be used when the above conditions are not satisfied. The A-D converter does not operate, also.
2: When multiple interrupts listed above are enabled, Stop mode is terminated by the interrupt request which occurs first.
3: Refer to **"CHAPTER 7. INTERRUPTS"** and the description of each internal peripheral device for details about each interrupt.

Before executing the **STP** instruction, interrupts used to terminate Stop mode must be enabled.
In addition, the interrupt priority level of the interrupt used to terminate Stop mode must be higher than the processor interrupt priority level (IPL) of the routine where the **STP** instruction is executed. When multiple interrupts in Table 5.3.2 are enabled, Stop mode is terminated by the first interrupt request.
There is a possibility that any of all interrupt requests occurs after the oscillation starts in ① and until supply of ϕ_{CPU} and ϕ starts in ③. The interrupt requests which occur during this period are accepted in order of priority after the watchdog timer's MSB becomes "0."
For interrupts not to be accepted, set their interrupt priority levels to level 0 (interrupt disabled) before executing the **STP** instruction.

EOL announced

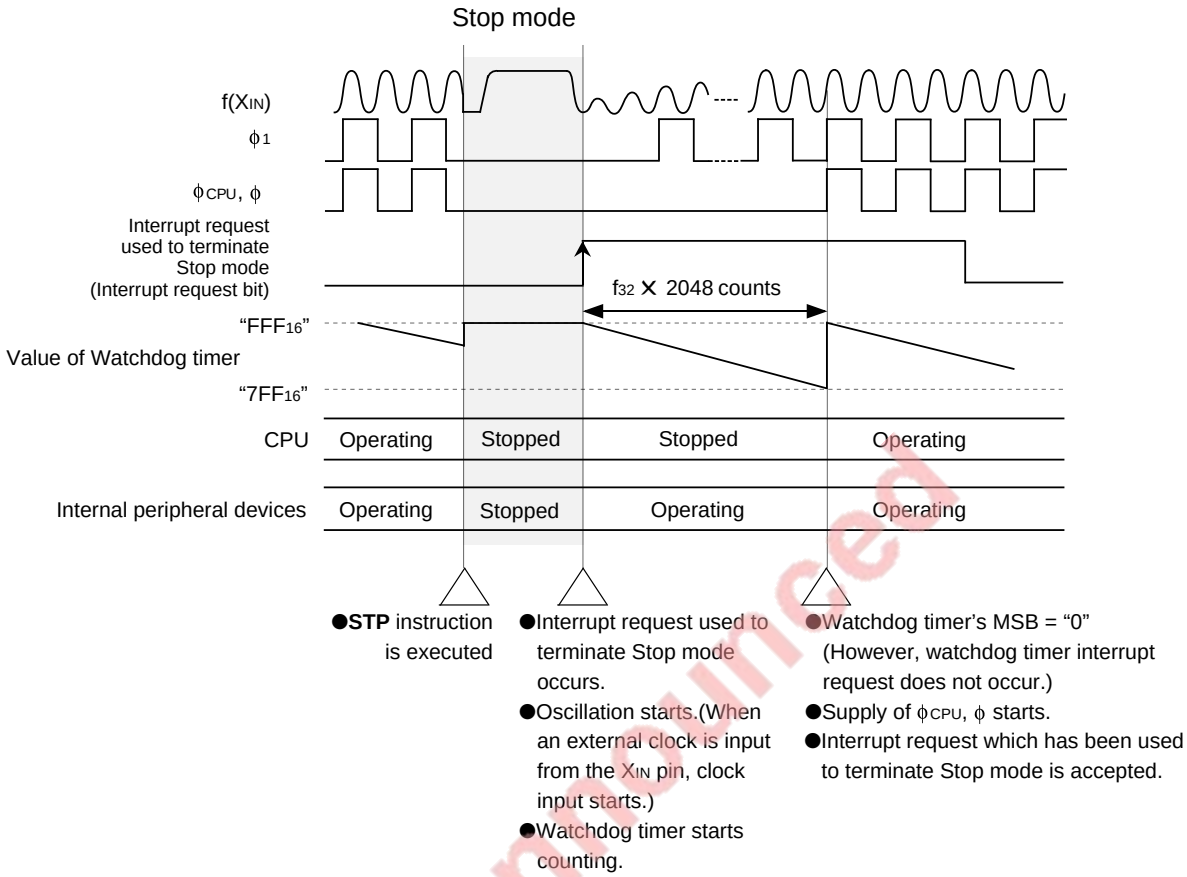


Fig. 5.3.1 Stop mode terminating sequence by interrupt request occurrence

(2) Termination by hardware reset

Supply "L" level to the $\overline{\text{RESET}}$ pin by using the external circuit until the oscillation of the oscillator is stabilized.

The CPU and the SFR area are initialized in the same way as system reset. However, the internal RAM area retains the same contents as that before executing the **STP** instruction. The terminating sequence is the same as the internal processing sequence which is performed after reset.

Refer to **"CHAPTER 4. RESET"** for details about reset.

EOL announced

5.3 Stop mode

[Precautions for Stop mode]

When executing the **STP** instruction after writing to an internal area or an external area, three **NOP** instructions must be inserted to complete the write operation before the **STP** instruction is executed. (Refer to **"Figure 5.3.2."**)

```
STA  A, XXXX ;  Write instruction
NOP          ;  NOP instruction inserted
NOP          ;
NOP          ;
STP          ;  STP instruction
```

Fig. 5.3.2 NOP instruction insertion example

EOL announced

5.4 Wait mode

Wait mode is used to stop ϕ_{CPU} and ϕ when there is no need to operate the central processing unit (CPU). The microcomputer enters Wait mode when the **WIT** instruction is executed.
Wait mode can be terminated by an interrupt request occurrence or the hardware reset.

5.4.1 Wait mode

When the **WIT** instruction is executed, ϕ_{CPU} and ϕ stop. The oscillator's oscillation is not stopped. This state is called "Wait mode."
In Wait mode, the microcomputer's power consumption is lowered though Vcc (power source voltage) is maintained.
Table 5.4.1 lists the microcomputer's state and operation in and after Wait mode.

Table 5.4.1 Microcomputer's state and operation in and after Wait mode

Item			State and Operation
State in Wait mode	Oscillation		Operating
	ϕ_{CPU}, ϕ		Stopped
	Clock ϕ_1, f_2 to f_{512}		Operating
	Internal peripheral devices	Timer A	Operating
		Timer B	
		Serial I/O	
		A-D converter	
		DMA controller	Stopped
		DRAM controller	Stopped **(Note)**
		Watchdog timer	Operating
		Pins	Retains the same state in which the **WIT** instruction was executed
Operation after terminating Wait mode	By interrupt request occurrence		Supply of ϕ_{CPU} and ϕ starts just after the termination.
	By hardware reset		Operates in the same way as hardware reset.

Note: The refresh timer operates, but DRAM refresh is not performed because the bus request (DRAMC) does not occur. (Refer to section **"Appendix 9. 7721 Group Q & A."**)

EOL announced

5.4 Wait mode

(1) Termination by interrupt request occurrence

① When an interrupt request occurs, supply of ϕ_{CPU} and ϕ starts.
② The interrupt request which occurred in ① is accepted.

The following interrupts are used to terminate Wait mode.
When a watchdog timer interrupt request occurs, Wait mode is also terminated.

•$\overline{INT_i}$ interrupt (i = 0 to 2)
•Timer Ai interrupt (i = 0 to 4)
•Timer Bi interrupt (i = 0 to 2)
•UARTi transmit interrupt (i = 0, 1)
•UARTi receive interrupt (i = 0, 1)
•A-D converter interrupt

Note: Refer to "**CHAPTER 7. INTERRUPTS**" and each functional description about interrupts.

Before executing the **WIT** instruction, interrupts used to terminate Wait mode must be enabled. In addition, the interrupt priority level of the interrupt used to terminate Wait mode must be higher than the processor interrupt priority level (IPL) of the routine where the **WIT** instruction is executed. When multiple interrupts listed above are enabled, Wait mode is terminated by the interrupt request which occurs first.

(2) Termination by hardware reset

The CPU and the SFR area are initialized in the same way as system reset. However, the internal RAM area retains the same contents as that before executing the **WIT** instruction. The terminating sequence is the same as the internal processing sequence which is performed after reset.
Refer to **"CHAPTER 4. RESET"** for details about reset.

EOL announced

[Precautions for Wait mode]

When executing the **WIT** instruction after writing to an internal area or an external area, three **NOP** instructions must be inserted to complete the write operation before the **WIT** instruction is executed. (Refer to **"Figure 5.4.1."**)

```
STA  A, XXXX ; Write instruction
NOP          ; NOP instruction inserted
NOP          ;
NOP          ;
WIT          ; WIT instruction
```

Fig. 5.4.1 NOP instruction insertion example

EOL announced

MEMORANDUM

EOL announced

CHAPTER 6

INPUT/OUTPUT PINS

6.1 Overview
6.2 Programmable I/O ports
6.3 Examples of handling unused pins

EOL announced

6.1 Overview

Input/output pins (hereafter called I/O pins) have functions as programmable I/O ports, internal peripheral devices's I/O pins, external buses, etc.

For the basic functions of each I/O pin, refer to section **"1.3 Pin description."** For the I/O functions of the internal peripheral devices, refer to relevant sections of each internal peripheral device. For the external address bus, external data bus, bus control signals, etc., refer to **"CHAPTER 3. CONNECTION WITH EXTERNAL DEVICES."**

This chapter describes the programmable I/O ports and examples of handling unused pins.

6.2 Programmable I/O ports

The programmable I/O ports have direction registers and port registers in the SFR area. Figure 6.2.1 shows the memory map of direction registers and port registers.

Addresses	
A_{16}	Port P4 register
B_{16}	Port P5 register
C_{16}	Port P4 direction register
D_{16}	Port P5 direction register
E_{16}	Port P6 register
F_{16}	Port P7 register
10_{16}	Port P6 direction register
11_{16}	Port P7 direction register
12_{16}	Port P8 register
13_{16}	Port P9 register
14_{16}	Port P8 direction register
15_{16}	Port P9 direction register
16_{16}	Port P10 register
17_{16}	
18_{16}	Port P10 direction register

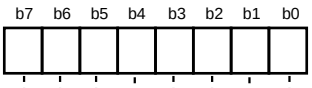
EOL announced

Fig. 6.2.1 Memory map of direction registers and port registers

6.2.1 Direction register

This register determines the I/O direction of programmable I/O ports. Each bit of this register corresponds one for one to each pin of the microcomputer.

Figure 6.2.2 shows the structure of port Pi (i = 4 to 10) direction register.



Port Pi direction register (i = 4 to 10)
(Addresses C_{16}, D_{16}, 10_{16}, 11_{16}, 14_{16}, 15_{16}, 18_{16})

Bit	Bit name	Functions	At reset	RW
0	Port Pi_0 direction bit	0 : Input mode (The port functions as an input port) 1 : Output mode (The port functions as an output port)	0	RW
1	Port Pi_1 direction bit		0	RW
2	Port Pi_2 direction bit		0	RW
3	Port Pi_3 direction bit		0	RW
4	Port Pi_4 direction bit		0	RW
5	Port Pi_5 direction bit		0	RW
6	Port Pi_6 direction bit		0	RW
7	Port Pi_7 direction bit		0	RW

Note: For bits 0 to 2 of the port P4 direction register, nothing is assigned and these bits are fixed to “0” at reading.

Fig. 6.2.2 Structure of port Pi (i = 4 to 10) direction register

EOL announced

6.2.2 Port register

Data is input from or output to the external by writing/reading data to/from a port register. A port register consists of a port latch which holds the output data and a circuit which reads the pin state. Each bit of the port register corresponds one for one to each pin of the microcomputer. Figure 6.2.3 shows the structure of the port Pi (i = 4 to 10) register.

● **When outputting data from programmable I/O port set to output mode**

① By writing data to the corresponding bit of the port register, the data is written into the port latch.
② The data is output from the pin according to the contents of the port latch.

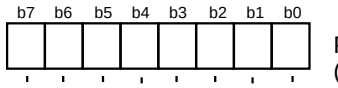
By reading the port register of a port set to the output mode, the contents of the port latch is read out, instead of the pin state. Accordingly, the output data is correctly read without being affected by an external load, etc. (Refer to **"Figures 6.2.4 and 6.2.5."**)

● **When inputting data from programmable I/O port set to input mode**

① A pin which is set to the input mode enters the floating state.
② By reading the corresponding bit of the port register, the data which is input from the pin can be read out.

By writing data to the port register of a programmable I/O port set to the input mode, the data is written only into the port latch and is not output to the external **(Note)**. The pin remains floating.

Note: When executing a read-modify-write instruction (**CLB, SEB, INC, DEC, ASL, ASR, LSR, ROL, ROR**) to the port register of a programmable I/O port set to the input mode, the instruction is executed to the data which is input from the pin and the result is written into the port register.



Port Pi register (i = 4 to 10)
(Addresses A_{16}, B_{16}, E_{16}, F_{16}, 12_{16}, 13_{16}, 16_{16})

Bit	Bit name	Functions	At reset	RW
0	Port P_{i0}'s pin	Data is input from or output to a pin by reading from or writing to the corresponding bit. 0 : "L" level 1 : "H" level	Undefined	RW
1	Port P_{i1}'s pin		Undefined	RW
2	Port P_{i2}'s pin		Undefined	RW
3	Port P_{i3}'s pin		Undefined	RW
4	Port P_{i4}'s pin		Undefined	RW
5	Port P_{i5}'s pin		Undefined	RW
6	Port P_{i6}'s pin		Undefined	RW
7	Port P_{i7}'s pin		Undefined	RW

Note: For bits 0 to 2 of the port P4 register, nothing is assigned and these bits are fixed to "0" at reading.

Fig. 6.2.3 Structure of port Pi (i = 4 to 10) register

EOL announced

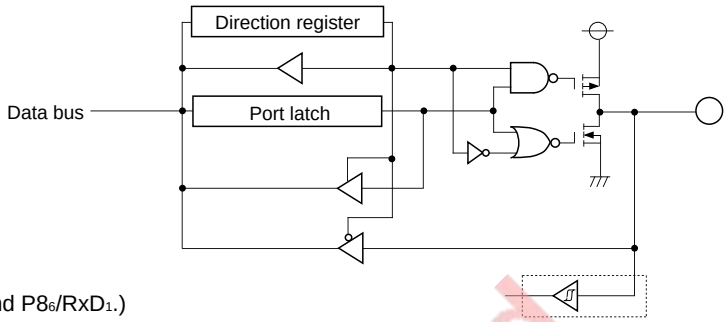
Figures 6.2.4 and 6.2.5 show the port peripheral circuits.

[Inside dotted-line not included]

Ports $P4_3$ to $P4_6$

[Inside dotted-line included]

Ports $P4_7$, $P5_1/TA2_{IN}$, $P5_3/TA3_{IN}$, $P5_5/TA4_{IN}$, $P5_6/TB0_{IN}$, $P5_7/TB1_{IN}$, $P8_2/RxD_0$, $P8_6/RxD_1$, $P9_1/\overline{DMAREQ0}$, $P9_3/\overline{DMAREQ1}$, $P9_5/\overline{DMAREQ2}$, $P9_7/\overline{DMAREQ3}$, $P10_0/\overline{INT_0}$, $P10_1/\overline{INT_1}$, $P10_2/\overline{INT_2}$,
(There is no hysteresis for $P8_2/RxD_0$ and $P8_6/RxD_1$.)

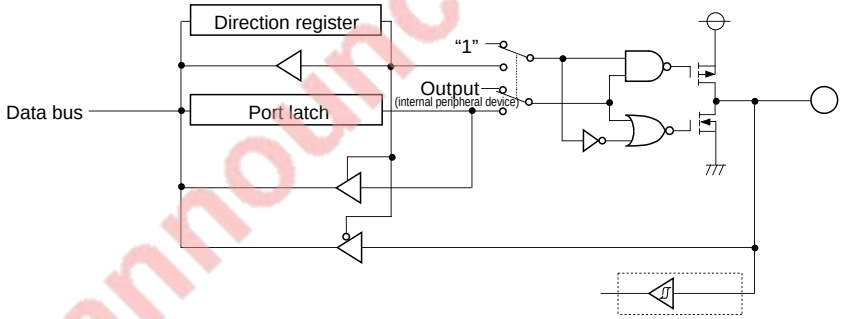


[Inside dotted-line not included]

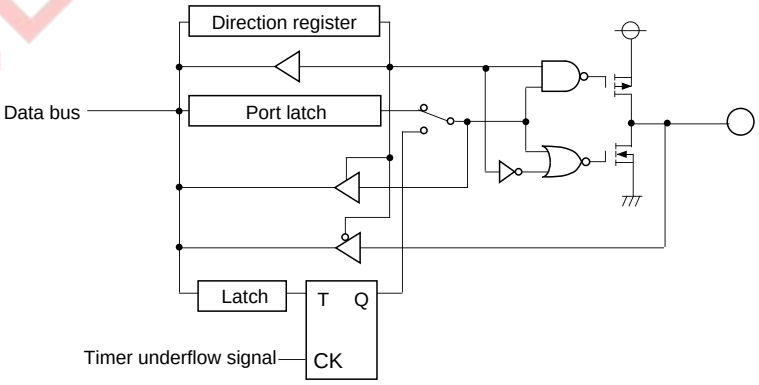
Ports $P8_3/TxD_0$, $P8_7/TxD_1$, $P9_0/\overline{DMAACK0}$, $P9_2/\overline{DMAACK1}$, $P9_4/\overline{DMAACK2}$, $P9_6/\overline{DMAACK3}$, $P10_4/\overline{CAS}$, $P10_5/\overline{RAS}$, $P10_6/MA_8$, $P10_7/MA_9$

[Inside dotted-line included]

Ports $P5_0/TA2_{OUT}$, $P5_2/TA3_{OUT}$, $P5_4/TA4_{OUT}$



Ports $P6_0/RTP0_0$ to $P6_7/RTP1_3$



EOL announced

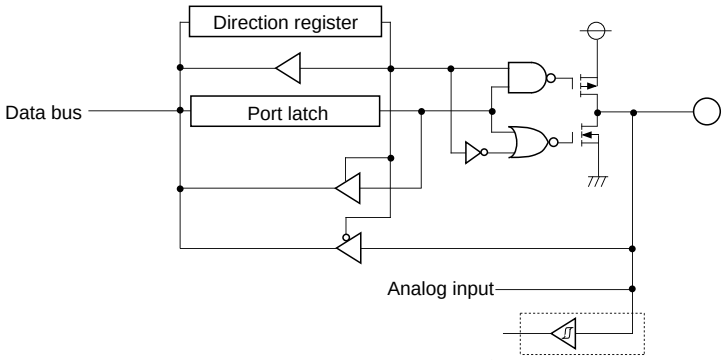
Fig. 6.2.4 Port peripheral circuits (1)

[Inside dotted-line not included]

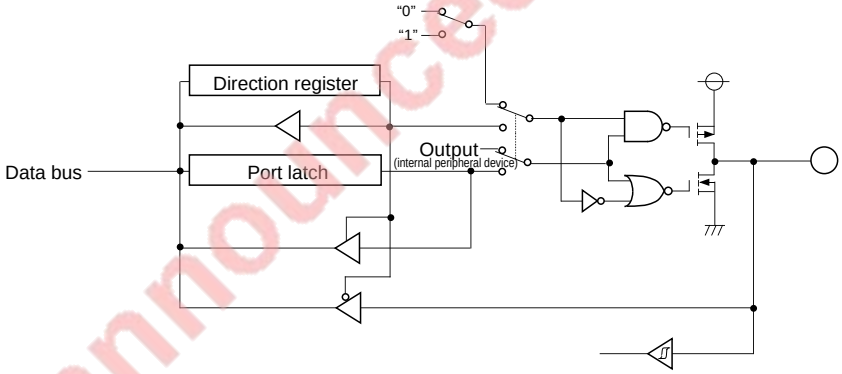
Ports $P7_0/AN_0$ to $P7_6/AN_6$

[Inside dotted-line included]

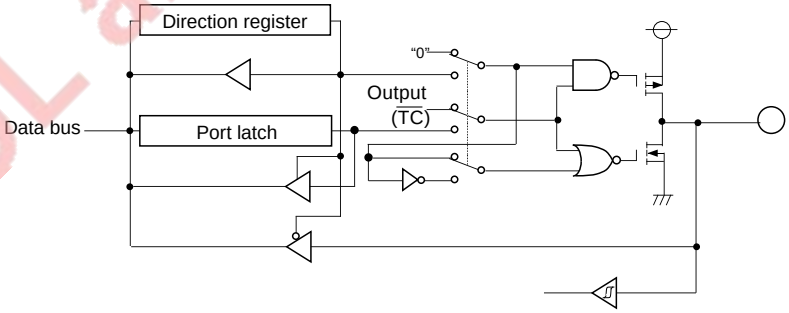
Port $P7_7/AN_7/\overline{AD_{TRG}}$



Ports $P8_0/\overline{CTS_0}/\overline{RTS_0}$, $P8_1/CLK_0$,
$P8_4/\overline{CTS_1}/\overline{RTS_1}$, $P8_5/CLK_1$



Port $P10_3/\overline{TC}$



$\overline{E}$ output pin

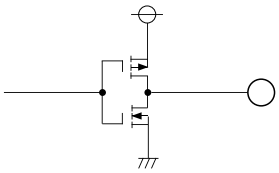


Fig. 6.2.5 Port peripheral circuits (2)

EOL announced

6.3 Examples of handling unused pins

When unusing an I/O pin, some handling is necessary for the pin. Examples of handling unused pins are described below.

The following are just examples. The user shall modify them according to the user's actual application and test them.

Table 6.3.1 Examples of handling unused pins

Pin name	Handling example
$P4_3$ to $P4_7$, P5 to P10	Set these pins to the input mode and connect each pin to Vcc or Vss via a resistor; or set these pins to the output mode and leave them open **(Notes 1 and 2).**
$\overline{BLE}$, $\overline{BHE}$, ALE, ϕ_1, ST0, ST1	Leave them open.
X_{OUT} **(Note 3)**	
$\overline{HOLD}$, $\overline{RDY}$	Connect these pins to Vcc via a resistor (pull-up) **(Note 2).**
CNVss	Connect this pin to Vcc or Vss.
AVcc	Connect this pin to Vcc.
AVss, V_{REF}	Connect these pins to Vss.

Notes 1: When leaving these pins open after they are set to the output mode, note the following: these pins function as input ports from reset until they are switched to the output mode by software. Therefore, voltage levels of these pins are undefined and the power source current may increase while these pins function as input ports. After reset, immediately set these ports to the output mode.
Software reliability can be enhanced by setting the contents of the above ports' direction registers periodically. This is because these contents may be changed by noise, a program runaway which occurs owing to noise, etc.

2: For unused pins, use the shortest possible wiring (within 20 mm from the microcomputer's pins).

3: This applies when a clock externally generated is input to the X_{IN} pin.

EOL announced

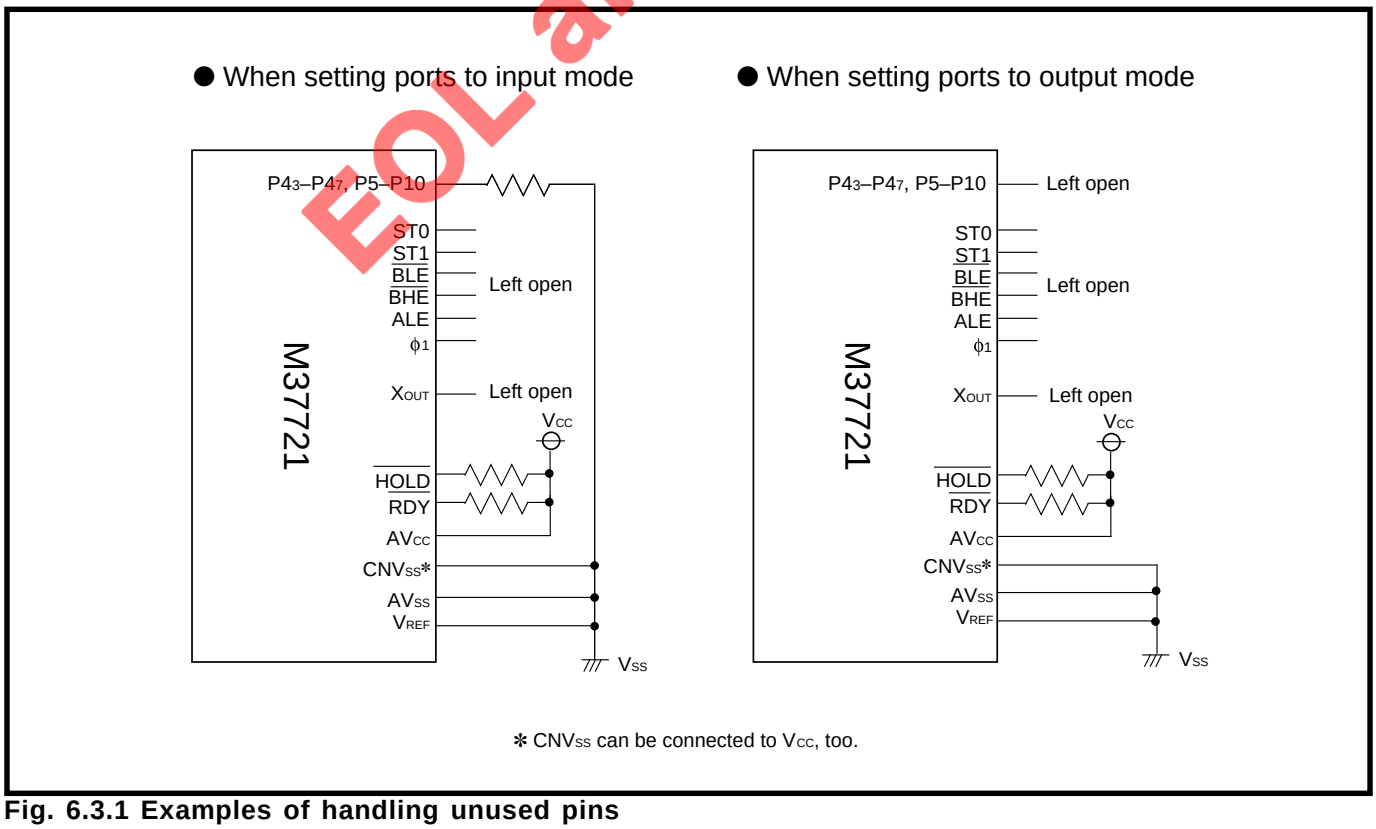


Fig. 6.3.1 Examples of handling unused pins

MEMORANDUM

EOL announced

CHAPTER 7
INTERRUPTS

EOL announced

7.1 Overview
7.2 Interrupt sources
7.3 Interrupt control
7.4 Interrupt priority level
7.5 Interrupt priority level detection circuit
7.6 Interrupt priority level detection time
7.7 Sequence from acceptance of interrupt request until execution of interrupt routine
7.8 Return from interrupt routine
7.9 Multiple interrupts
7.10 External interrupts ($\overline{INT_i}$ interrupt)
7.11 Precautions for interrupts

7.1 Overview

The M37721 provides 23 interrupt sources to generate interrupt requests.
Figure 7.1.1 shows the interrupt processing sequence.
When an interrupt request is accepted, a branch is made to the start address of the interrupt routine set in the interrupt vector table (addresses $FFCE_{16}$ to $FFFF_{16}$). Set the start address of each interrupt routine to the corresponding interrupt vector address in the interrupt vector table.

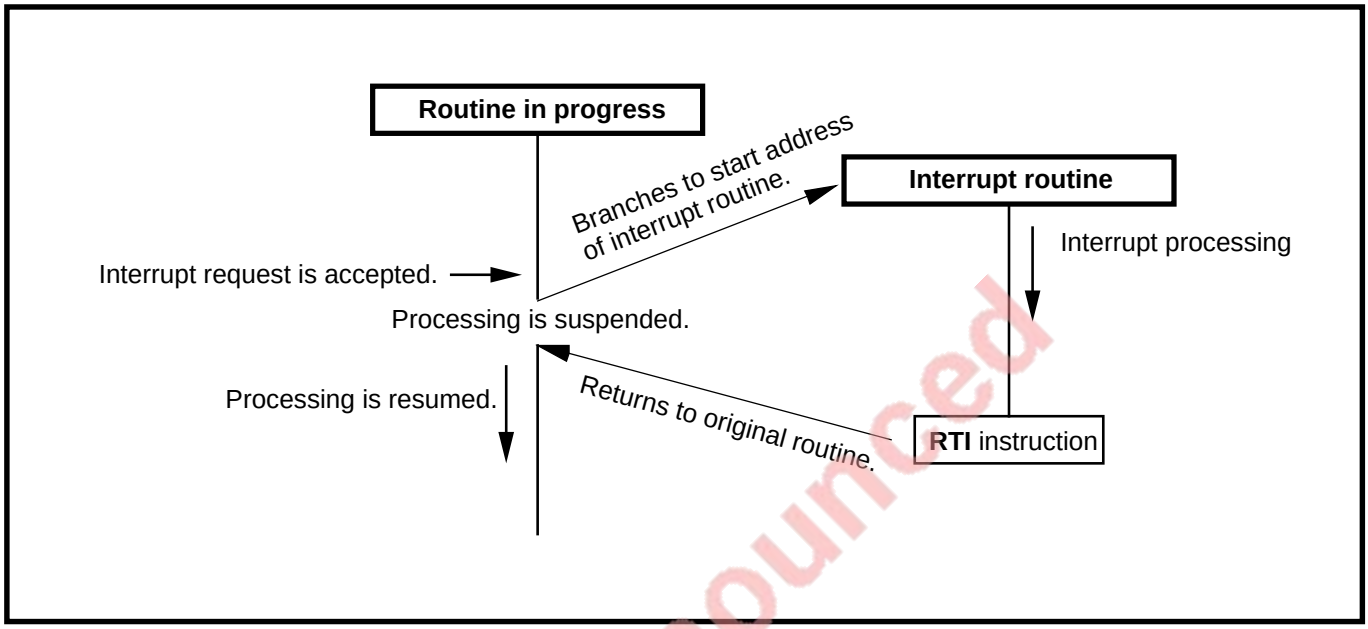


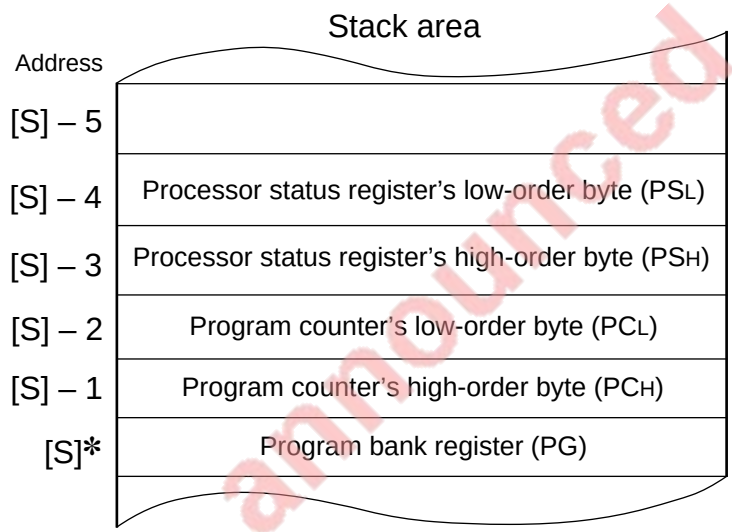
Fig. 7.1.1 Interrupt processing sequence

EOL announced

When an interrupt request is accepted, the following registers' contents just before acceptance of an interrupt request are automatically pushed onto the stack area ①→②→③ in that order.

① Program bank register (PG)
② Program counter (PC_L, PC_H)
③ Processor status register (PS_L, PS_H)

Figure 7.1.2 shows the state of the stack area just before entering the interrupt routine.
Execute the **RTI** instruction at the end of this interrupt routine to return to the routine that the microcomputer was executing before the interrupt request was accepted. By executing the **RTI** instruction, the register contents pushed onto the stack area are pulled ③→②→① in that order. Then, the suspended processing is resumed from where it left off.



* [S] is an initial address that the stack pointer (S) indicates when an interrupt request is accepted. The S's contents become "[S] – 5" after all of the above registers are pushed.

EoL announced

Fig. 7.1.2 State of stack area just before entering interrupt routine

7.2 Interrupt sources

Table 7.2.1 lists the interrupt sources and the interrupt vector addresses. When programming, set the start address of each interrupt routine at the vector addresses listed in this table.

Table 7.2.1 Interrupt sources and interrupt vector addresses

Interrupt source	Interrupt vector addresses		Remarks	Reference
	High-order address	Low-order address		
Reset	$FFFF_{16}$	$FFFE_{16}$	Non-maskable	4. RESET
Zero division	$FFFD_{16}$	$FFFC_{16}$	Non-maskable software interrupt	7700 Family Software Manual
BRK instruction	$FFFB_{16}$	$FFFA_{16}$	Non-maskable software interrupt	
$\overline{DBC}$ **(Note)**	$FFF9_{16}$	$FFF8_{16}$	Do not use.	
Watchdog timer	$FFF7_{16}$	$FFF6_{16}$	Non-maskable interrupt	15. WATCHDOG TIMER
$\overline{INT_0}$	$FFF5_{16}$	$FFF4_{16}$	Maskable external interrupts	7.10 External interrupts ($\overline{INTi}$ interrupt)
$\overline{INT_1}$	$FFF3_{16}$	$FFF2_{16}$		
$\overline{INT_2}$	$FFF1_{16}$	$FFF0_{16}$		
Timer A0	$FFEF_{16}$	$FFEE_{16}$	Maskable internal interrupts	8. TIMER A
Timer A1	$FFED_{16}$	$FFEC_{16}$		
Timer A2	$FFEB_{16}$	$FFEA_{16}$		
Timer A3	$FFE9_{16}$	$FFE8_{16}$		
Timer A4	$FFE7_{16}$	$FFE6_{16}$		
Timer B0	$FFE5_{16}$	$FFE4_{16}$	Maskable internal interrupts	9. TIMER B
Timer B1	$FFE3_{16}$	$FFE2_{16}$		
Timer B2	$FFE1_{16}$	$FFE0_{16}$		
UART0 receive	$FFDF_{16}$	$FFDE_{16}$	Maskable internal interrupts	11. SERIAL I/O
UART0 transmit	$FFDD_{16}$	$FFDC_{16}$		
UART1 receive	$FFDB_{16}$	$FFDA_{16}$		
UART1 transmit	$FFD9_{16}$	$FFD8_{16}$		
A-D conversion	$FFD7_{16}$	$FFD6_{16}$	Maskable internal interrupt	12. A-D CONVERTER
DMA0	$FFD5_{16}$	$FFD4_{16}$	Maskable internal interrupts	13. DMA CONTROLLER
DMA1	$FFD3_{16}$	$FFD2_{16}$		
DMA2	$FFD1_{16}$	$FFD0_{16}$		
DMA3	$FFCF_{16}$	$FFCE_{16}$		

Note: The $\overline{DBC}$ interrupt is used exclusively for debugger control.

EOL announced

●**Maskable interrupt:** An interrupt of which request's acceptance can be disabled by software.

●**Non-maskable interrupt** (including Zero division, **BRK** instruction, Watchdog timer interrupts)**:** An interrupt which is certain to be accepted when its request occurs. These interrupts do not have their interrupt control registers and are not affected by the interrupt disable flag (I).

7.3 Interrupt control

The maskable interrupts are controlled by the following :

- Interrupt request bit
- Interrupt priority level select bits

} Assigned to the interrupt control register of each interrupt.

- Processor interrupt priority level (IPL)
- Interrupt disable flag (I)

} Assigned to the processor status register (PS).

Figure 7.3.1 shows the memory assignment of the interrupt control registers, and Figure 7.3.2 shows their structures.

Address	
$6C_{16}$	DMA0 interrupt control register
$6D_{16}$	DMA1 interrupt control register
$6E_{16}$	DMA2 interrupt control register
$6F_{16}$	DMA3 interrupt control register
70_{16}	A-D conversion interrupt control register
71_{16}	UART0 transmit interrupt control register
72_{16}	UART0 receive interrupt control register
73_{16}	UART1 transmit interrupt control register
74_{16}	UART1 receive interrupt control register
75_{16}	Timer A0 interrupt control register
76_{16}	Timer A1 interrupt control register
77_{16}	Timer A2 interrupt control register
78_{16}	Timer A3 interrupt control register
79_{16}	Timer A4 interrupt control register
$7A_{16}$	Timer B0 interrupt control register
$7B_{16}$	Timer B1 interrupt control register
$7C_{16}$	Timer B2 interrupt control register
$7D_{16}$	$\overline{INT_0}$ interrupt control register
$7E_{16}$	$\overline{INT_1}$ interrupt control register
$7F_{16}$	$\overline{INT_2}$ interrupt control register

EOL announced

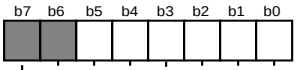
Fig. 7.3.1 Memory assignment of interrupt control registers

7.3 Interrupt control



DMA0 to DMA3, A-D conversion, UART0 and 1 transmit, UART0 and 1 receive, timers A0 to A4, timers B0 to B2 interrupt control registers (Addresses $6C_{16}$ to $7C_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Interrupt priority level select bits	b2 b1 b0 0 0 0 : Level 0 (Interrupt disabled) 0 0 1 : Level 1 0 1 0 : Level 2 0 1 1 : Level 3 1 0 0 : Level 4 1 0 1 : Level 5 1 1 0 : Level 6 1 1 1 : Level 7	0	RW
1			0	RW
2			0	RW
3	Interrupt request bit	0 : No interrupt requested 1 : Interrupt requested	0	RW
7 to 4	Nothing is assigned.		Undefined	—



$\overline{INT_0}$ to $\overline{INT_2}$ interrupt control registers (Addresses $7D_{16}$ to $7F_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Interrupt priority level select bits	b2 b1 b0 0 0 0 : Level 0 (Interrupt disabled) 0 0 1 : Level 1 0 1 0 : Level 2 0 1 1 : Level 3 1 0 0 : Level 4 1 0 1 : Level 5 1 1 0 : Level 6 1 1 1 : Level 7	0	RW
1			0	RW
2			0	RW
3	Interrupt request bit **(Note)**	0 : No interrupt requested 1 : Interrupt requested	0	RW
4	Polarity select bit	0 : Interrupt request bit is set to “1” at “H” level when level sense is selected; this bit is set to “1” at falling edge when edge sense is selected. 1 : Interrupt request bit is set to “1” at “L” level when level sense is selected; this bit is set to “1” at rising edge when edge sense is selected.	0	RW
5	Level sense/Edge sense select bit	0 : Edge sense 1 : Level sense	0	RW
7, 6	Nothing is assigned.		Undefined	—

Note: The interrupt request bits of $\overline{INT_0}$ to $\overline{INT_2}$ interrupts are invalid when the level sense is selected.

EOL announced

Fig. 7.3.2 Structures of interrupt control register

7.3.1 Interrupt disable flag (I)

All maskable interrupts can be disabled by this flag. When this flag is set to “1,” all maskable interrupts are disabled; when this flag is cleared to “0,” those interrupts are enabled. Because this flag is set to “1” at reset, clear this flag to “0” when enabling interrupts.

7.3.2 Interrupt request bit

When an interrupt request occurs, this bit is set to “1.” This bit remains set to “1” until the interrupt request is accepted; it is cleared to “0” when the interrupt request is accepted.
This bit can also be set to “0” or “1” by software.
The $\overline{INT_i}$ interrupt request bit (i = 0 to 2) is ignored when the $\overline{INT_i}$ interrupt is used with level sense.

7.3.3 Interrupt priority level select bits and processor interrupt priority level (IPL)

The interrupt priority level select bits are used to determine the priority level of each interrupt.
When an interrupt request occurs, its interrupt priority level is compared with the processor interrupt priority level (IPL). The requested interrupt is enabled only when the comparison result meets the following condition. Accordingly, an interrupt can be disabled by setting its interrupt priority level to 0.

Each interrupt priority level > Processor interrupt priority level (IPL)

Table 7.3.1 lists the setting of interrupt priority level, and Table 7.3.2 lists the interrupt enabled level corresponding to IPL contents.

The interrupt disable flag (I), interrupt request bit, interrupt priority level select bits, and processor interrupt priority level (IPL) are independent of one another; they do not affect one another. Interrupt requests are accepted only when the following conditions are satisfied.

- Interrupt disable flag (I) = “0”
- Interrupt request bit = “1”
- Interrupt priority level > Processor interrupt priority level (IPL)

EOL announced

7.3 Interrupt control

Table 7.3.1 Setting of interrupt priority level

Interrupt priority level select bits			Interrupt priority level	Priority
b2	b1	b0		
0	0	0	Level 0 (Interrupt disabled)	—
0	0	1	Level 1	Low
0	1	0	Level 2	
0	1	1	Level 3	
1	0	0	Level 4	
1	0	1	Level 5	
1	1	0	Level 6	
1	1	1	Level 7	High

Table 7.3.2 Interrupt enabled level corresponding to IPL contents

IPL_2	IPL_1	IPL_0	Enabled interrupt priority level
0	0	0	Enable level 1 and above interrupts.
0	0	1	Enable level 2 and above interrupts.
0	1	0	Enable level 3 and above interrupts.
0	1	1	Enable level 4 and above interrupts.
1	0	0	Enable level 5 and above interrupts.
1	0	1	Enable level 6 and level 7 interrupts.
1	1	0	Enable only level 7 interrupt.
1	1	1	Disable all maskable interrupts.

IPL_0: Bit 8 in processor status register (PS)
IPL_1: Bit 9 in processor status register (PS)
IPL_2: Bit 10 in processor status register (PS)

EOL announced

7.4 Interrupt priority level

When the interrupt disable flag (I) = “0” (interrupts enabled) and more than one interrupt request is detected at the same sampling timing, which means a timing to check whether an interrupt request exists or not, they are accepted in order of priority levels. In other words, the interrupt request with the highest priority level is accepted first.

Among a total of 23 interrupt sources, the user can set the desired priority levels for 20 interrupt sources except software interrupts (zero division and **BRK** instruction interrupts) and the watchdog timer interrupt. Use the interrupt priority level select bits to set their priority levels. Priority levels of reset, which is handled as the interrupt request with the highest priority, and the watchdog timer interrupt are set by hardware. Figure 7.4.1 shows the interrupt priority set by hardware.

Note that software interrupts are not affected by the interrupt priority levels. Whenever the instruction is executed, a program certainly branches to the interrupt routine.

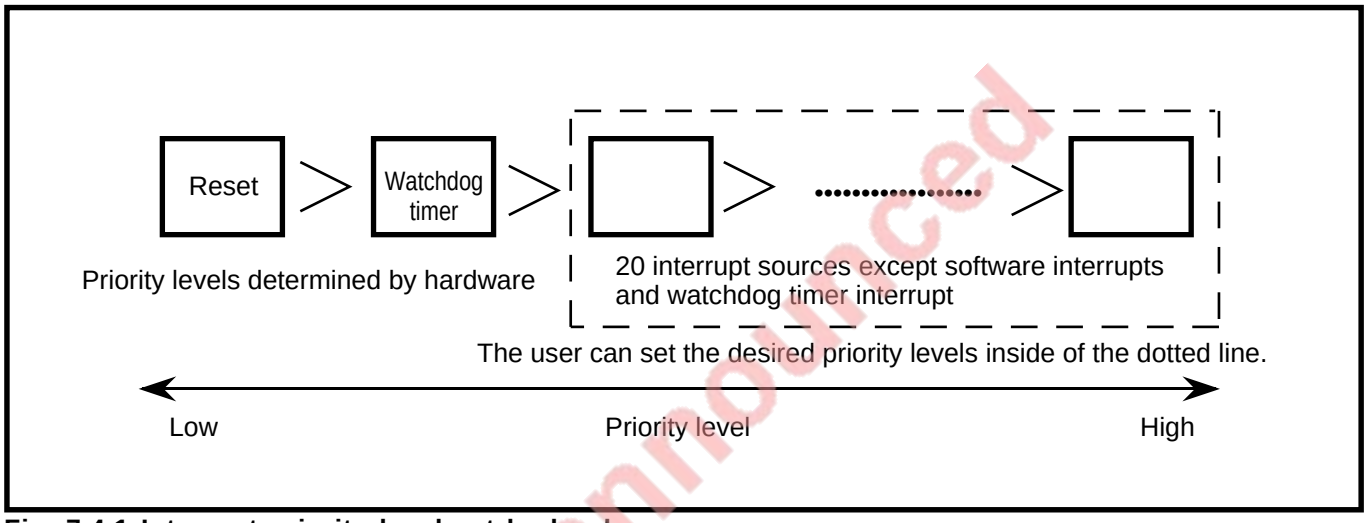


Fig. 7.4.1 Interrupt priority level set by hardware

EOL announced

7.5 Interrupt priority level detection circuit

The interrupt priority level detection circuit selects the interrupt with the highest priority level when more than one interrupt request occurs at the same sampling timing. Figure 7.5.1 shows the interrupt priority level detection circuit.

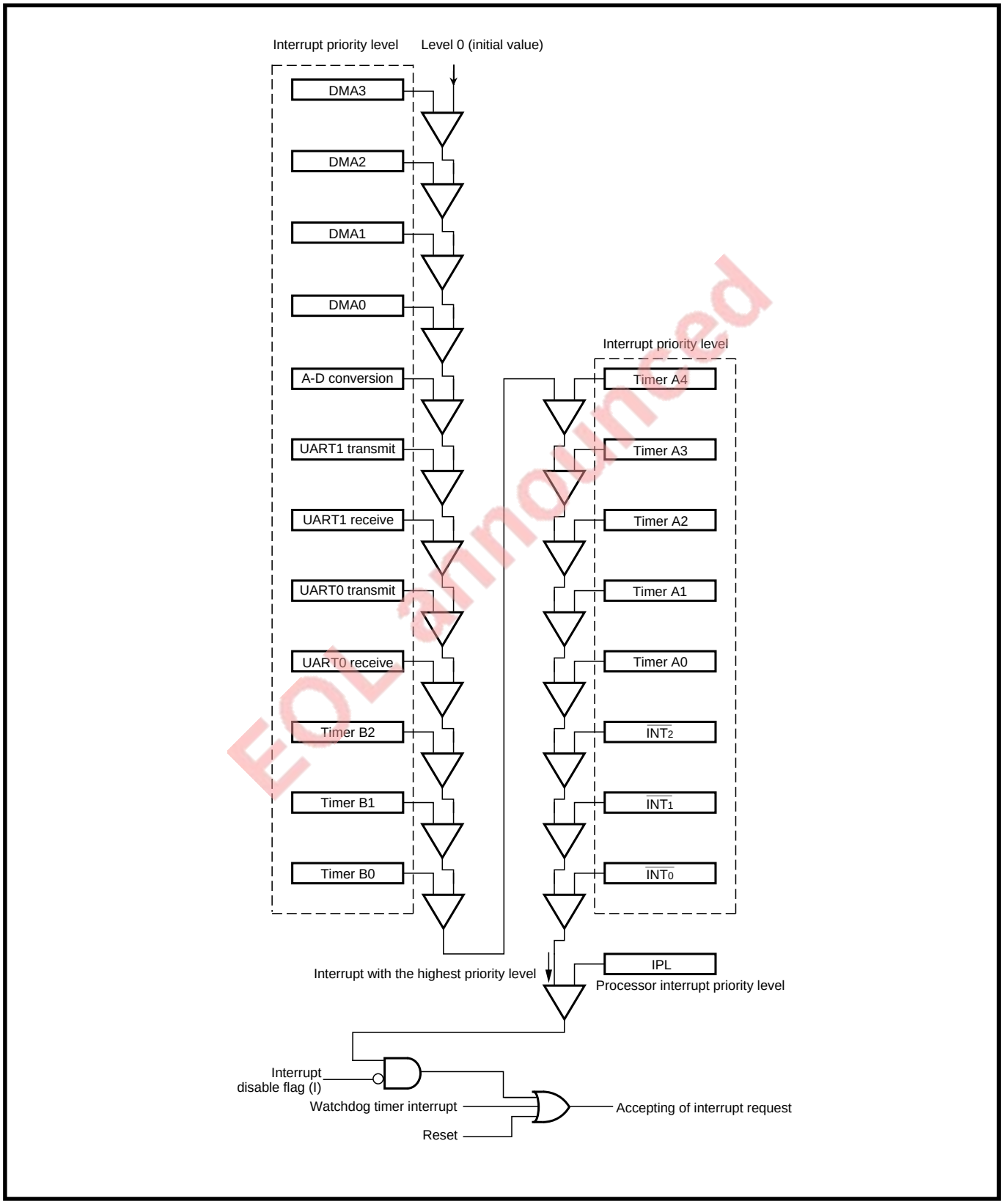


Fig. 7.5.1 Interrupt priority level detection circuit

EOL announced

The following explains the operation of the interrupt priority detection circuit using Figure 7.5.2.
The interrupt priority level of a requested interrupt (Y in Figure 7.5.2) is compared with the resultant priority level which is sent from the preceding comparator (X in Figure 7.5.2); the interrupt with the higher priority level is sent to the next comparator (Z in Figure 7.5.2). (Initial comparison value of "X" is "0.") For an interrupt which is not requested, the comparison is not performed and the priority level which is sent from the preceding comparator is forwarded to the next comparator as it is. When the two priority levels are found the same by comparison, the priority level which is sent from the preceding comparator is forwarded to the next comparator. Accordingly, when the same priority level is set by software, the interrupt priority levels are handled as follows:

DMA3 > DMA2 > DMA1 > DMA0 > A-D conversion > UART1 transmit > UART1 receive > UART0 transmit > UART0 receive > Timer B2 > Timer B1 > Timer B0 > Timer A4 > Timer A3 > Timer A2 > Timer A1 > Timer A0 > $\overline{INT_2}$ > $\overline{INT_1}$ > $\overline{INT_0}$

Among the multiple interrupt requests sampled at the same time, one request with the highest priority level is detected by the above comparison.

Then, this highest interrupt priority level is compared with the processor interrupt priority level (IPL). When this interrupt priority level is higher than the processor interrupt priority level (IPL) and the interrupt disable flag (I) is "0," the interrupt request is accepted. A interrupt request which is not accepted here is held until it is accepted or its interrupt request bit is cleared to "0" by software.
The interrupt priority is detected when the CPU fetches an op code, which is called the CPU's op-code fetch cycle. However, when an op-code fetch cycle starts during detection of an interrupt priority, a new interrupt priority detection does not start. (Refer to **"Figure 7.6.1."**) Since the state of the interrupt request bit and interrupt priority levels are latched during the interrupt priority detection, even if they change, the interrupt priority detection is performed for the previous state before the change occurred.

The interrupt priority level is detected when the CPU fetches an op code. Therefore, in the following execution or states, after the execution or state is terminated, no interrupt request is accepted until the CPU fetches the op code of the next instruction.
•Execution of an instruction which requires many cycles, such as the **MVN** or **MVP** instruction
•During DRAM refreshment
•During Hold state
•During DMA transfer

EOL announced

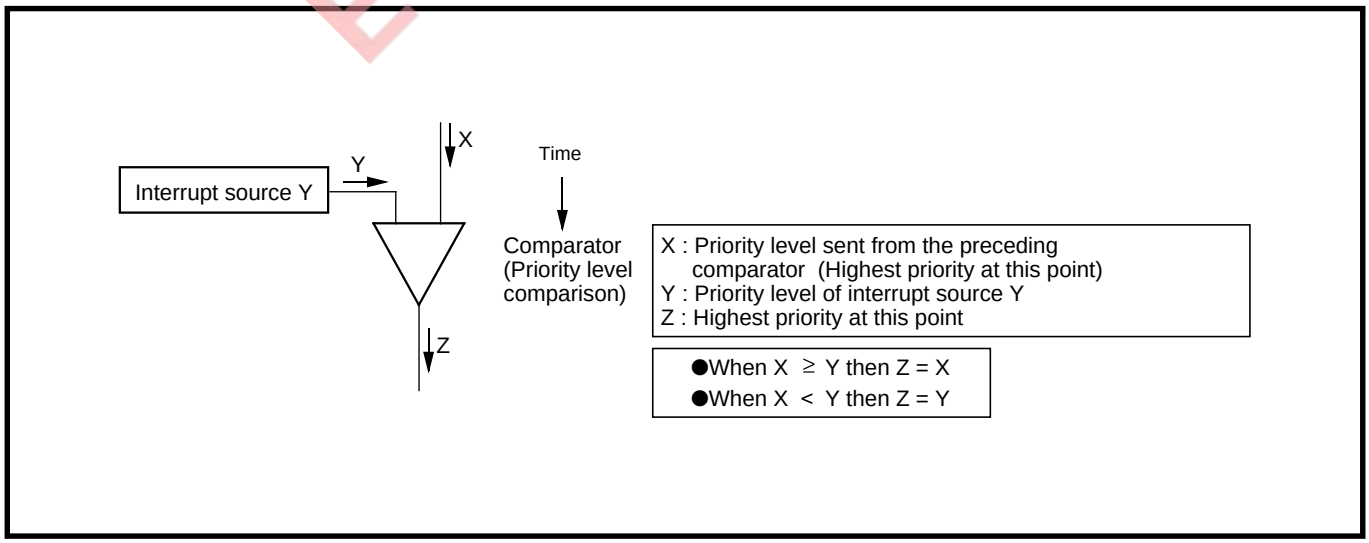
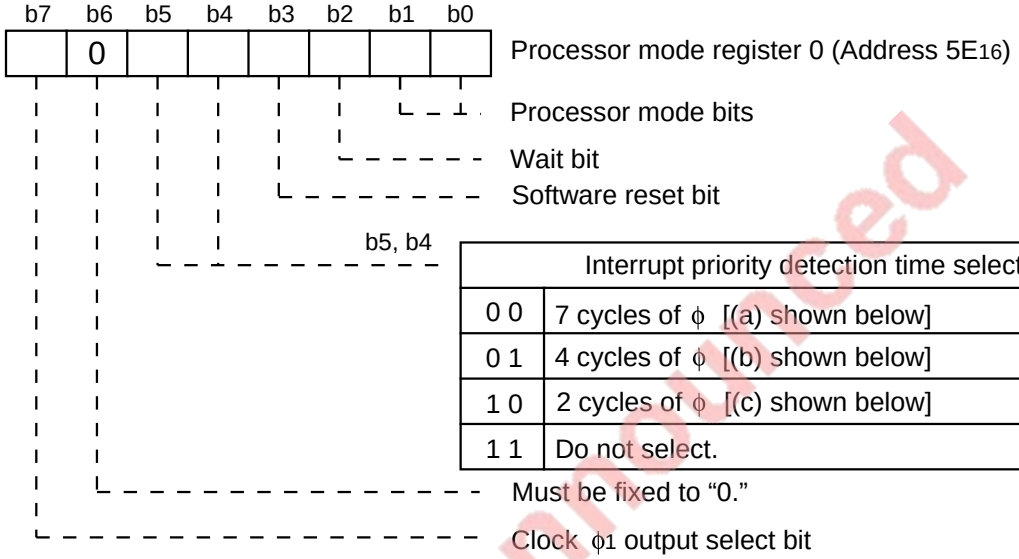


Fig. 7.5.2 Interrupt priority level detection model

7.6 Interrupt priority level detection time

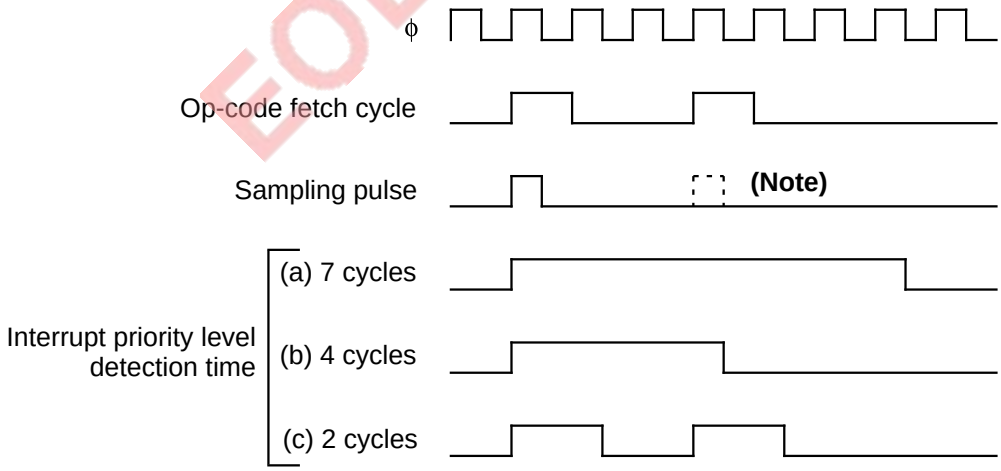
When the interrupt priority level detection time has passed after sampling starts, an interrupt request is accepted. The interrupt priority level detection time can be selected by software. Figure 7.6.1 shows the interrupt priority level detection time. Usually, select "2 cycles of ϕ" as the interrupt priority level detection time.

(1) Interrupt priority detection time select bits



Interrupt priority detection time select bits	
0 0	7 cycles of ϕ [(a) shown below]
0 1	4 cycles of ϕ [(b) shown below]
1 0	2 cycles of ϕ [(c) shown below]
1 1	Do not select.

(2) Interrupt priority level detection time



Note: The pulse resides when "2 cycles of ϕ" is selected.

Fig. 7.6.1 Interrupt priority level detection time

EOL announced

7.7 Sequence from acceptance of interrupt request until execution of interrupt routine

The sequence from the acceptance of interrupt request until the execution of the interrupt routine is described below.

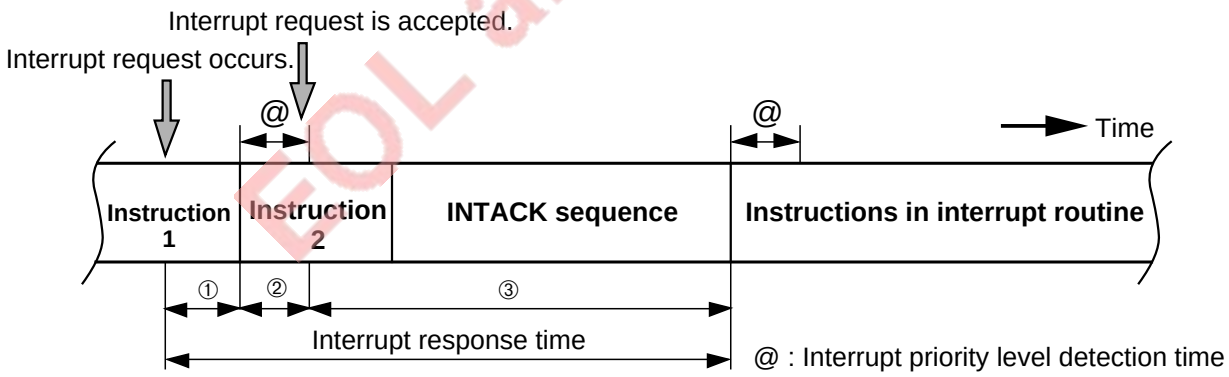
When an interrupt request is accepted, the interrupt request bit of the accepted interrupt is cleared to "0." And then, the interrupt processing starts from the cycle just after the completion of the instruction which was executed at accepting the interrupt request. Figure 7.7.1 shows the sequence from acceptance of interrupt request to execution of interrupt routine. After execution of an instruction at accepting the interrupt request is completed, an INTACK (Interrupt Acknowledge) sequence is executed, and a branch is made to the start address of the interrupt routine allocated in addresses 0_{16} to $FFFF_{16}$.

The INTACK sequence is automatically performed in the following order.

① The contents of the program bank register (PG) just before performing the INTACK sequence are pushed onto stack.
② The contents of the program counter (PC) just before performing the INTACK sequence are pushed onto stack.
③ The contents of the processor status register (PS) just before performing the INTACK sequence is pushed onto stack.
④ The interrupt disable flag (I) is set to "1."
⑤ The interrupt priority level of the accepted interrupt is set into the processor interrupt priority level (IPL).
⑥ The contents of the program bank register (PG) are cleared to "00_{16}," and the contents of the interrupt vector address are set into the program counter (PC).

Performing the INTACK sequence requires at least 13 cycles of internal clock ϕ. Figure 7.7.2 shows the INTACK sequence timing. After the INTACK sequence is completed, the instruction execution starts from the start address of the interrupt routine.

EOL announced



① Time from the occurrence of an interrupt request until the instruction execution which is in progress at that time is completed.
② Time from when execution of an instruction next to ① begins **(Note)** until the instruction execution which is in progress at completion of interrupt priority level detection.
Note: At this time, detection of interrupt priority level begins.
③ Time required to execute the INTACK sequence (13 cycles of ϕ at minimum)

Fig. 7.7.1 Sequence from acceptance of interrupt request until execution of interrupt routine

7.7 Sequence from acceptance of interrupt request until execution of interrupt routine

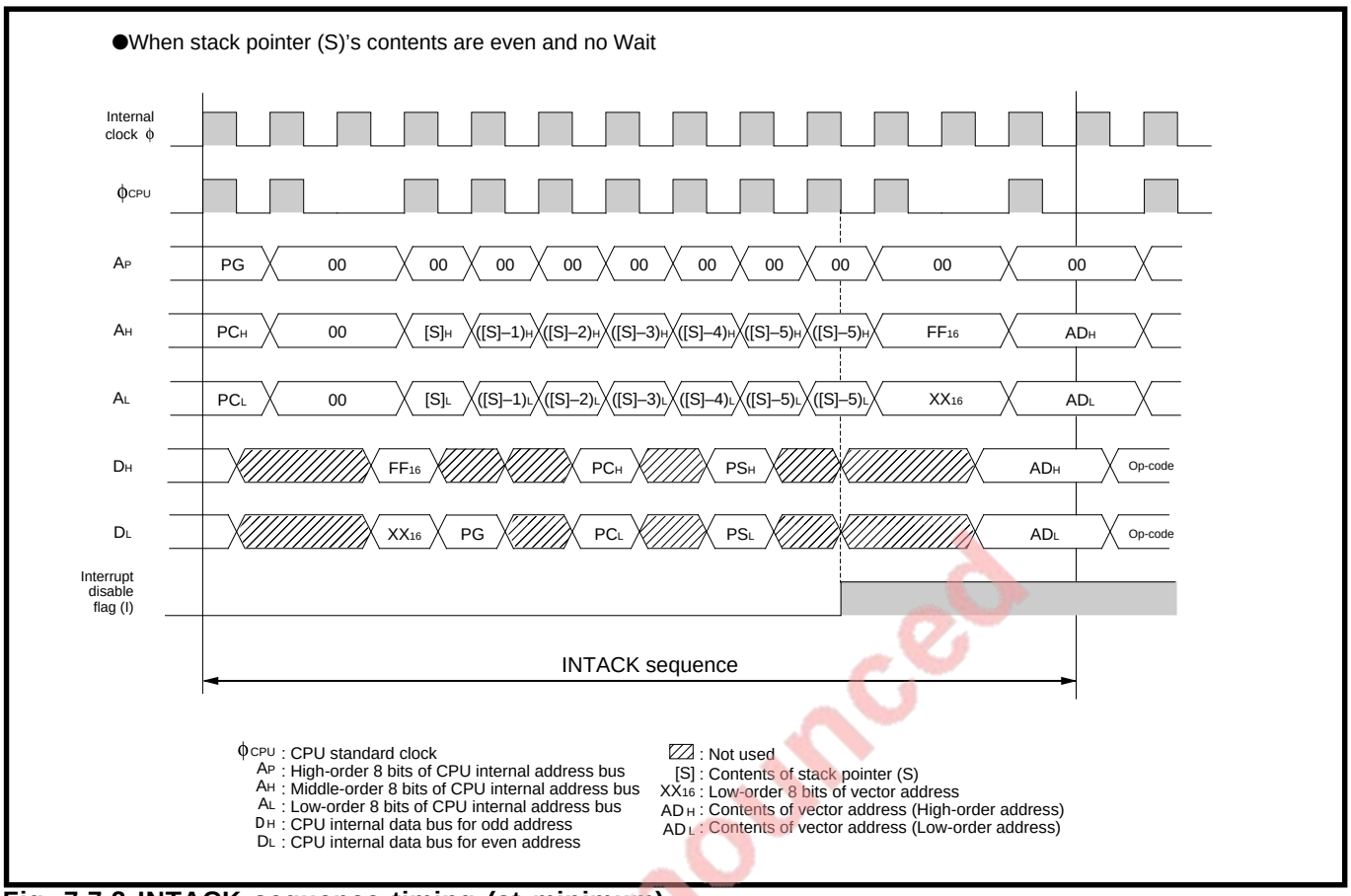


Fig. 7.7.2 INTACK sequence timing (at minimum)

7.7.1 Change in IPL at acceptance of interrupt request

When an interrupt request is accepted, the processor interrupt priority level (IPL) is replaced with the interrupt priority level of the accepted interrupt. This results in easy control of the processing for multiple interrupts. (Refer to section **"7.9 Multiple interrupts."**)

At reset or when a watchdog timer interrupt or a software interrupt is accepted, a value listed in Table 7.7.1 is set into the IPL.

Table 7.7.1 Change in IPL at acceptance of interrupt request

Interrupts	Change in IPL
Reset	Level 0 ("000_2") is set.
Watchdog timer	Level 7 ("111_2") is set.
Zero division	Not changed.
BRK instruction	Not changed.
Other interrupts	Accepted interrupt priority level is set.

EOL announced

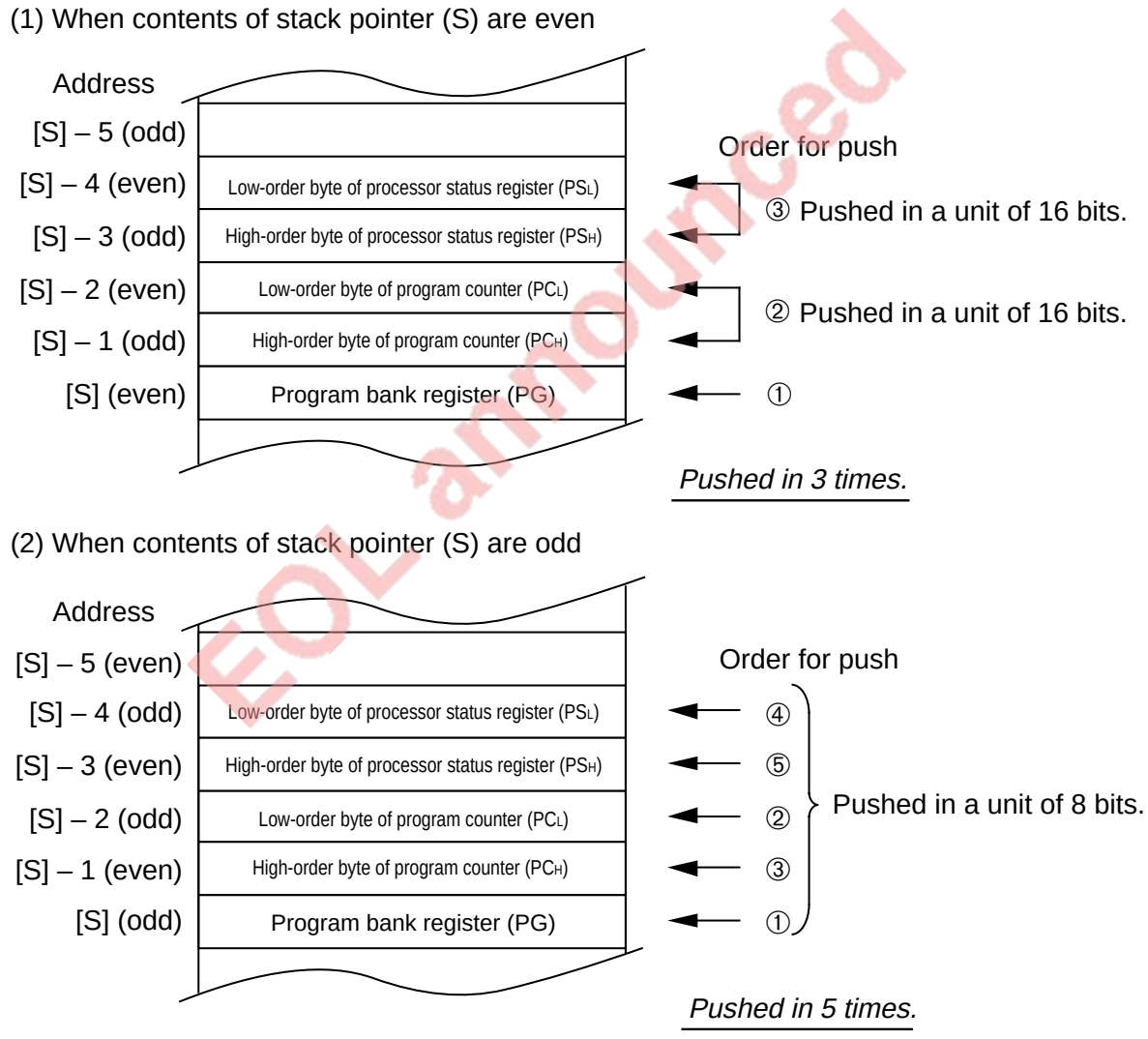
7.7.2 Push operation for registers

The push operation for registers performed in the INTACK sequence depends on whether the contents of the stack pointer (S) at acceptance of an interrupt request are even or odd.

When the contents of the stack pointer (S) are even, the contents of the program counter (PC) and the processor status register (PS) are simultaneously pushed in a unit of 16 bits. When the contents of the stack pointer (S) are odd, each of these registers is pushed in a unit of 8 bits. Figure 7.7.3 shows the push operation for registers.

In the INTACK sequence, only the contents of the program bank register (PG), program counter (PC), and processor status register (PS) are pushed onto the stack area. The other necessary registers must be pushed by software at the start of the interrupt routine.

By using the **PSH** instruction, all CPU registers except the stack pointer (S) can be pushed.



EOL announced

✻ [S] is an initial address that the stack pointer (S) indicates when an interrupt request is accepted. The S's contents become "[S] – 5" after all of the above registers are pushed.

Fig. 7.7.3 Push operation for registers

7.8 Return from interrupt routine

When the **RTI** instruction is executed at the end of the interrupt routine, the contents of the program bank register (PG), program counter (PC), and processor status register (PS) which were pushed onto the stack area just before the INTACK sequence, are automatically pulled. After this, the control returns to the original routine. And then, the suspended processing, which was in progress before the acceptance of the interrupt request, is resumed.

Before the **RTI** instruction is executed, pull registers which were pushed by software in the interrupt routine, using the **PUL** instruction, etc.

7.9 Multiple interrupts

Just after a branch is made to an interrupt routine, the following occur:

- Interrupt disable flag (I) = “1” (Interrupts are disabled)
- Interrupt request bit of accepted interrupt = “0”
- Processor interrupt priority level (IPL) = Interrupt priority level of accepted interrupt

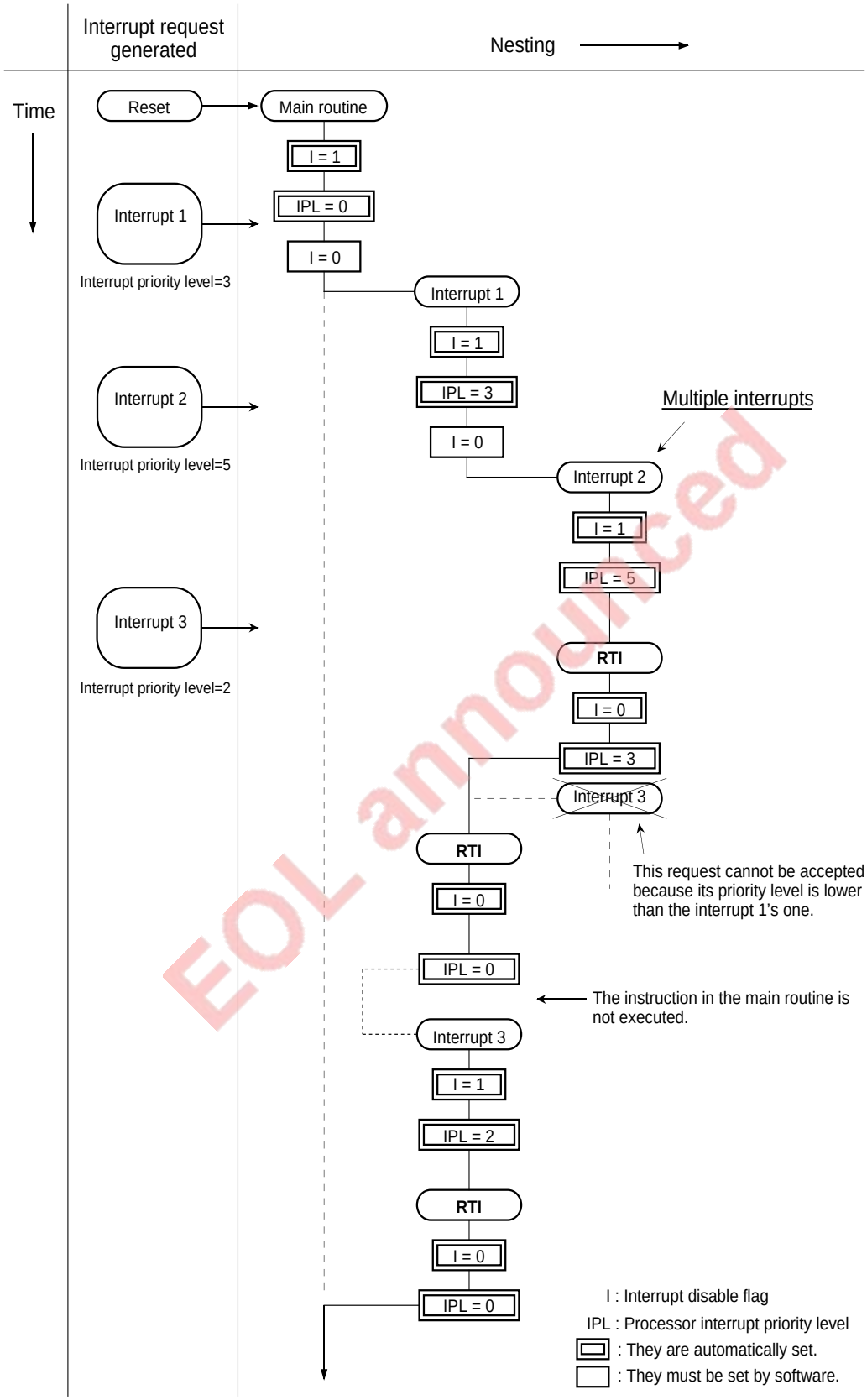
Accordingly, as long as the IPL remains unchanged, an interrupt request whose priority level is higher than that of the interrupt which is in progress can be accepted by clearing the interrupt disable flag (I) to “0” in an interrupt routine. In this way, multiple interrupts are processed.

Figure 7.9.1 shows the processing for multiple interrupts.

An interrupt request which has not been accepted because its priority level is lower is held. When the **RTI** instruction is executed, the interrupt priority level of the routine which was in progress at acceptance of an interrupt request is pulled into the IPL. Therefore, if the following relationship is satisfied when interrupt priority level detection is performed next, the held interrupt request is accepted.

Held interrupt request’s priority level > Processor interrupt priority level (IPL)

EOL announced



EOL announced

Fig. 7.9.1 Processing for multiple interrupts

7.10 External interrupts ($\overline{INT_i}$ interrupt)

An external interrupt request occurs by an input signal to the $\overline{INT_i}$ (i = 0 to 2) pin. The occurrence factor of the interrupt request can be selected by the level sense/edge sense select bit and the polarity select bit (bits 5 and 4 at addresses $7D_{16}$ to $7F_{16}$) shown in Figure 7.10.1. Table 7.10.1 lists the occurrence factor of $\overline{INT_i}$ interrupt request.

When using $P10_0/\overline{INT_0}$ to $P10_2/\overline{INT_2}$ pins as input pins of external interrupts, set the corresponding bits at address 18_{16} (port P10 direction register) to "0." (Refer to **"Figure 7.10.2."**)

The signals input to the $\overline{INT_i}$ pin require "H"- or "L"- level width of 250 ns or more, independent of $f(X_{IN})$. Additionally, even when using the pins $P10_0/\overline{INT_0}$ to $P10_2/\overline{INT_2}$ as the input pins of external interrupts, the user can obtain the pin's state by reading bits 0 to 2 at address 16_{16} (port P10 register).

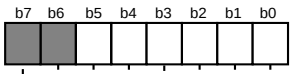
Note: When selecting an input signal's falling or "L" level as the occurrence factor of an interrupt request, make sure that the input signal is held "L" for 250 ns or more. When selecting an input signal's rising or "H" level as that, make sure that the input signal is held "H" for 250 ns or more.

Table 7.10.1 Occurrence factor of $\overline{INT_i}$ interrupt request

b5	b4	$\overline{INT_i}$ interrupt request occurrence factor
0	0	Interrupt request occurs at the falling edge of a signal input to pin $\overline{INT_i}$ (Edge sense).
0	1	Interrupt request occurs at the rising edge of a signal input to pin $\overline{INT_i}$ (Edge sense).
1	0	Interrupt request occurs when pin $\overline{INT_i}$ is at "H" level (Level sense).
1	1	Interrupt request occurs when pin $\overline{INT_i}$ is at "L" level (Level sense).

The $\overline{INT_i}$ interrupt request occurs by detecting the state of pin $\overline{INT_i}$ all the time. Therefore, when the user does not use the $\overline{INT_i}$ interrupt, set the $\overline{INT_i}$ interrupt's priority level to level 0.

EOL announced

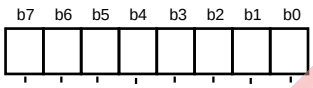


$\overline{INT_0}$ to $\overline{INT_2}$ interrupt control registers (Addresses $7D_{16}$ to $7F_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Interrupt priority level select bits	b2 b1 b0 0 0 0 : Level 0 (Interrupt disabled) 0 0 1 : Level 1 0 1 0 : Level 2 0 1 1 : Level 3 1 0 0 : Level 4 1 0 1 : Level 5 1 1 0 : Level 6 1 1 1 : Level 7	0	RW
1			0	RW
2			0	RW
3	Interrupt request bit **(Note)**	0 : No interrupt requested 1 : Interrupt requested	0	RW
4	Polarity select bit	0 : Interrupt request bit is set to "1" at "H" level when level sense is selected; this bit is set to "1" at falling edge when edge sense is selected. 1 : Interrupt request bit is set to "1" at "L" at level when level sense is selected; this bit is set to "1" at rising edge when edge sense is selected.	0	RW
5	Level sense/Edge sense select bit	0 : Edge sense 1 : Level sense	0	RW
7, 6	Nothing is assigned.		Undefined	—

Note: The interrupt request bits of $\overline{INT_0}$ to $\overline{INT_2}$ interrupts are invalid when the level sense is selected.

Fig. 7.10.1 Structure of $\overline{INT_i}$ (i=0 to 2) interrupt control register



Port P10 direction register (Address 18_{16})

Bit	Corresponding pin	Functions	At reset	RW
0	$\overline{INT_0}$ pin	0 : Input mode 1 : Output mode When using a pin as an input pin for an external interrupt,clear the corresponding bit to "0."	0	RW
1	$\overline{INT_1}$ pin		0	RW
2	$\overline{INT_2}$ pin		0	RW
3	$\overline{TC}$ pin		0	RW
4	$\overline{CAS}$ pin		0	RW
5	$\overline{RAS}$ pin		0	RW
6	$\overline{MA_8}$ pin		0	RW
7	$\overline{MA_9}$ pin		0	RW

: Bits 3 to 7 are not used for external interrupts.

Fig. 7.10.2 Relationship between port P10 direction register and input pins of external interrupt

EOL announced

7.10 External interrupts ($\overline{INT_i}$ interrupt)

7.10.1 Functions of $\overline{INT_i}$ interrupt request bit

(1) Functions when edge sense is selected

The interrupt request bit has the same functions as that of an internal interrupt. That is, when an interrupt request occurs, the interrupt request bit is set to "1" and retains this state until the interrupt request is accepted. When this bit is cleared to "0" by software, the interrupt request is cancelled; when this bit is set to "1" by software, the interrupt request can be generated.

(2) Functions when level sense is selected

The $\overline{INT_i}$ interrupt request bit is ignored.

Interrupt requests continuously occur while the level of the $\overline{INT_i}$ pin is the valid level*1; when the $\overline{INT_i}$ pin's level changes from the valid level to the invalid level*2 before the $\overline{INT_i}$ interrupt request is accepted, this interrupt request is not retained. (Refer to **"Figure 7.10.4."**)

Valid level*1: This means the level selected by the polarity select bit (bit 4 at addresses $7D_{16}$ to $7F_{16}$)
Invalid level*2: This means the reversed level of "valid level"

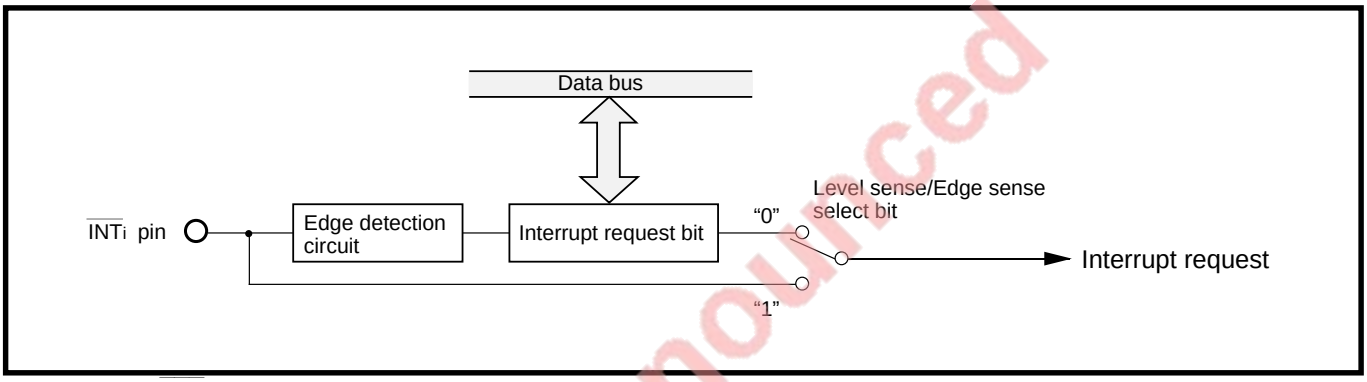


Fig. 7.10.3 $\overline{INT_i}$ Interrupt request

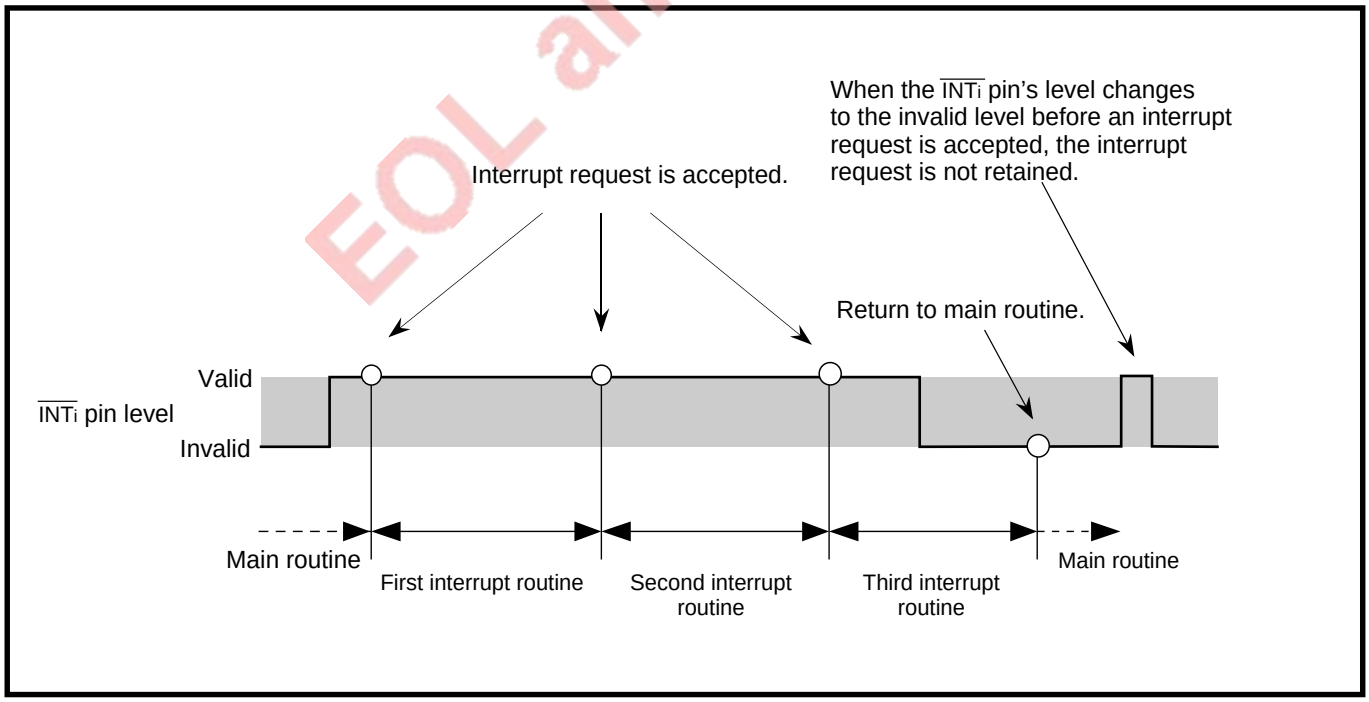


Fig. 7.10.4 Occurrence of $\overline{INT_i}$ interrupt request when level sense is selected

EOL announced

7.10.2 Switching of $\overline{INT_i}$ interrupt request occurrence factor

When the $\overline{INT_i}$ interrupt request occurrence factor is switched in one of the following ways, the interrupt request bit may be set to "1":

- Switching the level sense to the edge sense
- Switching polarity

Therefore, after this switching, make sure to clear the interrupt request bit to "0." Figure 7.10.5 shows an example of the switching procedure for the $\overline{INT_i}$ interrupt request occurrence factor.

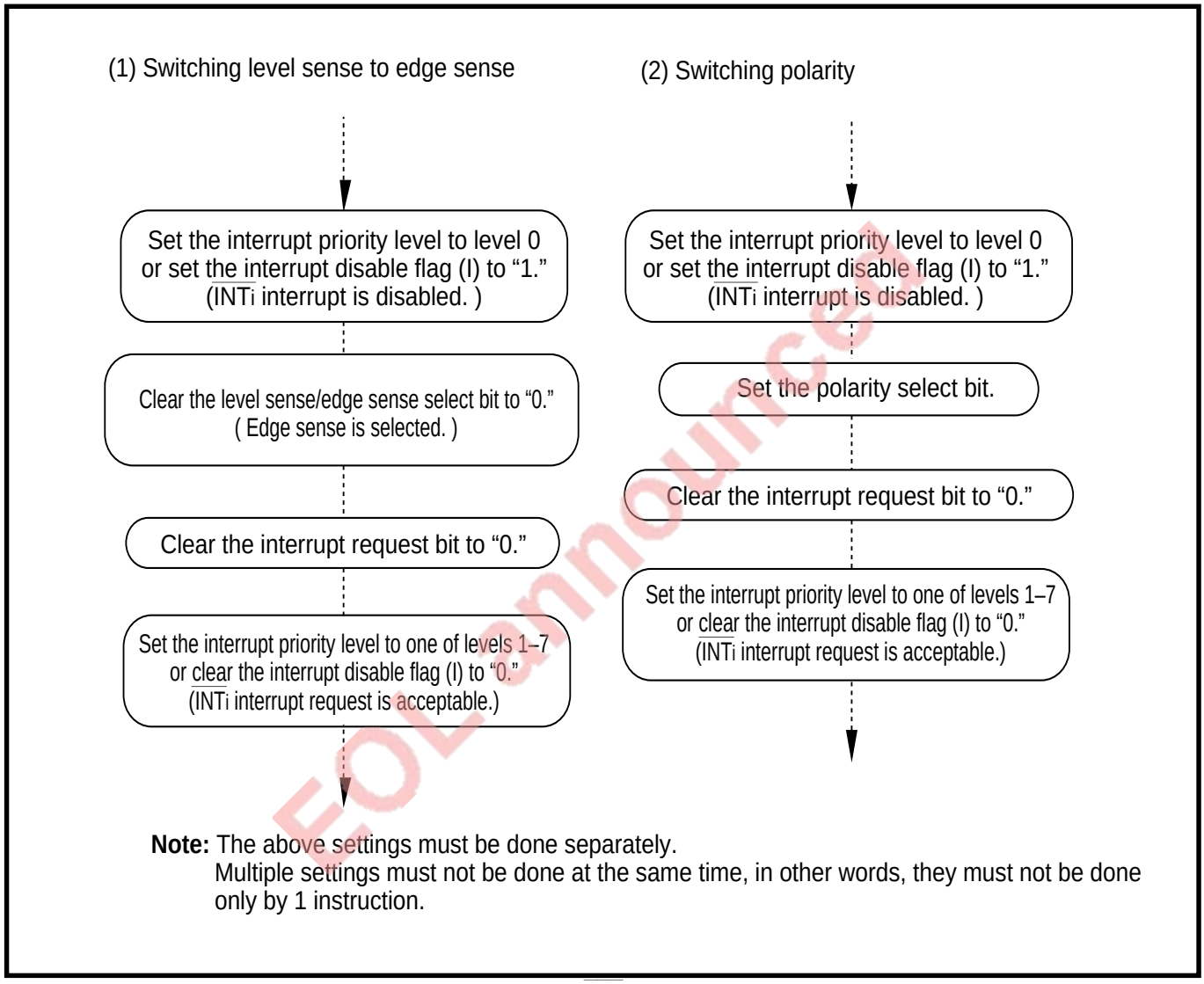


Fig. 7.10.5 Example of switching procedure for $\overline{INT_i}$ interrupt request occurrence factor

EOL announced

7.11 Precautions for interrupts

When changing the interrupt priority level select bits (bits 0 to 2 at addresses $6C_{16}$ to $7F_{16}$), 2 to 7 cycles of ϕ are required until the interrupt priority level is changed. Therefore, when the interrupt priority level of a certain interrupt source is repeatedly changed in a very short time, which consists of a few instructions, it is necessary to reserve the time required for the change by software. Figure 7.11.1 shows a program example to reserve the time required for the change. Note that the time required for the change depends on the contents of the interrupt priority detection time select bits (bits 4 and 5 at address $5E_{16}$). Table 7.11.1 lists the correspondence between the number of instructions inserted in Figure 7.11.1 and the interrupt priority detection time select bits.

```
:
LDM.B #0XH, 00XXH  ; Write instruction for the interrupt priority level select bits
NOP                ; Inserted NOP instruction (Note)
NOP                ;
NOP                ;
LDM.B #0XH, 00XXH  ; Write instruction for the interrupt priority level select bits
:
```

Note: Except the write instruction for address XX_{16}, any instruction which has the same cycles as the **NOP** instruction can also be inserted.
For the number of inserted **NOP** instructions, refer to **"Table 7.11.1."**

XX: any of 6C to 7F

Fig. 7.11.1 Program example to reserve time required for change of interrupt priority level

Table 7.11.1 Correspondence between number of instructions to be inserted in Figure 7.11.1 and interrupt priority detection time select bits

Interrupt priority detection time select bits **(Note)**		Interrupt priority level detection time	Number of inserted **NOP** instructions
b5	b4		
0	0	7 cycles of ϕ	4 or more
0	1	4 cycles of ϕ	2 or more
1	0	2 cycles of ϕ	1 or more
1	1	Do not select.	

Note: We recommend [b5 = "1", b4 = "0"].

EOL announced

CHAPTER 8
TIMER A

8.1 Overview
8.2 Block description
8.3 Timer mode
[Precautions for timer mode]
8.4 Event counter mode
[Precautions for event counter mode]
8.5 One-shot pulse mode
[Precautions for one-shot pulse mode]
8.6 Pulse width modulation (PWM) mode
[Precautions for pulse width modulation (PWM) mode]

EOL announced

8.1 Overview

Timer A consists of five counters, Timers A0 to A4, each equipped with a 16-bit reload function. Timers A0 to A4 operate independently of one another.
Timer A has four operating modes listed below. Timers A0 and A1 operate in the timer mode only. Timers A2 to A4 have selective four operating modes listed below.

(1) Timer mode (Timers A0 to A4)
The timer counts an internally generated count source. For Timers A2 to A4, the following functions can be used in this mode:
•Gate function
•Pulse output function

(2) Event counter mode (Timers A2 to A4)
The timer counts an external signal. The following functions can be used in this mode:
•Pulse output function
•Two-phase pulse signal processing function

(3) One-shot pulse mode (Timers A2 to A4)
The timer outputs a pulse which has an arbitrary width once.

(4) Pulse width modulation (PWM) mode (Timers A2 to A4)
Timer outputs pulses which have an arbitrary width in succession. The counter functions as one of the following pulse width modulators:
•16-bit pulse width modulator
•8-bit pulse width modulator

In this chapter, Timer Ai (i = 0 to 4) indicates Timers A0 to A4. Timer Aj (j = 2 to 4) indicates Timers A2 to A4; this is applies when the timer A's input/output pins are used etc. (Hereafter, input/output pins are called I/O pins.)

EOL announced

8.2 Block description

Figure 8.2.1 shows the block diagram of Timer A. Explanation of registers relevant to Timer A is described below.

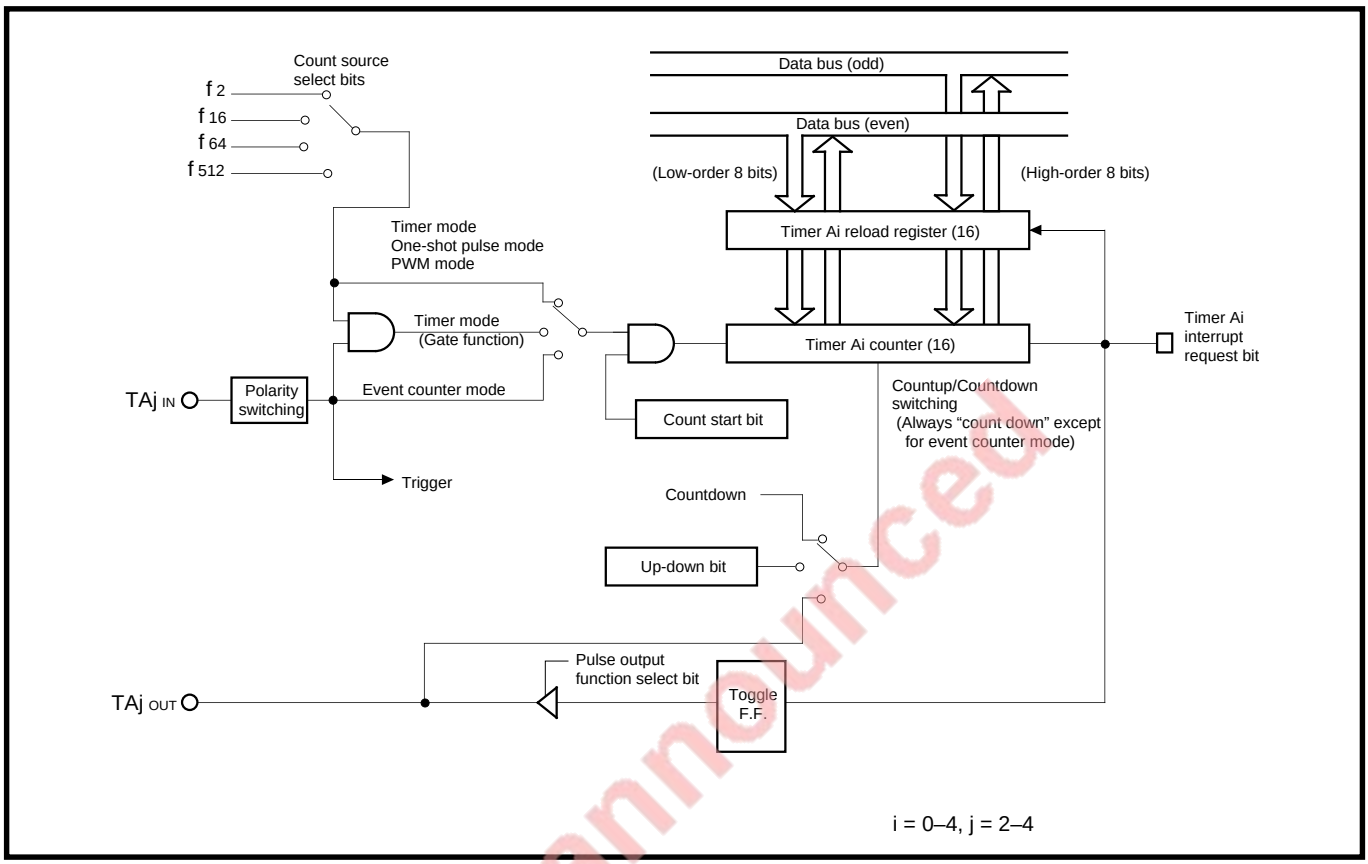


Fig. 8.2.1 Block diagram of Timer A

EOL announced

8.2 Block description

8.2.1 Counter and reload register (timer Ai register)

Each of timer Ai counter and reload register consists of 16 bits.

Countdown in the counter is performed each time the count source is input. In the event counter mode, it can also function as an up-counter.

The reload register is used to store the initial value of the counter. When a counter underflow or overflow occurs, the reload register's contents are reloaded into the counter.

A value is set to the counter and reload register by writing the value to the timer Ai register. Table 8.2.1 lists the memory assignment of the timer Ai register.

The value written into the timer Ai register while counting is not in progress is set to the counter and reload register. The value written into the timer Ai register while counting is in progress is set only to the reload register. In this case, the reload register's updated contents are transferred to the counter at the next reload time. The value obtained when reading out the timer Ai register varies according to the operating mode. Table 8.2.2 lists reading from and writing to the timer Ai register.

Table 8.2.1 Memory assignment of timer Ai register

Timer Ai register	High-order byte	Low-order byte
Timer A0 register	Address 47_{16}	Address 46_{16}
Timer A1 register	Address 49_{16}	Address 48_{16}
Timer A2 register	Address $4B_{16}$	Address $4A_{16}$
Timer A3 register	Address $4D_{16}$	Address $4C_{16}$
Timer A4 register	Address $4F_{16}$	Address $4E_{16}$

Note: At reset, the contents of the timer Ai register are undefined.

Table 8.2.2 Reading from and writing to timer Ai register

Operating mode	Read	Write
Timer mode	Counter value is read out. (**Note 1**)	<While counting> Written only to reload register. <While not counting> Written to both of the counter and reload register.
Event counter mode		
One-shot pulse mode	Undefined value is read out.	
Pulse width modulation (PWM) mode		

Notes 1: Also refer to **"[Precautions for timer mode]"** and **"[Precautions for event counter mode]."**

2: When reading from and writing to the timer Ai register, perform it in a unit of 16 bits.

EOL announced

8.2.2 Count start register

This register is used to start and stop counting. Each bit of this register corresponds to each timer. Figure 8.2.2 shows the structure of the count start register.



Count start register (Address 40_{16})

Bit	Bit name	Functions	At reset	RW
0	Timer A0 count start bit	0 : Stop counting 1 : Start counting	0	RW
1	Timer A1 count start bit		0	RW
2	Timer A2 count start bit		0	RW
3	Timer A3 count start bit		0	RW
4	Timer A4 count start bit		0	RW
5	Timer B0 count start bit		0	RW
6	Timer B1 count start bit		0	RW
7	Timer B2 count start bit		0	RW

: Bits 5 to 7 are not used for Timer A.

Fig. 8.2.2 Structure of count start register

EOL announced

8.2 Block description

8.2.3 Timer Ai mode register

Figure 8.2.3 shows the structure of the timer Ai mode register. The operating mode select bits are used to select the operating mode of Timer Ai. Bits 2 to 7 have different functions according to the operating mode. These bits are described in the paragraph of each operating mode.

Timer Ai mode register (i = 0 to 4) (Addresses 56_{16} to $5A_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 0 0 : Timer mode 0 1 : Event counter mode 1 0 : One-shot pulse mode 1 1 : Pulse width modulation (PWM) mode	0	RW
1			0	RW
2	These bits have different functions according to the operating mode.		0	RW
3			0	RW
4			0	RW
5			0	RW
6			0	RW
7			0	RW

Fig. 8.2.3 Structure of timer Ai mode register

EOL announced

8.2.4 Timer Ai interrupt control register

Figure 8.2.4 shows the structure of the timer Ai interrupt control register. For details about interrupts, refer to **"CHAPTER 7. INTERRUPTS."**



Timer Ai interrupt control register (i = 0 to 4) (Addresses 75_{16} to 79_{16})

Bit	Bit name	Functions	At reset	RW
0	Interrupt priority level select bits	b2 b1 b0 0 0 0 : Level 0 (Interrupt disabled) 0 0 1 : Level 1 Low level 0 1 0 : Level 2 0 1 1 : Level 3 1 0 0 : Level 4 1 0 1 : Level 5 1 1 0 : Level 6 1 1 1 : Level 7 High level	0	RW
1			0	RW
2			0	RW
3	Interrupt request bit	0 : No interrupt requested 1 : Interrupt requested	0	RW
7 to 4	Nothing is assigned.		Undefined	—

Fig. 8.2.4 Structure of timer Ai interrupt control register

(1) Interrupt priority level select bits (bits 2 to 0)

These bits select a timer Ai interrupt's priority level. When using timer Ai interrupts, select one of the priority levels (1 to 7). When a timer Ai interrupt request occurs, its priority level is compared with the processor interrupt priority level (IPL). The requested interrupt is enabled only when its priority level is higher than the IPL. (However, this applies when the interrupt disable flag (I) = "0.") To disable timer Ai interrupts, set these bits to "000_2" (level 0).

(2) Interrupt request bit (bit 3)

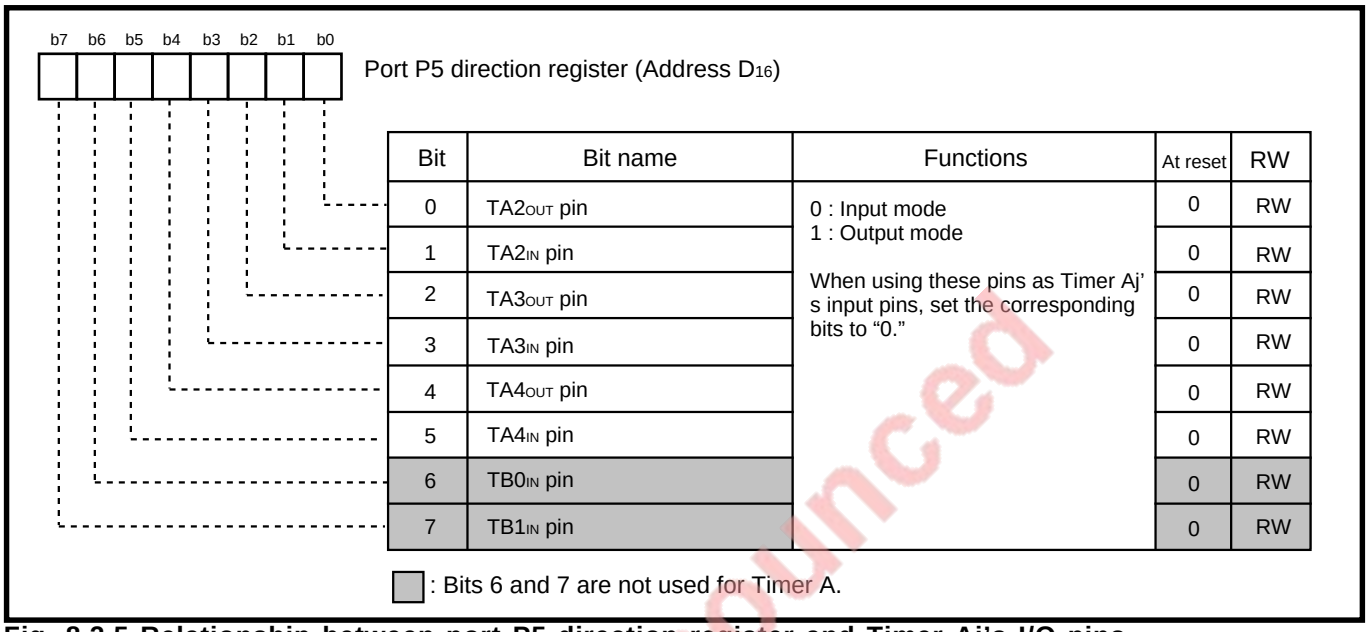
This bit is set to "1" when a timer Ai interrupt request occurs. This bit is automatically cleared to "0" when the timer Ai interrupt request is accepted. This bit can be set to "1" or cleared to "0" by software.

EOL announced

8.2 Block description

8.2.5 Port P5 direction register

The I/O pins of Timers A2 to A4 are multiplexed with port P5. When using these pins as Timer Aj's input pins, set the corresponding bits of the port P5 direction register to "0" to set these port pins for the input mode. When used as Timer Aj's output pins, these pins are forcibly set to the output pins of Timer Aj regardless of the direction registers's contents. Figure 8.2.5 shows the relationship between the port P5 direction register and the Timer Aj's I/O pins.



Bit	Bit name	Functions	At reset	RW
0	TA2OUT pin	0 : Input mode 1 : Output mode When using these pins as Timer Aj's input pins, set the corresponding bits to "0."	0	RW
1	TA2IN pin		0	RW
2	TA3OUT pin		0	RW
3	TA3IN pin		0	RW
4	TA4OUT pin		0	RW
5	TA4IN pin		0	RW
6	TB0IN pin		0	RW
7	TB1IN pin		0	RW

: Bits 6 and 7 are not used for Timer A.

Fig. 8.2.5 Relationship between port P5 direction register and Timer Aj's I/O pins

EOL announced

8.3 Timer mode

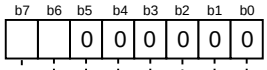
In this mode, the timer counts an internally generated count source. (Refer to **"Table 8.3.1."**) Figure 8.3.1 shows the structures of the timer Ai mode register and timer Ai register in the timer mode.

Table 8.3.1 Specifications of timer mode

Item	Specifications
Count source	f_2, f_{16}, f_{64}, or f_{512}
Count operation	• Countdown • When a counter underflow occurs, reload register's contents are reloaded, and counting continues.
Division ratio	$\frac{1}{(n+1)}$ n : Timer Ai register's set value
Count start condition	When the count start bit is set to "1."
Count stop condition	When the count start bit is cleared to "0."
Interrupt request occurrence timing	When a counter underflow occurs.
TAj_{IN} pin's function	Programmable I/O port or gate input
TAj_{OUT} pin's function	Programmable I/O port or pulse output
Read from timer Ai register	Counter value can be read out.
Write to timer Ai register	● While counting is stopped When a value is written to the timer Ai register, it is written to both of the reload register and counter. ● While counting is in progress When a value is written to the timer Ai register, it is written only to the reload register. (Transferred to the counter at the next reload timing.)

EOL announced

8.3 Timer mode



Timer A0 mode register (Address 56_{16})
Timer A1 mode register (Address 57_{16})

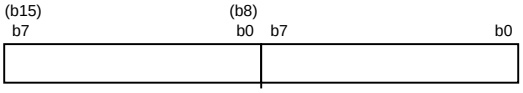
Bit	Bit name	Functions	At reset	RW
0	Fix these bits to "0."		0	RW
1			0	RW
2			0	RW
3			0	RW
4			0	RW
5			0	RW
6	Count source select bits	b7 b6 0 0 : f_2 0 1 : f_{16} 1 0 : f_{64} 1 1 : f_{512}	0	RW
7			0	RW



Timer Aj mode register (j = 2 to 4) (Addresses 58_{16} to $5A_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 0 0 : Timer mode	0	RW
1			0	RW
2	Pulse output function select bit	0 : No pulse output (TAj$_{OUT}$ pin functions as a programmable I/O port.) 1 : Pulse output (TAj$_{OUT}$ pin functions as a pulse output pin.)	0	RW
3	Gate function select bits	b4 b3 0 0 : No gate function (TAj$_{IN}$ pin functions as a programmable I/O port.) 0 1 : No gate function (TAj$_{IN}$ pin functions as a programmable I/O port.) 1 0 : Counter counts only while TAj$_{IN}$ pin's input signal is at "L" level. 1 1 : Counter counts only while TAj$_{IN}$ pin's input signal is at "H" level.	0	RW
4			0	RW
5	Fix this bit to "0" in timer mode.		0	RW
6	Count source select bits	b7 b6 0 0 : f_2 0 1 : f_{16} 1 0 : f_{64} 1 1 : f_{512}	0	RW
7			0	RW

EOL announced



Timer A0 register (Addresses 47_{16}, 46_{16})
Timer A1 register (Addresses 49_{16}, 48_{16})
Timer A2 register (Addresses $4B_{16}$, $4A_{16}$)
Timer A3 register (Addresses $4D_{16}$, $4C_{16}$)
Timer A4 register (Addresses $4F_{16}$, $4E_{16}$)

Bit	Functions	At reset	RW
15 to 0	These bits can be set to "0000_{16}" to "$FFFF_{16}$." Assuming that the set value = n, the counter divides the count source frequency by (n + 1). When reading, the register indicates the counter value.	Undefined	RW

Note: Read from or write to this register in a unit of 16 bits.

Fig. 8.3.1 Structures of timer Ai mode register and timer Ai register in timer mode

8.3.1 Setting for timer mode

Figures 8.3.2 and 8.3.3 show an initial setting example for registers relevant to the timer mode.
Note that when using interrupts, set up to enable the interrupts. For details, refer to section **"CHAPTER 7. INTERRUPTS."**

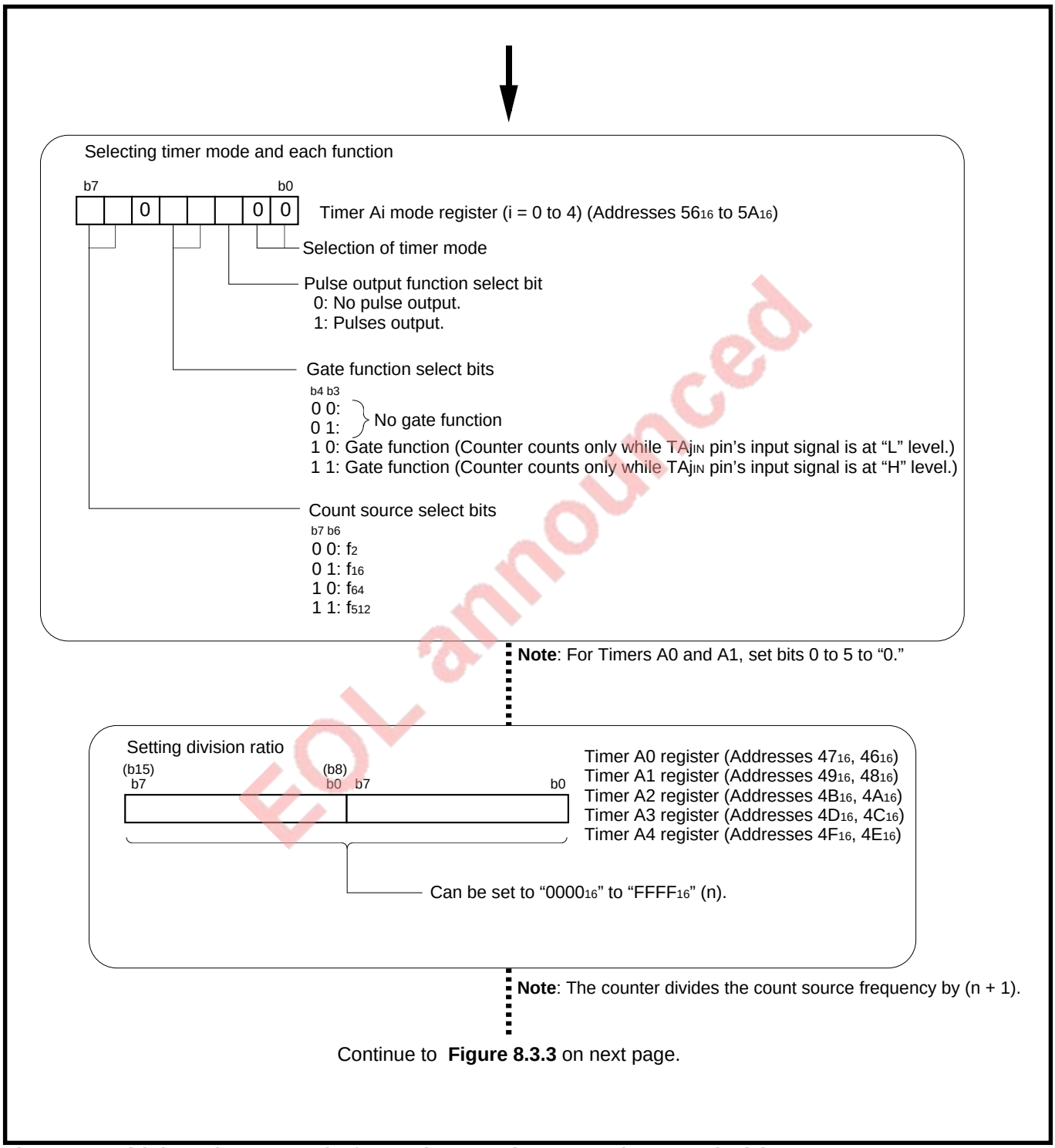
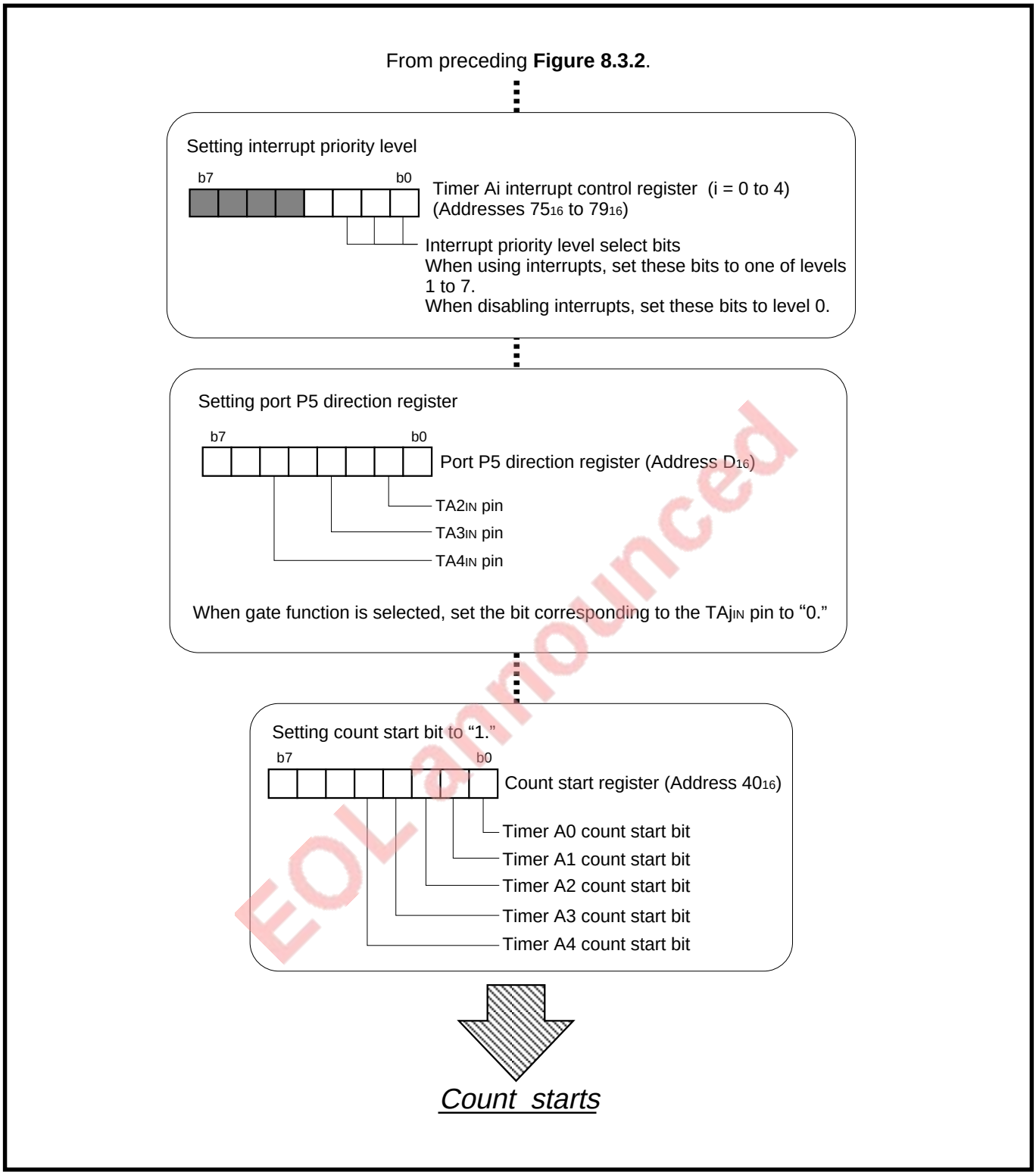


Fig. 8.3.2 Initial setting example for registers relevant to timer mode (1)

EOL announced



EOL announced

Fig. 8.3.3 Initial setting example for registers relevant to timer mode (2)

8.3.2 Count source

In the timer mode, the count source select bits (bits 6 and 7 at addresses 56_{16} to $5A_{16}$) select the count source. Table 8.3.2 lists the count source frequency.

Table 8.3.2 Count source frequency

Count source select bits		Count source	Count source frequency		
b7	b6		$f(X_{IN})$ = 8 MHz	$f(X_{IN})$ = 16 MHz	$f(X_{IN})$ = 25 MHz
0	0	f_2	4 MHz	8 MHz	12.5 MHz
0	1	f_{16}	500 kHz	1 MHz	1.5625 MHz
1	0	f_{64}	125 kHz	250 kHz	390.625 kHz
1	1	f_{512}	15625 Hz	31250 Hz	48.8281 kHz

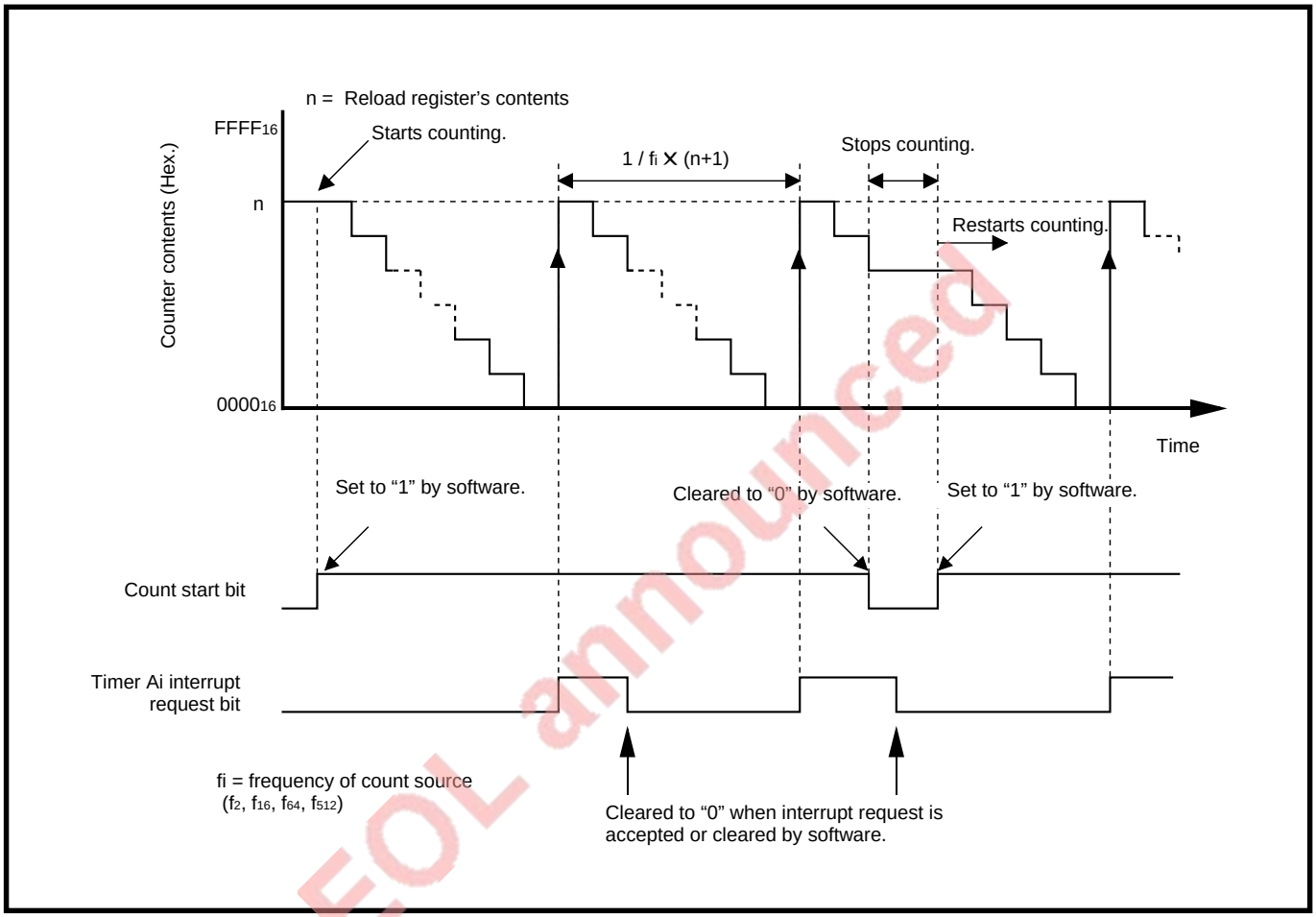
EOL announced

8.3 Timer mode

8.3.3 Operation in timer mode

① When the count start bit is set to “1,” the counter starts counting of the count source.
② When a counter underflow occurs, the reload register’s contents are reloaded, and counting continues.
③ The timer Ai interrupt request bit is set to “1” at the underflow in ②. The interrupt request bit remains set to “1” until the interrupt request is accepted or the interrupt request bit is cleared to “0” by software.

Figure 8.3.4 shows an example of operation in the timer mode.



EOL announced

Fig. 8.3.4 Example of operation in timer mode (without pulse output and gate functions)

8.3.4 Selectable functions

The following describes the selectable gate function for Timers A2 to A4 and pulse output function.

(1) Gate function

The gate function is selected by setting the gate function select bits (bits 4 and 3 at addresses 58_{16} to $5A_{16}$) to "10_2" or "11_2." The gate function makes it possible to start or stop counting depending on the TAj_{IN} pin's input signal. Table 8.3.3 lists the count valid levels. Figure 8.3.5 shows an example of operation with the gate function selected.

When selecting the gate function, set the port P5 direction registers' bits which correspond to the TAj_{IN} pin for the input mode. Additionally, make sure that the TAj_{IN} pin's input signal has a pulse width equal to or more than two cycles of the count source.

Table 8.3.3 Count valid levels

Gate function select bits		Count valid level (Duration while counter counts)
b4	b3	
1	0	While TAj_{IN} pin's input signal is at "L" level
1	1	While TAj_{IN} pin's input signal is at "H" level

Note: The counter does not count while the TAj_{IN} pin's input signal is not at the count valid level.

EOL announced

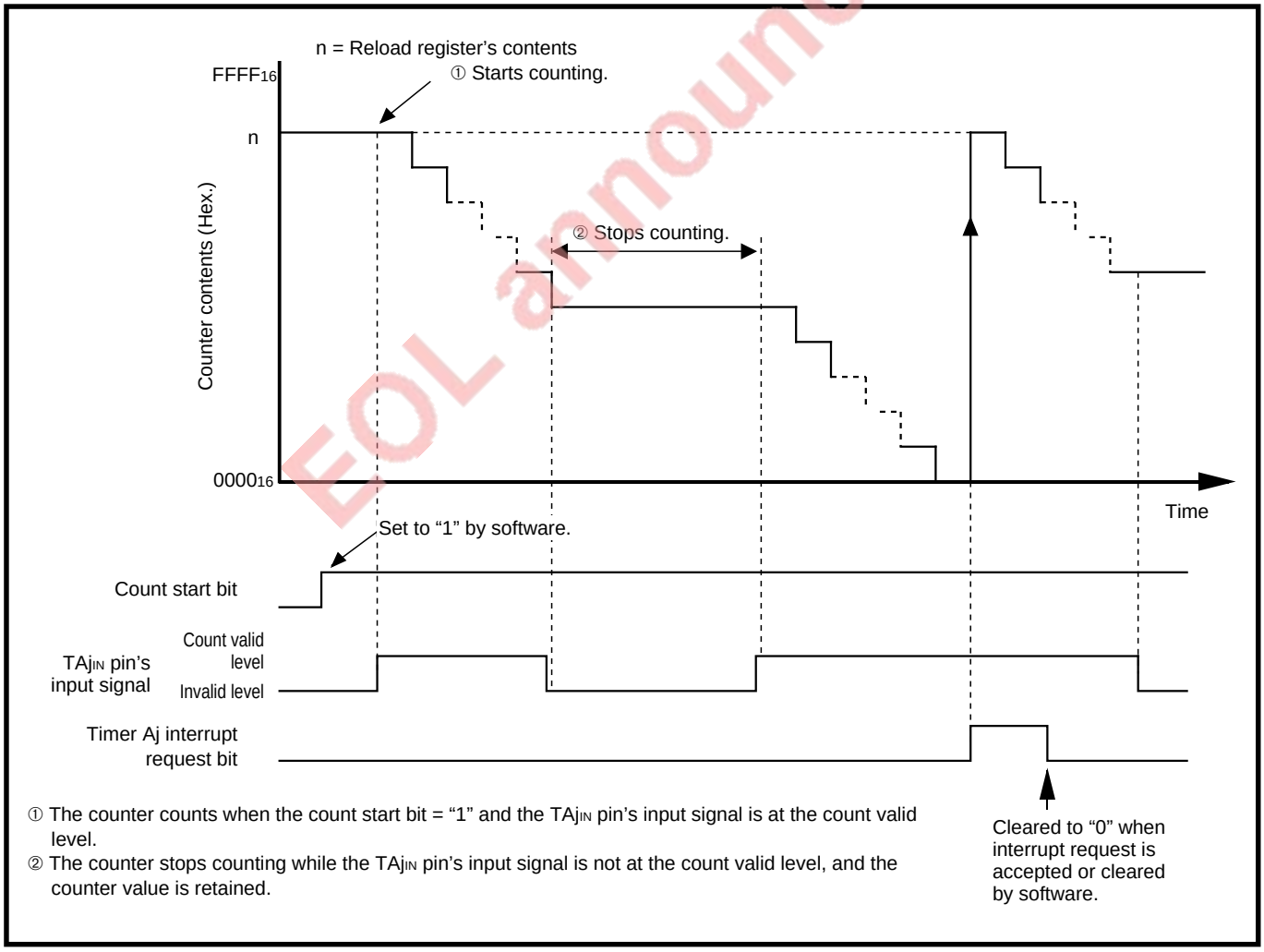


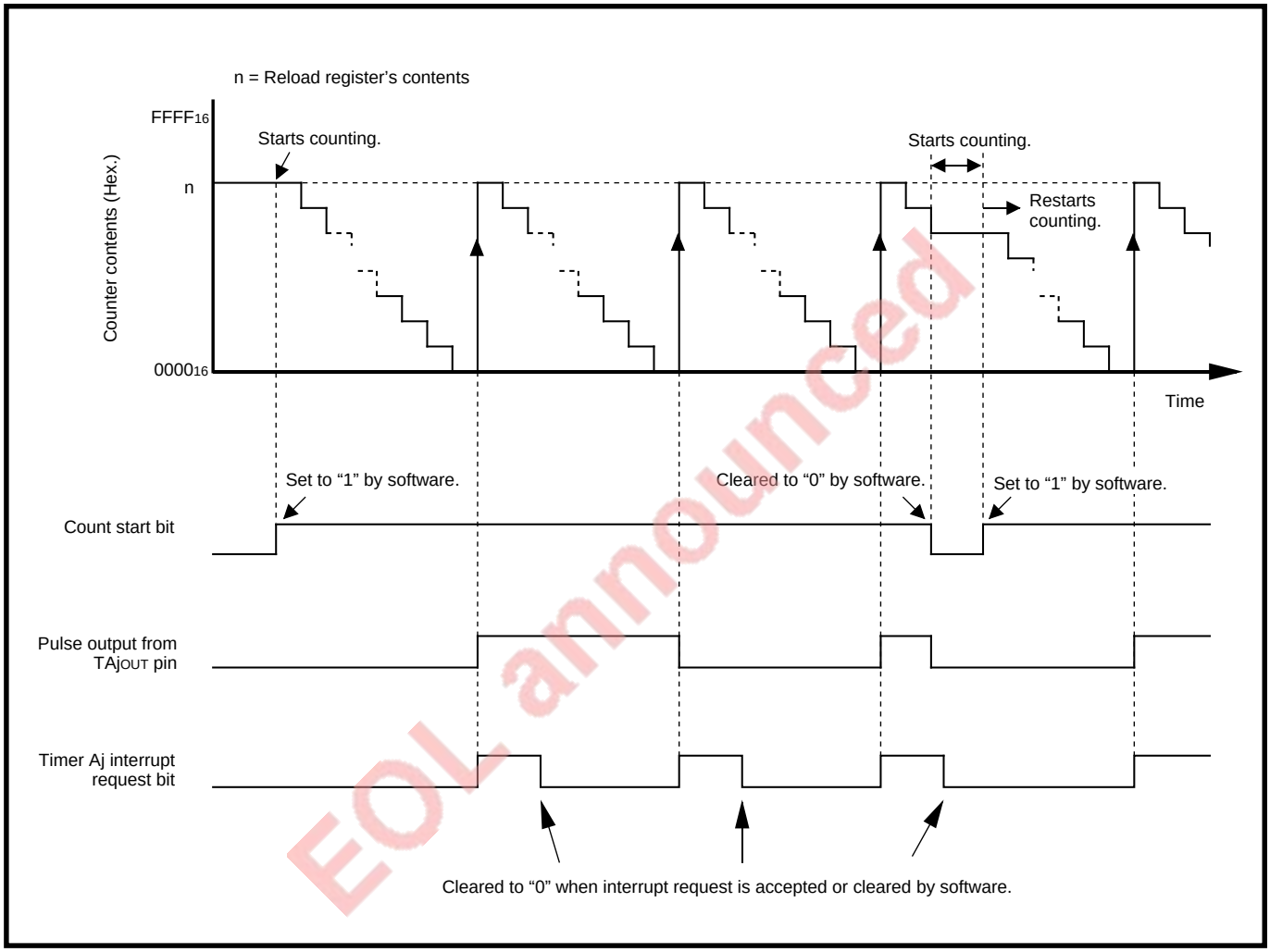
Fig. 8.3.5 Example of operation selecting gate function

8.3 Timer mode

(2) Pulse output function

The pulse output function is selected by setting the pulse output function select bit (bit 2 at addresses 58_{16} to $5A_{16}$) to "1." When this function is selected, the TAj$_{OUT}$ pin is forcibly set for the pulse output pin regardless of the corresponding bits of the port P5 direction register. The TAj$_{OUT}$ pin outputs pulses of which polarity is inverted each time a counter underflow occurs.

When the count start bit (address 40_{16}) is "0" (count stopped), the TAj$_{OUT}$ pin outputs "L" level. Figure 8.3.6 shows an example of operation with the pulse output function selected.



EOL announced

Fig. 8.3.6 Example of operation selecting pulse output function

[Precautions for timer mode]

By reading the timer Ai register, the counter value can be read out at any timing. However, if the timer Ai register is read at the reload timing shown in Figure 8.3.7, the value "$FFFF_{16}$" is read out. If reading is performed in the period from when a value is set into the timer Ai register with the counter stopped until the counter starts counting, the set value is correctly read out.

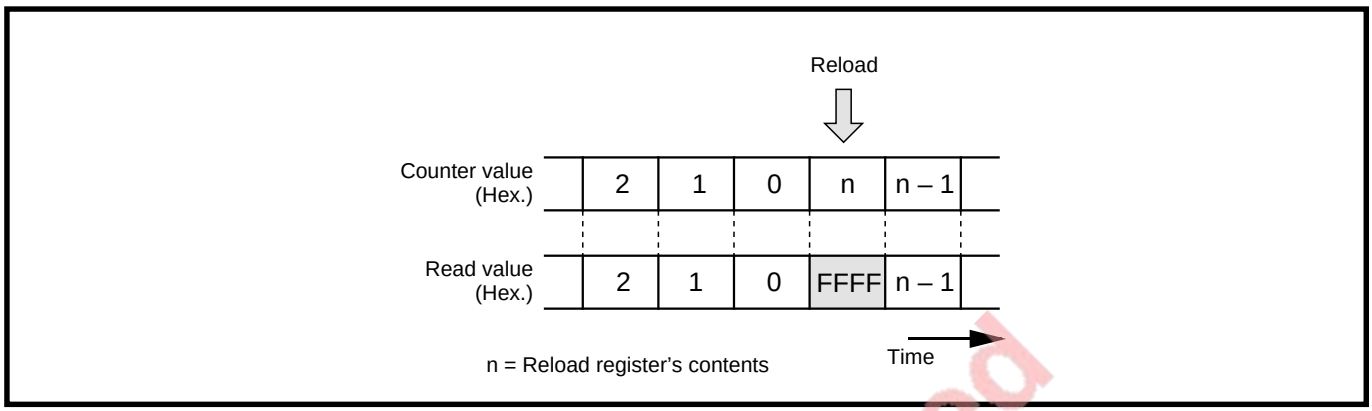


Fig. 8.3.7 Reading timer Ai register

EOL announced

8.4 Event counter mode

In this mode, the timer counts an external signal. (Refer to **"Tables 8.4.1 and 8.4.2."**) Timers A2 to A4 can be used in this mode. Figure 8.4.1 shows the structures of the timer Aj mode register and timer Aj register in the event counter mode.

Table 8.4.1 Specifications of event counter mode (when not using two-phase pulse signal processing function)

Item	Specifications
Count source	● External signal input to the TAj_{IN} pin ● The count source's valid edge can be selected from the falling edge and the rising edge by software.
Count operation	● Countup or countdown can be switched by external signal or software. ● When a counter overflow or underflow occurs, reload register's contents are reloaded, and counting continues.
Division ratio	● For countdown $\frac{1}{(n+1)}$ n : Timer Aj register's set value ● For countup $\frac{1}{(FFFF_{16}-n+1)}$
Count start condition	When the count start bit is set to "1."
Count stop condition	When the count start bit is cleared to "0."
Interrupt request occurrence timing	When a counter overflow or underflow occurs.
TAj_{IN} pin's function	Count source input
TAj_{OUT} pin's function	Programmable I/O port, pulse output, or countup/countdown switch signal input
Read from timer Aj register	Counter value can be read out.
Write to timer Aj register	● While counting is stopped When a value is written to the timer Aj register, it is written to both of the reload register and counter. ● While counting is in progress When a value is written to the timer Aj register, it is written only to the reload register. (Transferred to the counter at the next reload time.)

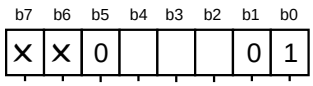
EOL announced

Table 8.4.2 Specifications of event counter mode (when using two-phase pulse signal processing function)

Item	Specifications
Count source	External signal (two-phase pulse) input to the TAj_{IN} or TAj_{OUT} pin
Count operation	● Countup or countdown can be switched by external signal (two-phase pulse). ● When a counter overflow or underflow occurs, reload register's contents are reloaded, and counting continues.
Division ratio	● For countdown $\frac{1}{(n+1)}$ n : Timer Aj register's set value ● For countup $\frac{1}{(FFFF_{16} - n + 1)}$
Count start condition	When the count start bit is set to "1."
Count stop condition	When the count start bit is cleared to "0."
Interrupt request occurrence timing	When a counter overflow or underflow occurs.
TAj_{IN}, TAj_{OUT} pin function	Two-phase pulse input
Read from timer Aj register	Counter value can be read out.
Write to timer Aj register	● While counting is stopped When a value is written to the timer Aj register, it is written to both of the reload register and counter. ● While counting is in progress When a value is written to the timer Aj register, it is written only to the reload register. (Transferred to the counter at the next reload time.)

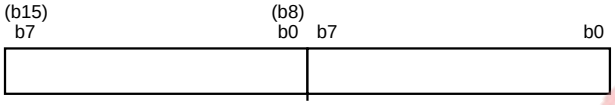
EOL announced

8.4 Event counter mode



Timer Aj mode register (j = 2 to 4) (Addresses 58_{16} to $5A_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 0 1 : Event counter mode	0	RW
1			0	RW
2	Pulse output function select bit	0 : No pulse output (TAj$_{OUT}$ pin functions as a programmable I/O port.) 1 : Pulse output (TAj$_{OUT}$ pin functions as a pulse output pin.)	0	RW
3	Count polarity select bit	0 : Counts at falling edge of external signal 1 : Counts at rising edge of external signal	0	RW
4	Up-down switching factor select bit	0 : Contents of up-down register 1 : Input signal to TAj$_{OUT}$ pin	0	RW
5	Fix this bit to “0” in event counter mode.		0	RW
6	These bits are invalid in event counter mode.		0	RW
7			0	RW



Timer A2 register (Addresses $4B_{16}$, $4A_{16}$)
Timer A3 register (Addresses $4D_{16}$, $4C_{16}$)
Timer A4 register (Addresses $4F_{16}$, $4E_{16}$)

Bit	Functions	At reset	RW
15 to 0	These bits can be set to “0000_{16}” to “$FFFF_{16}$.” Assuming that the set value = n, the counter divides the count source frequency by (n + 1) during countdown, or by ($FFFF_{16}$ – n + 1) during countup. When reading, the register indicates the counter value.	Undefined	RW

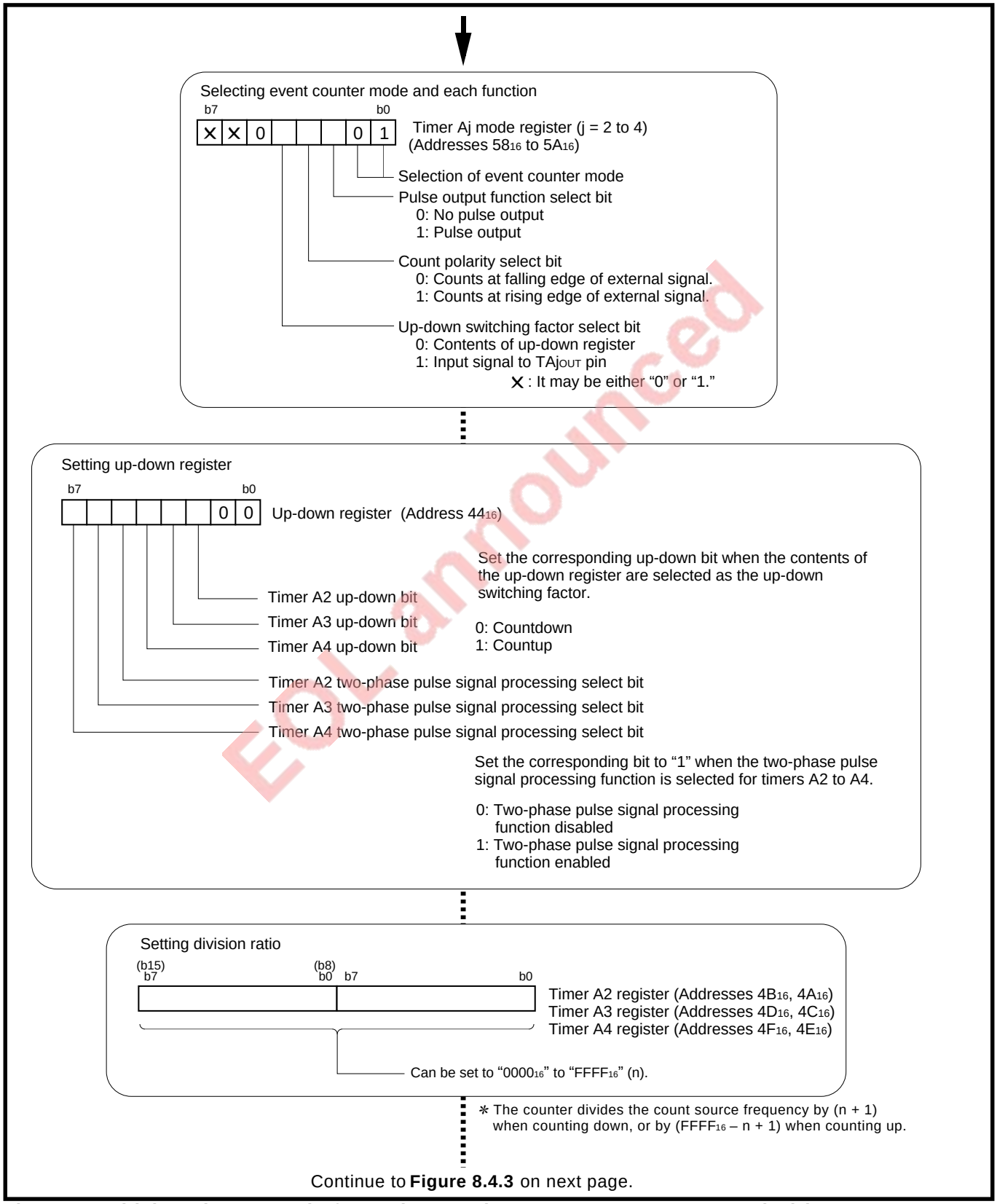
Note: Read from or write to this register in a unit of 16 bits.

EOL announced

Fig. 8.4.1 Structures of timer Aj mode register and timer Aj register in event counter mode

8.4.1 Setting for event counter mode

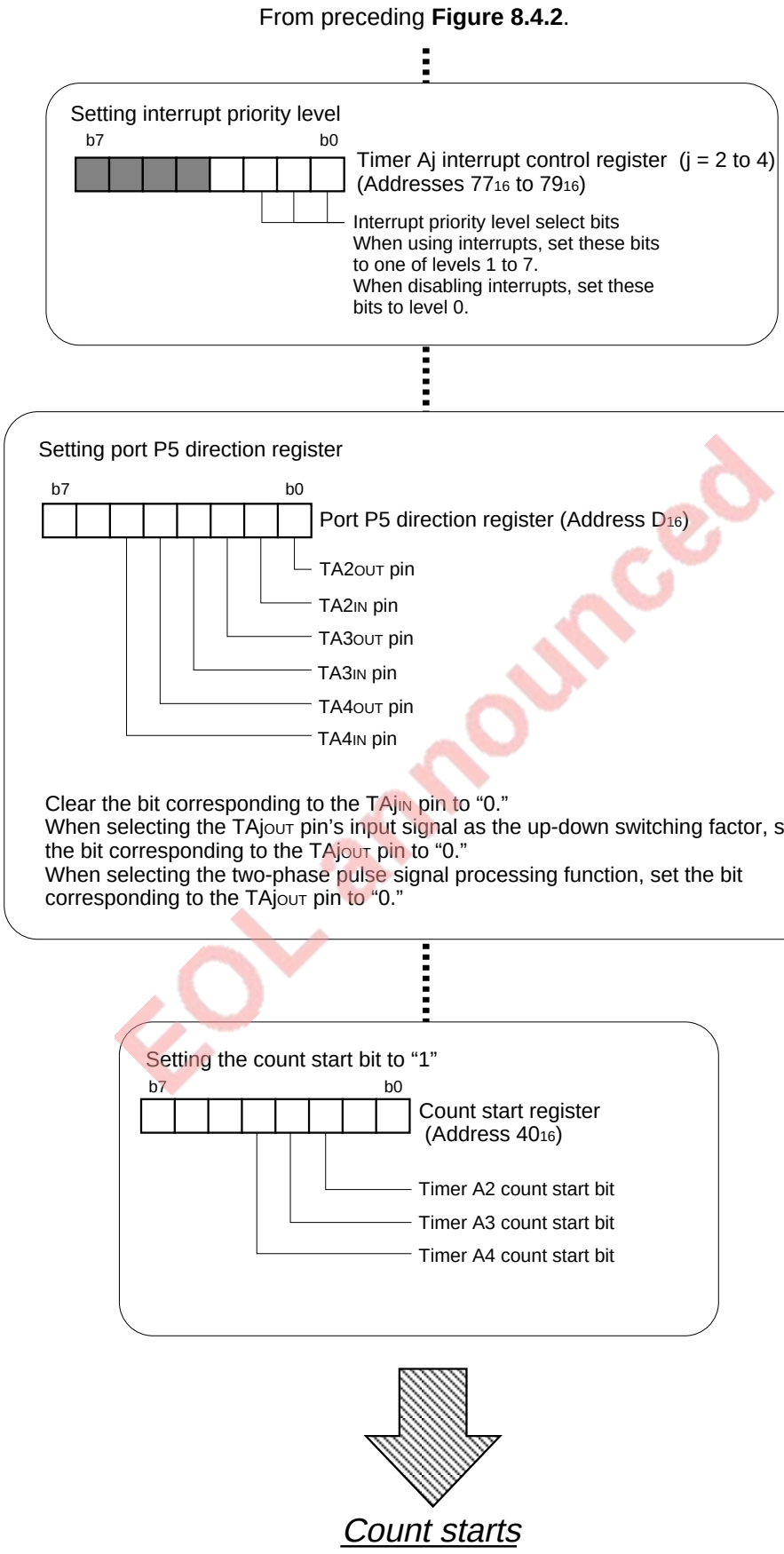
Figures 8.4.2 and 8.4.3 show an initial setting example for registers relevant to the event counter mode. Note that when using interrupts, set up to enable the interrupts. For details, refer to **"CHAPTER 7. INTERRUPTS."**



Continue to **Figure 8.4.3** on next page.

Fig. 8.4.2 Initial setting example for registers relevant to event counter mode (1)

EOL announced



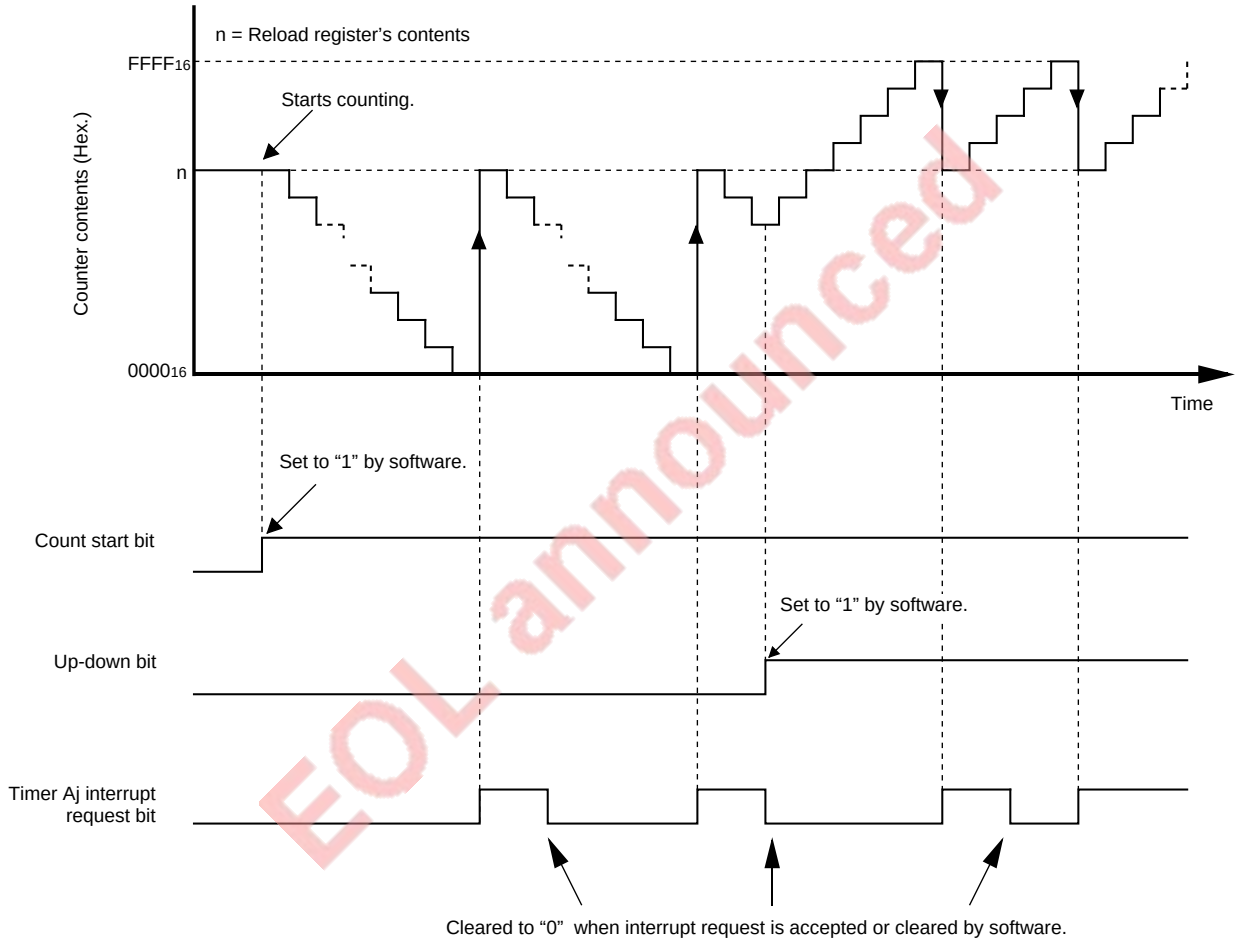
EOL announced

Fig. 8.4.3 Initial setting example for registers relevant to event counter mode (2)

8.4.2 Operation in event counter mode

① When the count start bit is set to “1,” the counter starts counting of the count source’s valid edges.
② When a counter underflow or overflow occurs, the reload register’s contents are reloaded, and counting continues.
③ The timer Aj interrupt request bit is set to “1” at the underflow or overflow in ②.
The interrupt request bit remains set to “1” until the interrupt request is accepted or the interrupt request bit is cleared to “0” by software.

Figure 8.4.4 shows an example of operation in the event counter mode.



EOL announced

Note: The above applies when the up-down bit’s contents are selected as the up-down switching factor (i.e., up-down switching factor select bit = “0”).

Fig. 8.4.4 Example of operation in event counter mode (without pulse output and two-phase pulse signal processing functions)

8.4 Event counter mode

8.4.3 Switching between countup and countdown

The up-down register (address 44_{16}) or the input signal from the TAj$_{OUT}$ pin is used to switch countup from and to countdown. This switching is performed by the up-down bit when the up-down switching factor select bit (bit 4 at addresses 58_{16} to $5A_{16}$) is "0," and by the input signal from the TAj$_{OUT}$ pin when the up-down switching factor select bit is "1."

When the switching between countup and countdown is set while counting is in progress, this switching is actually performed when the count source's next valid edge is input.

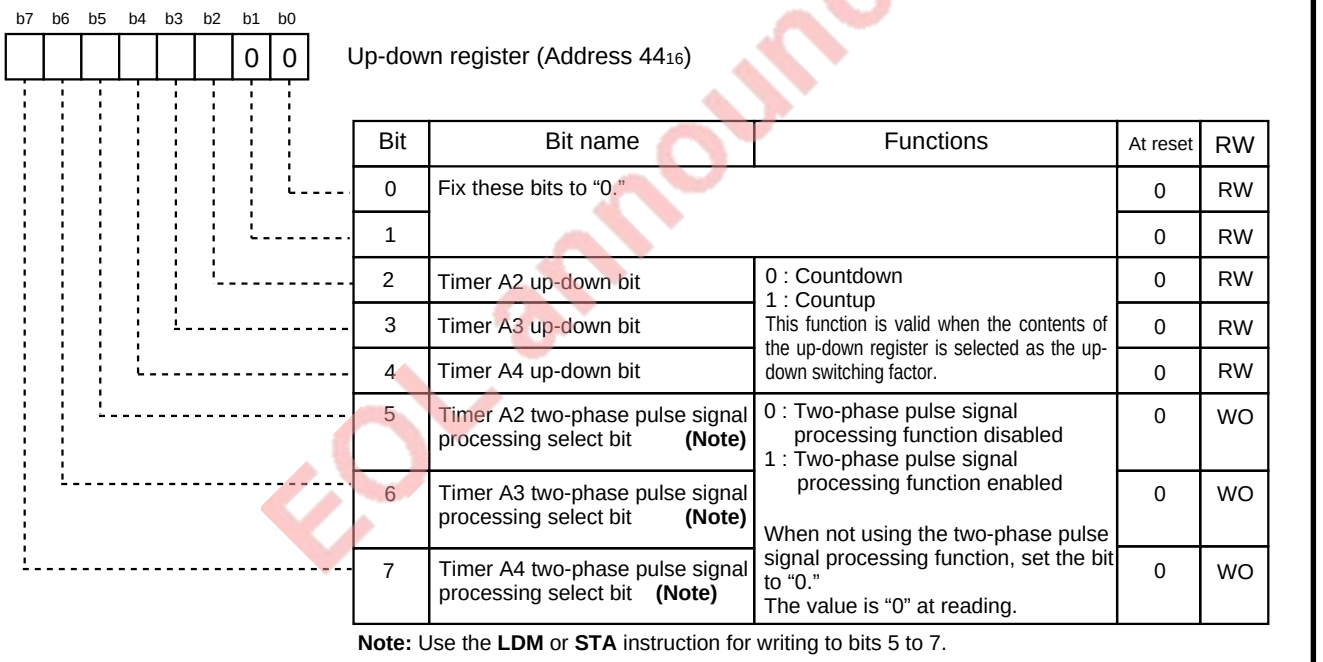
(1) Switching by up-down bit

Countdown is performed when the up-down bit is "0," and countup is performed when the up-down bit is "1." Figure 8.4.5 shows the structure of the up-down register.

(2) Switching by TAj$_{OUT}$ pin's input signal

Countdown is performed when the TAj$_{OUT}$ pin's input signal is at "L" level, and countup is performed when the TAj$_{OUT}$ pin's input signal is at "H" level.

When using the TAj$_{OUT}$ pin's input signal to switch countup from and to countdown, set the port P5 direction register's bit which corresponds to the TAj$_{OUT}$ pin for the input mode.



Bit	Bit name	Functions	At reset	RW
0	Fix these bits to "0."		0	RW
1			0	RW
2	Timer A2 up-down bit	0 : Countdown 1 : Countup This function is valid when the contents of the up-down register is selected as the up-down switching factor.	0	RW
3	Timer A3 up-down bit		0	RW
4	Timer A4 up-down bit		0	RW
5	Timer A2 two-phase pulse signal processing select bit **(Note)**	0 : Two-phase pulse signal processing function disabled 1 : Two-phase pulse signal processing function enabled When not using the two-phase pulse signal processing function, set the bit to "0." The value is "0" at reading.	0	WO
6	Timer A3 two-phase pulse signal processing select bit **(Note)**		0	WO
7	Timer A4 two-phase pulse signal processing select bit **(Note)**		0	WO

Note: Use the **LDM** or **STA** instruction for writing to bits 5 to 7.

Fig. 8.4.5 Structure of up-down register

EOL announced

8.4.4 Selectable functions

The following describes the selectable pulse output, and two-phase pulse signal processing functions.

(1) Pulse output function

The pulse output function is selected by setting the pulse output function select bit (bit 2 at addresses 58_{16} to $5A_{16}$) to "1." When this function is selected, the TAj$_{OUT}$ pin is forcibly set for the pulse output pin regardless of the corresponding bit of the port P5 direction register. The TAj$_{OUT}$ pin outputs pulses of which polarity is inverted each time a counter underflow or overflow occurs. (Refer to **"Figure 8.3.6."**)

When the count start bit (address 40_{16}) is "0" (count stopped), the TAj$_{OUT}$ pin outputs "L" level.

EOL announced

8.4 Event counter mode

(2) Two-phase pulse signal processing function

The two-phase pulse signal processing function is selected by setting the two-phase pulse signal processing select bits (bits 5 to 7 at address 44_{16}) to "1." (Refer to **"Figure 8.4.5.")** Figure 8.4.6 shows the timer Aj mode registers when the two-phase pulse signal processing function is selected. For timers with the two-phase pulse signal processing function selected, the timer counts two kinds of pulses of which phases differ by 90 degrees. There are two types of the two-phase pulse signal processing: normal processing and quadruple processing. In Timers A2 and A3, normal processing is performed; in timer A4, quadruple processing is performed.

For some bits of the port P5 direction register correspond to pins used for two-phase pulse input, set these bits for the input mode.

b7	b6	b5	b4	b3	b2	b1	b0
X	X	0	1	0	0	0	1

Timer A2 mode register (Address 58_{16})
Timer A3 mode register (Address 59_{16})
Timer A4 mode register (Address $5A_{16}$)

X : It may be either "0" or "1."

Fig. 8.4.6 Timer Aj mode registers when two-phase pulse signal processing function is selected

●Normal processing

Countup is performed at the rising edges input to the TAk_{IN} pin when the phase has the relationship that the TAk_{IN} pin's input signal level goes from "L" to "H" while the TAk_{OUT} (k = 2 and 3) pin's input signal is at "H" level.

Countdown is performed at the falling edges input to the TAk_{IN} pin when the phase has the relationship that the TAk_{IN} pin's input signal level goes from "H" to "L" while the TAk_{OUT} pin's input signal is at "H" level. (Refer to **"Figure 8.4.7."**)

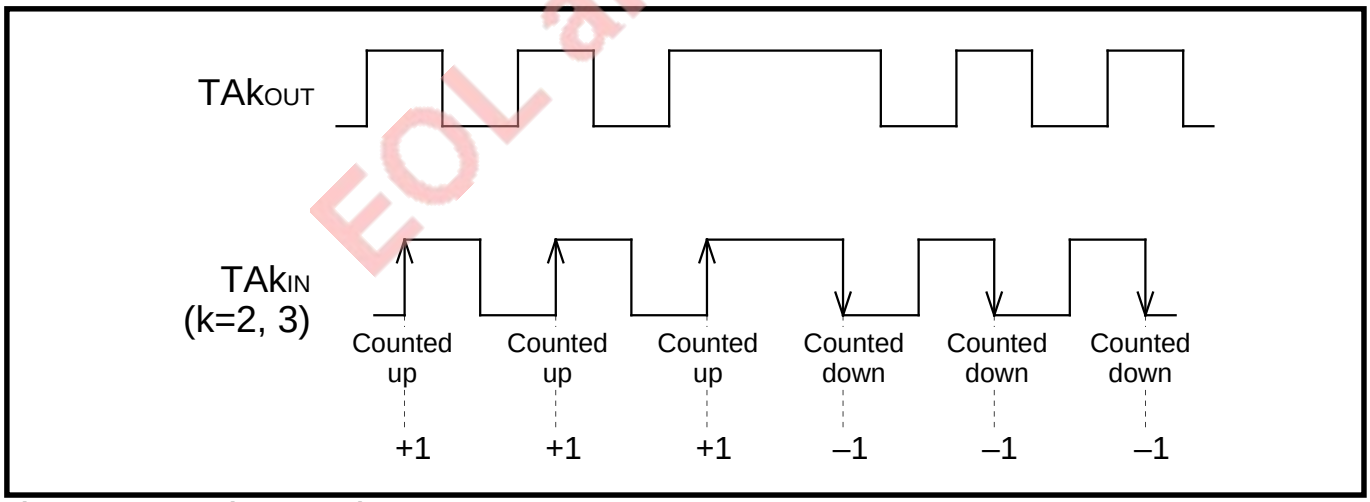


Fig. 8.4.7 Normal processing

EOL announced

●**Quadruple processing**

Countup is performed at all rising and falling edges input to the $TA4_{OUT}$ and $TA4_{IN}$ pins when the phase has the relationship that the $TA4_{IN}$ pin's input signal level goes from "L" to "H" while the $TA4_{OUT}$ pin's input signal is at "H" level.

Countdown is performed at all rising and falling edges input to the $TA4_{OUT}$ and $TA4_{IN}$ pins when the phase has the relationship that the $TA4_{IN}$ pin's input signal level goes from "H" to "L" while the $TA4_{OUT}$ pin's input signal is at "H" level. (Refer to **"Figure 8.4.8."**)

Table 8.4.3 lists the relationship between the input signals to the $TA4_{OUT}$ and $TA4_{IN}$ pins and count operation when the quadruple processing is selected.

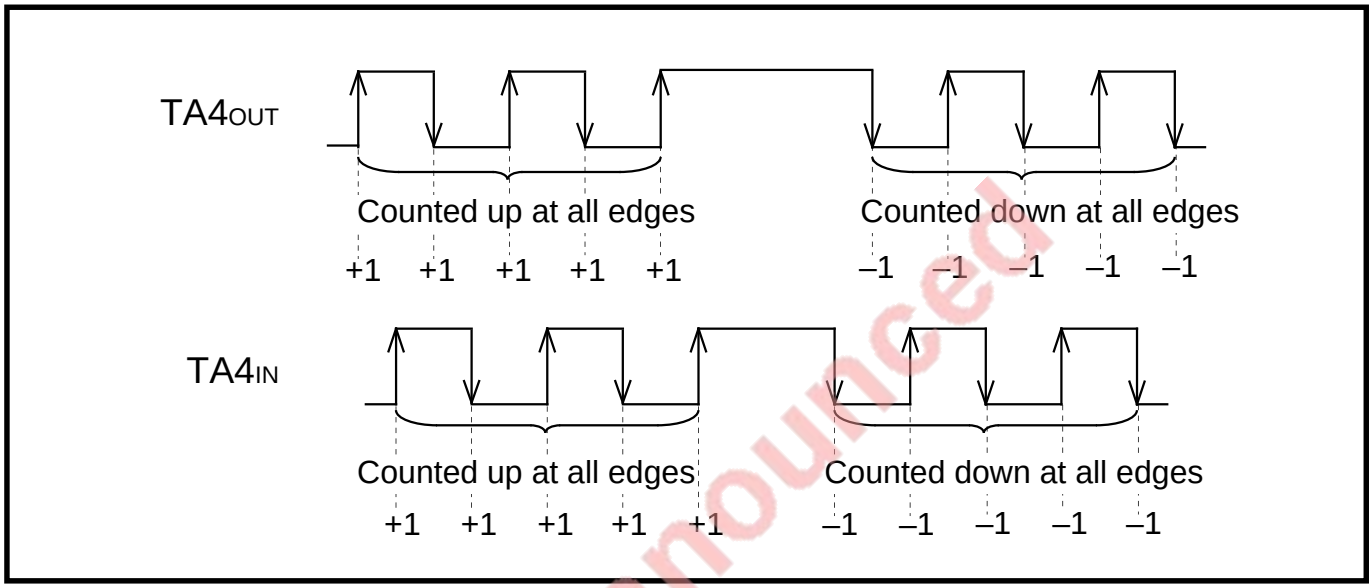


Fig. 8.4.8 Quadruple processing

Table 8.4.3 Relationship between input signals to $TA4_{OUT}$ and $TA4_{IN}$ pins and count operation when quadruple processing is selected

	Input signal to $TA4_{OUT}$ pin	Input signal to $TA4_{IN}$ pin
Up-count	"H" level	Rising edge
	"L" level	Falling edge
	Rising edge	"L" level
	Falling edge	"H" level
Down-count	"H" level	Falling edge
	"L" level	Rising edge
	Rising edge	"H" level
	Falling edge	"L" level

EOL announced

8.4 Event counter mode

[Precautions for event counter mode]

1. While counting is in progress, by reading the timer Aj register, the counter value can be read out at any timing. However, if the timer Aj register is read at the reload timing shown in Figure 8.4.9, the value "$FFFF_{16}$" (at an underflow) or "0000_{16}" (at an overflow) is read out. If reading is performed in the period from when a value is set into the timer Aj register with the counter stopped until the counter starts counting, the set value is correctly read out.

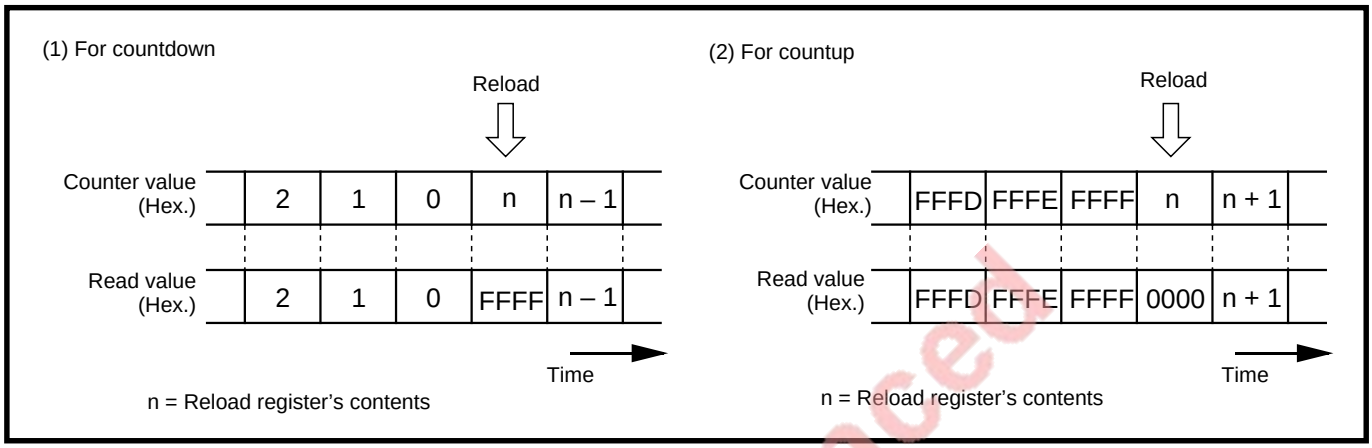


Fig. 8.4.9 Reading timer Aj register

2. The TAj_{OUT} pin is used for all functions listed below. Accordingly, only one of these functions can be selected for each timer.
   - •Switching between countup and countdown by TAj_{OUT} pin's input signal
   - •Pulse output function
   - •Two-phase pulse signal processing function

EOL announced

8.5 One-shot pulse mode

In this mode, the timer outputs a pulse which has an arbitrary width once. (Refer to **"Table 8.5.1."**) Timers A2 to A4 can be used in this mode. When a trigger occurs, the timer outputs "H" level from the TAj_{OUT} pin for an arbitrary time. Figure 8.5.1 shows the structures of the timer Aj mode register and timer Aj register in the one-shot pulse mode.

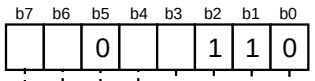
Table 8.5.1 Specifications of one-shot pulse mode

Item	Specifications
Count source	f_2, f_{16}, f_{64}, or f_{512}
Count operation	● Countdown ● When the counter value becomes "0000_{16}," reload register's contents are reloaded, and counting stops. ● If a trigger occurs during counting, reload register's contents are reloaded, and counting continues.
Output pulse width ("H")	$\frac{n}{f_i}$ [S] n : Timer Aj register's set value
Count start condition	● When a trigger occurs. (**Note**) ● Internal or external trigger can be selected by software.
Count stop condition	● When the counter value becomes "0000_{16}" ● When the count start bit is cleared to "0"
Interrupt request occurrence timing	When counting stops.
TAj_{IN} pin's function	Programmable I/O port or trigger input
TAj_{OUT} pin's function	One-shot pulse output
Read from timer Aj register	An undefined value is read out.
Write to timer Aj register	● While counting is stopped When a value is written to the timer Aj register, it is written to both of the reload register and counter. ● While counting is in progress When a value is written to the timer Aj register, it is written only to the reload register. (Transferred to the counter at the next reload time.)

Note: The trigger is generated with the count start bit = "1."

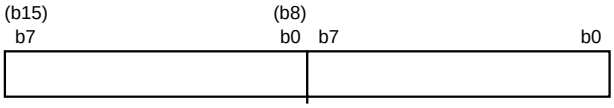
EOL announced

8.5 One-shot pulse mode



Timer Aj mode register (j = 2 to 4) (Addresses 58_{16} to $5A_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 1 0 : One-shot pulse mode	0	RW
1			0	RW
2	Fix this bit to "1" in one-shot pulse mode.		0	RW
3	Trigger select bits	b4 b3 0 0 : Writing "1" to one-shot start register 0 1 : (TAj_{IN} pin functions as a programmable I/O port.) 1 0 : Falling edge of TAj_{IN} pin's input signal 1 1 : Rising edge of TAj_{IN} pin's input signal	0	RW
4			0	RW
5	Fix this bit to "0" in one-shot pulse mode.		0	RW
6	Count source select bits	b7 b6 0 0 : f_2 0 1 : f_{16} 1 0 : f_{64} 1 1 : f_{512}	0	RW
7			0	RW



Timer A2 register (Addresses $4B_{16}$, $4A_{16}$)
Timer A3 register (Addresses $4D_{16}$, $4C_{16}$)
Timer A4 register (Addresses $4F_{16}$, $4E_{16}$)

Bit	Functions	At reset	RW
15 to 0	These bits can be set to "0001_{16}" to "$FFFF_{16}$." Assuming that the set value = n, the "H" level width of the one-shot pulse output from the TAj_{OUT} pin is expressed as follows : $\frac{n}{f_i}$	Undefined	WO

f_i: Frequency of count source (f_2, f_{16}, f_{64}, or f_{512})

Note: Use the **LDM** or **STA** instruction for writing to this register. Read from or write to this register in a unit of 16 bits.

EOL announced

Fig. 8.5.1 Structures of timer Aj mode register and timer Aj register in one-shot pulse mode

8.5.1 Setting for one-shot pulse mode

Figures 8.5.2 and 8.5.3 show an initial setting example for registers relevant to the one-shot pulse mode. Note that when using interrupts, set up to enable the interrupts. For details, refer to **"CHAPTER 7. INTERRUPTS."**

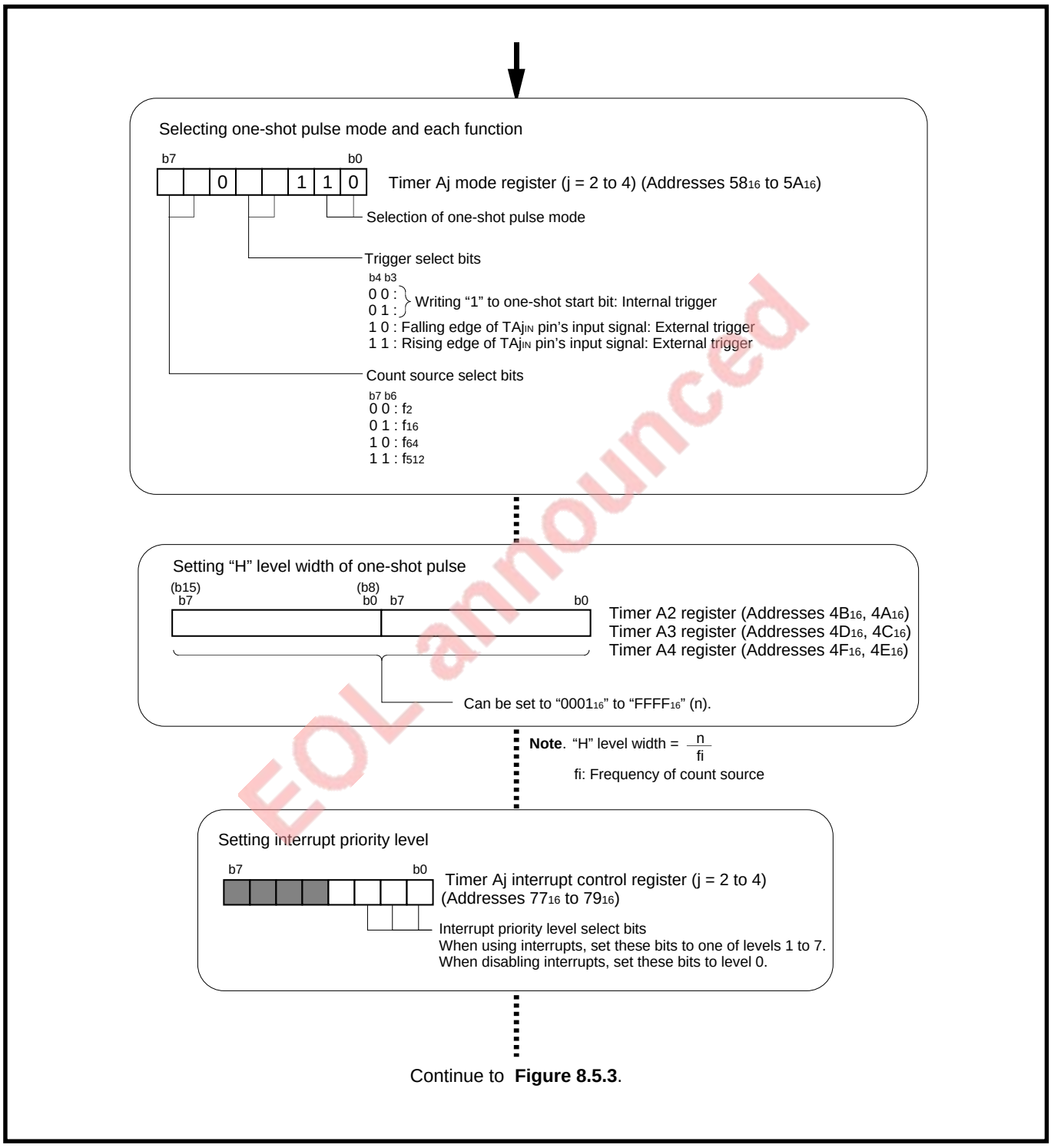


Fig. 8.5.2 Initial setting example for registers relevant to one-shot pulse mode (1)

EOL announced

8.5 One-shot pulse mode

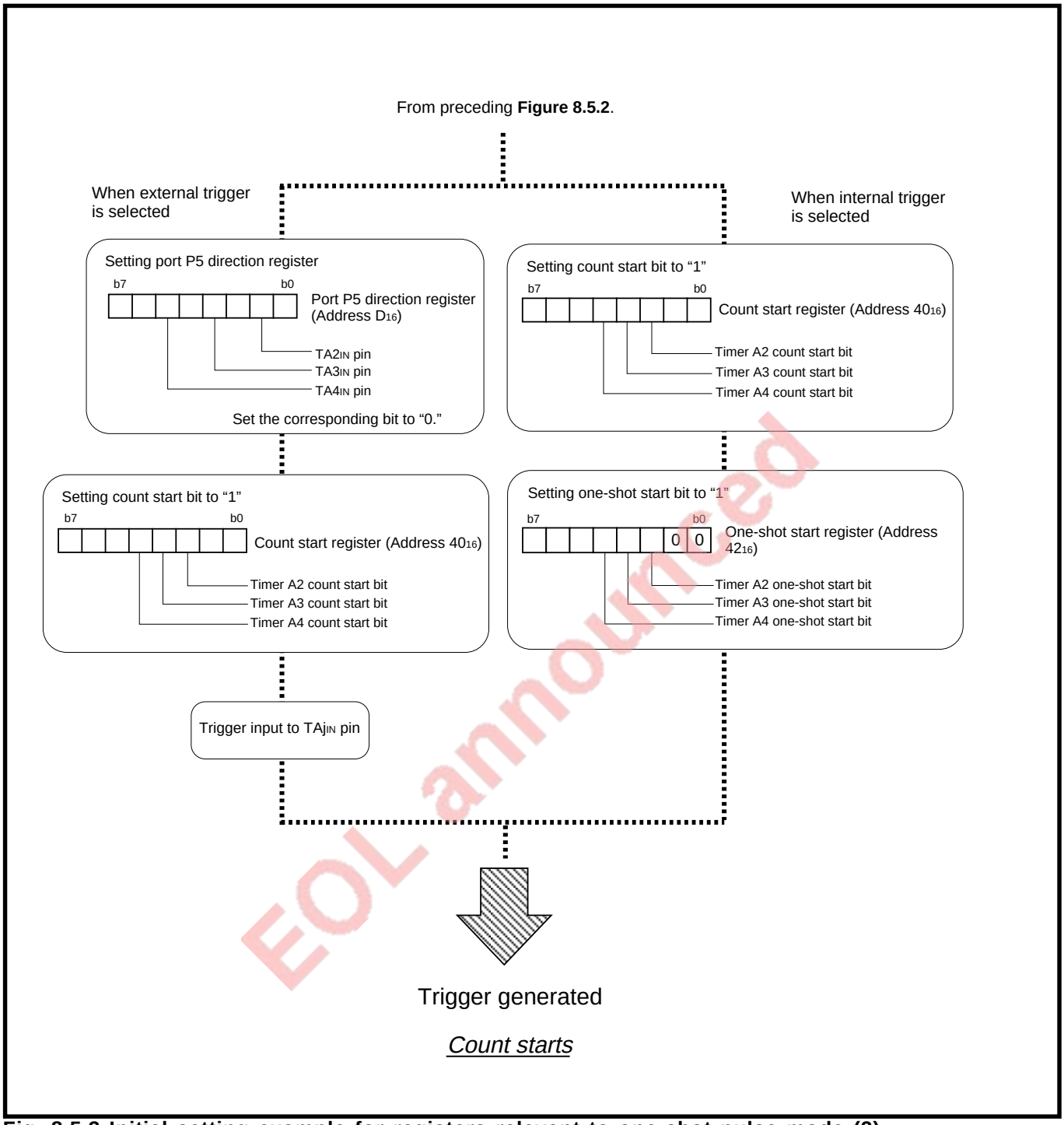


Fig. 8.5.3 Initial setting example for registers relevant to one-shot pulse mode (2)

8.5.2 Count source

In the one-shot pulse mode, the count source select bits (bits 6 and 7 at addresses 58_{16} to $5A_{16}$) select the count source. Table 8.5.2 lists the count source frequency.

Table 8.5.2 Count source frequency

Count source select bits		Count source	Count source frequency		
b7	b6		$f(X_{IN})$ = 8 MHz	$f(X_{IN})$ = 16 MHz	$f(X_{IN})$ = 25 MHz
0	0	f_2	4 MHz	8 MHz	12.5 MHz
0	1	f_{16}	500 kHz	1 MHz	1.5625 MHz
1	0	f_{64}	125 kHz	250 kHz	390.625 kHz
1	1	f_{512}	15625 Hz	31250 Hz	48.8281 kHz

EOL announced

8.5 One-shot pulse mode

8.5.3 Trigger

The counter is enabled for counting when the count start bit (address 40_{16}) is set to "1." The counter starts counting when a trigger is generated after counting has been enabled. An internal or external trigger can be selected as that trigger.

An internal trigger is selected when the trigger select bits (bits 4 and 3 at addresses 58_{16} to $5A_{16}$) are "00_2" or "01_2"; an external trigger is selected when the bits are "10_2" or "11_2."

If a trigger is generated during counting, the reload register's contents are reloaded and the counter continues counting. If generating a trigger during counting, make sure that a certain time which is equivalent to one cycle of the timer's count source or more has passed between the previously generated trigger and a new trigger.

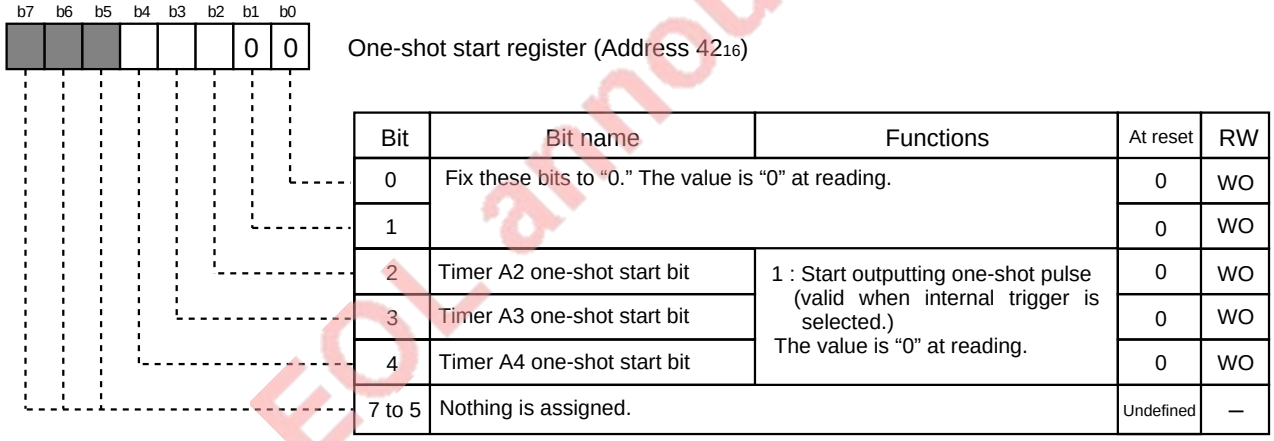
(1) When selecting internal trigger

A trigger is generated when writing "1" to the one-shot start bit (bits 2 to 4 at address 42_{16}). Figure 8.5.4 shows the structure of the one-shot start register.

(2) When selecting external trigger

A trigger is generated at the falling edge of the TAj_{IN} pin's input signal when bit 3 at addresses 58_{16} to $5A_{16}$ is "0," or at its rising edge when bit 3 is "1."

When using an external trigger, set the port P5 direction registers' bits which correspond to the TAj_{IN} pins for the input mode.



Bit	Bit name	Functions	At reset	RW
0	Fix these bits to "0." The value is "0" at reading.		0	WO
1			0	WO
2	Timer A2 one-shot start bit	1 : Start outputting one-shot pulse (valid when internal trigger is selected.) The value is "0" at reading.	0	WO
3	Timer A3 one-shot start bit		0	WO
4	Timer A4 one-shot start bit		0	WO
7 to 5	Nothing is assigned.		Undefined	–

EOL announced

Fig. 8.5.4 Structure of one-shot start register

8.5.4 Operation in one-shot pulse mode

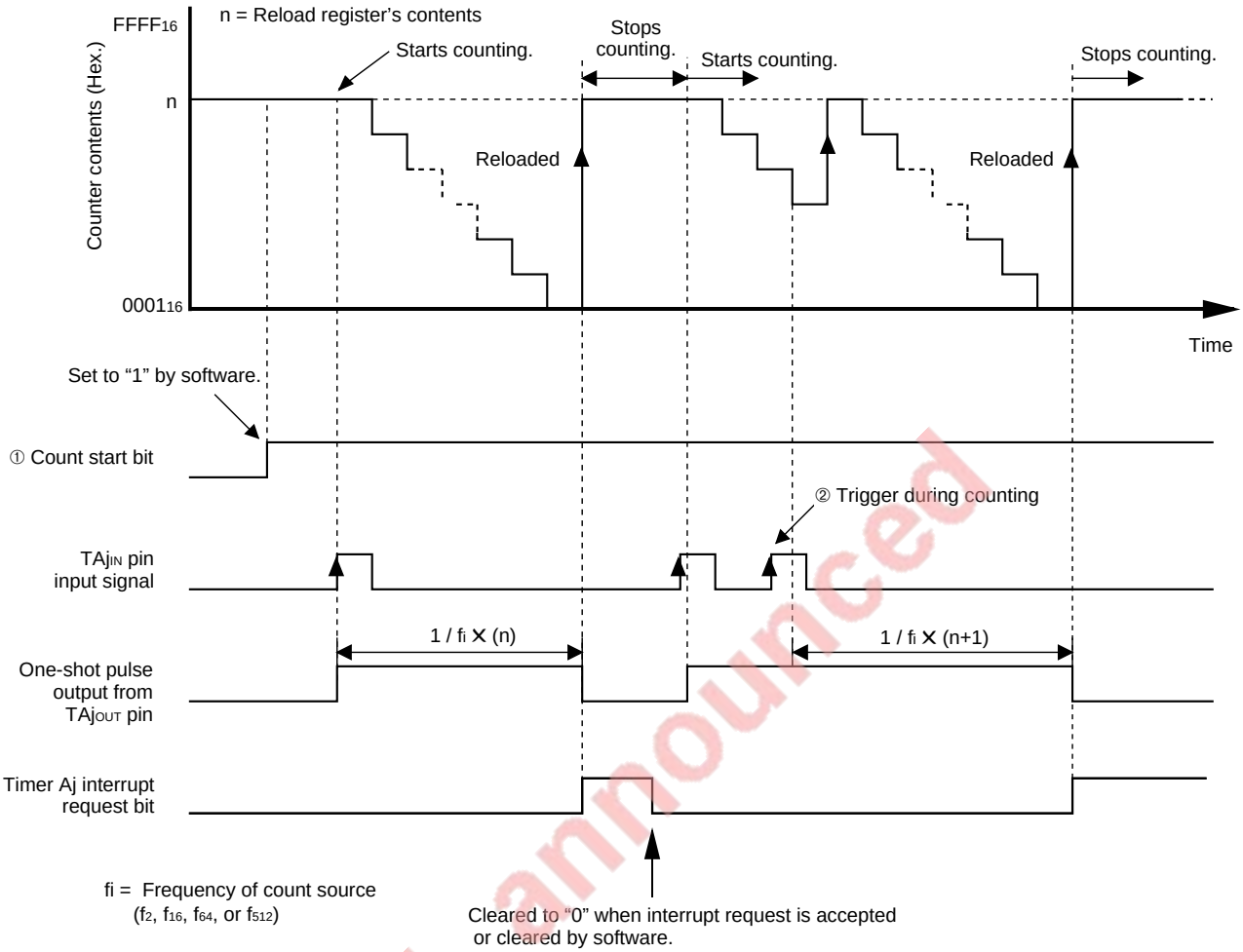
① When the one-shot pulse mode is selected with the operating mode select bits, the TAj$_{OUT}$ pin outputs "L" level.
② When the count start bit is set to "1," the counter is enabled for counting. After that, counting starts when a trigger is generated.
③ When the counter starts counting, the TAj$_{OUT}$ pin outputs "H" level.
④ When the counter value becomes "0000_{16}," the output from the TAj$_{OUT}$ pin becomes "L" level. Additionally, the reload register's contents are reloaded and the counter stops counting there.
⑤ Simultaneously with ④, the timer Aj interrupt request bit is set to "1."
This interrupt request bit remains set to "1" until the interrupt request is accepted or the interrupt request bit is cleared to "0" by software.

Figure 8.5.5 shows an example of operation in the one-shot pulse mode.
When a trigger is generated after ④ above, the counter and TAj$_{OUT}$ pin perform the same operations beginning from ② again. Furthermore, if a trigger is generated during counting, the counter performs countdown once after this new trigger is generated, and it continues counting with the reload register's contents reloaded. If generating a trigger during counting, make sure that a certain time which is equivalent to one cycle of the timer's count source or more has passed between the previously generated trigger and a new trigger.

The one-shot pulse output from the TAj$_{OUT}$ pin can be disabled by clearing the timer Aj mode register's bit 2 to "0." Accordingly, Timer Aj can be also used as an internal one-shot timer that does not perform the pulse output. In this case, the TAj$_{OUT}$ pin functions as a programmable I/O port.

EOL announced

8.5 One-shot pulse mode



① When the count start bit = "0" (counting stopped), the TAjOUT pin outputs "L" level.
② When a trigger is generated during counting, the counter counts the count source (n + 1) times after a new trigger is generated.

Note: The above applies when an external trigger (rising of TAjIN pin's input signal) is selected.

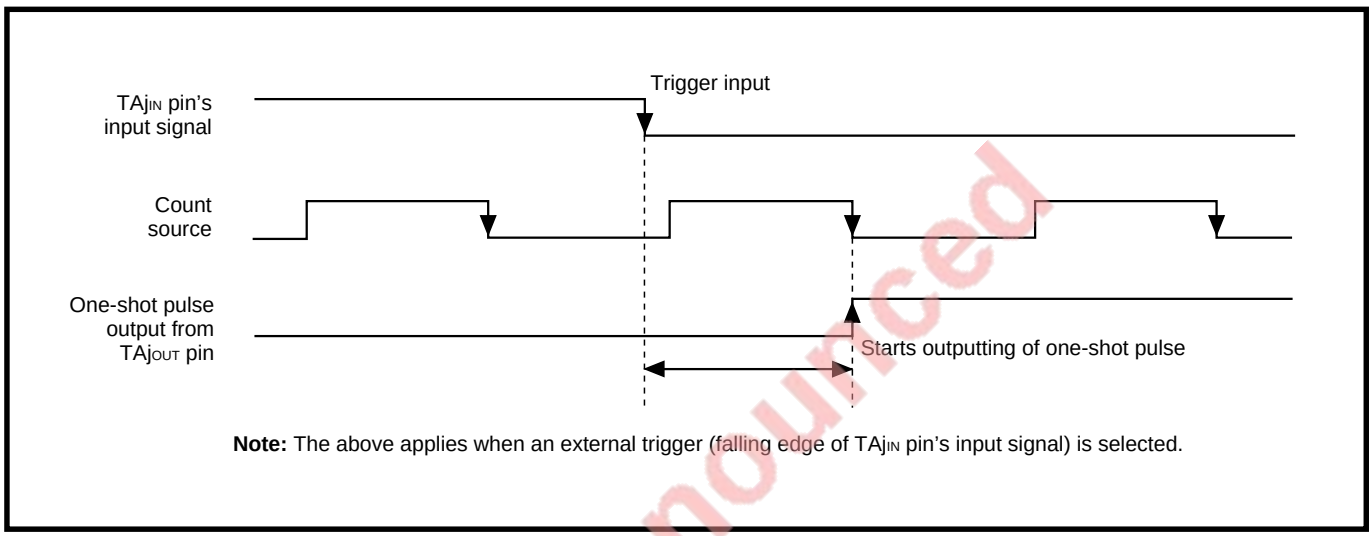
EOL announced

Fig. 8.5.5 Example of operation in one-shot pulse mode (selecting external trigger)

[Precautions for one-shot pulse mode]

1. If the count start bit is cleared to "0" during counting, the counter becomes as follows:
   - •The counter stops counting, and the reload register's contents are reloaded into the counter.
   - •The TAj_{OUT} pin's output level becomes "L."
   - •The timer Aj interrupt request bit is set to "1."

2. A one-shot pulse is output synchronously with an internally generated count source. Accordingly, when selecting an external trigger, there will be a delay equivalent to one cycle of the count source at maximum from when a trigger is input to the TAj_{IN} pin until a one-shot pulse is output.



Note: The above applies when an external trigger (falling edge of TAj_{IN} pin's input signal) is selected.

Fig. 8.5.6 Output delay in one-shot pulse output

3. When the timer's operating mode is set by one of the following procedures, the timer Aj interrupt request bit is set to "1."
   - ●When the one-shot pulse mode is selected after reset
   - ●When the operating mode is switched from the timer mode to the one-shot pulse mode
   - ●When the operating mode is switched from the event counter mode to the one-shot pulse mode

   Accordingly, when using the timer Aj interrupt (interrupt request bit), be sure to clear the timer Aj interrupt request bit to "0" after the above setting.

4. Don not set "0000_{16}" to the timer Aj register.

EOL announced

8.6 Pulse width modulation (PWM) mode

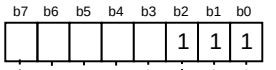
In this mode, the timer continuously outputs pulses which have an arbitrary width. (Refer to **"Table 8.6.1."**) Timers A2 to A4 can be used in this mode. Figure 8.6.1 shows the structures of the timer Aj mode registers and timer Aj registers in the PWM mode.

Table 8.6.1 Specifications of PWM mode

Item	Specifications
Count source	f_2, f_{16}, f_{64}, or f_{512}
Count operation	● Countdown (operating as an 8-bit or 16-bit pulse width modulator) ● Reload register's contents are reloaded at rising edge of PWM pulse, and counting continues. ● A trigger generated during counting does not affect the counting.
PWM period/"H" level width	<16-bit pulse width modulator> Period = $\frac{(2^{16}-1)}{fi}$ [s] "H" level width = $\frac{n}{fi}$ [s] n : Timer Aj register's set value <8-bit pulse width modulator> Period = $\frac{(m+1)(2^8-1)}{fi}$ [s] "H" level width = $\frac{n(m+1)}{fi}$ [s] m : Timer Aj register low-order 8 bits' set value n : Timer Aj register high-order 8 bits' set value
Count start condition	● When a trigger is generated. **(Note)** ● Internal or external trigger can be selected by software.
Count stop condition	When the count start bit is cleared to "0."
Interrupt request occurrence timing	At falling edge of PWM pulse
TAj_{IN} pin's function	Programmable I/O port or trigger input
TAj_{OUT} pin's function	PWM pulse output
Read from timer Aj register	An undefined value is read out.
Write to timer Aj register	● While counting is stopped When a value is written to the timer Aj register, it is written to both of the reload register and counter. ● While counting is in progress When a value is written to the timer Aj register, it is written only to the reload register. (Transferred to the counter at the next reload time.)

Note: The trigger is generated with the count start bit = "1."

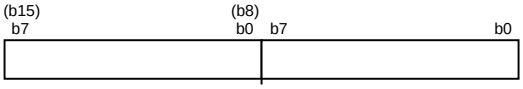
EOL announced



Timer Aj mode register (j = 2 to 4) (Addresses 58_{16} to $5A_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 1 1 : PWM mode	0	RW
1			0	RW
2	Fix this bit to "1" in PWM mode.		0	RW
3	Trigger select bits	b4 b3 0 0 : } Writing "1" to count start register 0 1 : } (TAj_{IN} pin functions as a programmable I/O port.) 1 0 : Falling edge of TAj_{IN} pin's input signal 1 1 : Rising edge of TAj_{IN} pin's input signal	0	RW
4			0	RW
5	16/8-bit PWM mode select bit	0 : 16-bit pulse width modulator 1 : 8-bit pulse width modulator	0	RW
6	Count source select bits	b7 b6 0 0 : f_2 0 1 : f_{16} 1 0 : f_{64} 1 1 : f_{512}	0	RW
7			0	RW

<When operating as a 16-bit pulse width modulator>

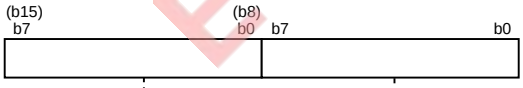


Timer A2 register (Addresses $4B_{16}$, $4A_{16}$)
Timer A3 register (Addresses $4D_{16}$, $4C_{16}$)
Timer A4 register (Addresses $4F_{16}$, $4E_{16}$)

Bit	Functions	At reset	RW
15 to 0	These bits can be set to "0000_{16}" to "$FFFE_{16}$." Assuming that the set value = n, the "H" level width of the PWM pulse output from the TAj_{OUT} pin is expressed as follows: $\frac{n}{f_i}$ (PWM pulse period = $\frac{n^{16}-1}{f_i}$)	Undefined	WO

f_i: Frequency of count source (f_2, f_{16}, f_{64}, or f_{512})
Note: Use the **LDM** or **STA** instruction for writing to this register. Read from or write to this register in a unit of 16 bits.

<When operating as an 8-bit pulse width modulator>



Timer A2 register (Addresses $4B_{16}$, $4A_{16}$)
Timer A3 register (Addresses $4D_{16}$, $4C_{16}$)
Timer A4 register (Addresses $4F_{16}$, $4E_{16}$)

Bit	Functions	At reset	RW
7 to 0	These bits can be set to "00_{16}" to "FF_{16}." Assuming that the set value = m, PWM pulse's period output from the TAj_{OUT} pin is expressed as follows: $\frac{(m+1)(2^8-1)}{f_i}$	Undefined	WO
15 to 8	These bits can be set to "00_{16}" to "FE_{16}." Assuming that the set value = n, the "H" level width of the PWM pulse output from the TAj_{OUT} pin is expressed as follows: $\frac{n(m+1)}{f_i}$	Undefined	WO

f_i: Frequency of count source (f_2, f_{16}, f_{64}, or f_{512})
Note: Use the **LDM** or **STA** instruction for writing to this register. Read from or write to this register in a unit of 16 bits.

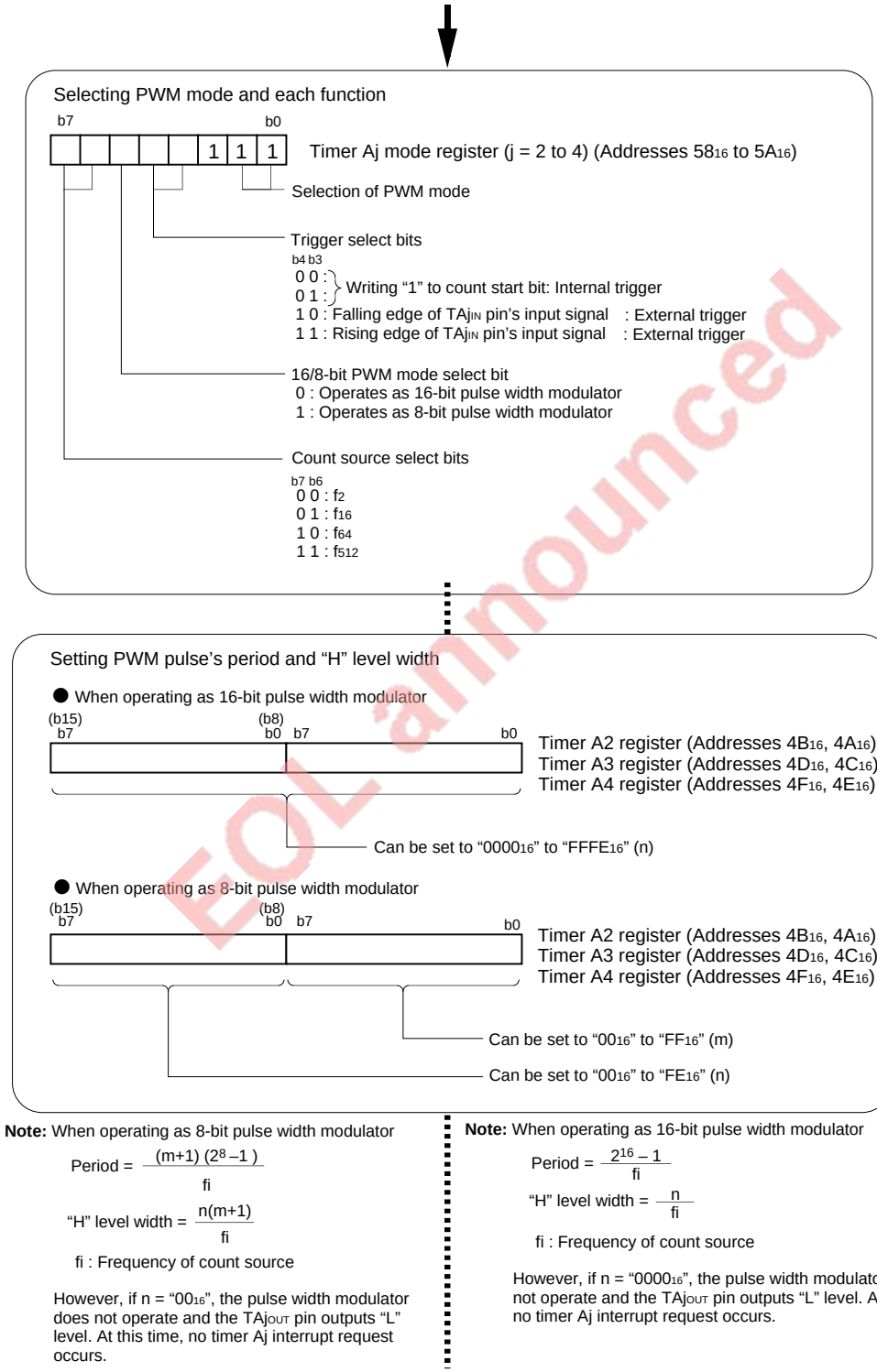
EOL announced

Fig. 8.6.1 Structures of timer Aj mode registers and timer Aj registers in PWM mode

8.6 Pulse width modulation (PWM) mode

8.6.1 Setting for PWM mode

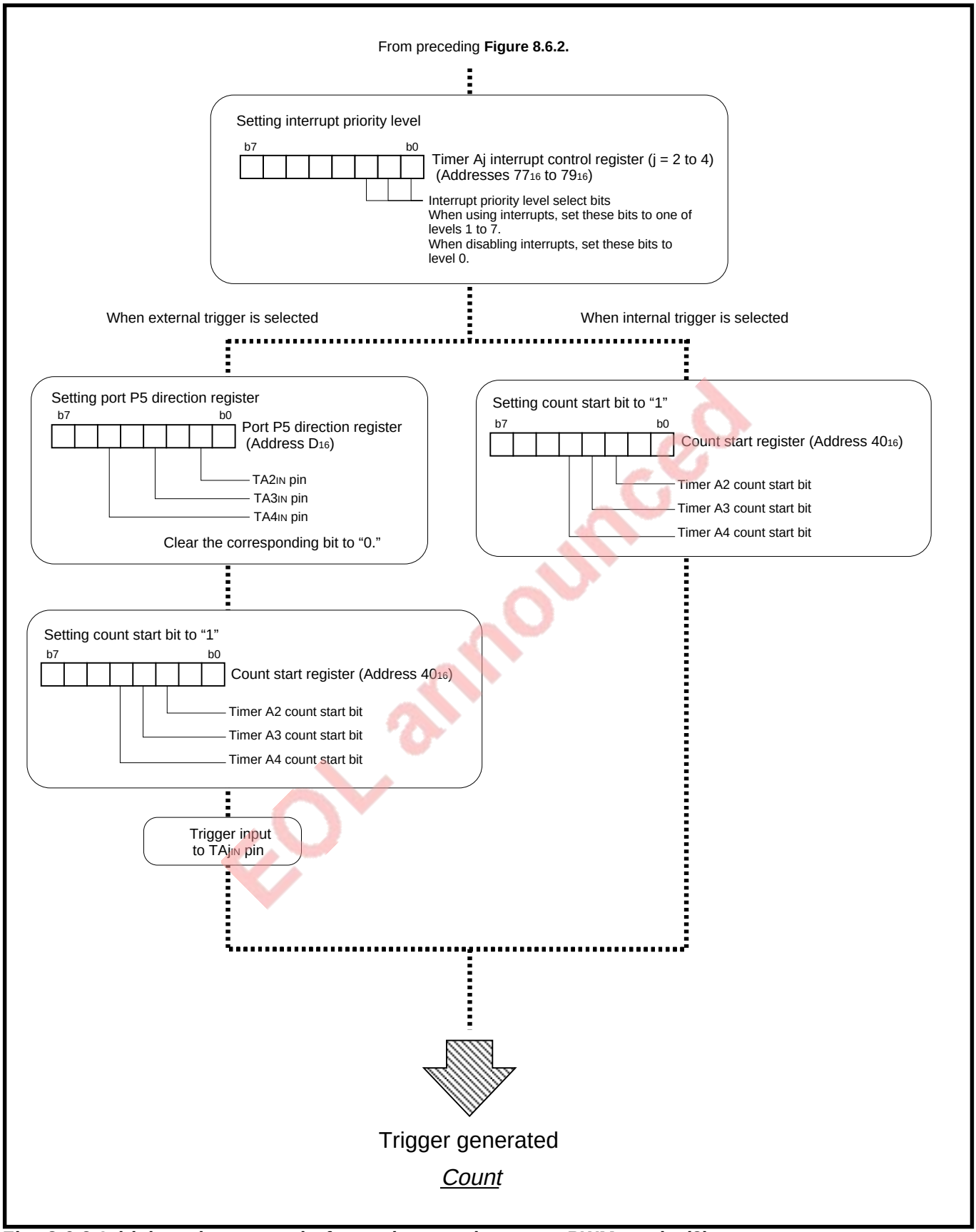
Figures 8.6.2 and 8.6.3 show an initial setting example for registers relevant to the PWM mode.
Note that when using interrupts, set up to enable the interrupts. For details, refer to **"CHAPTER 7. INTERRUPTS."**



EOL announced

Continue to **Figure 8.6.3**.

Fig. 8.6.2 Initial setting example for registers relevant to PWM mode (1)



EOL announced

Fig. 8.6.3 Initial setting example for registers relevant to PWM mode (2)

8.6 Pulse width modulation (PWM) mode

8.6.2 Count source

In the PWM mode, the count source select bits (bits 6 and 7 at addresses 58_{16} to $5A_{16}$) select the count source. Table 8.6.2 lists the count source frequency.

Table 8.6.2 Count source frequency

Count source select bits		Count source	Count source frequency		
b7	b6		$f(X_{IN})$ = 8 MHz	$f(X_{IN})$ = 16 MHz	$f(X_{IN})$ = 25 MHz
0	0	f_2	4 MHz	8 MHz	12.5 MHz
0	1	f_{16}	500 kHz	1 MHz	1.5625 MHz
1	0	f_{64}	125 kHz	250 kHz	390.625 kHz
1	1	f_{512}	15625 Hz	31250 Hz	48.8281 kHz

8.6.3 Trigger

When a trigger is generated, the TAj_{OUT} pin starts outputting PWM pulses. An internal or an external trigger can be selected as that trigger.

An internal trigger is selected when the trigger select bits (bits 4 and 3 at addresses 58_{16} to $5A_{16}$) are "00_2" or "01_2"; an external trigger is selected when the bits are "10_2" or "11_2."

A trigger generated during outputting of PWM pulses is invalid and it does not affect the pulse output operation.

(1) When selecting internal trigger

A trigger is generated when "1" is written to the count start bit (address 40_{16}).

(2) When selecting external trigger

A trigger is generated at the falling edge of the TAj_{IN} pin's input signal when bit 3 at addresses 58_{16} to $5A_{16}$ is "0," or at its rising edge when bit 3 is "1." However, the trigger input is accepted only when the count start bit is "1."

When using an external trigger, set the port P5 direction registers' bits which correspond to the TAj_{IN} pins for the input mode.

EOL announced

8.6.4 Operation in PWM mode

① When the PWM mode is selected with the operating mode select bits, the TAj$_{OUT}$ pin outputs "L" level.
② When a trigger is generated, the counter (pulse width modulator) starts counting and the TAj$_{OUT}$ pin outputs a PWM pulse (**Notes 1 and 2**).
③ The timer Aj interrupt request bit is set to "1" each time the PWM pulse level goes from "H" to "L." The interrupt request bit remains set to "1" until the interrupt request is accepted or the interrupt request bit is cleared to "0" by software.
④ Each time a PWM pulse has been output for one period, the reload register's contents are reloaded and the counter continues counting.

The following explains operations of the pulse width modulator.

(1) 16-bit pulse width modulator

When the 16/8-bit PWM mode select bit is set to "0," the counter operates as a 16-bit pulse width modulator. Figures 8.6.4 and 8.6.5 show operation examples of the 16-bit pulse width modulator.

(2) 8-bit pulse width modulator

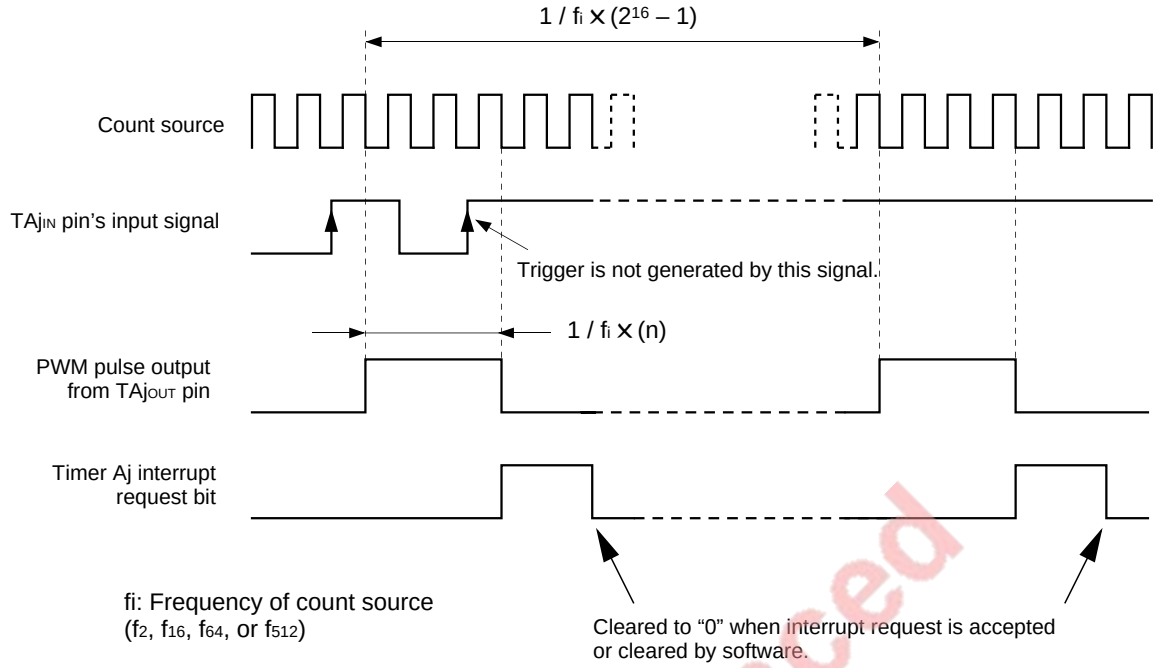
When the 16/8-bit PWM mode select bit is set to "1," the counter is divided into 8-bit halves. Then, the high-order 8 bits operate as an 8-bit pulse width modulator, and the low-order 8 bits operate as an 8-bit prescaler. Figures 8.6.6 and 8.6.7 show operation examples of the 8-bit pulse width modulator.

Notes 1: If a value "0000_{16}" is set into the timer Aj register when the counter operates as a 16-bit pulse width modulator, the pulse width modulator does not operate and the output from the TAj$_{OUT}$ pin remains "L" level. The timer Aj interrupt request does not occur. Similarly, if a value "00_{16}" is set into the high-order 8 bits of the timer Aj register when the counter operates as an 8-bit pulse width modulator, the same is performed.

2: When the counter operates as an 8-bit pulse width modulator, after a trigger is generated, the TAj$_{OUT}$ pin outputs "L" level which has the same width as "H" level width of the PWM pulse, which was set. After that, the PWM pulse output starts from the TAj$_{OUT}$ pin.

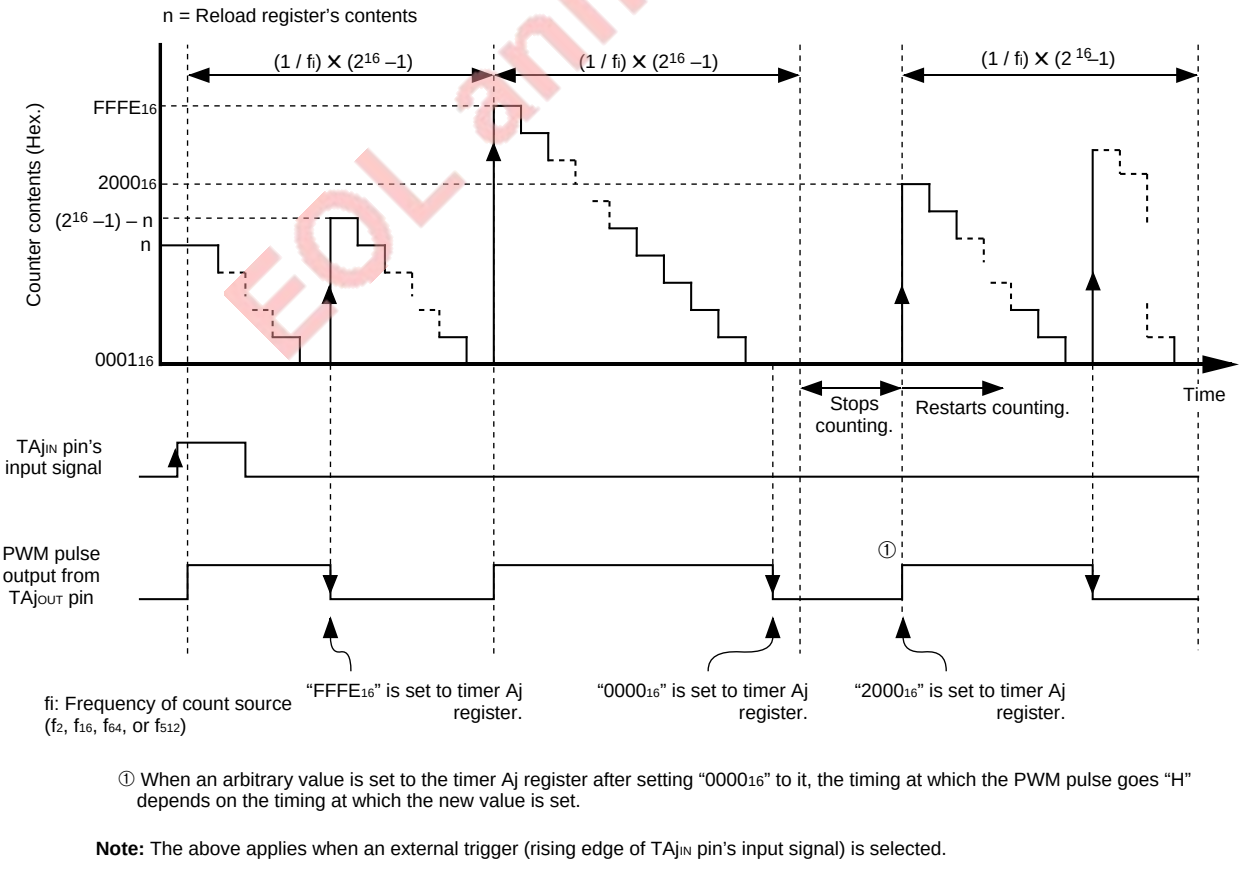
EOL announced

8.6 Pulse width modulation (PWM) mode



Note: The above applies when reload register (n) = "0003_{16}" and an external trigger (rising edge of TAjIN pin's input signal) is selected.

Fig. 8.6.4 Operation example of 16-bit pulse width modulator

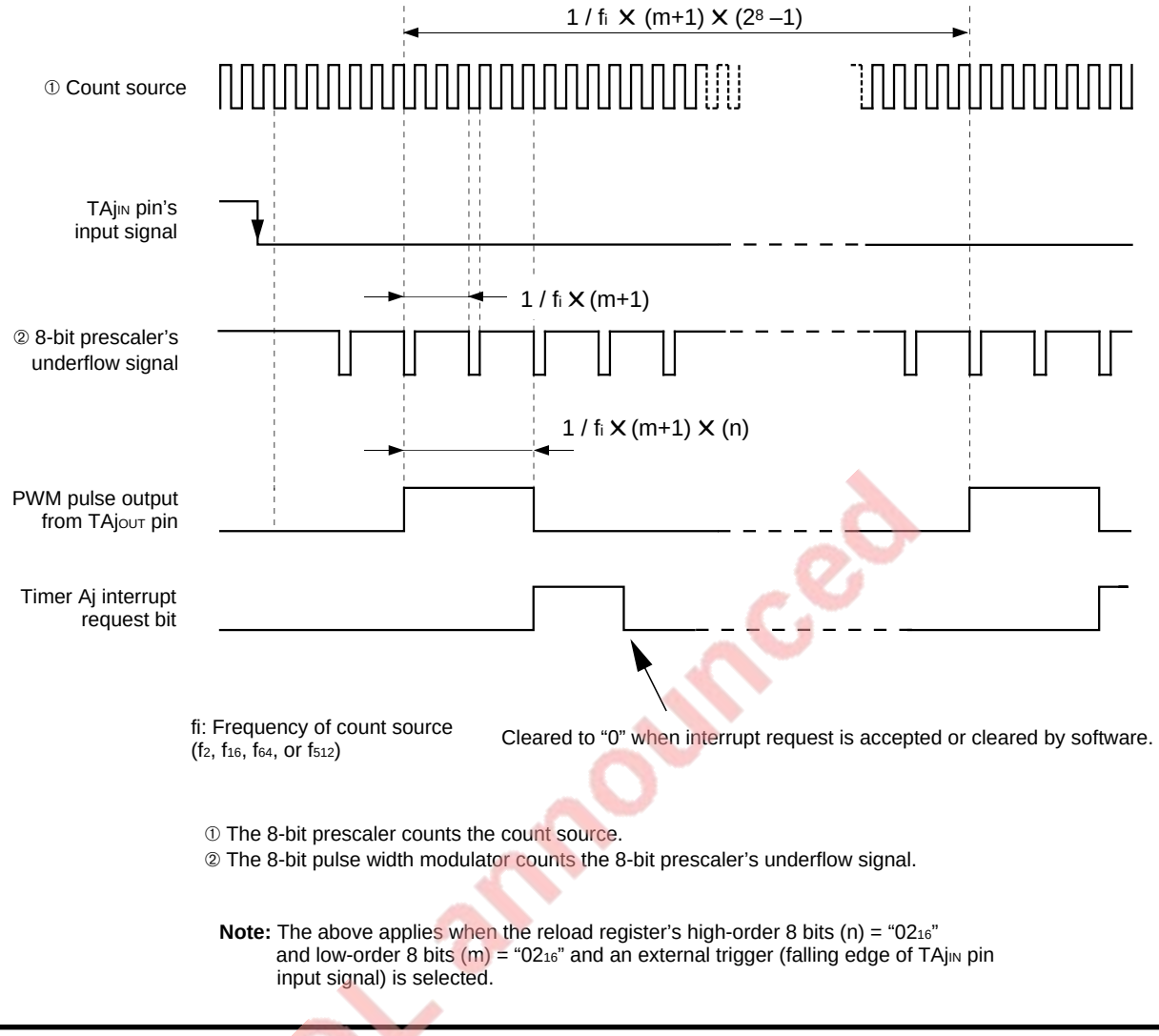


① When an arbitrary value is set to the timer Aj register after setting "0000_{16}" to it, the timing at which the PWM pulse goes "H" depends on the timing at which the new value is set.

Note: The above applies when an external trigger (rising edge of TAjIN pin's input signal) is selected.

Fig. 8.6.5 Operation example of 16-bit pulse width modulator (when counter value is updated during pulse output)

EOL announced



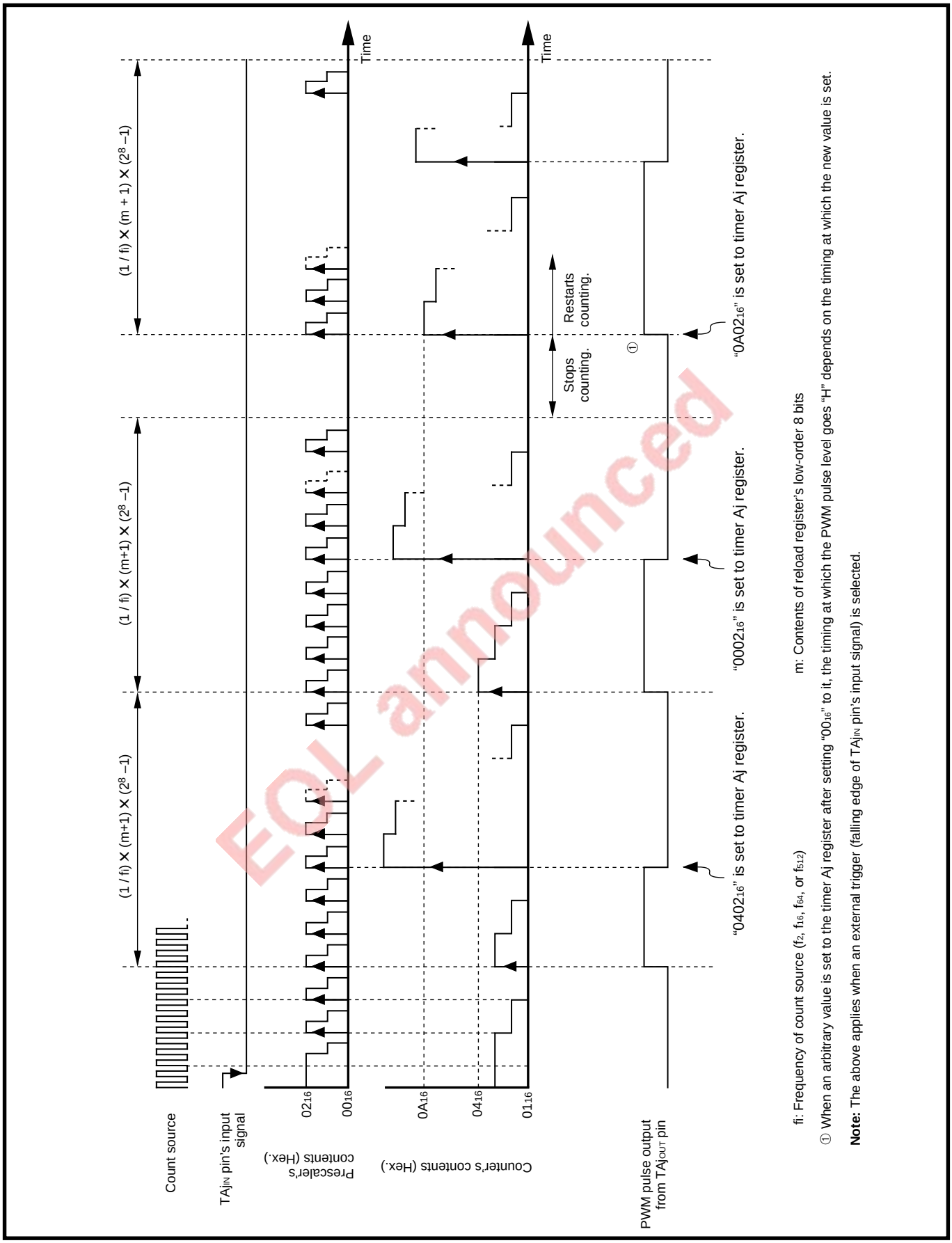
① The 8-bit prescaler counts the count source.
② The 8-bit pulse width modulator counts the 8-bit prescaler's underflow signal.

Note: The above applies when the reload register's high-order 8 bits (n) = "02_{16}" and low-order 8 bits (m) = "02_{16}" and an external trigger (falling edge of TAj_{IN} pin input signal) is selected.

Fig. 8.6.6 Operation example of 8-bit pulse width modulator

EOL announced

8.6 Pulse width modulation (PWM) mode



EOL announced

Fig. 8.6.7 Operation example of 8-bit pulse width modulator (when counter value is updated during pulse output)

[Precautions for PWM mode]

1. If the count start bit is cleared to “0” while outputting PWM pulses, the counter stops counting. When the TAj$_{OUT}$ pin was outputting “H” level at that time, the output level becomes “L” and the timer Aj interrupt request bit is set to “1.” When the TAj$_{OUT}$ pin was outputting “L” level, the output level does not change and a timer Aj interrupt request does not occur.

2. When the timer’s operating mode is set by one of the following procedures, the timer Aj interrupt request bit is set to “1.”
   - ●When the PWM mode is selected after reset
   - ●When the operating mode is switched from the timer mode to the PWM mode
   - ●When the operating mode is switched from the event counter mode to the PWM mode

   Accordingly, when using the timer Aj interrupt (interrupt request bit), be sure to clear the timer Aj interrupt request bit to “0” after the above setting.

EOL announced

MEMORANDUM

EOL announced

CHAPTER 9
TIMER B

9.1 Overview
9.2 Block description
9.3 Timer mode
[Precautions for timer mode]
9.4 Event counter mode
[Precautions for event counter mode]
9.5 Pulse period/Pulse width measurement mode
[Precautions for pulse period/pulse width measurement (PWM) mode]

EOL announced

9.1 Overview

Timer B consists of three counters, Timers B0 to B2, each equipped with a 16-bit reload function. Timers B0 to B2 operate independently of one another.

Timer B has three operating modes listed below. Timers B0 and B1 have selective three operating modes listed below. Timer B2 operates only in the timer mode.

(1) Timer mode (Timers B0 to B2)
The timer counts an internally generated count source.

(2) Event counter mode (Timers B0 and B1)
The timer counts an external signal.

(3) Pulse period/Pulse width measurement mode (Timers B0 and B1)
The timer measures an external signal's pulse period or pulse width.

In this chapter, Timer Bi (i = 0 to 2) indicates Timers B0 to B2. Timer Bj (j = 0, 1) indicates Timers B0 and B1; this is used when the timer B's input/output pins are used etc. (Hereafter, input/output pins are called I/O pins.)

9.2 Block description

Figure 9.2.1 shows the block diagram of Timer B. Explanation of registers relevant to Timer B is described below.

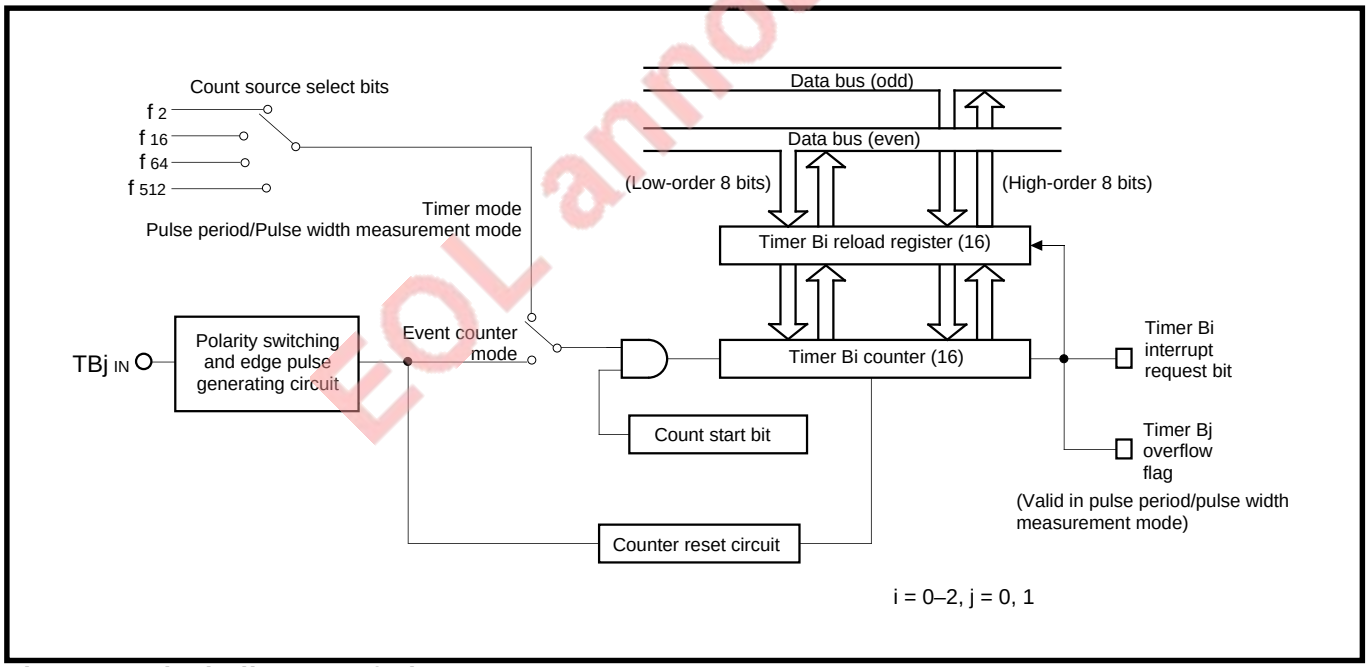


Fig. 9.2.1 Block diagram of Timer B

EOL announced

9.2.1 Counter and reload register (timer Bi register)

Each of timer Bi counter and reload register consists of 16 bits and has the following functions.

(1) Functions in timer mode and event counter mode

Countdown in the counter is performed each time the count source is input. The reload register is used to store the initial value of the counter. When a counter underflow occurs, the reload register's contents are reloaded into the counter.

A value is set to the counter and reload register by writing the value to the timer Bi register. Table 9.2.1 lists the memory assignment of the timer Bi register.

The value written into the timer Bi register when the counting is not in progress is set to the counter and reload register. The value written into the timer Bi register when the counting is in progress is set only to the reload register. In this case, the reload register's updated contents are transferred to the counter when the next underflow occurs. The counter value is read out by reading out the timer Bi register.

Note: When reading from or writing to the timer Bi register, perform it in a unit of 16 bits. For more information about the value obtained by reading the timer Bi register, refer to **"[Precautions for timer mode]"** and **"[Precautions for event counter mode]."**

(2) Functions in pulse period/pulse width measurement mode

Countup in the counter is performed each time the count source is input. The reload register is used to retain the pulse period or pulse width measurement result. When a valid edge is input to the TBj_{IN} pin, the counter value is transferred to the reload register. In this mode, the value obtained by reading the timer Bj register is the reload register's contents, so that the measurement result is obtained.

Note: When reading from the timer Bj register, perform it in a unit of 16 bits.

Table 9.2.1 Memory assignment of timer Bi registers

Timer Bi register	High-order byte	Low-order byte
Timer B0 register	Address 51_{16}	Address 50_{16}
Timer B1 register	Address 53_{16}	Address 52_{16}
Timer B2 register	Address 55_{16}	Address 54_{16}

Note: At reset, the contents of the timer Bi register are undefined.

EOL announced

9.2 Block description

9.2.2 Count start register

This register is used to start and stop counting. Each bit of this register corresponds to each timer. Figure 9.2.2 shows the structure of the count start register.



Count start register (Address 40_{16})

Bit	Bit name	Functions	At reset	RW
0	Timer A0 count start bit	0 : Stop counting 1 : Start counting	0	RW
1	Timer A1 count start bit		0	RW
2	Timer A2 count start bit		0	RW
3	Timer A3 count start bit		0	RW
4	Timer A4 count start bit		0	RW
5	Timer B0 count start bit		0	RW
6	Timer B1 count start bit		0	RW
7	Timer B2 count start bit		0	RW

: Bits 0 to 4 are not used for Timer B.

Fig. 9.2.2 Structure of count start register

EOL announced

9.2.3 Timer Bi mode register

Figure 9.2.3 shows the structure of the timer Bi mode register. The operating mode select bits are used to select the operating mode of Timer Bi. Bits 2, 3, and bits 5 to 7 have different functions according to the operating mode. These bits are described in the paragraph of each operating mode.



Timer Bi mode register (i = 0 to 2) (Addresses $5B_{16}$ to $5D_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 0 0 : Timer mode 0 1 : Event counter mode 1 0 : Pulse period/Pulse width measurement mode 1 1 : Do not select.	0	RW
1			0	RW
2	These bits have different functions according to the operating mode.		0	RW
3			0	RW
4	Nothing is assigned.		Undefined	–
5	These bits have different functions according to the operating mode.		Undefined	RO **(Note)**
6			0	RW
7			0	RW

Note: Bit 5 is invalid in the timer and event counter modes; its value is undefined at reading.

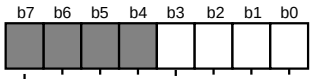
Fig. 9.2.3 Structure of timer Bi mode register

EOL announced

9.2 Block description

9.2.4 Timer Bi interrupt control register

Figure 9.2.4 shows the structure of the timer Bi interrupt control register. For details about interrupts, refer to **"CHAPTER 7. INTERRUPTS."**



Timer Bi interrupt control register (i = 0 to 2) (Addresses $7A_{16}$ to $7C_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Interrupt priority level select bits	b2 b1 b0 0 0 0 : Level 0 (Interrupt disabled) 0 0 1 : Level 1 (Low level) 0 1 0 : Level 2 0 1 1 : Level 3 1 0 0 : Level 4 1 0 1 : Level 5 1 1 0 : Level 6 1 1 1 : Level 7 (High level)	0	RW
1			0	RW
2			0	RW
3	Interrupt request bit	0 : No interrupt requested 1 : Interrupt requested	0	RW
7 to 4	Nothing is assigned.		Undefined	—

Fig. 9.2.4 Structure of timer Bi interrupt control register

(1) Interrupt priority level select bits (bits 2 to 0)

These bits select a timer Bi interrupt's priority level. When using timer Bi interrupts, select one of the priority levels (1 to 7). When a timer Bi interrupt request occurs, its priority level is compared with the processor interrupt priority level (IPL). The requested interrupt is enabled only when its priority level is higher than the IPL. (However, this applies when the interrupt disable bit (I) = "0.") To disable timer Bi interrupts, set these bits to "000_2" (level 0).

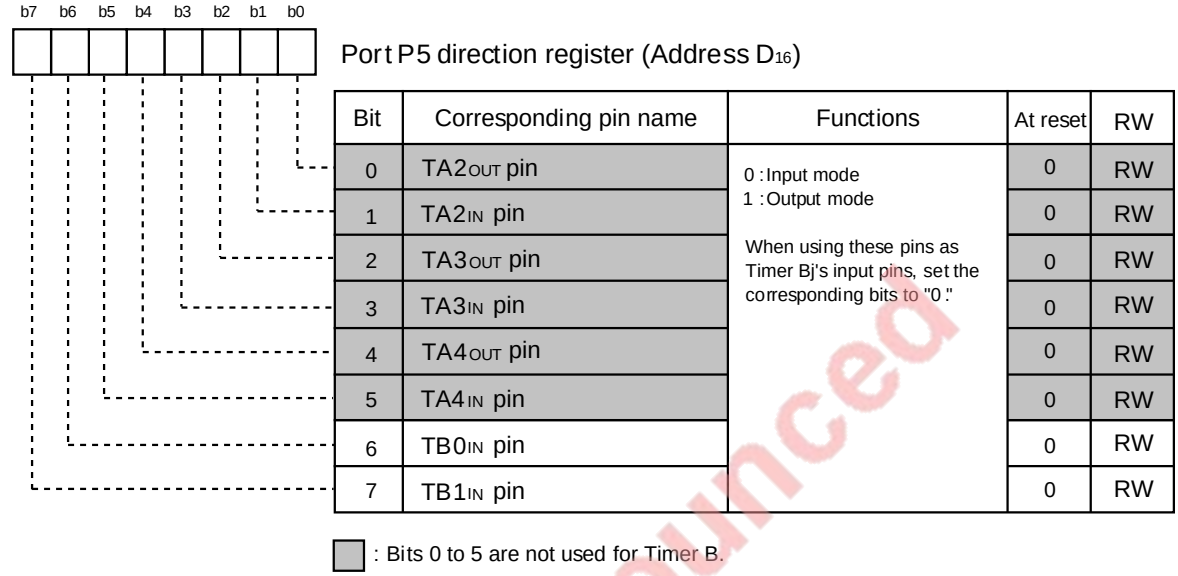
(2) Interrupt request bit (bit 3)

This bit is set to "1" when a timer Bi interrupt request occurs. This bit is automatically cleared to "0" when the timer Bi interrupt request is accepted. This bit can be set to "1" or cleared to "0" by software.

EOL announced

9.2.5 Port P5 direction register

Input pins of Timer Bj are multiplexed with port P5. When using these pins as Timer Bj's input pins, set the corresponding bits of the port P5 direction register to "0" to set these port pins for the input mode. Figure 9.2.5 shows the relationship between port P5 direction register and the Timer Bj's input pins.



Bit	Corresponding pin name	Functions	At reset	RW
0	TA2OUT pin	0 : Input mode 1 : Output mode When using these pins as Timer Bj's input pins, set the corresponding bits to "0."	0	RW
1	TA2IN pin		0	RW
2	TA3OUT pin		0	RW
3	TA3IN pin		0	RW
4	TA4OUT pin		0	RW
5	TA4IN pin		0	RW
6	TB0IN pin		0	RW
7	TB1IN pin		0	RW

: Bits 0 to 5 are not used for Timer B.

Fig. 9.2.5 Relationship between port P5 direction register and Timer Bj's input pins

EOL announced

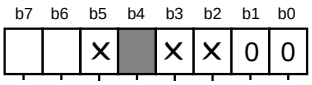
9.3 Timer mode

In this mode, the timer counts an internally generated count source. (Refer to **"Table 9.3.1."**) Figure 9.3.1 shows the structures of the timer Bi mode register and timer Bi register in the timer mode.

Table 9.3.1 Specifications of timer mode

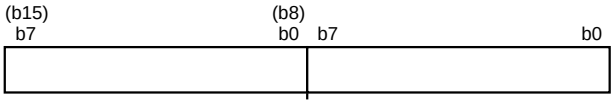
Item	Specifications
Count source	f_2, f_{16}, f_{64}, or f_{512}
Count operation	•Countdown •When a counter underflow occurs, reload register's contents are reloaded, and counting continues.
Division ratio	$\frac{1}{(n+1)}$ n : Timer Bi register's set value
Count start condition	When the count start bit is set to "1."
Count stop condition	When the count start bit is cleared to "0."
Interrupt request occurrence timing	When a counter underflow occurs.
TBj_{IN} pin's function	Programmable I/O port
Read from timer Bi register	Counter value can be read out.
Write to timer Bi register	● While counting is stopped When a value is written to the timer Bi register, it is written to both of the reload register and counter. ● While counting is in progress When a value is written to the timer Bi register, it is written only to the reload register. (Transferred to the counter at the next reload time.)

EOL announced



Timer Bi mode register (i = 0 to 2) (Addresses $5B_{16}$ to $5D_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 0 0 : Timer mode	0	RW
1			0	RW
2	These bits are invalid in timer mode.		0	RW
3			0	RW
4	Nothing is assigned.		Undefined	–
5	This bit is invalid in timer mode; its value is undefined at reading.		Undefined	RO
6	Count source select bits	b7 b6 0 0 : f_2 0 1 : f_{16} 1 0 : f_{64} 1 1 : f_{512}	0	RW
7			0	RW



Timer B0 register (Addresses 51_{16}, 50_{16})
Timer B1 register (Addresses 53_{16}, 52_{16})
Timer B2 register (Addresses 55_{16}, 54_{16})

Bit	Functions	At reset	RW
15 to 0	These bits can be set to "0000_{16}" to "$FFFF_{16}$." Assuming that the set value = n, the counter divides the count source frequency by (n + 1). When reading, the register indicates the counter value.	Undefined	RW

Note: Read from or write to this register in a unit of 16 bits.

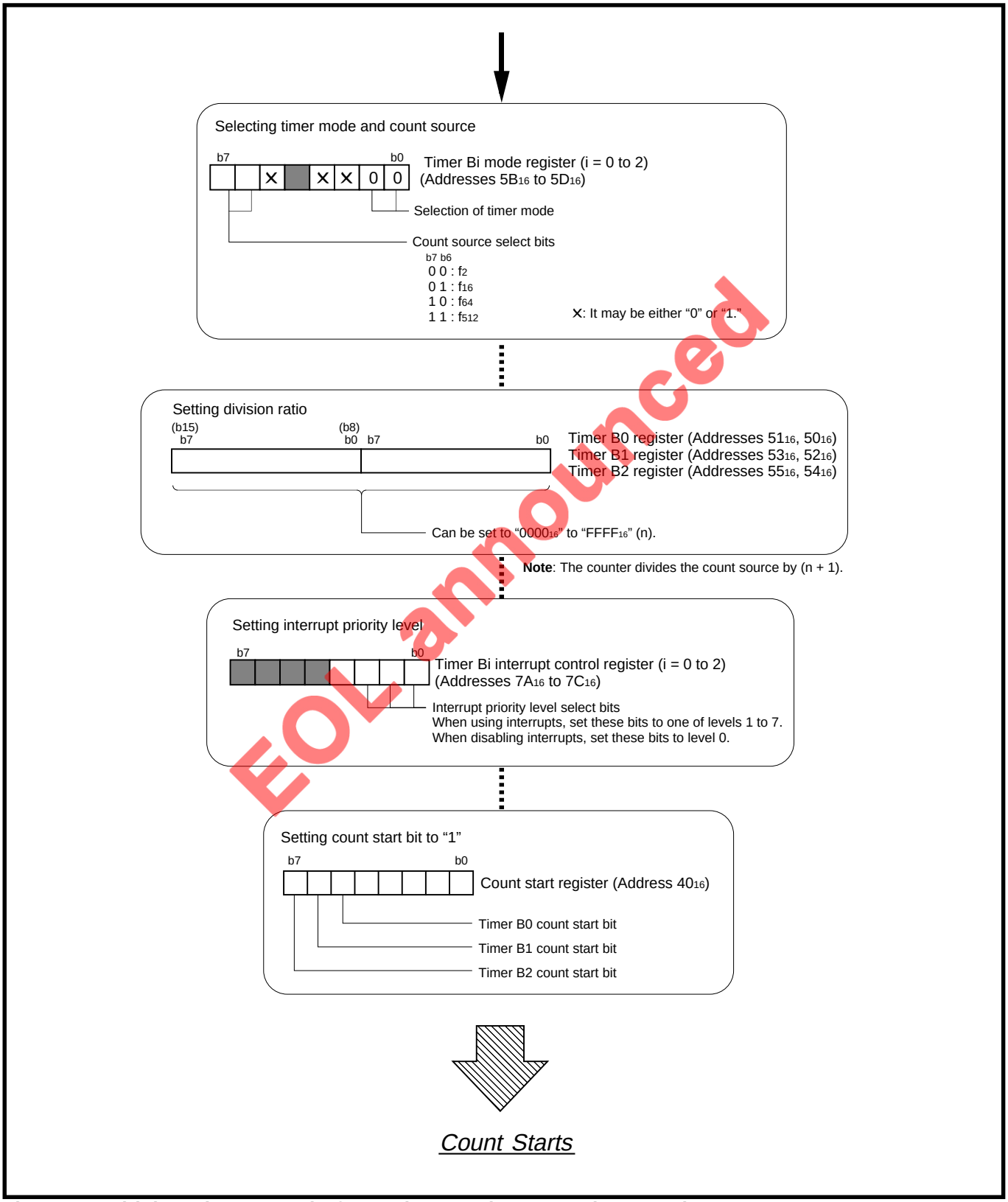
Fig. 9.3.1 Structures of timer Bi mode register and timer Bi register in timer mode

EOL announced

9.3 Timer mode

9.3.1 Setting for timer mode

Figure 9.3.2 shows an initial setting example for registers relevant to the timer mode.
Note that when using interrupts, set up to enable the interrupts. For details, refer to **"CHAPTER 7. INTERRUPTS."**



EOL announced

Fig. 9.3.2 Initial setting example for registers relevant to timer mode

9.3.2 Count source

In the timer mode, the count source select bits (bits 6 and 7 at addresses $5B_{16}$ to $5D_{16}$) select the count source. Table 9.3.2 lists the count source frequency.

Table 9.3.2 Count source frequency

Count source select bits		Count source	Count source frequency		
b7	b6		$f(X_{IN})$ = 8 MHz	$f(X_{IN})$ = 16 MHz	$f(X_{IN})$ = 25 MHz
0	0	f_2	4 MHz	8 MHz	12.5 MHz
0	1	f_{16}	500 kHz	1 MHz	1.5625 MHz
1	0	f_{64}	125 kHz	250 kHz	390.625 kHz
1	1	f_{512}	15625 Hz	31250 Hz	48.8281 kHz

EOL announced

9.3 Timer mode

9.3.3 Operation in timer mode

① When the count start bit is set to “1,” the counter starts counting of the count source.
② When a counter underflow occurs, the reload register’s contents are reloaded, and counting continues.
③ The timer Bi interrupt request bit is set to “1” at the underflow in ②. The interrupt request bit remains set to “1” until the interrupt request is accepted or the interrupt request bit is cleared to “0” by software.

Figure 9.3.3 shows an example of operation in the timer mode.

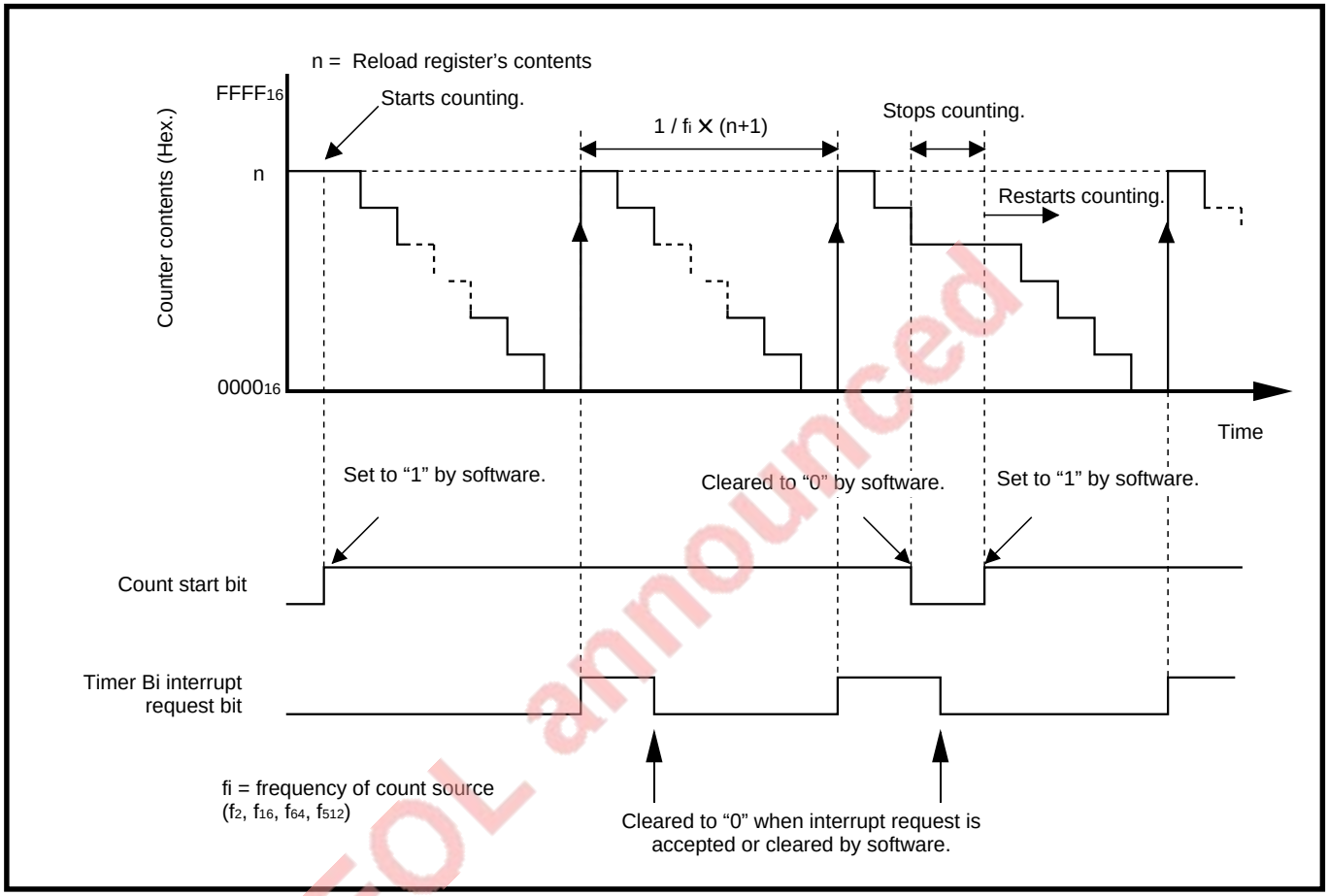


Fig. 9.3.3 Example of operation in timer mode

EOL announced

[Precautions for timer mode]

While counting is in progress, by reading the timer Bi register, the counter value can be read out at any timing. However, if the timer Bi register is read at the reload timing shown in Figure 9.3.4, the value "FFFF16" is read out. If reading is performed in the period from when a value is set into the timer Bi register with the counter stopped until the counter starts counting, the set value is correctly read out.

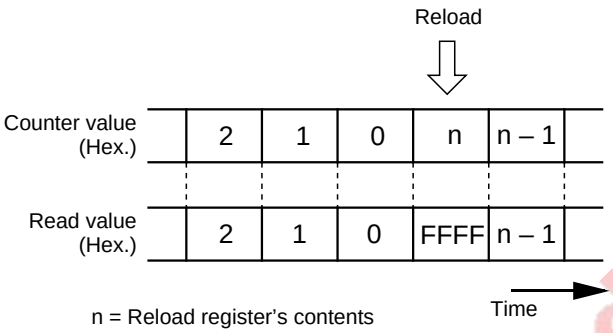


Fig. 9.3.4 Reading timer Bi register

EOL announced

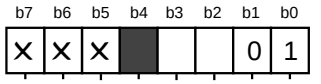
9.4 Event counter mode

In this mode, the timer counts an external signal. (Refer to **"Table 9.4.1."**) Figure 9.4.1 shows the structures of the timer Bj mode register and the timer Bj register in the event counter mode.

Table 9.4.1 Specifications of event counter mode

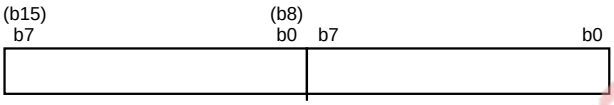
Item	Specifications
Count source	•External signal input to the TBj_{IN} pin •The count source's valid edge can be selected from the falling edge, the rising edge, and both of the falling and rising edges by software.
Count operation	•Countdown •When a counter underflow occurs, reload register's contents are reloaded, and counting continues.
Division ratio	$\frac{1}{(n+1)}$ n : Timer Bj register's set value
Count start condition	When the count start bit is set to "1."
Count stop condition	When the count start bit is cleared to "0."
Interrupt request occurrence timing	When a counter underflow occurs.
TBj_{IN} pin's function	Count source input
Read from timer Bj register	Counter value can be read out.
Write to timer Bj register	● While counting is stopped When a value is written to the timer Bj register, it is written to both of the reload register and counter. ● While counting is in progress When a value is written to the timer Bj register, it is written only to the reload register. (Transferred to the counter at the next reload time.)

EOL announced



Timer Bj mode register (j = 0, 1) (Addresses $5B_{16}$, $5C_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 0 1 : Event counter mode	0	RW
1			0	RW
2	Count polarity select bits	b3 b2 0 0 : Count at falling edge of external signal 0 1 : Count at rising edge of external signal 1 0 : Counts at both falling and rising edges of external signal 1 1 : Do not select.	0	RW
3			0	RW
4	Nothing is assigned.		Undefined	—
5	This bit is invalid in event counter mode; its value is undefined at reading.		Undefined	RO
6	These bits are invalid in event counter mode.		0	RW
7			0	RW



Timer B0 register (Addresses 51_{16}, 50_{16})
Timer B1 register (Addresses 53_{16}, 52_{16})

Bit	Functions	At reset	RW
15 to 0	These bits can be set to "0000_{16}" to "$FFFF_{16}$." Assuming that the set value = n, the counter divides the count source frequency by (n + 1). When reading, the register indicates the counter value.	Undefined	RW

Note: Read from or write to this register in a unit of 16 bits.

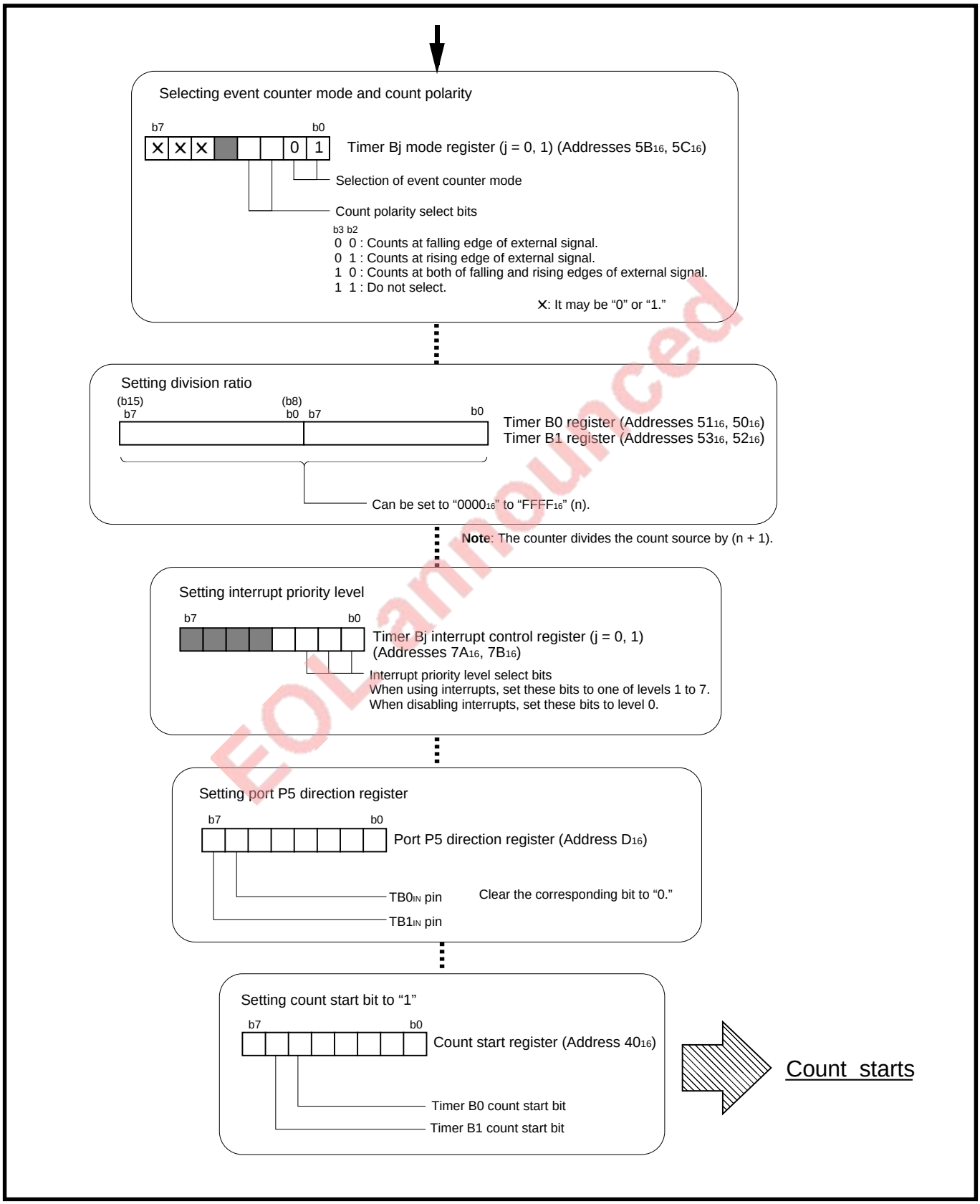
EOL announced

Fig. 9.4.1 Structures of timer Bj mode register and timer Bj register in event counter mode

9.4 Event counter mode

9.4.1 Setting for event counter mode

Figure 9.4.2 shows an initial setting example for registers relevant to the event counter mode.
Note that when using interrupts, set up to enable the interrupts. For details, refer to **"CHAPTER 7. INTERRUPTS."**



EOL announced

Fig. 9.4.2 Initial setting example for registers relevant to event counter mode

9.4.2 Operation in event counter mode

① When the count start bit is set to “1,” the counter starts counting of the count source’s valid edges.
② When a counter underflow occurs, the reload register’s contents are reloaded, and counting continues.
③ The timer Bj interrupt request bit is set to “1” at the underflow in ②.
The interrupt request bit remains set to “1” until the interrupt request is accepted or the interrupt request bit is cleared to “0” by software.

Figure 9.4.3 shows an example of operation in the event counter mode.

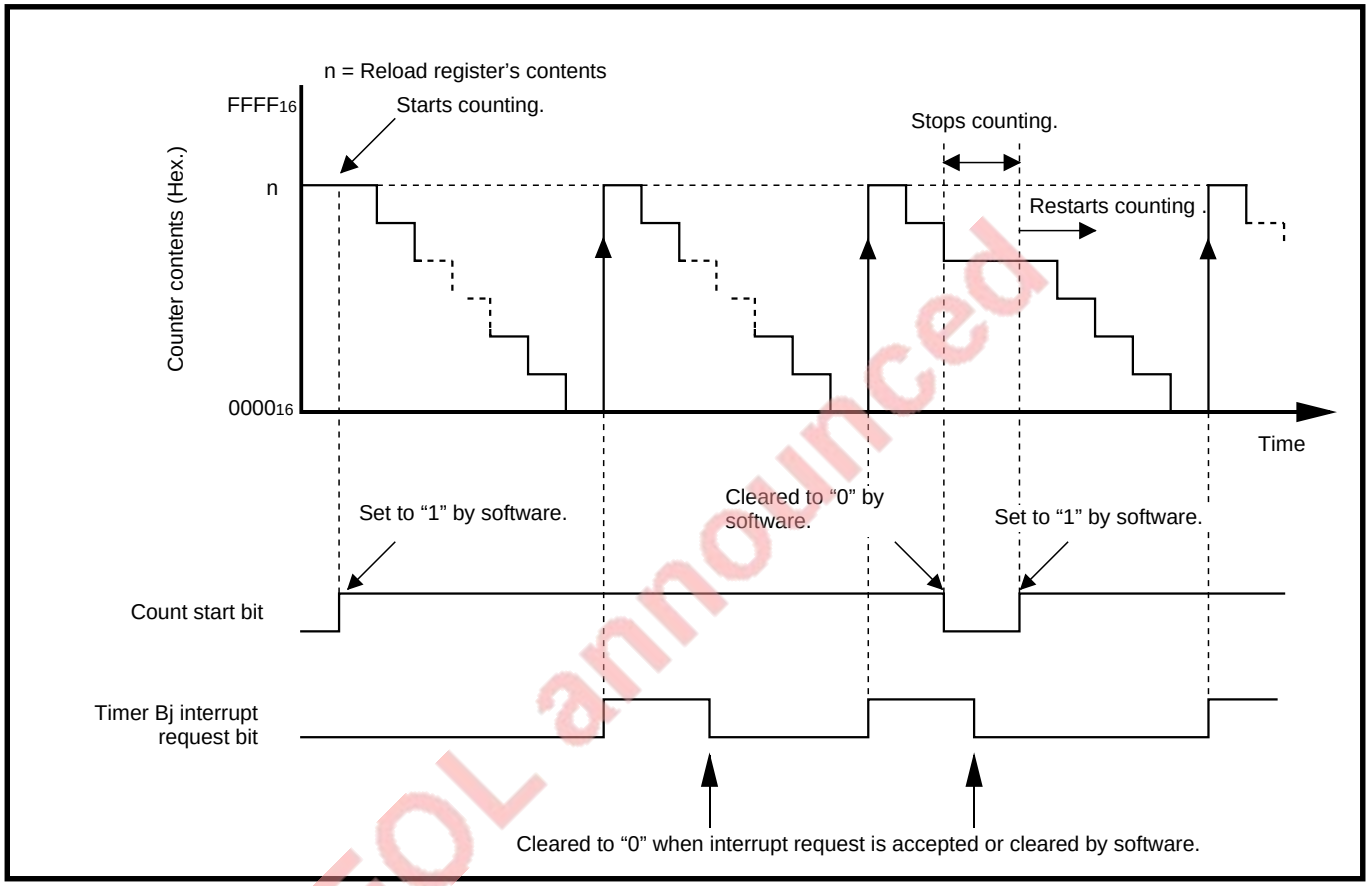


Fig. 9.4.3 Example of operation in event counter mode

EOL announced

9.4 Event counter mode

[Precautions for event counter mode]

While counting is in progress, by reading the timer Bj register, the counter value can be read out at any timing. However, if the timer Bj register is read at the reload timing shown in Figure 9.4.4, the value "FFFF16" is read out. If reading is performed in the period from when a value is set into the timer Bj register with the counter stopped until the counter starts counting, the set value is correctly read out.

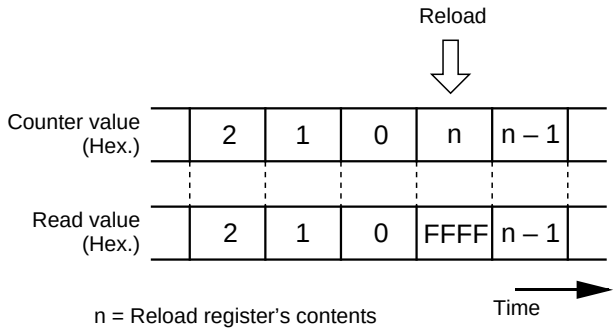


Fig. 9.4.4 Reading timer Bj register

EOL announced

9.5 Pulse period/Pulse width measurement mode

In this mode, the timer measures an external signal's pulse period or pulse width. (Refer to **"Table 9.5.1."**) Timers B0 and B1 can be used in this mode. Figure 9.5.1 shows the structures of the timer Bj mode register and timer Bj register in the pulse period/pulse width measurement mode.

- **Pulse period measurement**
  The timer measures the pulse period of the external signal that is input to the TBj_{IN} pin.

- **Pulse width measurement**
  The timer measures the pulse width ("L" level and "H" level widths) of the external signal that is input to the TBj_{IN} pin.

Table 9.5.1 Specifications of pulse period/pulse width measurement mode

Item	Specifications
Count source	f_2, f_{16}, f_{64}, or f_{512}
Count operation	● Countup ● Counter value is transferred to the reload register at valid edge of measurement pulse, and counting continues after clearing the counter value to "0000_{16}."
Count start condition	When the count start bit is set to "1."
Count stop condition	When the count start bit is cleared to "0."
Interrupt request occurrence timing	● When valid edge of measurement pulse is input (**Note 1**). ● When a counter overflow occurs (Timer Bj overflow flag* is set to "1" simultaneously.)
TBj_{IN} pin's function	Measurement pulse input
Read from timer Bj register	The value obtained by reading timer Bj register is the reload register's contents (Measurement result) (**Note 2**).
Write to timer Bj register	Invalid

Timer Bj overflow flag*: The bit used to identify the source of an interrupt request occurrence.

Notes 1: No interrupt request occurs when the first valid edge is input after the counter starts counting.

2: The value read out from the timer Bj register is undefined after the counter starts counting until the second valid edge is input.

EOL announced

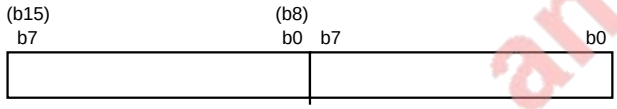
9.5 Pulse period/Pulse width measurement mode



Timer Bj mode register (j = 0, 1) (Addresses $5B_{16}$, $5C_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 1 0 : Pulse period/Pulse width measurement mode	0	RW
1			0	RW
2	Measurement mode select bits	b3 b2 0 0 : Pulse period measurement (Interval between falling edges of measurement pulse) 0 1 : Pulse period measurement (Interval between rising edges of measurement pulse) 1 0 : Pulse width measurement (Interval from a falling edge to a rising edge, and from a rising edge to a falling edge of measurement pulse) 1 1 : Do not select.	0	RW
3			0	RW
4	Nothing is assigned.		Undefined	–
5	Timer Bj overflow flag **(Note)**	0 : No overflow 1 : Overflowed	Undefined	RO
6	Count source select bits	b7 b6 0 0 : f_2 0 1 : f_{16} 1 0 : f_{64} 1 1 : f_{512}	0	RW
7			0	RW

Note: The timer Bj overflow flag is cleared to "0" at the next count timing of the count source when a value is written to the timer Bj mode register with the count start bit = "1."



Timer B0 register (Addresses 51_{16}, 50_{16})
Timer B1 register (Addresses 53_{16}, 52_{16})

Bit	Functions	At reset	RW
15 to 0	The measurement result of pulse period or pulse width is read out.	Undefined	RO

Note: Read from this register in a unit of 16 bits.

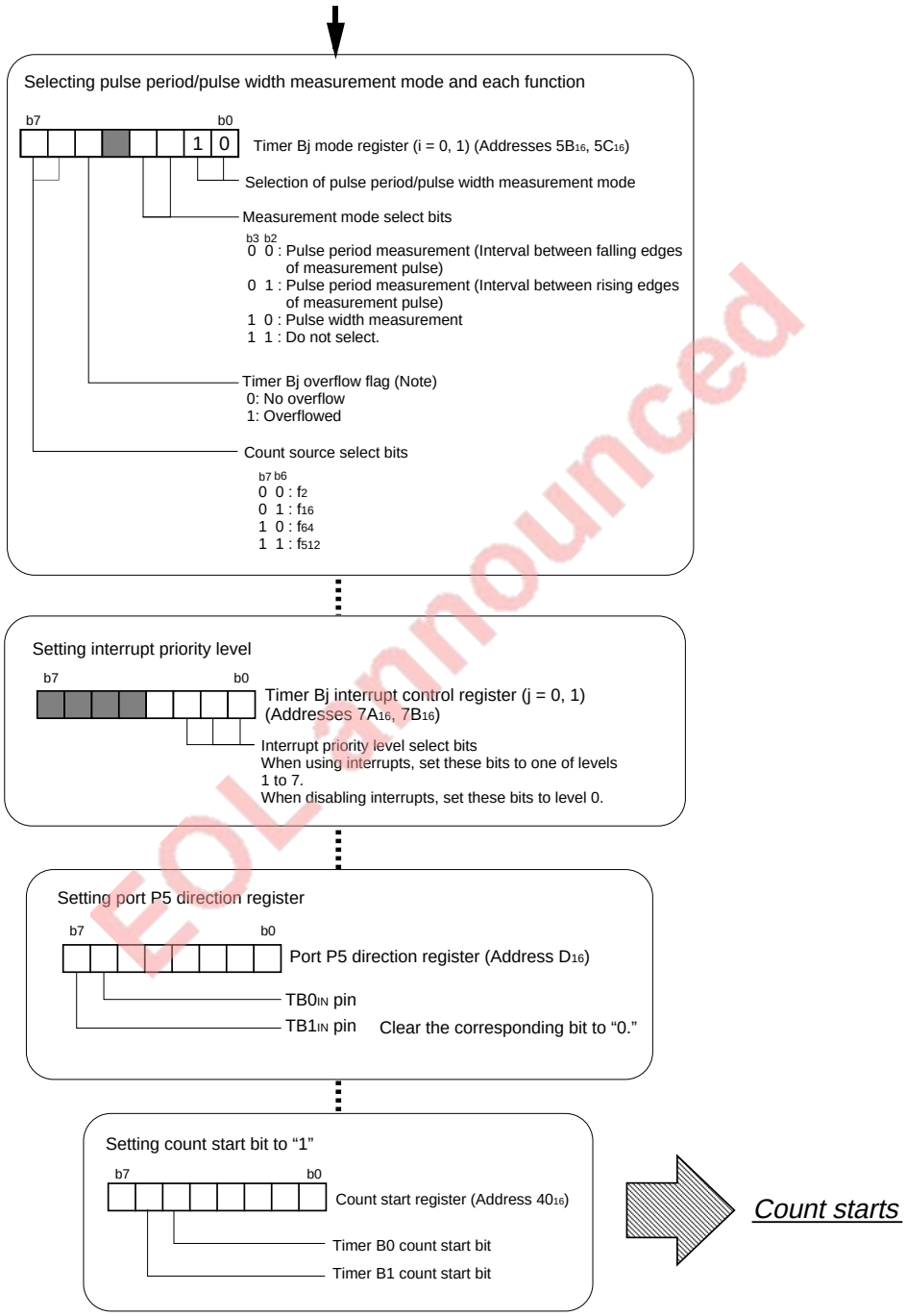
EOL announced

Fig. 9.5.1 Structures of timer Bj mode register and timer Bj register in pulse period/pulse width measurement mode

9.5.1 Setting for pulse period/pulse width measurement mode

Figure 9.5.2 shows an initial setting example for registers relevant to the pulse period/pulse width measurement mode.

Note that when using interrupts, set up to enable the interrupts. For details, refer to **"CHAPTER 7. INTERRUPTS."**



EOL announced

Note: The timer Bj overflow flag is a read-only bit. This bit is undefined after reset. When a value is written to the timer Bj mode register with the count start bit = "1," this bit is cleared to "0" at the next count timing of the count source.

Fig. 9.5.2 Initial setting example for registers relevant to pulse period/pulse width measurement mode

9.5 Pulse period/Pulse width measurement mode

9.5.2 Count source

In the pulse period/pulse width measurement mode, the count source select bits (bits 6 and 7 at addresses $5B_{16}$ and $5C_{16}$) select the count source.
Table 9.5.2 lists the count source frequency.

Table 9.5.2 Count source frequency

Count source select bits		Count source	Count source frequency		
b7	b6		$f(X_{IN})$ = 8 MHz	$f(X_{IN})$ = 16 MHz	$f(X_{IN})$ = 25 MHz
0	0	f_2	4 MHz	8 MHz	12.5 MHz
0	1	f_{16}	500 kHz	1 MHz	1.5625 MHz
1	0	f_{64}	125 kHz	250 kHz	390.625 kHz
1	1	f_{512}	15625 Hz	31250 Hz	48.8281 kHz

EOL announced

9.5.3 Operation in pulse period/pulse width measurement mode

① When the count start bit is set to "1," the counter starts counting of the count source.
② The counter value is transferred to the reload register when an valid edge of the measurement pulse is detected. (Refer to section **"(1) Pulse period/Pulse width measurement."**)
③ The counter value is cleared to "0000_{16}" after the transfer in ②, and the counter continues counting.
④ The timer Bj interrupt request bit is set to "1" when the counter value is cleared to "0000_{16}" in ③ (**Note**). The interrupt request bit remains set to "1" until the interrupt request is accepted or the interrupt request bit is cleared to "0" by software.
⑤ The timer repeats operations ② to ④ above.

Note: No timer Bj interrupt request occurs when the first valid edge is input after the counter starts counting.

(1) Pulse period/Pulse width measurement

The measurement mode select bits (bits 2 and 3 at addresses $5B_{16}$ and $5C_{16}$) specify whether the pulse period of an external signal is measured or its pulse width is done. Table 9.5.3 lists the relationship between the measurement mode select bits and the pulse period/pulse width measurements. Make sure that the measurement pulse interval from the falling edge to the rising edge, and vice versa are two cycles of the count source or more. Additionally, use software to identify whether the measurement result indicates the "H" level or the "L" level width.

Table 9.5.3 Relationship between measurement mode select bits and pulse period/pulse width measurements

b3	b2	Pulse period/Pulse width measurement	Measurement interval (Valid edges)
0	0	Pulse period measurement	From falling edge to falling edge (Falling edges)
0	1		From rising edge to rising edge (Rising edges)
1	0	Pulse width measurement	From falling edge to rising edge, and vice versa (Falling and rising edges)

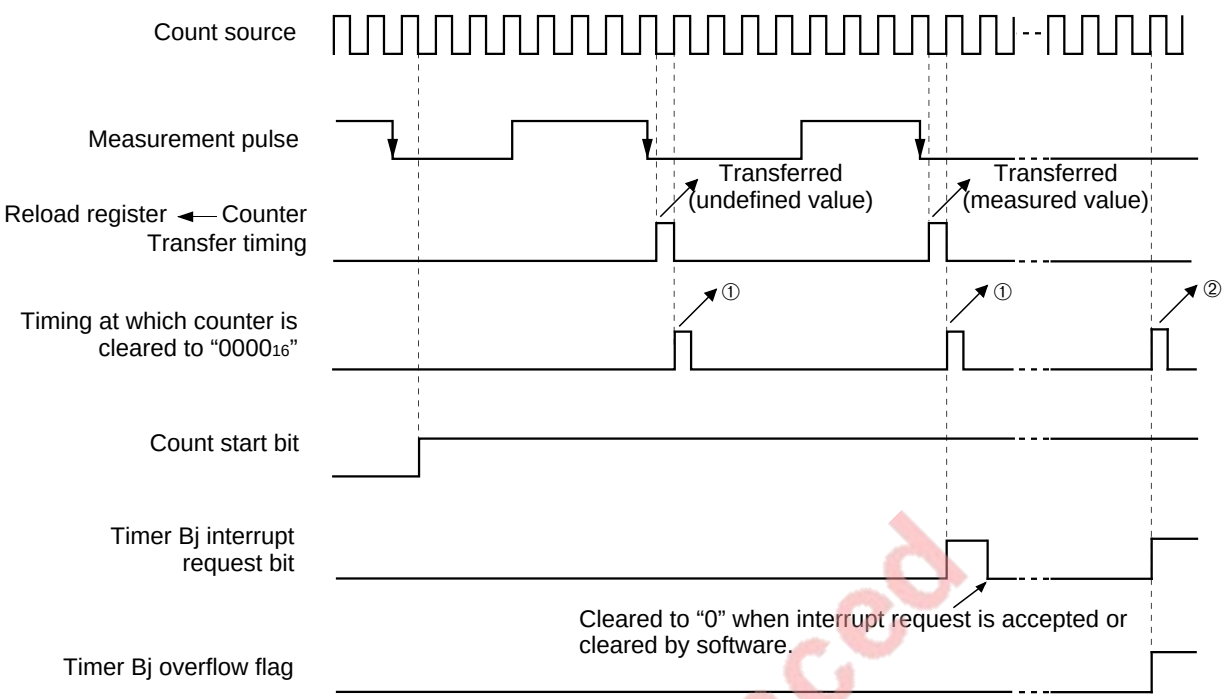
(2) Timer Bj overflow flag

A timer Bj interrupt request occurs when a measurement pulse's valid edge is input or a counter overflow occurs. The timer Bj overflow flag is used to identify the cause of the interrupt request, that is, whether it is an overflow occurrence or a valid edge input.
The timer Bj overflow flag is set to "1" by an overflow. Accordingly, the cause of the interrupt request occurrence is identified by checking the timer Bj overflow flag in the interrupt routine. When a value is written to the timer Bj mode register with the count start bit = "1," the timer Bj overflow flag is cleared to "0" at the next count timing of the count source
The timer Bj overflow flag is a read-only bit.
Use the timer Bi interrupt request bit to detect the overflow timing. Do not use the timer Bi overflow flag for this detection.

Figure 9.5.3 shows the operation during pulse period measurement. Figure 9.5.4 shows the operation during pulse width measurement.

PDL announced

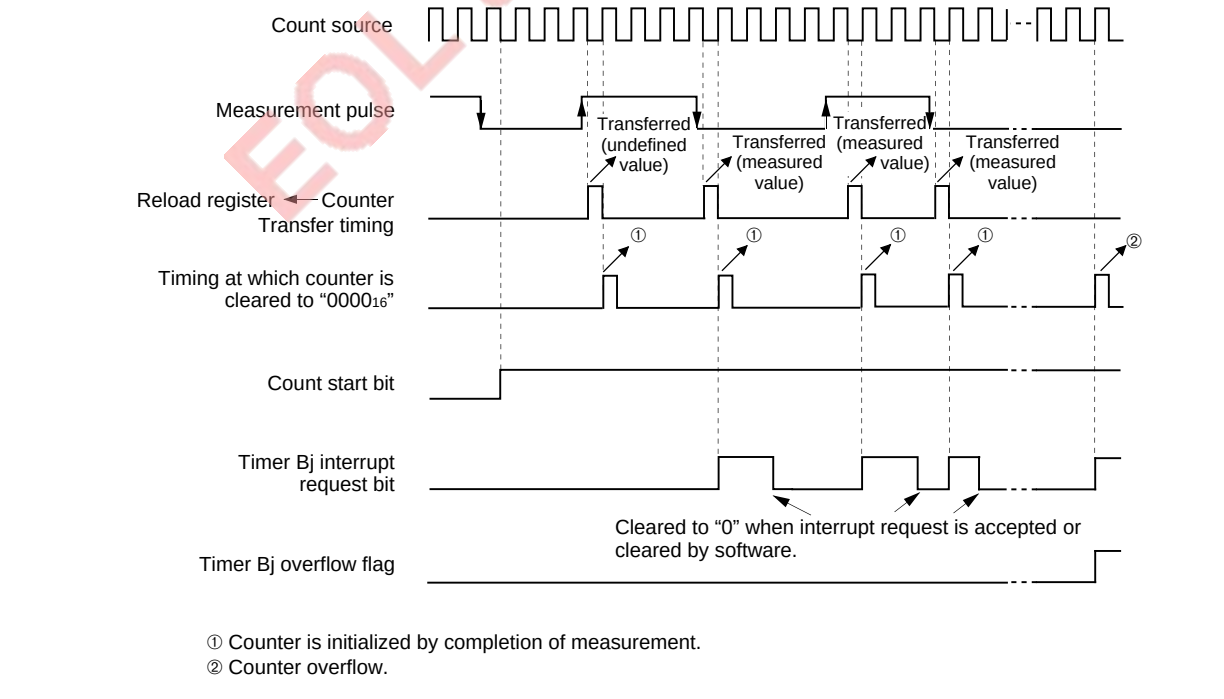
9.5 Pulse period/Pulse width measurement mode



① Counter is initialized by completion of measurement.
② Counter overflow.

Note: The above applies when measurement is performed for an interval from one falling edge to the next falling edge of the measurement pulse.

Fig. 9.5.3 Operation during pulse period measurement



① Counter is initialized by completion of measurement.
② Counter overflow.

Fig. 9.5.4 Operation during pulse width measurement

EOL announced

[Precautions for pulse period/pulse width measurement mode]

1. A timer Bj interrupt request is generated by the following sources:
   - Input of measured pulse's valid edge
   - Counter overflow

   When the overflow generates the interrupt request, the timer Bj overflow flag is set to "1."

2. After reset, the timer Bj overflow flag is undefined. When a value is written to the timer Bj mode register with the count start bit = "1," this flag is cleared to "0" at the next count timing of the count source.

3. An undefined value is transferred to the reload register when the first valid edge is input after the counter starts counting. In this case, no timer Bj interrupt request occurs.

4. The counter value at start of counting is undefined. Accordingly, a timer Bj interrupt request may be generated by an overflow immediately after the counter starts counting.

5. If the contents of the measurement mode select bits are changed after the counter starts counting, the timer Bj interrupt request bit is set to "1." When the same value which has been set in these bits are written again, the timer Bj interrupt request bit is not changed, that is, the bit retains the state.

6. If the input signal to the TBj_{IN} pin is affected by noise, etc., the counter may not perform the exact measurement. We recommend to verify, by software, that the measurement values are within a constant range.

EOL announced

MEMORANDUM

EOL announced

CHAPTER 10
REAL-TIME OUTPUT

10.1 Overview
10.2 Block description
10.3 Setting of real-time output
10.4 Real-time output operation

EOL announced

10.1 Overview

The real-time output has the function of changing the output level of several pins simultaneously at every period of the timer. Figure 10.1.1 shows the block diagram of real-time output per bit. Real-time output has two operating modes described below.

(1) Pulse mode 0

The 8-bit pulse output pins serve for two independent 4-bit outputs. Figure 10.1.2 shows the configuration of real-time output in the pulse mode 0.

(2) Pulse mode 1

The 8-bit pulse output pins serve for a 2-bit and a 6-bit outputs. Figure 10.1.3 shows the configuration of real-time output in the pulse mode 1.

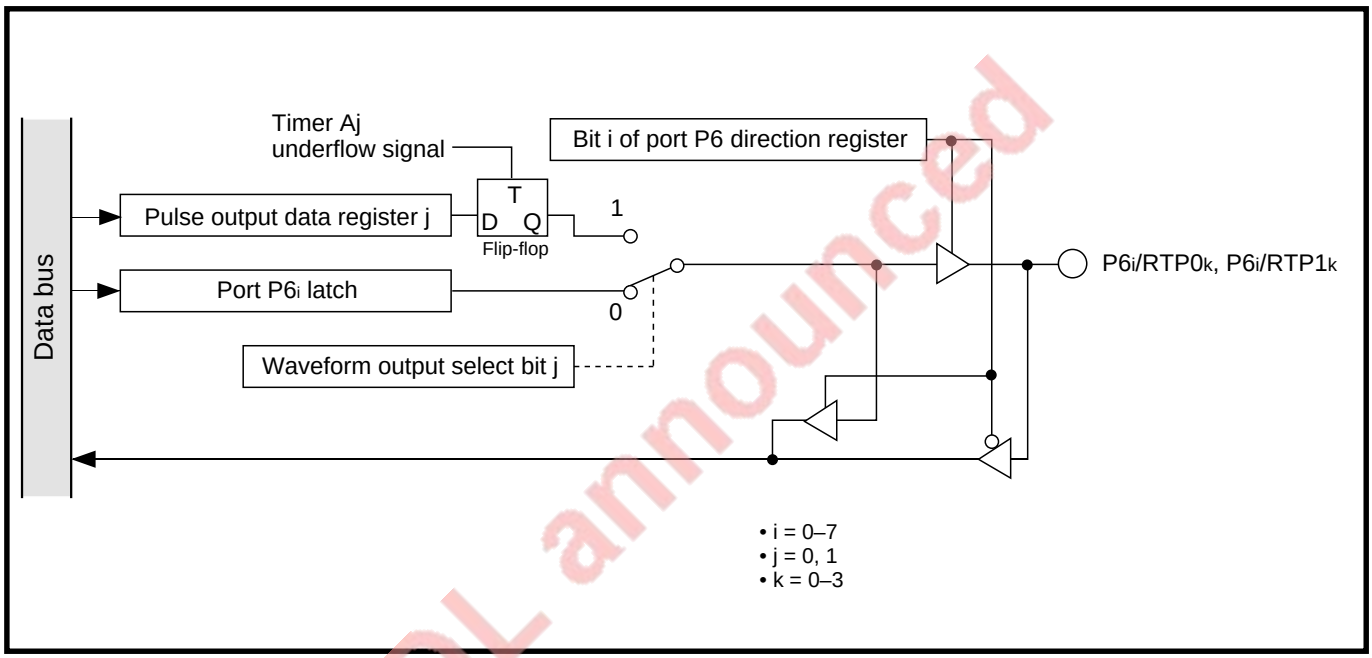


Fig. 10.1.1 Block diagram of real-time output per bit

EOL announced

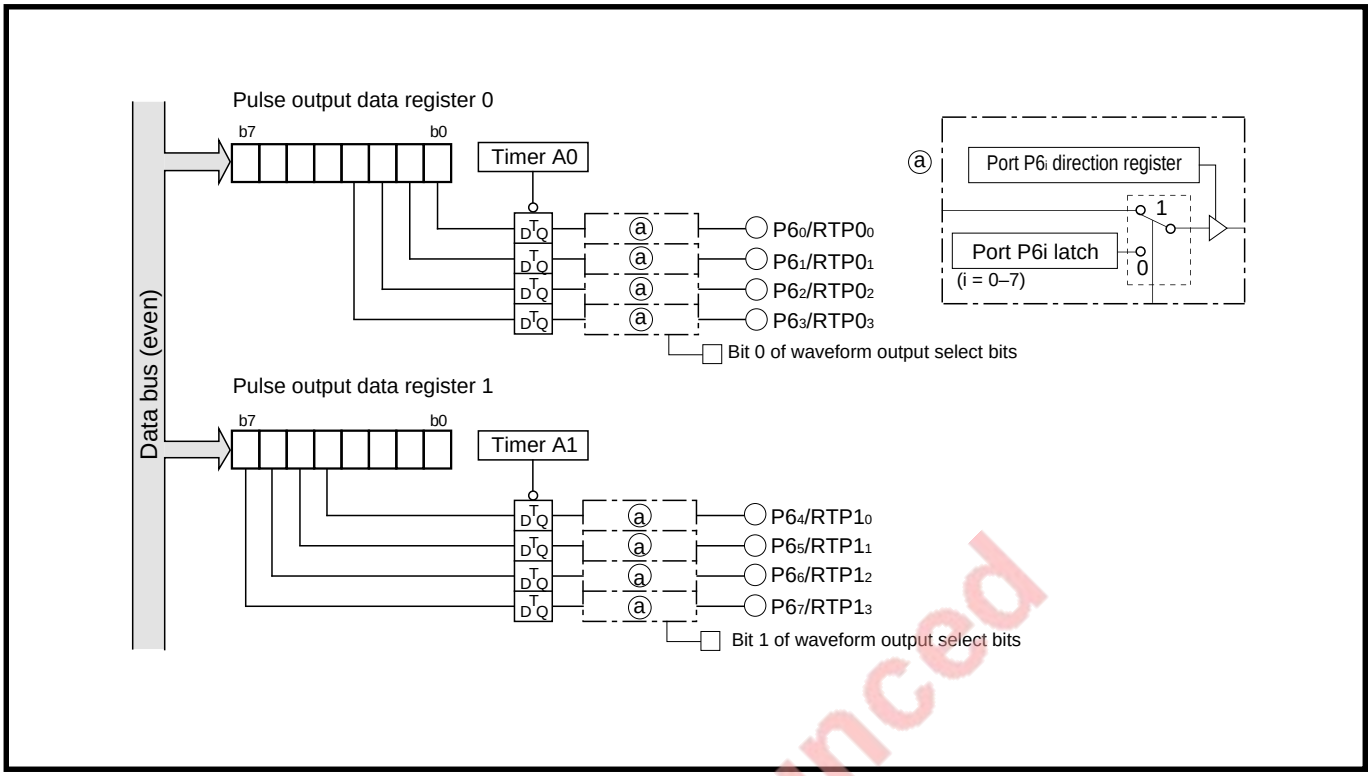


Fig. 10.1.2 Configuration of real-time output in pulse mode 0

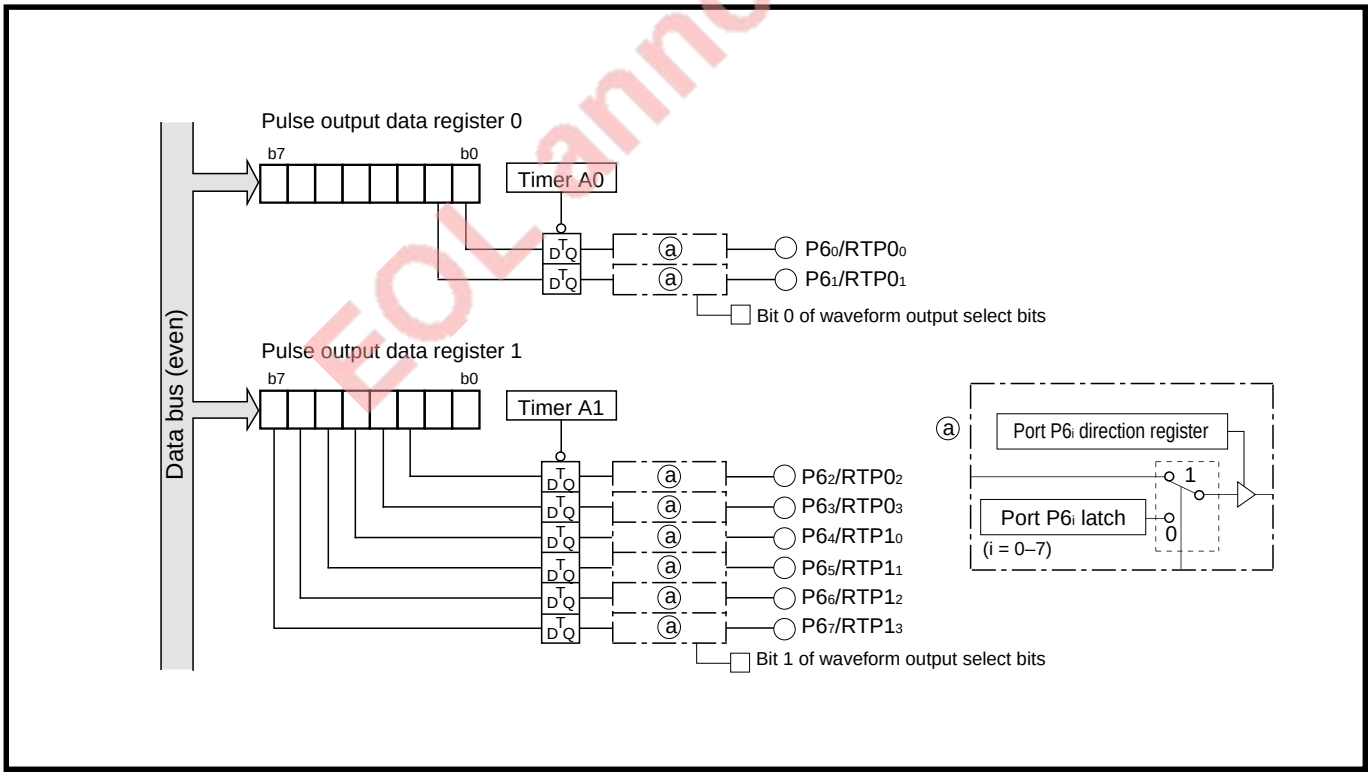


Fig. 10.1.3 Configuration of real-time output in pulse mode 1

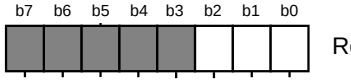
EOL announced

10.2 Block description

Relevant registers to real-time output are described below.

10.2.1 Real-time output control register

Figure 10.2.1 shows the structure of the real-time output control register.



Real-time output control register (Address 62_{16})

Bit	Bit name	Functions	At reset	RW
0	Waveform output select bits	See the following Table.	0	RW
1			0	RW
2	Pulse output mode select bit	0 : Pulse mode 0 1 : Pulse mode 1	0	RW
7 to 3	Nothing is assigned. The value is “0” at reading.		Undefined	–

Note: When using the $P6_0$–$P6_7$ pins as the pulse output pins for real-time output, set the corresponding bits of the port P6 direction register (address 10_{16}) to “1.”

b1 b0	00	01	10	11
When pulse mode 0 is selected	$P6_7$/RTP1$_3$ $P6_6$/RTP1$_2$ $P6_5$/RTP1$_1$ $P6_4$/RTP1$_0$ — Port	$P6_7$/RTP1$_3$ $P6_6$/RTP1$_2$ $P6_5$/RTP1$_1$ $P6_4$/RTP1$_0$ — Port	$P6_7$/RTP1$_3$ $P6_6$/RTP1$_2$ $P6_5$/RTP1$_1$ $P6_4$/RTP1$_0$ — RTP	$P6_7$/RTP1$_3$ $P6_6$/RTP1$_2$ $P6_5$/RTP1$_1$ $P6_4$/RTP1$_0$ — RTP
	$P6_3$/RTP0$_3$ $P6_2$/RTP0$_2$ $P6_1$/RTP0$_1$ $P6_0$/RTP0$_0$ — Port	$P6_3$/RTP0$_3$ $P6_2$/RTP0$_2$ $P6_1$/RTP0$_1$ $P6_0$/RTP0$_0$ — RTP	$P6_3$/RTP0$_3$ $P6_2$/RTP0$_2$ $P6_1$/RTP0$_1$ $P6_0$/RTP0$_0$ — Port	$P6_3$/RTP0$_3$ $P6_2$/RTP0$_2$ $P6_1$/RTP0$_1$ $P6_0$/RTP0$_0$ — RTP
When pulse mode 1 is selected	$P6_7$/RTP1$_3$ $P6_6$/RTP1$_2$ $P6_5$/RTP1$_1$ $P6_4$/RTP1$_0$ $P6_3$/RTP0$_3$ $P6_2$/RTP0$_2$ — Port	$P6_7$/RTP1$_3$ $P6_6$/RTP1$_2$ $P6_5$/RTP1$_1$ $P6_4$/RTP1$_0$ $P6_3$/RTP0$_3$ $P6_2$/RTP0$_2$ — Port	$P6_7$/RTP1$_3$ $P6_6$/RTP1$_2$ $P6_5$/RTP1$_1$ $P6_4$/RTP1$_0$ $P6_3$/RTP0$_3$ $P6_2$/RTP0$_2$ — RTP	$P6_7$/RTP1$_3$ $P6_6$/RTP1$_2$ $P6_5$/RTP1$_1$ $P6_4$/RTP1$_0$ $P6_3$/RTP0$_3$ $P6_2$/RTP0$_2$ — RTP
	$P6_1$/RTP0$_1$ $P6_0$/RTP0$_0$ — Port	$P6_1$/RTP0$_1$ $P6_0$/RTP0$_0$ — RTP	$P6_1$/RTP0$_1$ $P6_0$/RTP0$_0$ — Port	$P6_1$/RTP0$_1$ $P6_0$/RTP0$_0$ — RTP

Port : This functions as a programmable I/O port.
RTP : This functions as a pulse output pin.

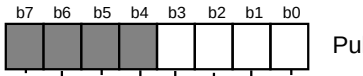
Not announced

Fig. 10.2.1 Structure of real-time output control register

10.2.2 Pulse output data registers 0 and 1

Figure 10.2.2 shows the structure of the pulse output data registers 0 and 1. The bit position of the $RTP0_2$ and $RTP0_3$ pulse output data bits differs according to the pulse mode. Before setting the pulse output data registers 0 and 1, set of the pulse output mode select bit (bit 2 at address 62_{16}).

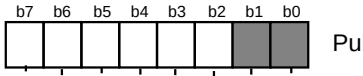
The data written into the pulse output data registers 0 and 1 is output from the corresponding pulse output pins every underflow of Timers A0 and A1.



Pulse output data register 0 (Address $1A_{16}$)

Bit	Bit name	Functions	At reset	RW
0	$RTP0_0$ pulse output data bit	0 : "L" level output 1 : "H" level output	Undefined	WO
1	$RTP0_1$ pulse output data bit		Undefined	WO
2	$RTP0_2$ pulse output data bit (Valid in pulse mode 0)		Undefined	WO
3	$RTP0_3$ pulse output data bit (Valid in pulse mode 0)		Undefined	WO
7 to 4	Nothing is assigned.		Undefined	—

Note: Use the **LDM** or **STA** instruction for writing to this register



Pulse output data register 1 (Address $1C_{16}$)

Bit	Bit name	Functions	At reset	RW
0, 1	Nothing is assigned.		Undefined	—
2	$RTP0_2$ pulse output data bit (Valid in pulse mode 1)	0 : "L" level output 1 : "H" level output	Undefined	WO
3	$RTP0_3$ pulse output data bit (Valid in pulse mode 1)		Undefined	WO
4	$RTP1_0$ pulse output data bit		Undefined	WO
5	$RTP1_1$ pulse output data bit		Undefined	WO
6	$RTP1_2$ pulse output data bit		Undefined	WO
7	$RTP1_3$ pulse output data bit		Undefined	WO

Note: Use the **LDM** or **STA** instruction for writing to this register.

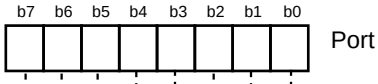
Announced

Fig. 10.2.2 Structure of pulse output data registers 0 and 1

10.2 Block description

10.2.3 Port P6 direction register

The pulse output pins are shared with port P6. When using these pins as pulse output pins of real-time output, set the corresponding bits of the port P6 direction register to “1” to set these ports for the output mode. Figure 10.2.3 shows the relationship between the port P6 direction register and the pulse output pins.



Port P6 direction register (Address 10_{16})

Bit	Bit name	Functions	At reset	RW
0	$RTP0_0$ pin	0 : Input mode 1 : Output mode When using these pins as pulse output pins, set the corresponding bits to “0.”	0	RW
1	$RTP0_1$ pin		0	RW
2	$RTP0_2$ pin		0	RW
3	$RTP0_3$ pin		0	RW
4	$RTP1_0$ pin		0	RW
5	$RTP1_1$ pin		0	RW
6	$RTP1_2$ pin		0	RW
7	$RTP1_3$ pin		0	RW

Note: When setting these bits to “0,” the corresponding pins serve as input port (floated) regardless of the state of the waveform output select bits (bits 0 and 1 at address 62_{16}).

Fig. 10.2.3 Relationship between port P6 direction register and pulse output pins

After reset, the state of the port P6 pins are floated since these pins are in the input mode. The output levels of the pulse output pins are undefined until Timer A0 or A1 underflows first after the data for the timer is written. Because the pulse output data registers 0 and 1 are undefined after reset.
When these conditions should be avoided, follow the procedure “Processing of avoiding undefined output before starting pulse output” in Figures 10.3.1 and 10.3.2.
When reading the port P6 register (address E_{16}), the output values of the real time output pins can be read out.

10.2.4 Timers A0 and A1

The data written into the pulse output registers 0 and 1 is output from the pulse output pins every underflow of Timer A0 or A1. Refer to section **“8.3 Timer mode”** for the setting of Timers A0 and A1.

Not announced

10.3 Setting of real-time output

Figures 10.3.1 to 10.3.3 show an initial setting example for registers relevant to the real-time output. Note that when using interrupts, set up to enable the interrupts. For details, refer to **"CHAPTER 7. INTERRUPTS."**

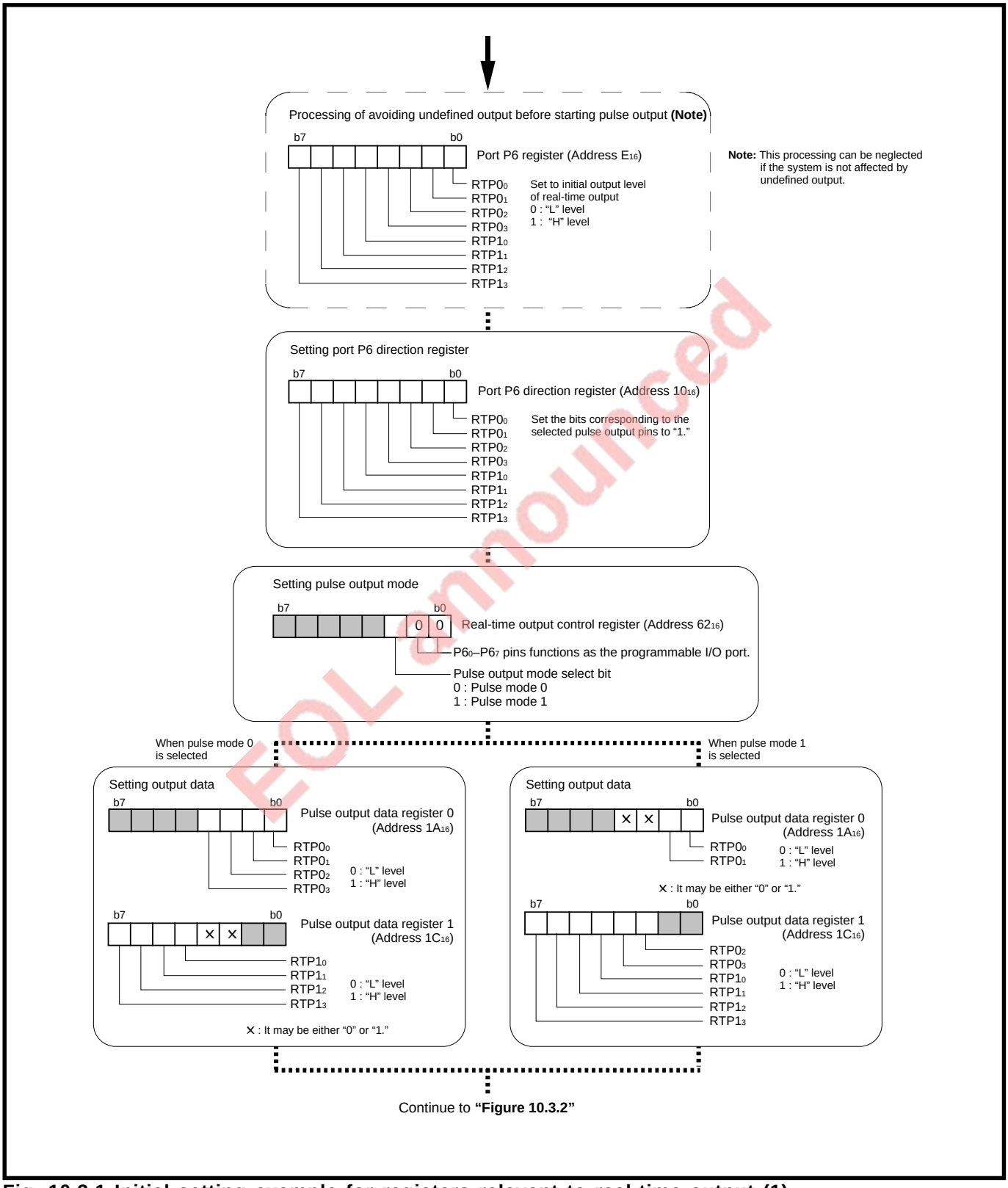
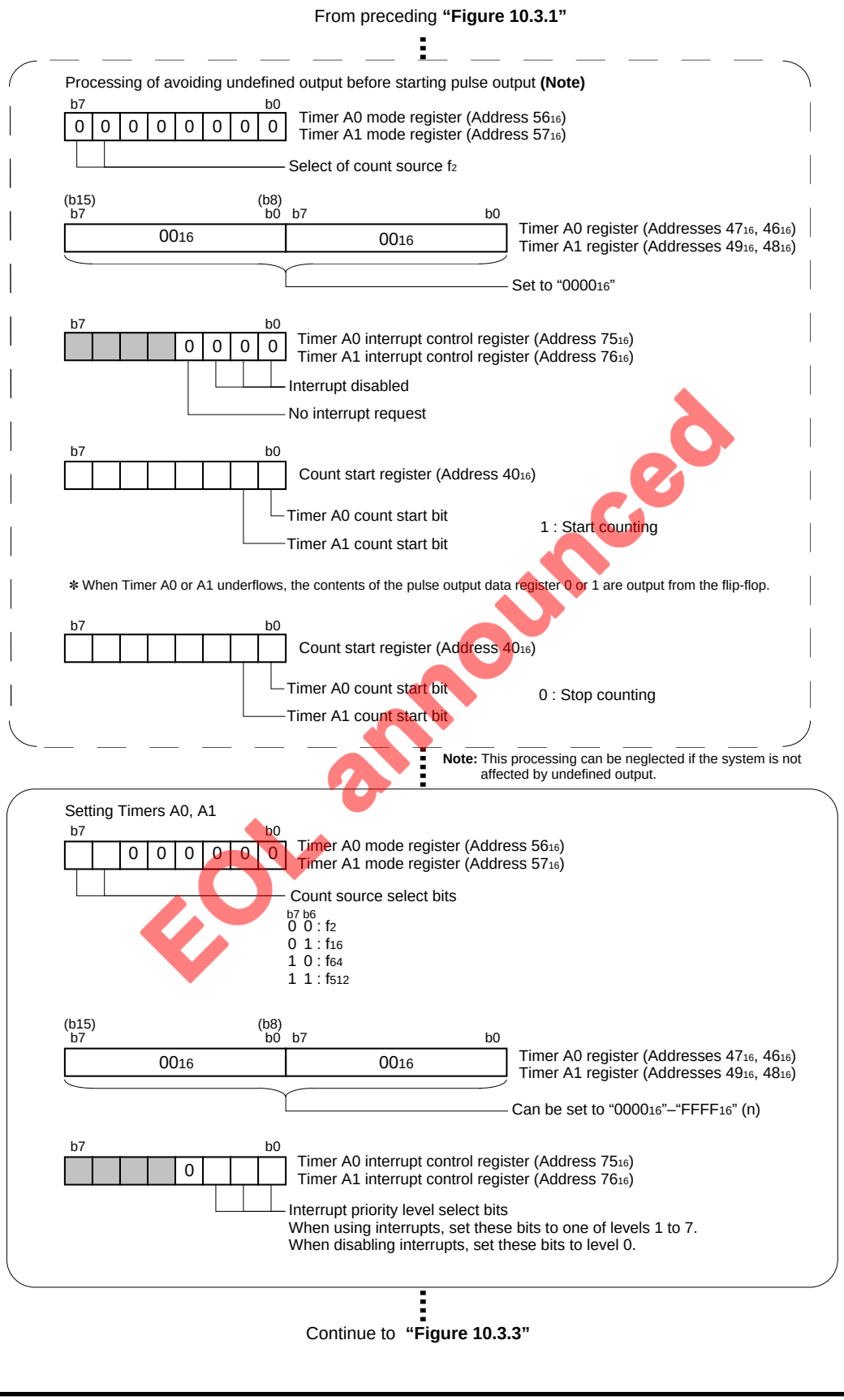


Fig. 10.3.1 Initial setting example for registers relevant to real-time output (1)

EOL announced

10.3 Setting of real-time output



EOL announced

Fig. 10.3.2 Initial setting example for registers relevant to real-time output (2)

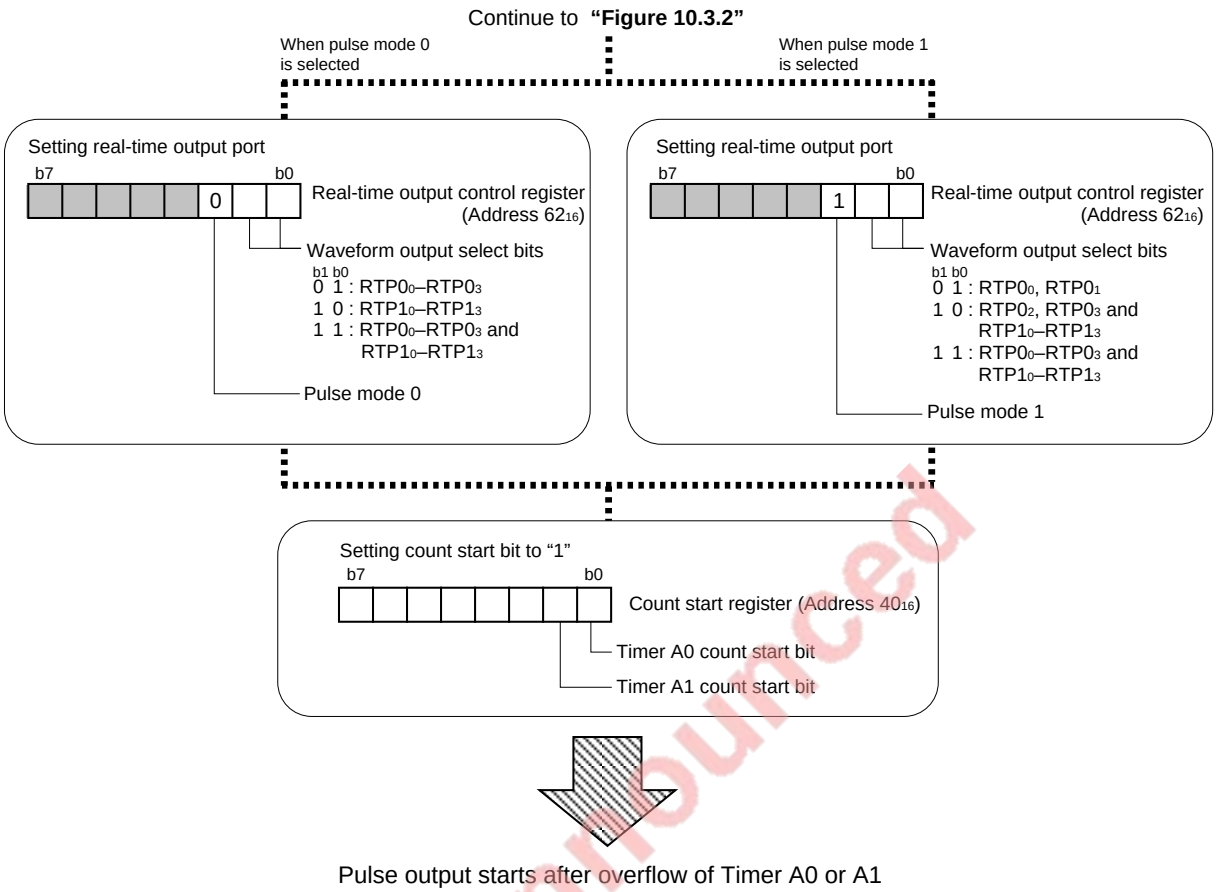


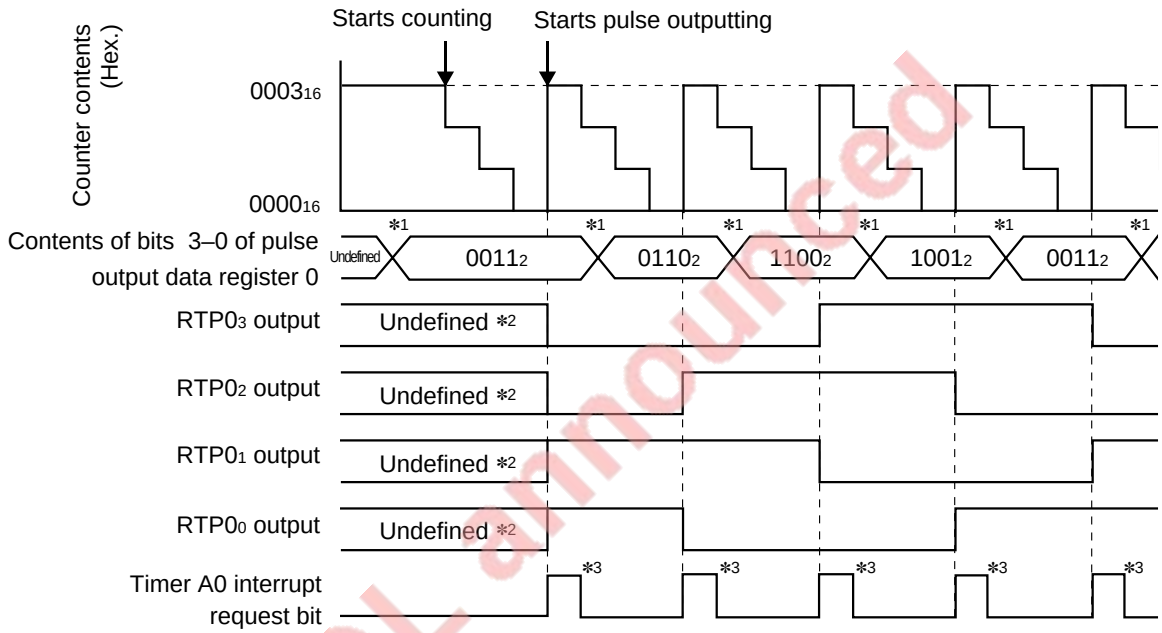
Fig. 10.3.3 Initial setting example for registers relevant to real-time output (3)

EOL announced

10.4 Real-time output operation

① When the timer Ai (i = 0, 1) count start bit is set to “1,” the counter starts counting of the count source.
② The contents of pulse output data register i are output from the pulse output pins at every underflow of Timer Ai. The timer is reloaded with the contents of the reload register and continues counting.
③ The timer Ai interrupt request bit is set to “1” when the counter underflows in ②. The interrupt request bit retains “1” until the interrupt request is accepted or it is cleared by software.
④ Write the next output data into the pulse output data register i during the timer Ai interrupt routine or after the recognition of the timer Ai interrupt request occurrence.

Figure 10.4.1 shows an example of real-time output operation.



∗1 : Written by software
∗2 : To avoid undefined output for these terms, follow the procedure “Processing of avoiding undefined output before starting pulse output” in Figures 10.3.1 and 10.3.2.
∗3 : Cleared to “0” when interrupt request is accepted or cleared by software.

The above figure shows an example of he following conditions:
•Pulse mode 0 selected
•RTP0₀–RTP0₃ selected
•Timer A0 register set value n = 0003_{16}

Fig. 10.4.1 Example of real-time output operation

EOL announced

CHAPTER 11
SERIAL I/O

11.1 Overview
11.2 Block description
11.3 Clock synchronous serial I/O mode
[Precautions for clock synchronous serial I/O mode]
11.4 Clock asynchronous serial I/O (UART) mode

EOL announced

11.1 Overview

Serial I/O consists of 2 channels: UART0 and UART1. They each have a transfer clock generating timer for the exclusive use of them and can operate independently. UART0 and UART1 have the same functions. UARTi (i = 0 and 1) has the following 2 operating modes:

(1) Clock synchronous serial I/O mode
Transmitter and receiver use the same clock as the transfer clock. Transfer data has a length of 8 bits.

(2) Clock asynchronous serial I/O (UART) mode
Transfer rate and transfer data format can arbitrarily be set. The user can select a transfer data length of 7 bits, 8 bits, or 9 bits.

Figure 11.1.1 shows the transfer data formats in each operating mode.

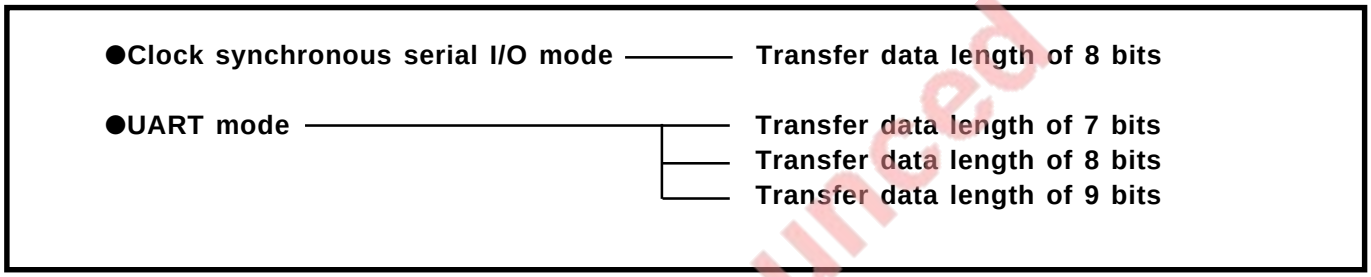


Fig. 11.1.1 Transfer data formats in each operating mode

EOL announced

11.2 Block description

Figure 11.2.1 shows the block diagram of Serial I/O. Registers relevant to Serial I/O are described below.

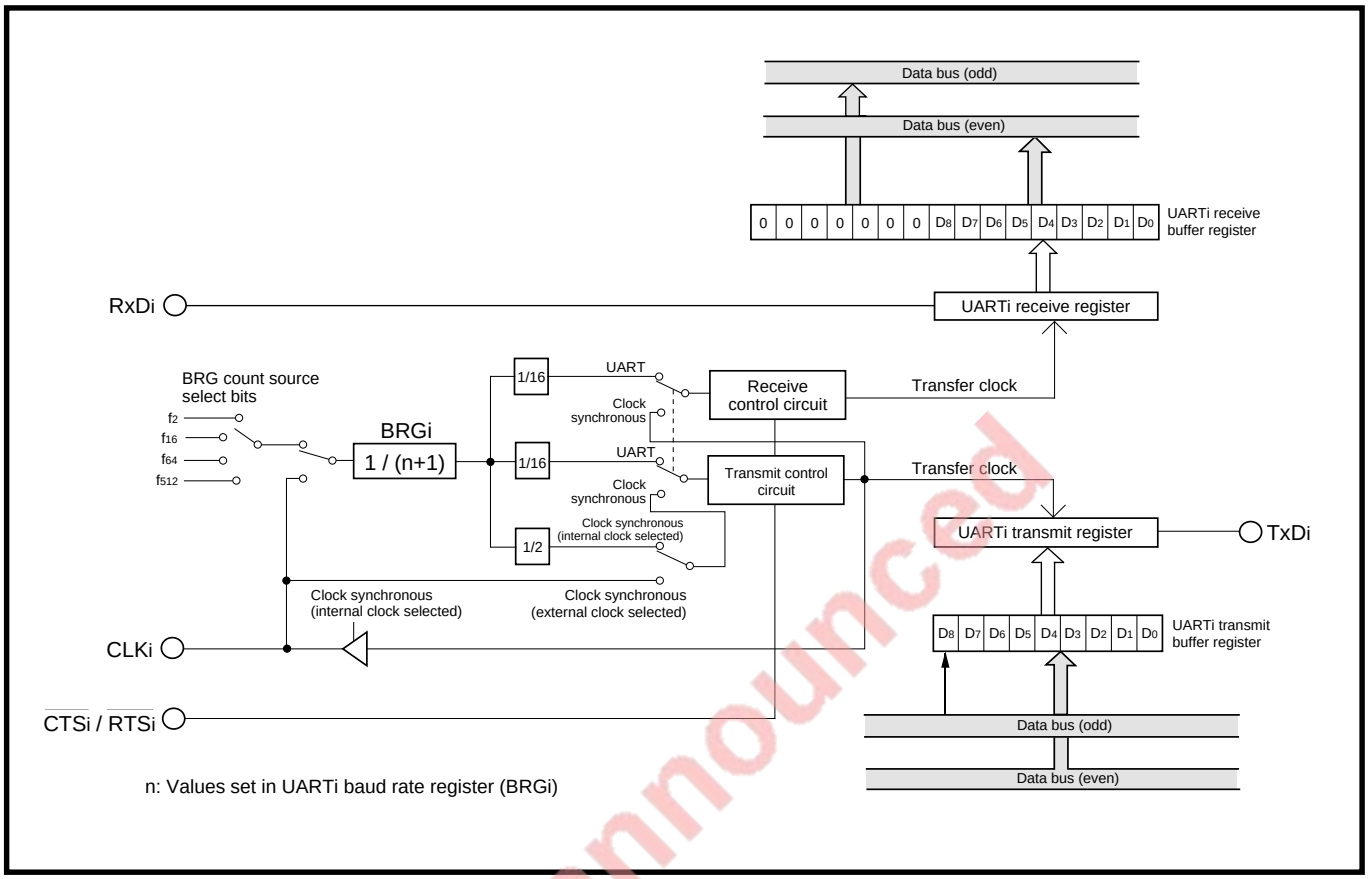


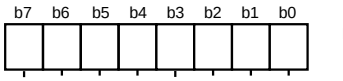
Fig. 11.2.1 Block diagram of Serial I/O

EOL announced

11.2 Block description

11.2.1 UARTi transmit/receive mode register

Figure 11.2.2 shows the structure of UARTi transmit/receive mode register. The serial I/O mode select bits are used to select a UARTi's operating mode. Bits 4 to 6 are described in section **"11.4.2 Transfer data format**," and bit 7 is done in section "**11.4.8 Sleep mode.**"



UART0 transmit/receive mode register (Address 30_{16})
UART1 transmit/receive mode register (Address 38_{16})

Bit	Bit name	Functions	At reset	RW
0	Serial I/O mode select bits	b2 b1 b0 0 0 0 : Serial I/O disabled (P8 functions as a programmable I/O port.) 0 0 1 : Clock synchronous serial I/O mode 0 1 0 : Do not select. 0 1 1 : Do not select. 1 0 0 : UART mode (Transfer data length = 7 bits) 1 0 1 : UART mode (Transfer data length = 8 bits) 1 1 0 : UART mode (Transfer data length = 9 bits) 1 1 1 : Do not select.	0	RW
1			0	RW
2			0	RW
3	Internal/External clock select bit	0 : Internal clock 1 : External clock	0	RW
4	Stop bit length select bit (Valid in UART mode) (**Note**)	0 : One stop bit 1 : Two stop bits	0	RW
5	Odd/Even parity select bit (Valid in UART mode when parity enable bit is "1") (**Note**)	0 : Odd parity 1 : Even parity	0	RW
6	Parity enable bit (Valid in UART mode) (**Note**)	0 : Parity disabled 1 : Parity enabled	0	RW
7	Sleep select bit (Valid in UART mode) (**Note**)	0 : Sleep mode terminated (Invalid) 1 : Sleep mode selected	0	RW

Note: Bits 4 to 6 are invalid in the clock synchronous serial I/O mode. (They may be either "0" or "1.") Additionally, fix bit 7 to "0."

For announced

Fig. 11.2.2 Structure of UARTi transmit/receive mode register

(1) Internal/External clock select bit (bit 3)

■ Clock synchronous serial I/O mode

By clearing this bit to "0" in order to select an internal clock, the clock which is selected with the BRG count source select bits (bits 0 and 1 at addresses 34_{16}, $3C_{16}$) becomes the count source of the BRGi (described later). The BRGi's output divided by 2 becomes the transfer clock. Additionally, the transfer clock is output from the CLK_i pin.

By setting this bit to "1" in order to select an external clock, the clock input to the CLK_i pin becomes the transfer clock.

■ UART mode

By clearing this bit to "0" in order to select an internal clock, the clock which is selected with the BRG count source select bits (bits 0 and 1 at addresses 34_{16}, $3C_{16}$) becomes the count source of the BRGi (described later). Then, the CLK_i pin functions as a programmable I/O port.

By setting this bit to "1" in order to select an external clock, the clock input to the CLK_i pin becomes the count source of BRGi.

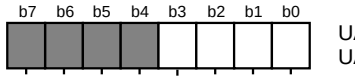
Always in the UART mode, the BRGi's output divided by 16 becomes the transfer clock.

EOL announced

11.2 Block description

11.2.2 UARTi transmit/receive control register 0

Figure 11.2.3 shows the structure of UARTi transmit/receive control register 0. For bits 0 and 1, refer to section **"11.2.1 (1) Internal/External clock select bit (bit 3)."**



UART0 transmit/receive control register 0 (Address 34_{16})
UART1 transmit/receive control register 0 (Address $3C_{16}$)

Bit	Bit name	Functions	At reset	RW
0	BRG count source select bits	b1 b0 0 0 : f_2 0 1 : f_{16} 1 0 : f_{64} 1 1 : f_{512}	0	RW
1			0	RW
2	$\overline{CTS}/\overline{RTS}$ select bit	0 : $\overline{CTS}$ function selected 1 : RTS function selected	0	RW
3	Transmit register empty flag	0 : Data present in transmit register (During transmission) 1 : No data present in transmit register (Transmission completed)	1	RO
7 to 4	Nothing is assigned.		Undefined	—

Fig. 11.2.3 Structure of UARTi transmit/receive control register 0

(1) $\overline{CTS}/\overline{RTS}$ select bit (bit 2)

By clearing this bit to "0" in order to select the $\overline{CTS}$ function, pins $P8_0$ and $P8_4$ function as $\overline{CTS}$ input pins, and the input signal of "L" level to these pins becomes one of the transmission conditions.

By setting this bit to "1" in order to select the $\overline{RTS}$ function, pins $P8_0$ and $P8_4$ become $\overline{RTS}$ output pins. When the receive enable bit (bit 2 at addresses 35_{16}, $3D_{16}$) is "0" (reception disabled), the $\overline{RTS}$ output pin outputs "H" level.

In the clock synchronous serial I/O mode, the output level of the $\overline{RTS}$ pin becomes "L" when reception conditions are satisfied, and it becomes "H" when reception starts. Note that, when an internal clock is selected (bit 3 at addresses 30_{16}, 38_{16} = "0"), the $\overline{RTS}$ output is undefined. Accordingly, do not select the $\overline{RTS}$ function.

In the clock asynchronous serial I/O mode, the output level of the $\overline{RTS}$ pin becomes "L" when the receive enable bit is set to "1." It becomes "H" when reception starts and it becomes "L" when reception is completed.

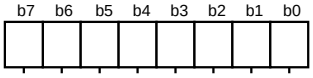
(2) Transmit register empty flag (bit 3)

This flag is cleared to "0" when the UARTi transmit buffer register's contents are transferred to the UARTi transmit register. When transmission is completed and the UARTi transmit register becomes empty, this flag is set to "1."

Not recommended for new design

11.2.3 UARTi transmit/receive control register 1

Figure 11.2.4 shows the structure of UARTi transmit/receive control register 1. For bits 4 to 7, refer to section **"11.3.6 Processing on detecting overrun error"** and **"11.4.7 Processing on detecting error."**



UART0 transmit/receive control register 1 (Address 35_{16})
UART1 transmit/receive control register 1 (Address $3D_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Transmit enable bit	0 : Transmission disabled 1 : Transmission enabled	0	RW
1	Transmit buffer empty flag	0 : Data present in transmit buffer register 1 : No data present in transmit buffer register	1	RO
2	Receive enable bit	0 : Reception disabled 1 : Reception enabled	0	RW
3	Receive complete flag	0 : No data present in receive buffer register 1 : Data present in receive buffer register	0	RO
4	Overrun error flag **(Note 1)**	0 : No overrun error 1 : Overrun error detected	0	RO
5	Framing error flag **(Notes 1, 2)** (Valid in UART mode)	0 : No framing error 1 : Framing error detected	0	RO
6	Parity error flag **(Notes 1, 2)** (Valid in UART mode)	0 : No parity error 1 : Parity error detected	0	RO
7	Error sum flag **(Notes 1, 2)** (Valid in UART mode)	0 : No error 1 : Error detected	0	RO

Notes 1: Bit 4 is cleared to "0" when the receive enable bit is cleared to "0" or when the serial I/O mode select bits (bits 2 to 0 at addresses 30_{16}, 38_{16}) are cleared to "000_2."
Bits 5 and 6 are cleared to "0" when one of the following is performed:
•Clearing the receive enable bit to "0"
•Reading the low-order byte of the UARTi receive buffer register (addresses 36_{16}, $3E_{16}$) out
•Clearing the serial I/O mode select bits (bits 2 to 0 at addresses 30_{16}, 38_{16}) to "000_2"
Bit 7 is cleared to "0" when all of bits 4 to 6 become "0."
2: Bits 5 to 7 are invalid in the clock synchronous serial I/O mode.

EOL announced

Fig. 11.2.4 Structure of UARTi transmit/receive control register 1

11.2 Block description

(1) Transmit enable bit (bit 0)

By setting this bit to “1,” UARTi enters the transmission enable state. By clearing this bit to “0” during transmission, UARTi enters the transmission disable state after the transmission which is in progress at that time is completed.

(2) Transmit buffer empty flag (bit 1)

This flag is set to “1” when data set in the UARTi transmit buffer register is transferred from the UARTi transmit buffer register to the UARTi transmit register. This flag is cleared to “0” when data is set in the UARTi transmit buffer register.

(3) Receive enable bit (bit 2)

By setting this bit to “1,” UARTi enters the reception enable state. By clearing this bit to “0” during reception, UARTi quits the reception immediately and enters the reception disable state.

(4) Receive complete flag (bit 3)

This flag is set to “1” when data is ready in the UARTi receive register and that is transferred to the UARTi receive buffer register (i.e., when reception is completed). This flag is cleared to “0” when one of the following is performed:

- Reading the low-order byte of the UARTi receive buffer register out
- Clearing the receive enable bit (bit 2) to “0”
- Clearing the serial I/O mode select bits (bits 2 to 0 at addresses 30_{16}, 38_{16}) to “000_2.”

EOL announced

11.2.4 UARTi transmit register and UARTi transmit buffer register

Figure 11.2.5 shows the block diagram for the transmitter; Figure 11.2.6 shows the structure of UARTi transmit buffer register.

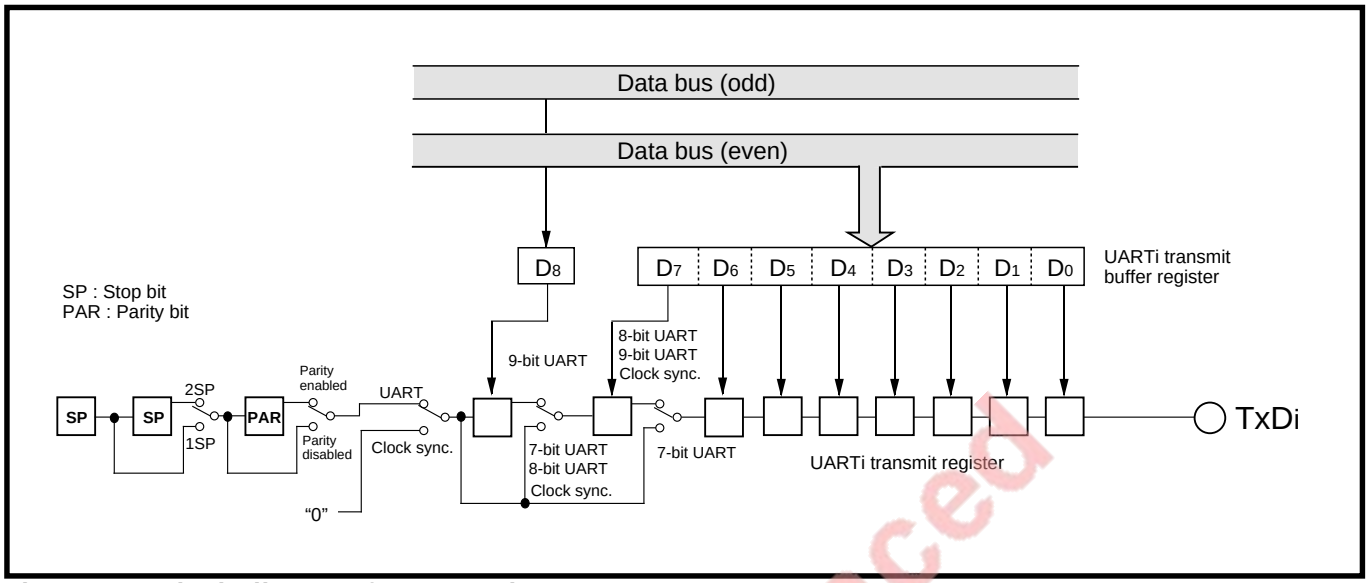
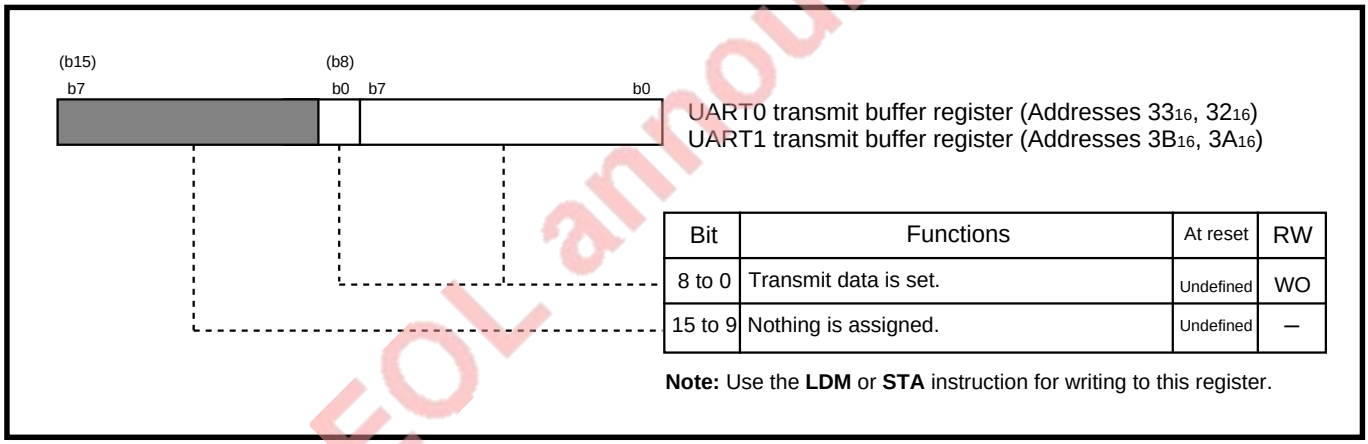


Fig. 11.2.5 Block diagram for transmitter



Bit	Functions	At reset	RW
8 to 0	Transmit data is set.	Undefined	WO
15 to 9	Nothing is assigned.	Undefined	—

Note: Use the **LDM** or **STA** instruction for writing to this register.

Fig. 11.2.6 Structure of UARTi transmit buffer register

EOL announced

11.2 Block description

Transmit data is set into the UARTi transmit buffer register. Set the transmit data into the low-order byte of this register when the microcomputer operates in the clock synchronous serial I/O mode or when a 7-bit or 8-bit length of transfer data is selected in the UART mode. When a 9-bit length of transfer data is selected in the UART mode, set the transmit data into the UARTi transmit buffer register as follows:
•Bit 8 of the transmit data into bit 0 of high-order byte of this register.
•Bits 7 to 0 of the transmit data into the low-order byte of this register.

The transmit data which is set in the UARTi transmit buffer register is transferred to the UARTi transmit register when the transmission conditions are satisfied, and then it is output from the TxD_i pin synchronously with the transfer clock. The UARTi transmit buffer register becomes empty when the data which is set in the UARTi transmit buffer register is transferred to the UARTi transmit register. Accordingly, the user can set the next transmit data.
When quitting the transmission which is in progress and setting the UARTi transmit buffer register again, follow the procedure described bellow:

① Clear the serial I/O mode select bits (bits 2 to 0 at addresses 30_{16}, 38_{16}) to "000_2" (Serial I/O disabled).
② Set the serial I/O mode select bits again.
③ Set the transmit enable bit (bit 0 at addresses 35_{16}, $3D_{16}$) to "1" (transmission enabled) and set transmit data in the UARTi transmit buffer register.

EOL announced

11.2.5 UARTi receive register and UARTi receive buffer register

Figure 11.2.7 shows the block diagram for the receiver; Figure 11.2.8 shows the structure of UARTi receive buffer register.

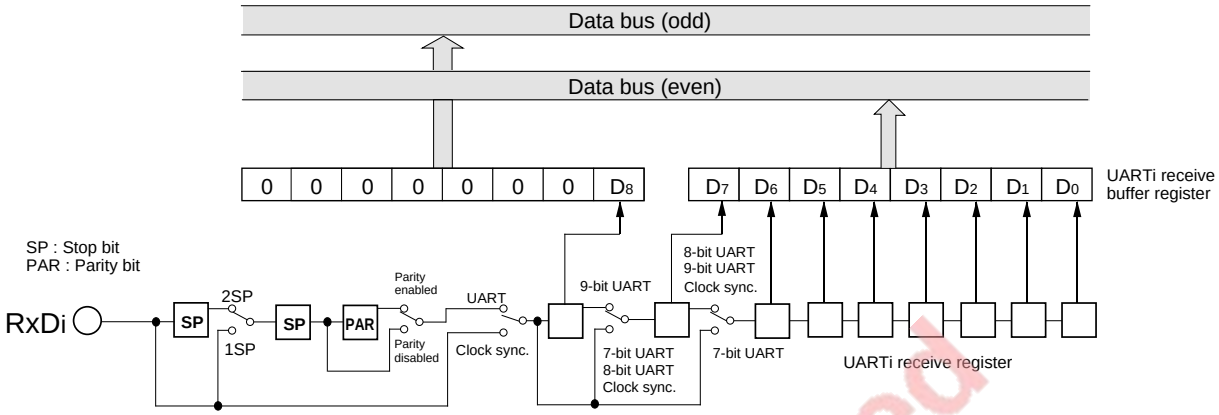
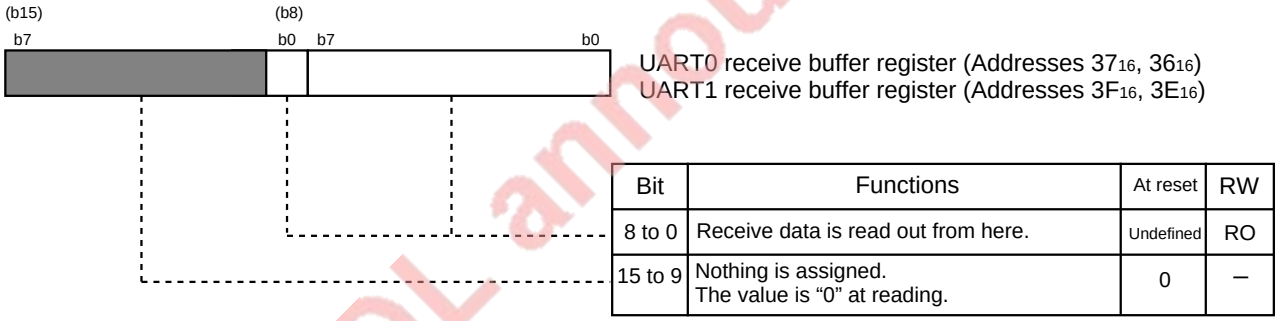


Fig. 11.2.7 Block diagram for receiver



Bit	Functions	At reset	RW
8 to 0	Receive data is read out from here.	Undefined	RO
15 to 9	Nothing is assigned. The value is “0” at reading.	0	–

Fig. 11.2.8 Structure of UARTi receive buffer register

EOL announced

11.2 Block description

The UARTi receive register is used to convert serial data which is input to the RxD_i pin into parallel data. This register takes in the signal input to the RxD_i pin in a unit of 1 bit synchronously with the transfer clock. The UARTi receive buffer register is used to read out receive data. When reception is completed, the receive data which is taken in the UARTi receive register is automatically transferred to the UARTi receive buffer register. Note that the contents of the UARTi receive buffer register is updated when the next data is ready in the UARTi receive register before the data which has been transferred to the UARTi receive buffer register is read out. (i.e., an overrun error occurs.)
The UARTi receive buffer register is initialized by setting the receive enable bit (bit 2 at addresses 35_{16}, $3D_{16}$) to "1" after clearing it to "0."
Figure 11.2.9 shows the contents of the UARTi receive buffer register when reception is completed.

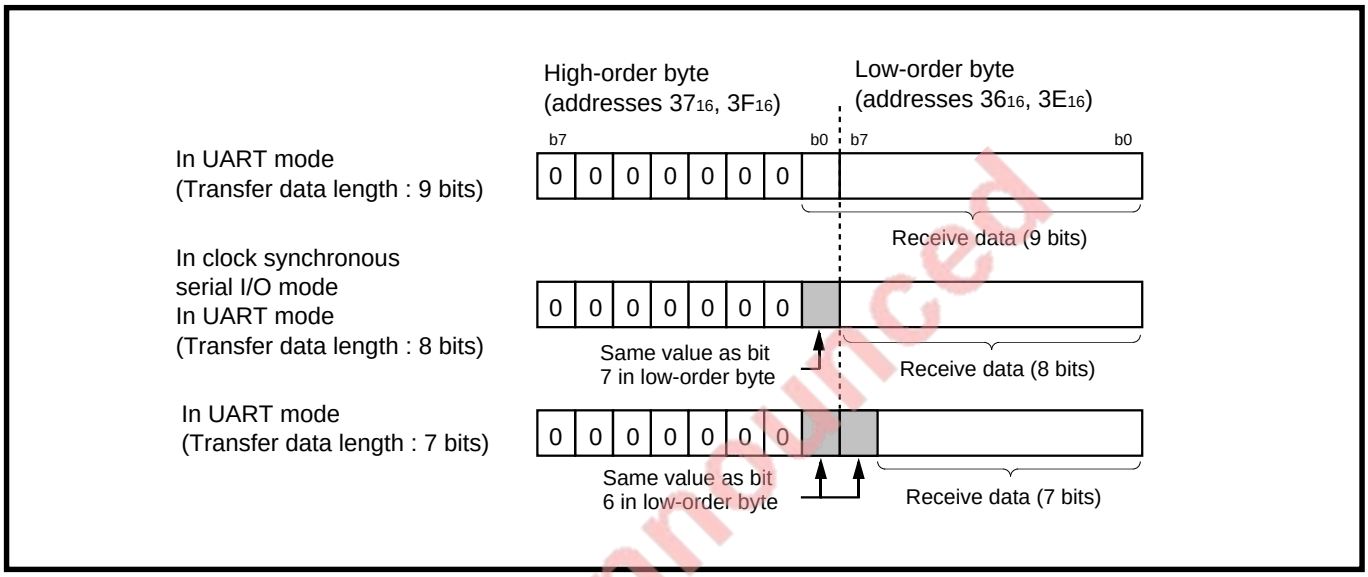


Fig. 11.2.9 Contents of UARTi receive buffer register when reception is completed

EOL announced

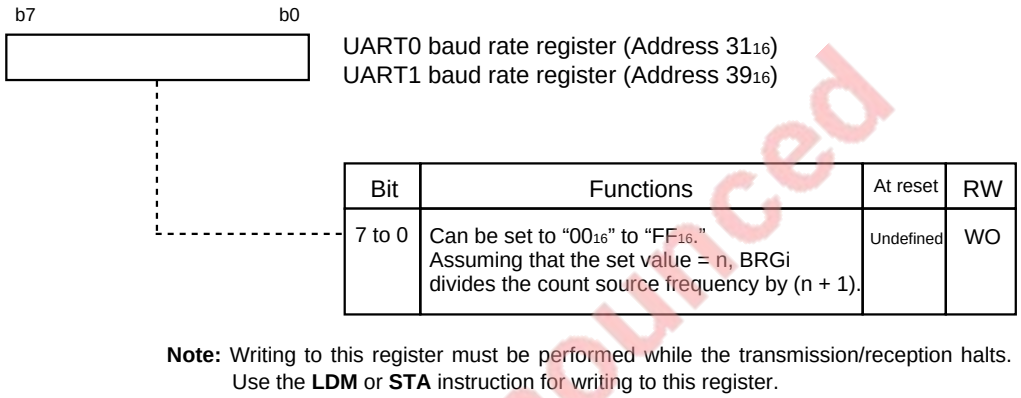
11.2.6 UARTi baud rate register (BRGi)

The UARTi baud rate register (BRGi) is an 8-bit timer exclusively used for UARTi to generate a transfer clock. It has a reload register. Assuming that the value set in the BRGi is "n" (n = "00_{16}" to "FF_{16}"), the BRGi divides the count source frequency by (n + 1).

In the clock synchronous serial I/O mode, the BRGi is valid when an internal clock is selected, and the BRGi's output divided by 2 becomes the transfer clock. In the UART mode, the BRGi is always valid, and the BRGi's output divided by 16 becomes the transfer clock.

The data which is written to the UARTi baud rate register (BRGi) is written to both the timer and the reload register whether transmission/reception is in progress or not. Accordingly, writing to these register must be performed while transmission/reception is stopped.

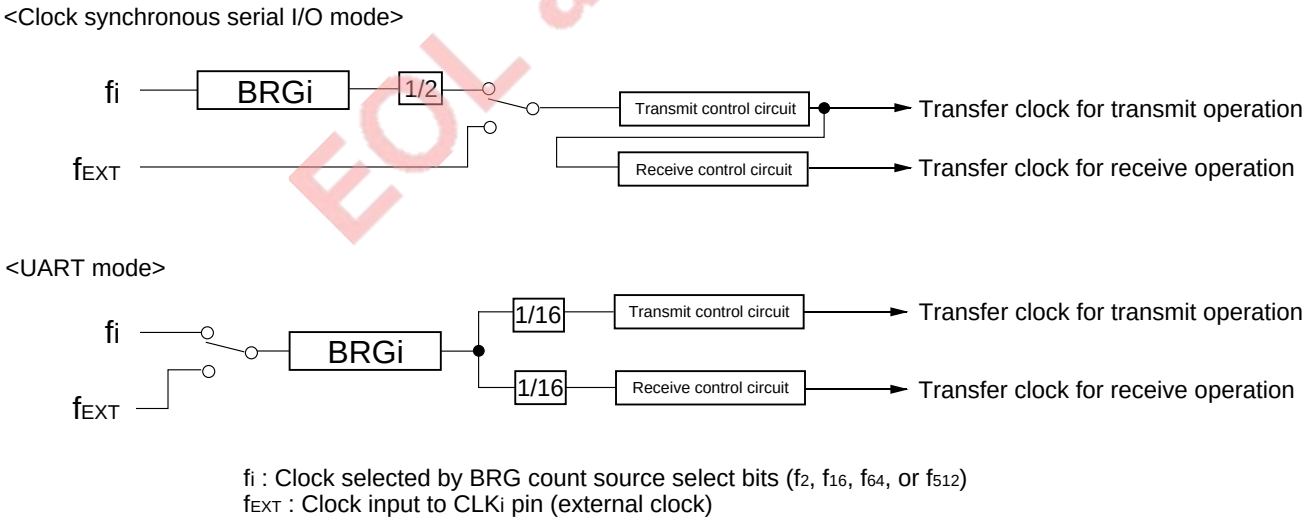
Figure 11.2.10 shows the structure of the UARTi baud rate register (BRGi); Figure 11.2.11 shows the block diagram of transfer clock generating section.



Bit	Functions	At reset	RW
7 to 0	Can be set to "00_{16}" to "FF_{16}." Assuming that the set value = n, BRGi divides the count source frequency by (n + 1).	Undefined	WO

Note: Writing to this register must be performed while the transmission/reception halts. Use the **LDM** or **STA** instruction for writing to this register.

Fig. 11.2.10 Structure of UARTi baud rate register (BRGi)



fi : Clock selected by BRG count source select bits (f_2, f_{16}, f_{64}, or f_{512})
f_{EXT} : Clock input to CLKi pin (external clock)

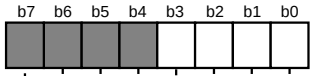
Fig. 11.2.11 Block diagram of transfer clock generating section

EOL announced

11.2 Block description

11.2.7 UARTi transmit interrupt control and UARTi receive interrupt control registers

When using UARTi, 2 types of interrupts, which are UARTi transmit and UARTi receive interrupts, can be used. Each interrupt has its corresponding interrupt control register. Figure 11.2.12 shows the structure of UARTi transmit interrupt control and UARTi receive interrupt control registers.
For details about interrupts, refer to **"CHAPTER 7. INTERRUPTS."**



UART0 transmit interrupt control register (Address 71_{16})
UART0 receive interrupt control register (Address 72_{16})
UART1 transmit interrupt control register (Address 73_{16})
UART1 receive interrupt control register (Address 74_{16})

Bit	Bit name	Functions	At reset	RW
0	Interrupt priority level select bits	b2 b1 b0 0 0 0 : Level 0 (Interrupt disabled) 0 0 1 : Level 1 (Low level) 0 1 0 : Level 2 0 1 1 : Level 3 1 0 0 : Level 4 1 0 1 : Level 5 1 1 0 : Level 6 1 1 1 : Level 7 (High level)	0	RW
1			0	RW
2			0	RW
3	Interrupt request bit	0 : No interrupt requested 1 : Interrupt requested	0	RW
7 to 4	Nothing is assigned.		Undefined	–

Fig. 11.2.12 Structure of UARTi transmit interrupt control and UARTi receive interrupt control registers

(1) Interrupt priority level select bits (bits 0 to 2)

These bits select a priority level of the UARTi transmit interrupt or UARTi receive interrupt. When using UARTi transmit/receive interrupts, select one of the priority levels (1 to 7). When a UARTi transmit/receive interrupt request occurs, its priority level is compared with the processor interrupt priority level (IPL). The requested interrupt is enabled only when its priority level is higher than the IPL. (However, this applies when the interrupt disable flag (I) = "0.") To disable UARTi transmit/receive interrupts, set these bits to "000_2" (level 0).

(2) Interrupt request bit (bit 3)

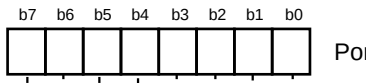
The UARTi transmit interrupt request bit is set to "1" when data is transferred from the UARTi transmit buffer register to the UARTi transmit register. The UARTi receive interrupt request bit is set to "1" when data is transferred from the UARTi receive register to the UARTi receive buffer register. (However, when an overrun error occurs, it does not change.)

Each interrupt request bit is automatically cleared to "0" when its corresponding interrupt request is accepted. This bit can be set to "1" or "0" by software.

Announced

11.2.8 Port P8 direction register

I/O pins of UARTi are multiplexed with port P8. When using pins $P8_2$ and $P8_6$ as serial data input pins (RxD_i), set the corresponding bits of the port P8 direction register to "0" to set these pins for the input mode. When using pins $P8_0$, $P8_1$, $P8_3$ to $P8_5$ and $P8_7$ as I/O pins ($\overline{CTS_i}/\overline{RTS_i}$, CLK_i, TxD_i) of UARTi, these pins are forcibly set as I/O pins of UARTi regardless of the port P8 direction register's contents. Figure 11.2.13 shows the relationship between the port P8 direction register and UARTi's I/O pins. For details, refer to the description of each operating mode.



Port P8 direction register (Address 14_{16})

Bit	Corresponding pin	Functions	At reset	RW
0	$\overline{CTS_0}/\overline{RTS_0}$ pin	0 : Input mode 1 : Output mode When using pins $P8_2$ and $P8_6$ as serial data input pins (RxD_0, RxD_1), set the corresponding bits to "0."	0	RW
1	CLK_0 pin		0	RW
2	RxD_0 pin		0	RW
3	TxD_0 pin		0	RW
4	$\overline{CTS_1}/\overline{RTS_1}$ pin		0	RW
5	CLK_1 pin		0	RW
6	RxD_1 pin		0	RW
7	TxD_1 pin		0	RW

Fig. 11.2.13 Relationship between port P8 direction register and UARTi's I/O pins

EOL announced

11.3 Clock synchronous serial I/O mode

Table 11.3.1 lists the performance overview in the clock synchronous serial I/O mode, and Table 11.3.2 lists the functions of I/O pins in this mode.

Table 11.3.1 Performance overview in clock synchronous serial I/O mode

Item		Functions
Transfer data format		Transfer data has a length of 8 bits. LSB first
Transfer rate	When selecting internal clock	BRGi's output divided by 2
	When selecting external clock	Maximum 5 Mbps
Transmit/Receive control		$\overline{CTS}$ function or $\overline{RTS}$ function can be selected by software.

Table 11.3.2 Functions of I/O pins in clock synchronous serial I/O mode

Pin name	Functions	Method of selection
TxD_i ($P8_3$, $P8_7$) **(Note)**	Serial data output	—————————— (Dummy data is output when performing only reception.)
RxD_i ($P8_2$, $P8_6$)	Serial data input	Port P8 direction register[*1]'s corresponding bit = "0" (Can be used as an I/O port when performing only transmission.)
CLK_i ($P8_1$, $P8_5$)	Transfer clock output	Internal/External clock select bit[*2] = "0"
	Transfer clock input	Internal/External clock select bit = "1"
$\overline{CTS}_i/\overline{RTS}_i$ ($P8_0$, $P8_4$)	$\overline{CTS}$ input	$\overline{CTS}/\overline{RTS}$ select bit[*3] = "0"
	$\overline{RTS}$ output	$\overline{CTS}/\overline{RTS}$ select bit = "1"

Port P8 direction register[*1]: address 14_{16}
Internal/External clock select bit[*2]: bit 3 at addresses 30_{16}, 38_{16}
$\overline{CTS}/\overline{RTS}$ select bit[*3]: bit 2 at addresses 34_{16}, $3C_{16}$

Note: The TxD_i pin outputs "H" level until transmission starts after UARTi's operating mode is selected.

EOL announced

11.3.1 Transfer clock (Synchronizing clock)

Data transfer is performed synchronously with the transfer clock. For the transfer clock, the user can select whether to generate the transfer clock internally or to input it from the external.
The transfer clock is generated by operation of the transmit control circuit. Accordingly, even when performing only reception, set the transmit enable bit to "1," and set dummy data in the UARTi transmit buffer register in order to make the transmit control circuit active.

(1) Internal generation of transfer clock

The count source selected with the BRG count source select bits is divided by the BRGi, and the BRGi output is further divided by 2. This is the transfer clock. The transfer clock is output from the CLK_i pin.

[Setting for relevant registers]

- Select an internal clock (bit 3 at addresses 30_{16}, 38_{16} = "0").
- Select the BRGi's count source (bits 0 and 1 at addresses 34_{16}, $3C_{16}$)
- Set "division value – 1" (= n; 00_{16} to FF_{16}) to the BRGi (addresses 31_{16}, 39_{16}).

$$\text{Transfer clock's frequency} = \frac{f_i}{2\ (n+1)}$$

f_i: Frequency of BRGi's count source (f_2, f_{16}, f_{64}, f_{512})

- Enable transmission (bit 0 at addresses 35_{16}, $3D_{16}$ = "1").
- Set data to the UARTi transmit buffer register (addresses 32_{16}, $3A_{16}$)

[Pin's state]

- A transfer clock is output from the CLK_i pin.
- Serial data is output from the TxDi pin. (Dummy data is output when performing only reception.)

(2) Input of transfer clock from the external

A clock input from the CLK_i pin is the transfer clock.

[Setting for relevant registers]

- Select an external clock (bit 3 at addresses 30_{16}, 38_{16} = "1").
- Enable transmission (bit 0 at addresses 35_{16}, $3D_{16}$ = "1").
- Set data to the UARTi transmit buffer register (addresses 32_{16}, $3A_{16}$).

[Pin's state]

- A transfer clock is input from the CLK_i pin.
- Serial data is output from the TxD_i pin. (Dummy data is output when performing only reception.)

EOL announced

11.3 Clock synchronous serial I/O mode

11.3.2 Method of transmission

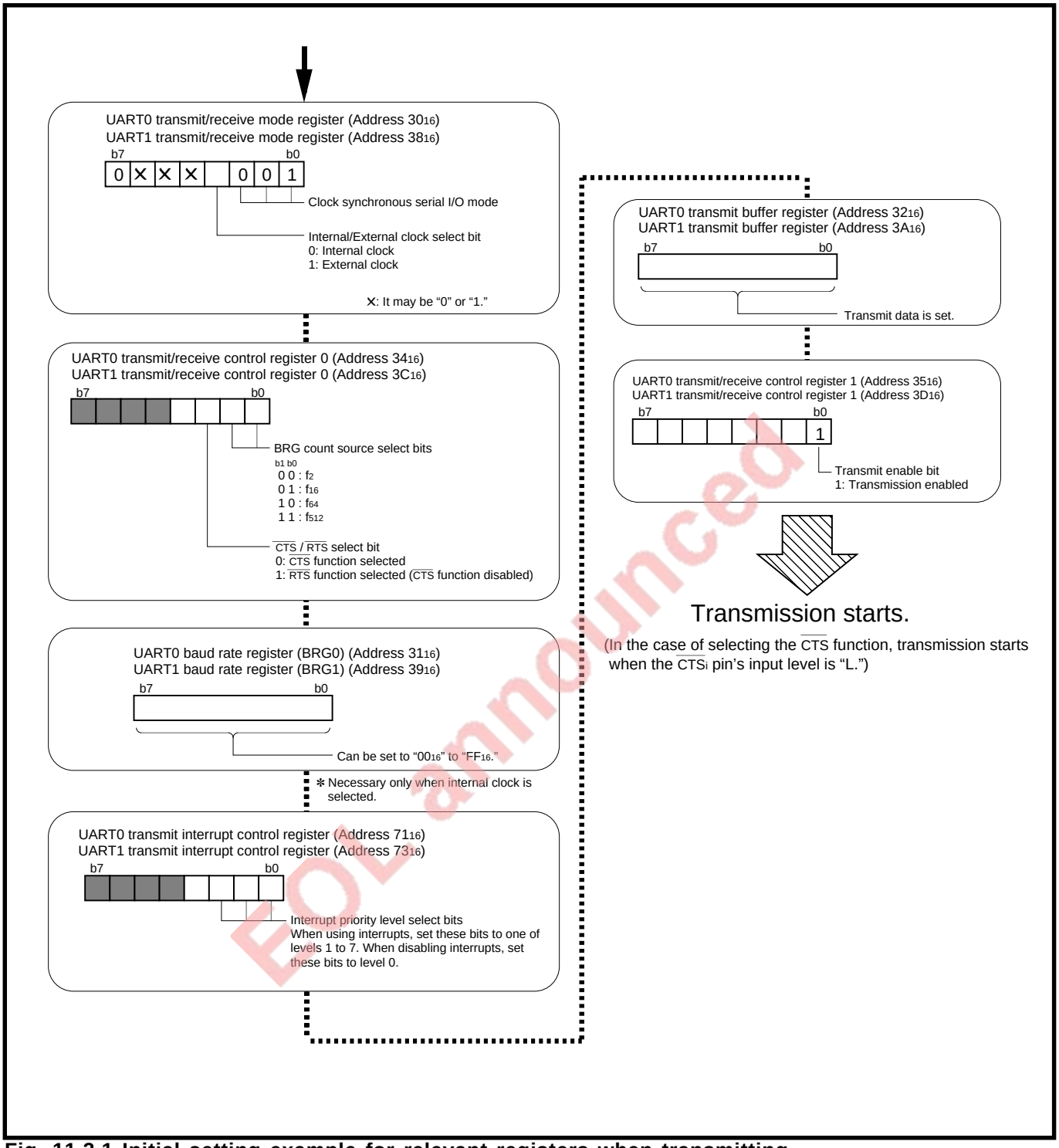
Figure 11.3.1 shows an initial setting example for relevant registers when transmitting. Transmission is started when all of the following conditions (① to ③) are satisfied. When an external clock is selected, satisfy conditions ① to ③ with the following precondition satisfied.

<Precondition>
The CLK_i pin's input is at "H" level
Note: When an internal clock is selected, the above precondition is ignored.

① Transmission is enabled (transmit enable bit = "1").
② Transmit data is present in the UARTi transmit buffer register (transmit buffer empty flag = "0")
③ The $\overline{CTS_i}$ pin's input is at "L" level (when the $\overline{CTS}$ function selected).
Note: When the $\overline{CTS}$ function is not selected, condition ③ is ignored.

By connecting the $\overline{RTS_i}$ pin (receiver side) and $\overline{CTS_i}$ pin (transmitter side), the timing of transmission and that of reception can be matched. For details, refer to section **"11.3.5 Receive operation."**
When using interrupts, it is necessary to set the relevant registers to enable interrupts. For details, refer to **"CHAPTER 7. INTERRUPTS."**
Figure 11.3.2 shows writing data after start of transmission, and Figure 11.3.3 shows detection of transmit completion.

EOL announced



EOL announced

Fig. 11.3.1 Initial setting example for relevant registers when transmitting

11.3 Clock synchronous serial I/O mode

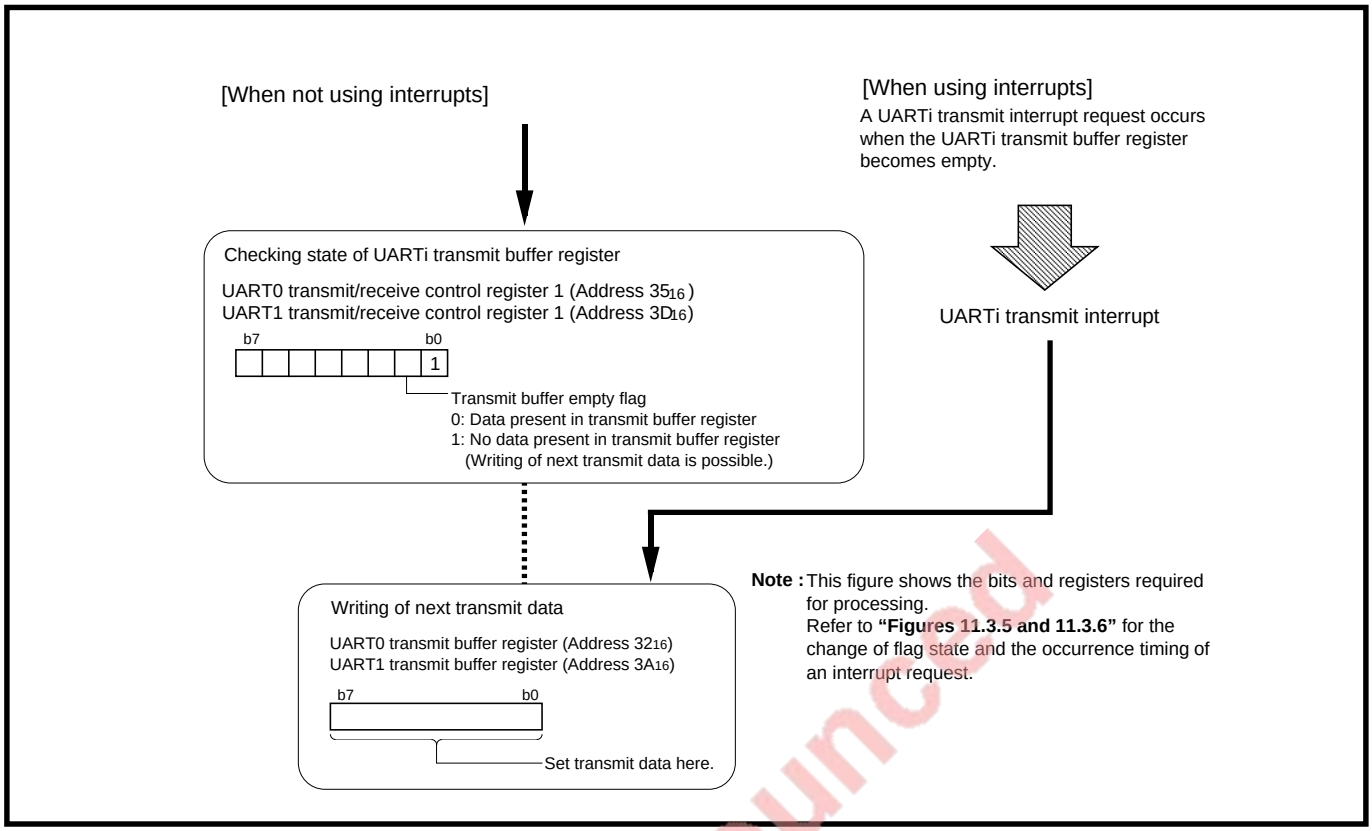


Fig. 11.3.2 Writing data after start of transmission

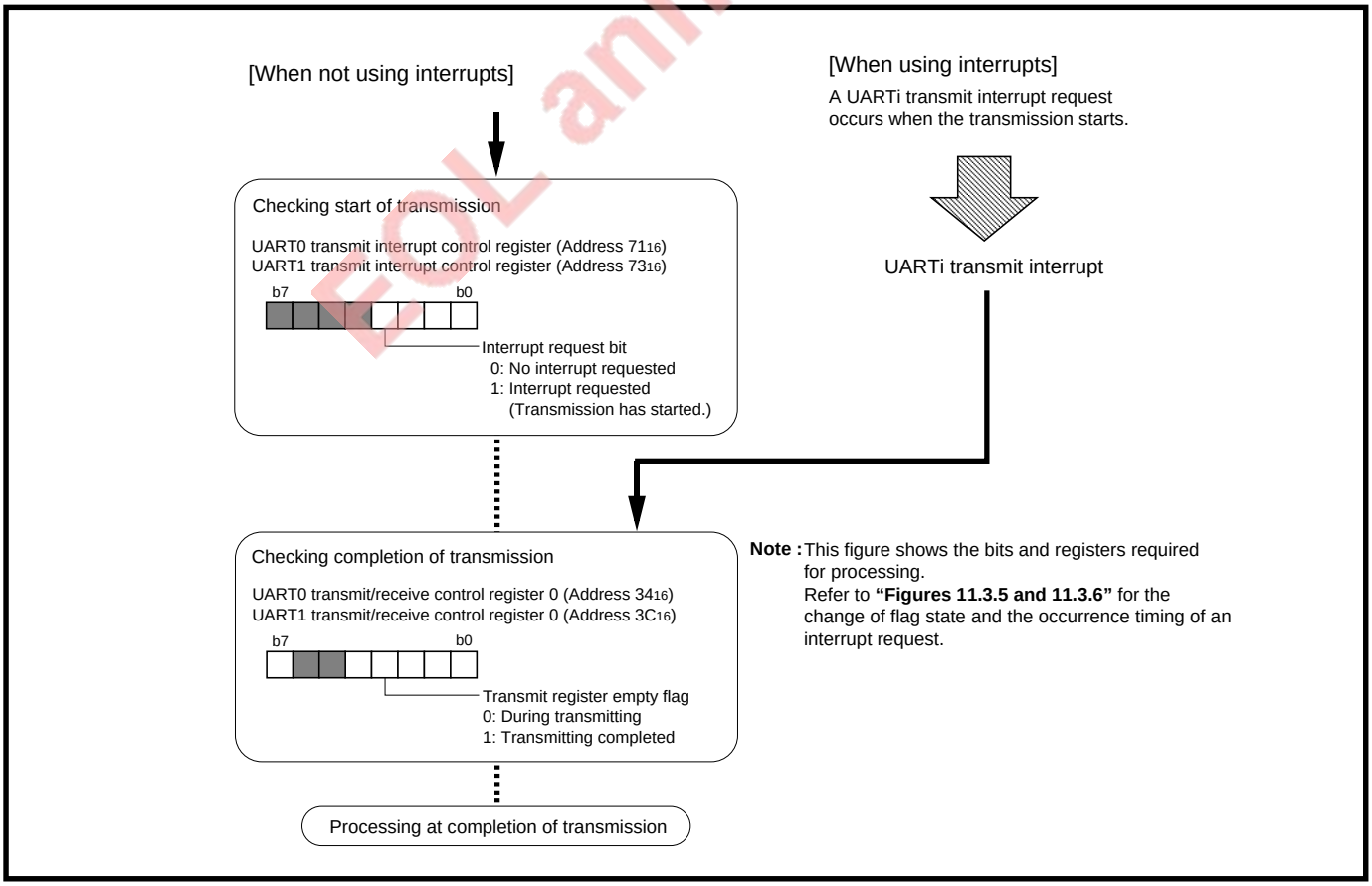


Fig. 11.3.3 Detection of transmit completion

EOL announced

11.3.3 Transmit operation

When the transmit conditions described in section **"11.3.2 Method of transmission"** are satisfied in the case of selecting an internal clock, a transfer clock is generated and the following operations are automatically performed after 1 cycle of the transfer clock has passed. When the transmit conditions are satisfied and the external clock is input to the CLKi pin in the case of selecting an external clock, the following operations are automatically performed.

- The UARTi transmit buffer register's contents are transferred to the UARTi transmit register.
- The transmit buffer empty flag is set to "1."
- The transmit register empty flag is cleared to "0."
- 8 transfer clocks are generated (when an internal clock is selected).
- A UARTi transmit interrupt request occurs, and the interrupt request bit is set to "1."

The transmit operations are described below:

① Data in the UARTi transmit register is transmitted from the TxD_i pin synchronously with the falling edge of the transfer clock.
② This data is transmitted bit by bit sequentially beginning with the least significant bit.
③ When 1-byte data has been transmitted, the transmit register empty flag is set to "1." This indicates the completion of transmission.

Figure 11.3.4 shows the transmit operation.
When an internal clock is selected, when the transmit conditions for the next data are satisfied at completion of the transmission, the transfer clock is generated continuously. Accordingly, when performing transmission continuously, set the next transmit data to the UARTi transmit buffer register during transmission (when the transmit register empty flag = "0"). When the transmit conditions for the next data are not satisfied, the transfer clock stops at "H" level.
Figures 11.3.5 and 11.3.6 show examples of transmit timing.

EOL announced

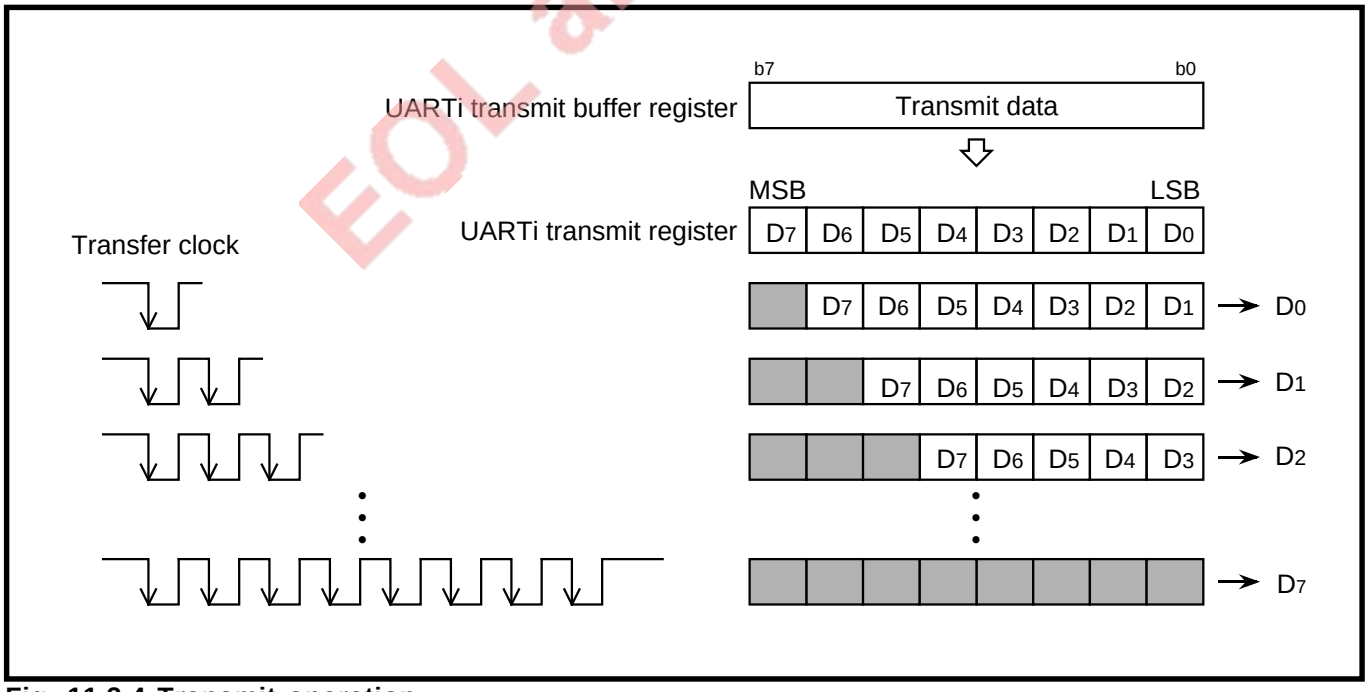


Fig. 11.3.4 Transmit operation

11.3 Clock synchronous serial I/O mode

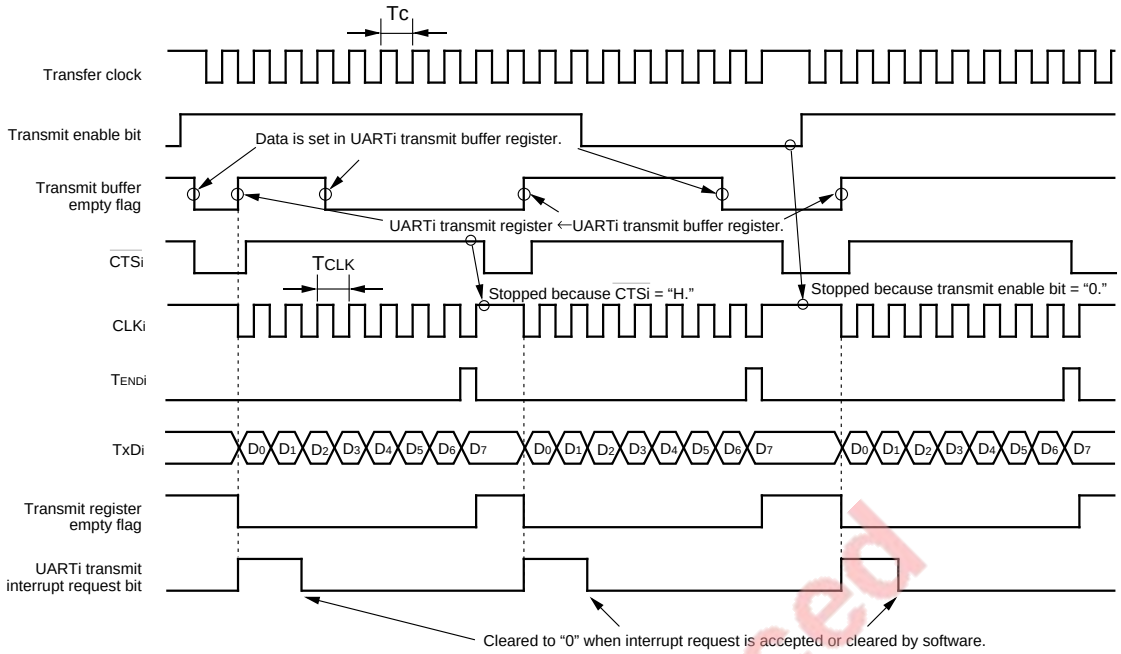


Fig. 11.3.5 Example of transmit timing (when selecting internal clock, selecting $\overline{\text{CTS}}$ function)

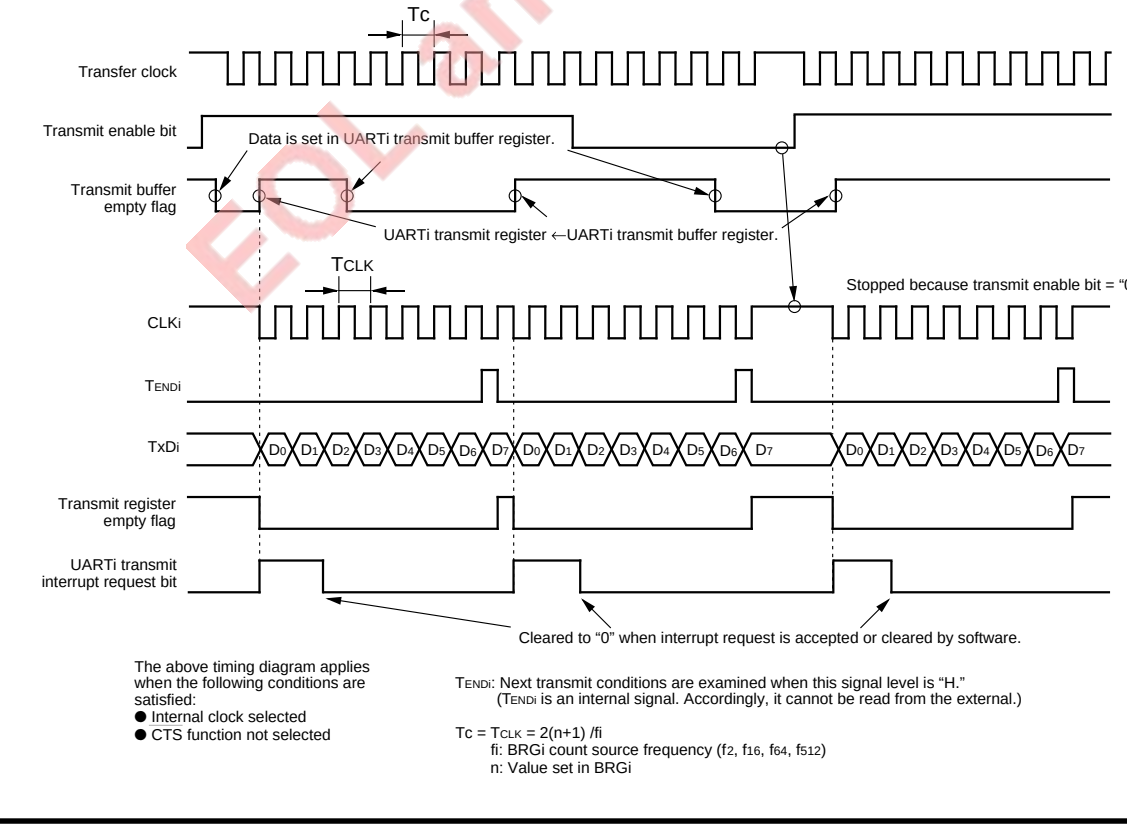


Fig. 11.3.6 Example of transmit timing (when selecting internal clock, not selecting $\overline{\text{CTS}}$ function)

EOL announced

11.3.4 Method of reception

Figures 11.3.7 and 11.3.8 show initial setting examples for relevant registers when receiving. Reception is started when all of the following conditions (① to ③) are satisfied. When an external clock is selected, satisfy conditions ① to ③ with the following precondition satisfied.

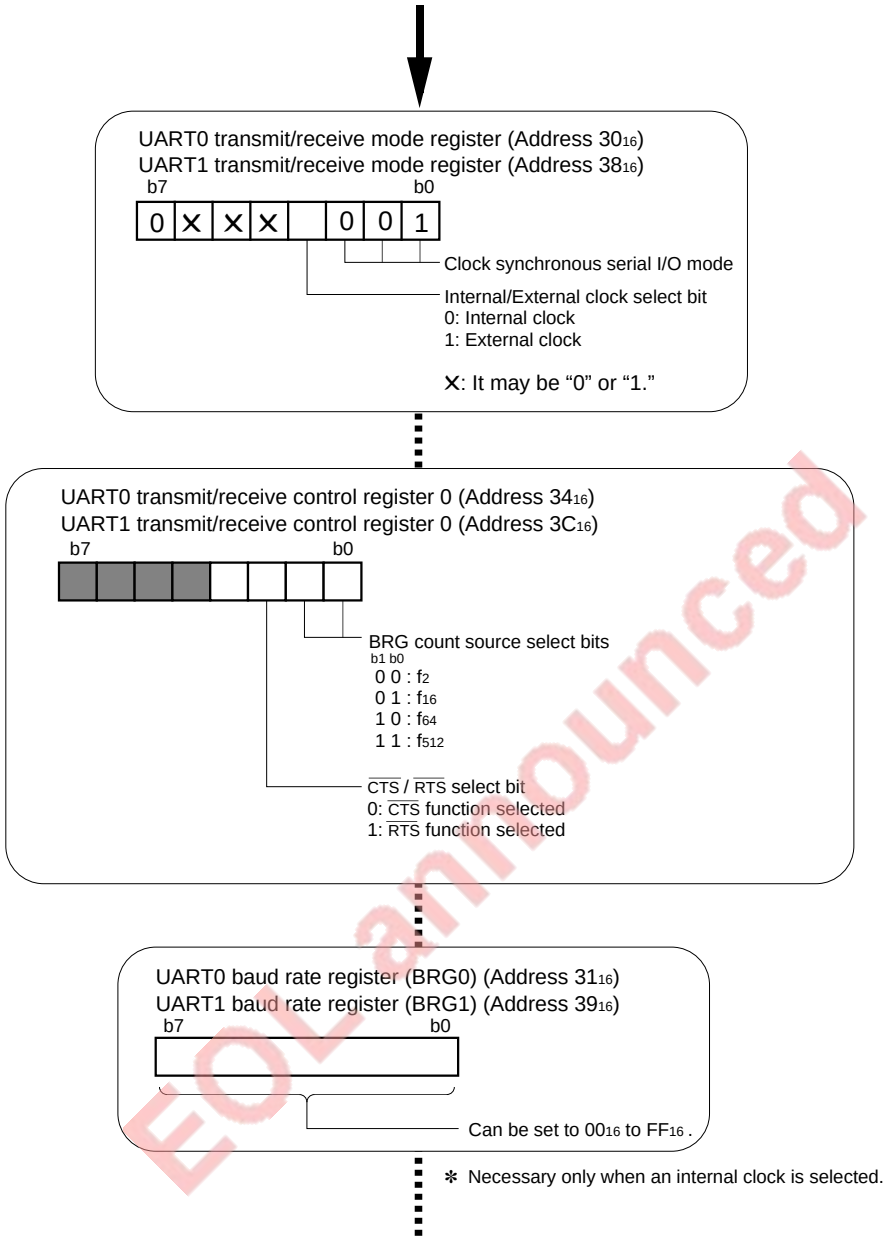
<Precondition>
The CLK_i pin's input is at "H" level.
Note: When an internal clock is selected, the above precondition is ignored.

① Reception is enabled (receive enable bit = "1").
② Transmission is enabled (transmit enable bit = "1").
③ Dummy data is present in the UARTi transmit buffer register (transmit buffer empty flag = "0")

By connecting the $\overline{RTS_i}$ pin (receiver side) and $\overline{CTS_i}$ pin (transmitter side), the timing of transmission and that of reception can be matched. For details, refer to section **"11.3.5 Receive operation."**
When using interrupts, it is necessary to set the relevant registers to enable interrupts. For details, refer to **"CHAPTER 7. INTERRUPTS."**
Figure 11.3.9 shows processing after receive completion.

EOL announced

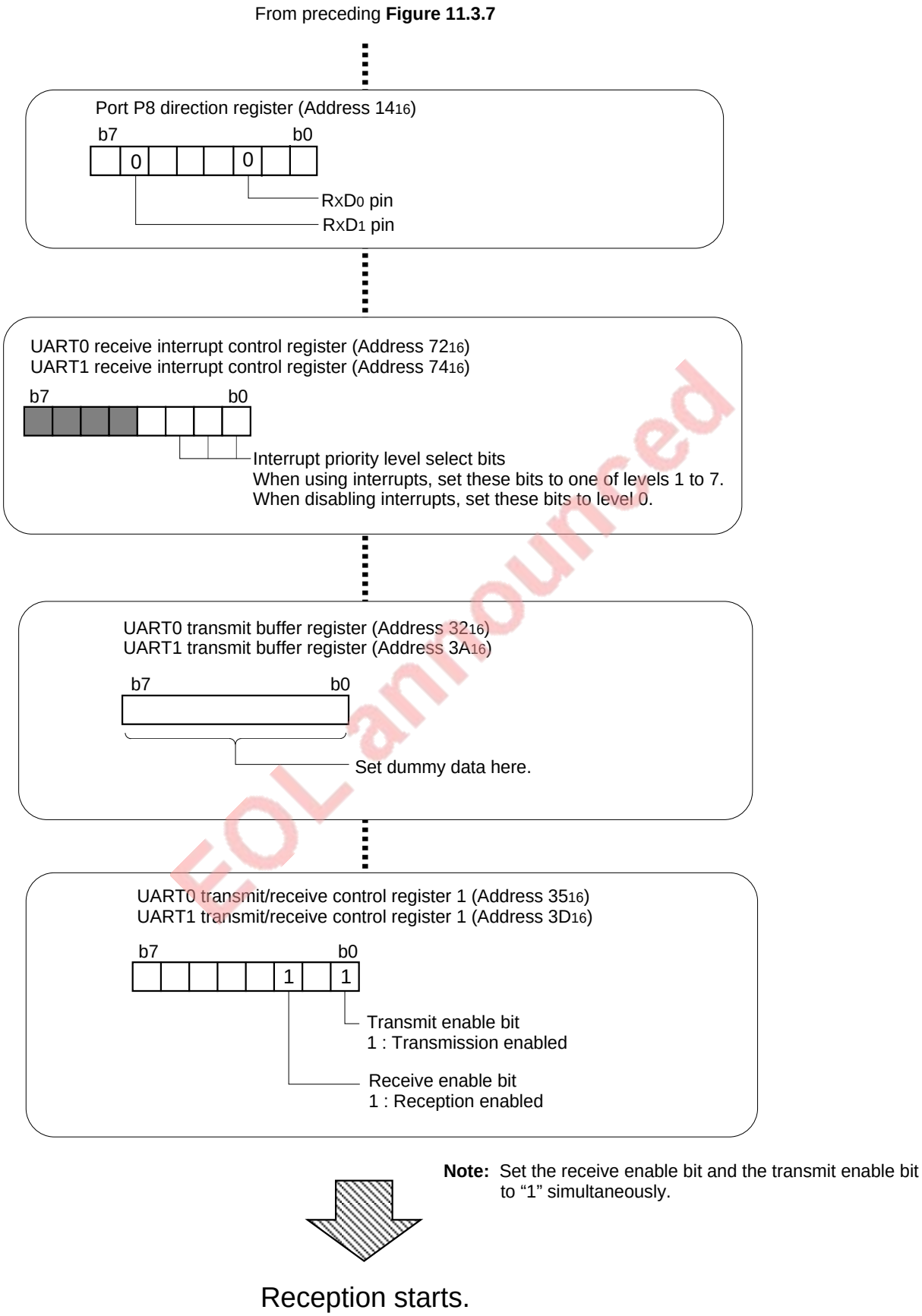
11.3 Clock synchronous serial I/O mode



Continued to **Figure 11.3.8** on next page.

EOL announced

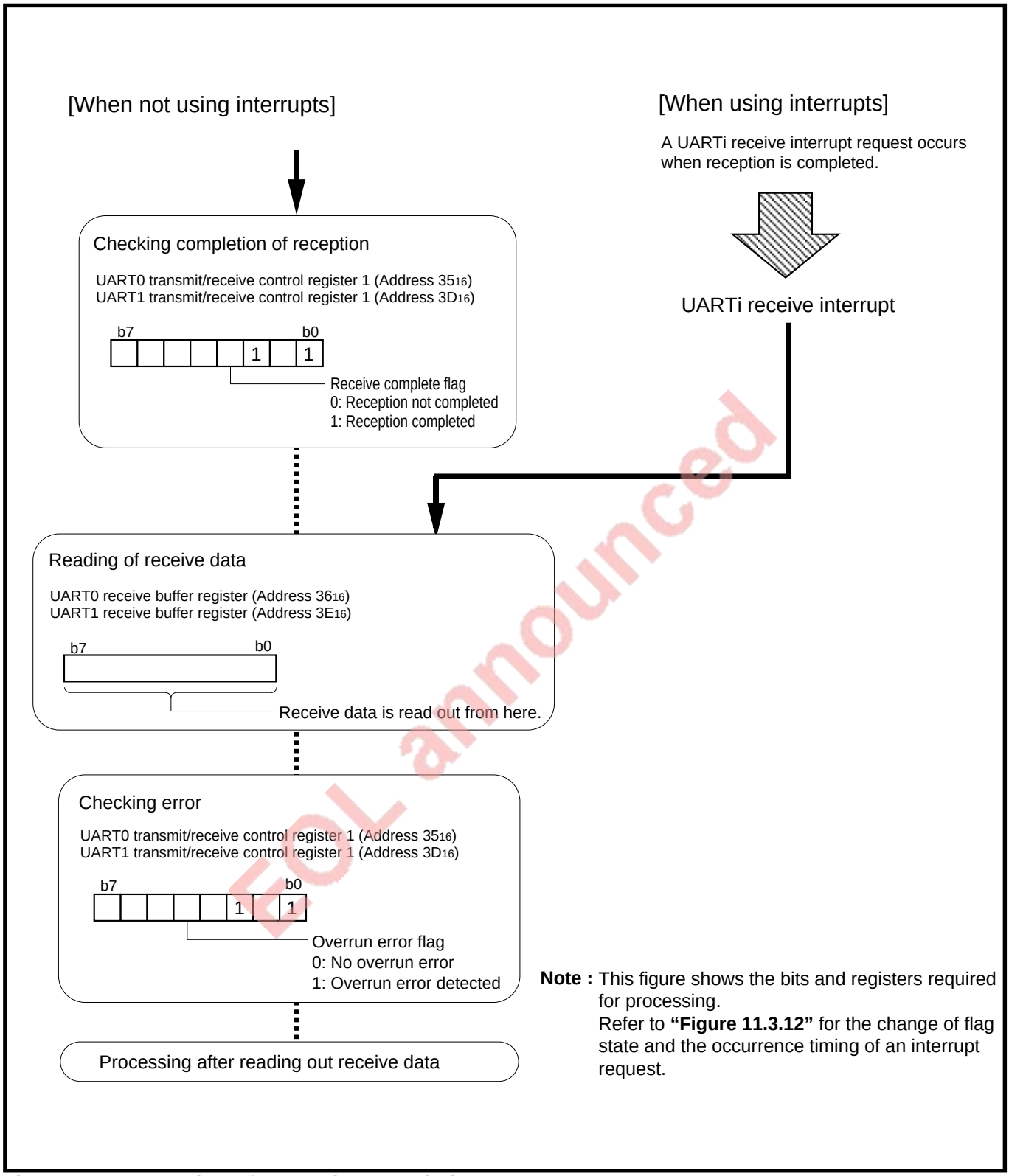
Fig. 11.3.7 Initial setting example for relevant registers when receiving (1)



EOL announced

Fig. 11.3.8 Initial setting example for relevant registers when receiving (2)

11.3 Clock synchronous serial I/O mode



EOL announced

Fig. 11.3.9 Processing after receive completion

11.3.5 Receive operation

In the case of selecting an internal clock, when the receive conditions described in section **"11.3.4 Method of reception"** are satisfied, a transfer clock is generated and the reception is started after 1 cycle of the transfer clock has passed.

In the case of selecting an external clock, when the receive conditions are satisfied, the UARTi enters the receive enable state and reception is started by input of an external clock to the CLKi pin.

In the case of selecting an external clock and the $\overline{\text{RTS}}$ function, when the UARTi enters the receive enable state, the $\overline{\text{RTS}}_i$ pin's output level becomes "L" to inform the transmitter side that reception is enabled. When reception is started, the $\overline{\text{RTS}}_i$ pin's output level becomes "H." Accordingly, by connecting the $\overline{\text{RTS}}_i$ pin to the $\overline{\text{CTS}}_i$ pin of the transmitter side, the timing of transmission and that of reception can be matched. When an internal clock is selected, do not use the $\overline{\text{RTS}}$ function. It is because the $\overline{\text{RTS}}$ output becomes undefined. Figure 11.3.10 shows a connection example.

The receive operations are described below:

① The input signal of the RxD_i pin is taken into the most significant bit of the UARTi receive register synchronously with the rising edge of the transfer clock.
② The contents of the UARTi receive register are shifted by 1 bit to the right.
③ Steps ① and ② are repeated at each rising edge of the transfer clock.
④ When 1-byte data is prepared in the UARTi receive register, the contents of this register are transferred to the UARTi receive buffer register.
⑤ Simultaneously with step ④, the receive complete flag is set to "1," and a UARTi receive interrupt request occurs and its interrupt request bit is set to "1."

The receive complete flag is cleared to "0" when the low-order byte of the UARTi receive buffer register is read out. The $\overline{\text{RTS}}_i$ pin outputs "H" level until the receive conditions are next satisfied (when selecting the $\overline{\text{RTS}}$ function). Figure 11.3.11 shows the receive operation, and Figure 11.3.12 shows an example of receive timing (when selecting an external clock).

EOL announced

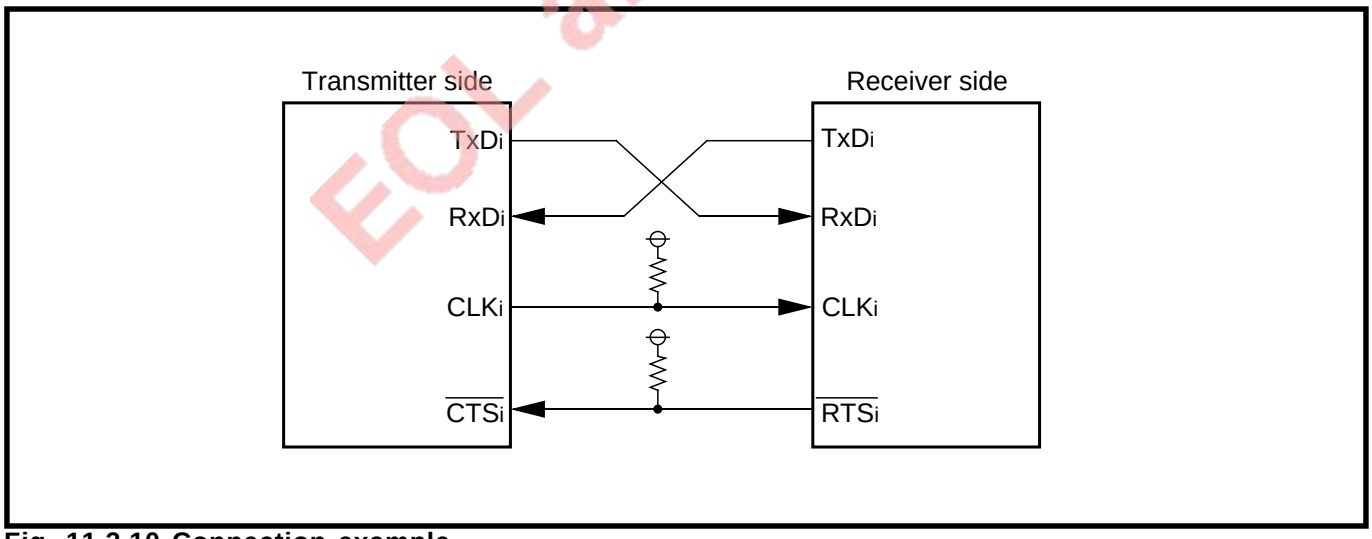


Fig. 11.3.10 Connection example

11.3 Clock synchronous serial I/O mode

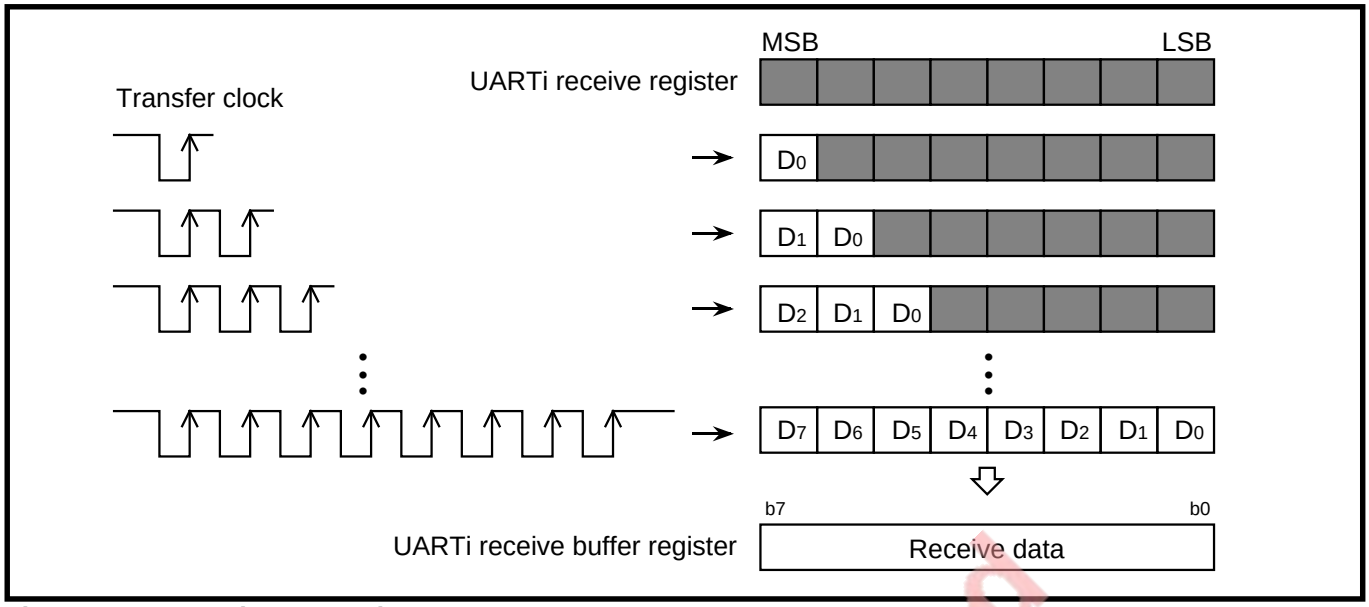


Fig. 11.3.11 Receive operation

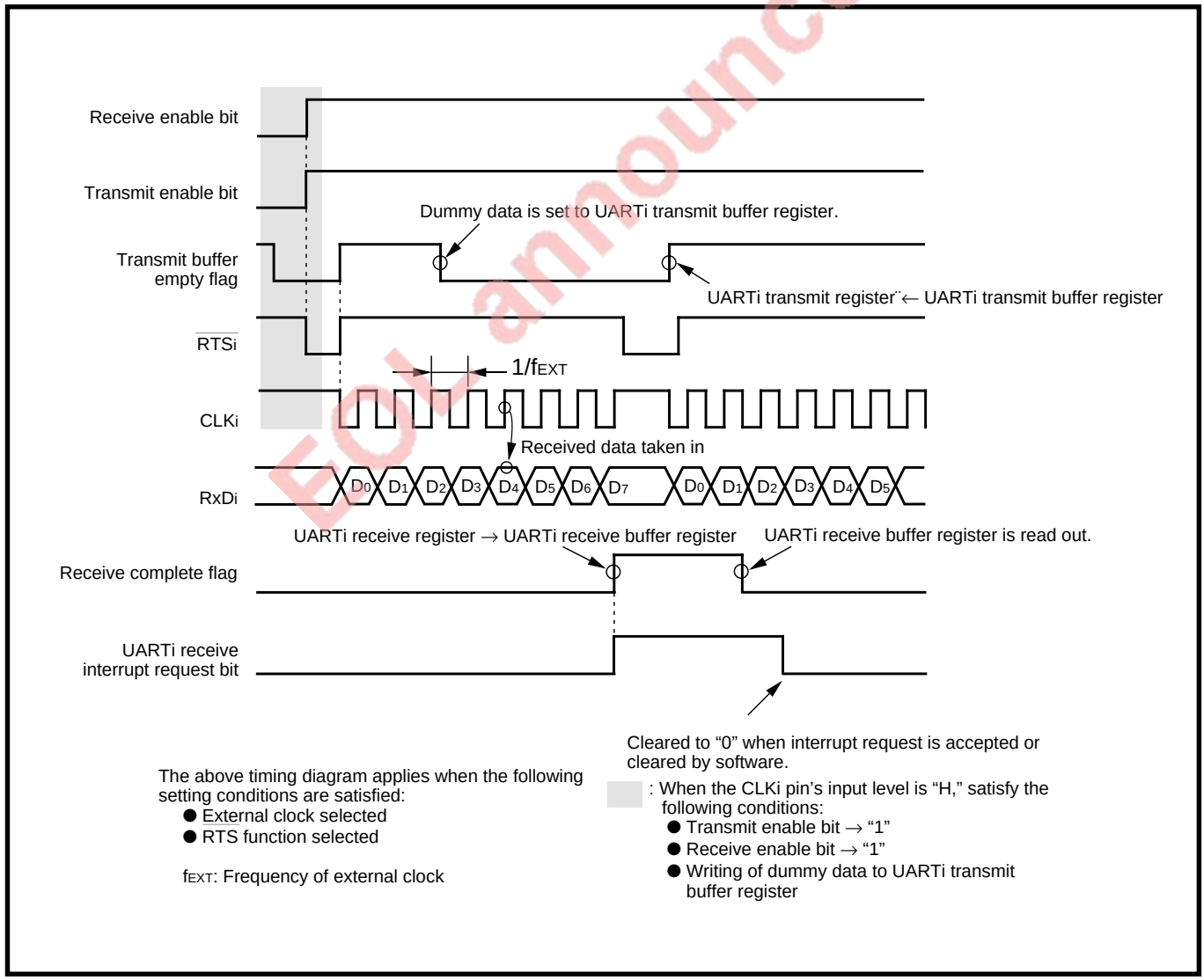


Fig. 11.3.12 Example of receive timing (when selecting external clock)

EOL announced

11.3.6 Processing on detecting overrun error

In the clock synchronous serial I/O mode, an overrun error can be detected.
An overrun error occurs when the next data is prepared in the UARTi receive register with the receive complete flag = "1" (data is present in the UARTi receive buffer register) and next data is transferred to the UARTi receive buffer register, in other words, when the next data is prepared before reading out the contents of the UARTi receive buffer register. When an overrun error occurs, the next receive data is written into the UARTi receive buffer register, and the UARTi receive interrupt request bit is not changed.
An overrun error is detected when data is transferred from the UARTi receive register to the UARTi receive buffer register and the overrun error flag is set to "1." The overrun error flag is cleared to "0" by clearing the serial I/O mode select bits to "000_2" or clearing the receive enable bit to "0."

When an overrun error occurs during reception, initialize the overrun error flag and the UARTi receive buffer register before performing reception again. When it is necessary to perform retransmission owing to an overrun error which occurs in the receiver side, set the UARTi transmit buffer register again before starting transmission again.
The method of initializing the UARTi receive buffer register and that of setting the UARTi transmit buffer register again are described below.

(1) Method of initializing UARTi receive buffer register

① Clear the receive enable bit to "0" (Reception disabled).
② Set the receive enable bit to "1" again (Reception enabled).

(2) Method of setting UARTi transmit buffer register again

① Clear the serial I/O mode select bits to "000_2" (Serial I/O invalid).
② Set the serial I/O mode select bits to "001_2" again.
③ Set the transmit enable bit to "1" (Transmission enabled), and set the transmit data to the UARTi transmit buffer register.

EOL announced

11.3 Clock synchronous serial I/O mode

[Precautions for clock synchronous serial I/O mode]

1. The transfer clock is generated by operation of the transmit control circuit. Accordingly, even when performing only reception, transmit operation (setting for transmission) must be performed. In this case, dummy data is output from the TxD_i pin.

2. When receiving, simultaneously set the receive enable bit and the transmit enable bit to “1.”

3. When receiving data, write dummy data to the low-order byte of the UARTi transmit buffer register for each reception of 1-byte data.

4. When selecting an external clock, satisfy the following 3 conditions with the input to the CLK_i pin = “H” level.

   <When transmitting>
   ① Set the transmit enable bit to “1.”
   ② Write transmit data to the UARTi transmit buffer register.
   ③ Input “L” level to the $\overline{CTS_i}$ pin (when selecting the $\overline{CTS}$ function).

   <When receiving>
   ① Set the receive enable bit to “1.”
   ② Set the transmit enable bit to “1.”
   ③ Write dummy data to the UARTi transmit buffer register.

EOL announced

11.4 Clock asynchronous serial I/O (UART) mode

Table 11.4.1 lists the performance overview in the UART mode, and Table 11.4.2 lists the functions of I/O pins in this mode.

Table 11.4.1 Performance overview in UART mode

Item		Functions
Transfer data format	Start bit	1 bit
	Character bit (Transfer data)	7 bits, 8 bits, or 9 bits
	Parity bit	0 bit or 1 bit (Odd or even can be selected.)
	Stop bit	1 bit or 2 bits
Transfer rate	When selecting internal clock	BRGi's output divided by 16
	When selecting external clock	Maximum 312.5 kbps
Error detection		4 types (Overrun, Framing, Parity, and Summing) Presence of error can be detected only by checking error sum flag.

Table 11.4.2 Functions of I/O pins in UART mode

Pin name	Functions	Method of selection
TxD_i ($P8_3$, $P8_7$) **(Note 1)**	Serial data output	— (Cannot be used as a programmable I/O port even when performing only reception.)
RxD_i ($P8_2$, $P8_6$)	Serial data input	Port P8 direction register*1's corresponding bit = "0" (Can be used as a programmable I/O port when performing only transmission.)
CLK_i ($P8_1$, $P8_5$)	Programmable I/O port	Internal/External clock select bit*2 = "0"
	BRGi's count source input	Internal/External clock select bit = "1"
$\overline{CTS_i}/\overline{RTS_i}$ ($P8_0$, $P8_4$) **(Note 2)**	$\overline{CTS}$ input	$\overline{CTS}/\overline{RTS}$ function select bit*3 = "0"
	$\overline{RTS}$ output	$\overline{CTS}/\overline{RTS}$ function select bit = "1"

Port P8 direction register*1: address 14_{16}
Internal/External clock select bit*2: bit 3 at addresses 30_{16}, 38_{16}
$\overline{CTS}/\overline{RTS}$ select bit*3: bit 2 at addresses 34_{16}, $3C_{16}$

Notes 1: The TxD_i pin outputs "H" level while transmission is not performed after selecting UARTi's operating mode.

2: The $\overline{CTS_i}/\overline{RTS_i}$ pin can be used as an input port when performing only reception and not using the $\overline{RTS}$ function (when selecting $\overline{CTS}$ function).

Eol announced

11.4 Clock asynchronous serial I/O (UART) mode

11.4.1 Transfer rate (Frequency of transfer clock)

The transfer rate is determined by the BRGi (addresses 31_{16}, 39_{16}).

When setting "n" into BRGi, BRGi divides the count source frequency by (n + 1). The BRGi's output is further divided by 16, and the resultant clock becomes the transfer clock. Accordingly, "n" is expressed by the following formula.

$$n = \frac{F}{16 \times B} - 1$$

n: Value set in BRGi (00_{16} to FF_{16})
F: BRGi's count source frequency (Hz)
B: Transfer rate (bps)

An internal clock or an external clock can be selected as the BRGi's count source with the internal/external clock select bit (bit 3 at addresses 30_{16}, 38_{16}). When an internal clock is selected, the clock selected with the BRG count source select bits (bits 0 and 1 at addresses 34_{16}, $3C_{16}$) becomes the BRGi's count source. When an external clock is selected, the clock input to the CLK_i pin becomes the BRGi's count source.

Set the same transfer rate for both transmitter and receiver sides. Tables 11.4.3 and 11.4.4 list the setting examples of transfer rate.

Table 11.4.3 Setting examples of transfer rate (1)

Transfer rate (bps)	$f(X_{IN})$ = 24.576 MHz			$f(X_{IN})$ = 25 MHz		
	BRGi's count source	BRGi's set value : n	Actual time (bps)	BRGi's count source	BRGi's set value : n	Actual time (bps)
300	f_{64}	79 ($4F_{16}$)	300.00	f_{64}	80 (50_{16})	301.41
600	f_{16}	159 ($9F_{16}$)	600.00	f_{16}	162 ($A2_{16}$)	599.12
1200	f_{16}	79 ($4F_{16}$)	1200.00	f_{16}	80 (50_{16})	1205.63
2400	f_{16}	39 (27_{16})	2400.00	f_{16}	40 (28_{16})	2381.86
4800	f_2	159 ($9F_{16}$)	4800.00	f_2	162 ($A2_{16}$)	4792.94
9600	f_2	79 ($4F_{16}$)	9600.00	f_2	80 (50_{16})	9645.06
14400	f_2	52 (34_{16})	14490.57	f_2	53 (35_{16})	14467.59
19200	f_2	39 (27_{16})	19200.00	f_2	40 (28_{16})	19054.58
31250				f_2	24 (18_{16})	31250.00
38400	f_2	19 (13_{16})	38400.00			

Table 11.4.4 Setting examples of transfer rate (2)

Transfer rate (bps)	$f(X_{IN})$ = 22.1184 MHz		
	BRGi's count source	BRGi's set value : n	Actual time (bps)
300	f_{64}	71 (47_{16})	300.00
600	f_{16}	143 ($8F_{16}$)	600.00
1200	f_{16}	71 (47_{16})	1200.00
2400	f_{16}	35 (23_{16})	2400.00
4800	f_2	143 ($8F_{16}$)	4800.00
9600	f_2	71 (47_{16})	9600.00
14400	f_2	47 ($2F_{16}$)	14400.00
19200	f_2	35 (23_{16})	19200.00
28800	f_2	23 (17_{16})	28800.00
31250	f_2	21 (15_{16})	31418.18
38400	f_2	17 (11_{16})	38400.00
57600	f_2	11 ($0B_{16}$)	57600.00
115200	f_2	5 (05_{16})	115200.00
230400	f_2	2 (02_{16})	230400.00

PDL announced

11.4.2 Transfer data format

The transfer data format can be selected from formats shown in Figure 11.4.1. Bits 4 to 6 at addresses 30_{16} and 38_{16} select the transfer data format. (Refer to **"Figure 11.2.2."**) Set the same transfer data format for both transmitter and receiver sides.

Figure 11.4.2 shows an example of transfer data format. Table 11.4.5 lists each bit in transmit data.

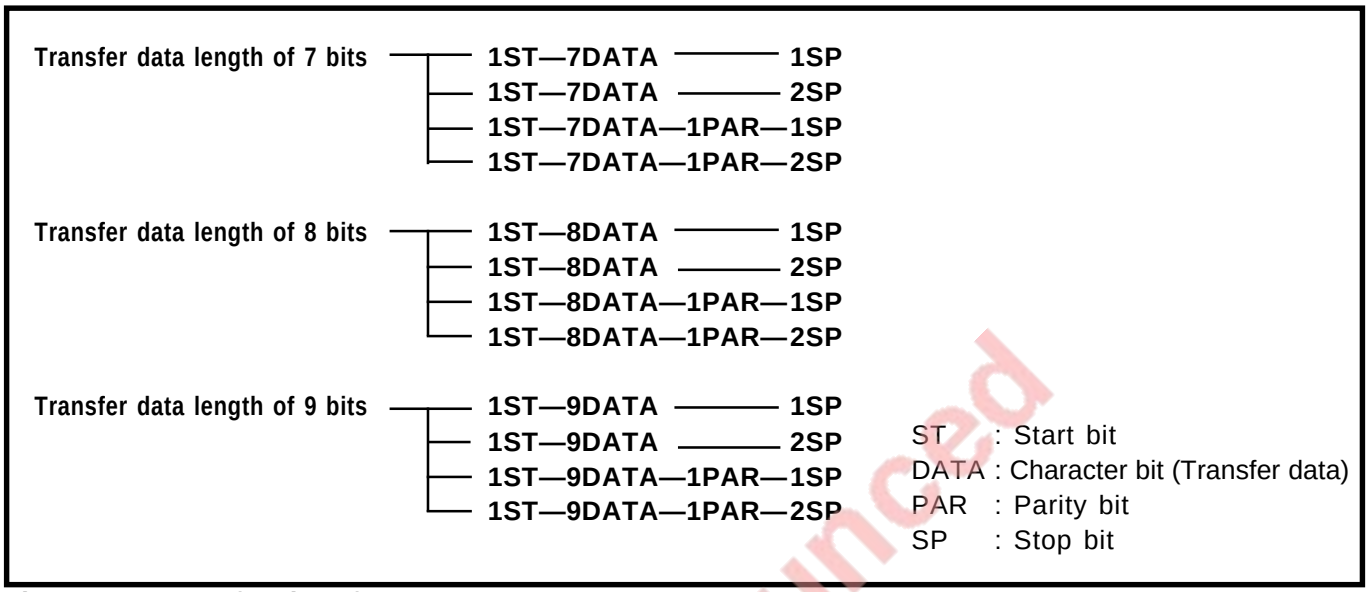


Fig. 11.4.1 Transfer data format

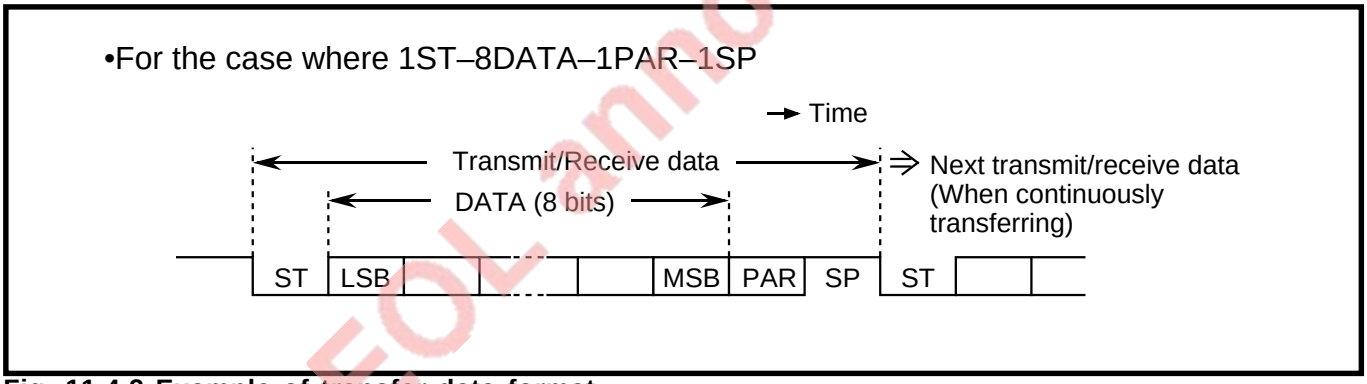


Fig. 11.4.2 Example of transfer data format

EOL announced

Table 11.4.5 Each bit in transmit data

Name	Functions
ST Start bit	"L" signal equivalent to 1 character bit which is added immediately before the character bits. It indicates start of data transmission.
DATA Character bit	Transmit data which is set in the UARTi transmit buffer register.
PAR Parity bit	A signal that is added immediately after the character bits in order to improve data reliability. The level of this signal changes according to selection of odd/even parity in such a way that the sum of "1"s in this bit and character bits is always an odd or even number.
SP Stop bit	"H" level signal equivalent to 1 or 2 character bits which is added immediately after the character bits (or parity bit when parity is enabled). It indicates finish of data transmission.

11.4 Clock asynchronous serial I/O (UART) mode

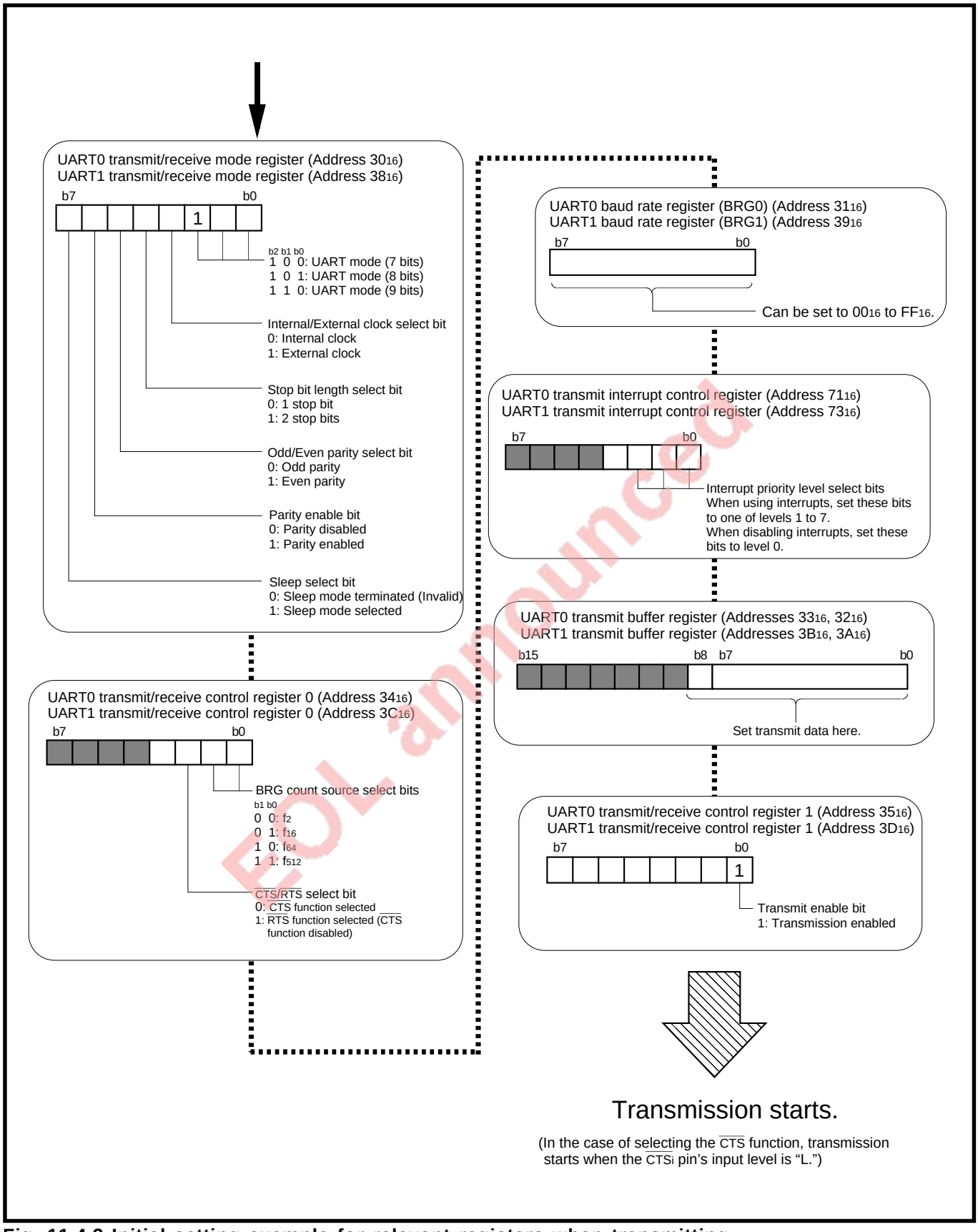
11.4.3 Method of transmission

Figure 11.4.3 shows an initial setting example for relevant registers when transmitting.
The difference due to selection of transfer data length (7 bits, 8 bits, or 9 bits) is only that data length. When selecting a 7- or 8-bit data length, set the transmit data into the low-order byte of the UARTi transmit buffer register. When selecting a 9-bit data length, set the transmit data into the low-order byte and bit 0 of the high-order byte.
Transmission is started when all of the following conditions (① to ③) are satisfied:

① Transmit is enabled (transmit enable bit = "1").
② Transmit data is present in the UARTi transmit buffer register (transmit buffer empty flag = "0").
③ The $\overline{CTS_i}$ pin's input is at "L" level (when the $\overline{CTS}$ function selected).
Note: When the $\overline{CTS}$ function is not selected, condition ③ is ignored.

By connecting the $\overline{RTS_i}$ pin (receiver side) and $\overline{CTS_i}$ pin (transmitter side), the timing of transmission and that of reception can be matched. For details, refer to section **"11.4.6 Receive operation."**
When using interrupts, it is necessary to set the relevant registers to enable interrupts. For details, refer to **"CHAPTER 7. INTERRUPTS."**
Figure 11.4.4 shows writing data after start of transmission, and Figure 11.4.5 shows detection of transmit completion.

EOL announced



EOL announced

Fig. 11.4.3 Initial setting example for relevant registers when transmitting

11.4 Clock asynchronous serial I/O (UART) mode

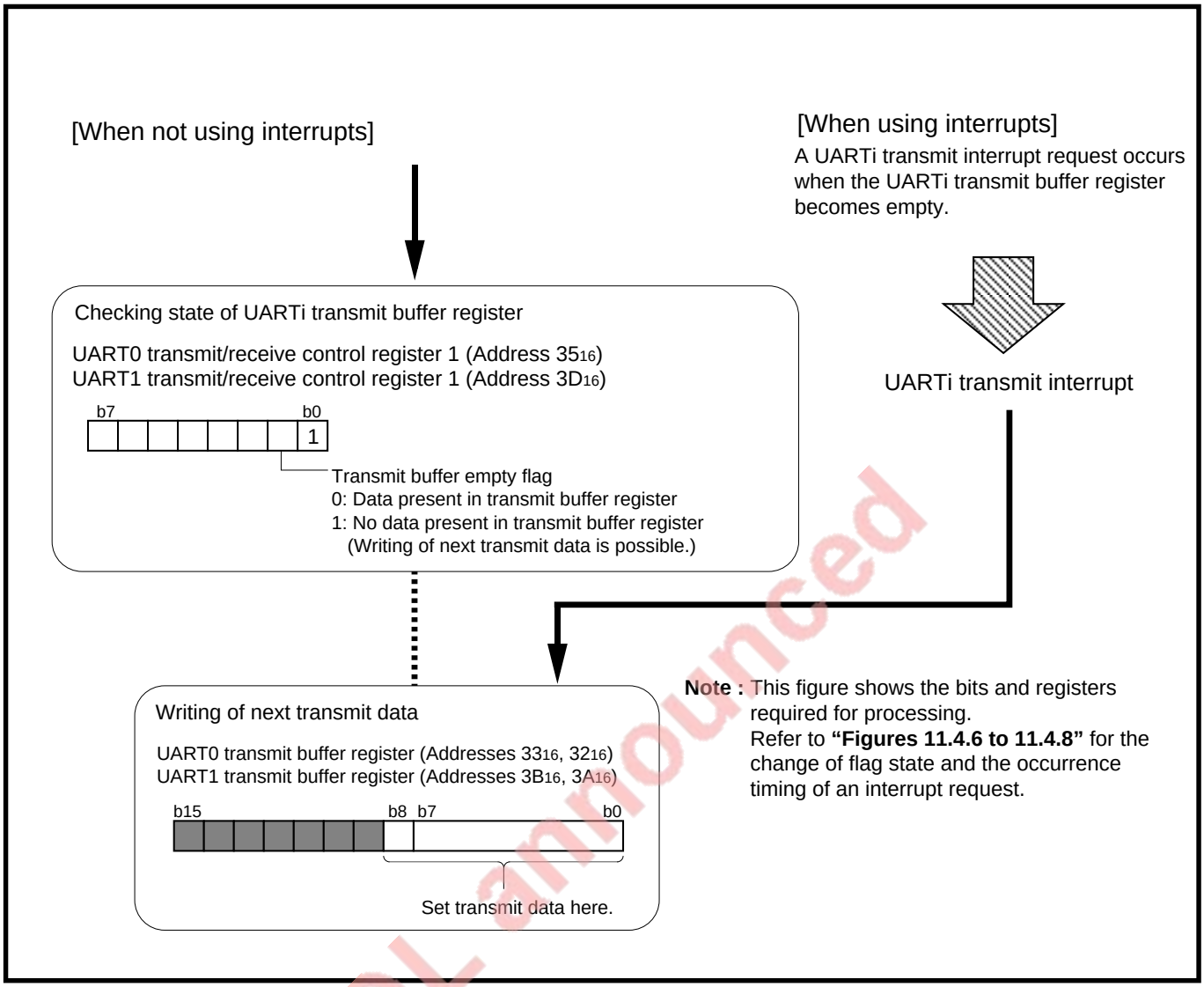


Fig. 11.4.4 Writing data after start of transmission

EOL announced

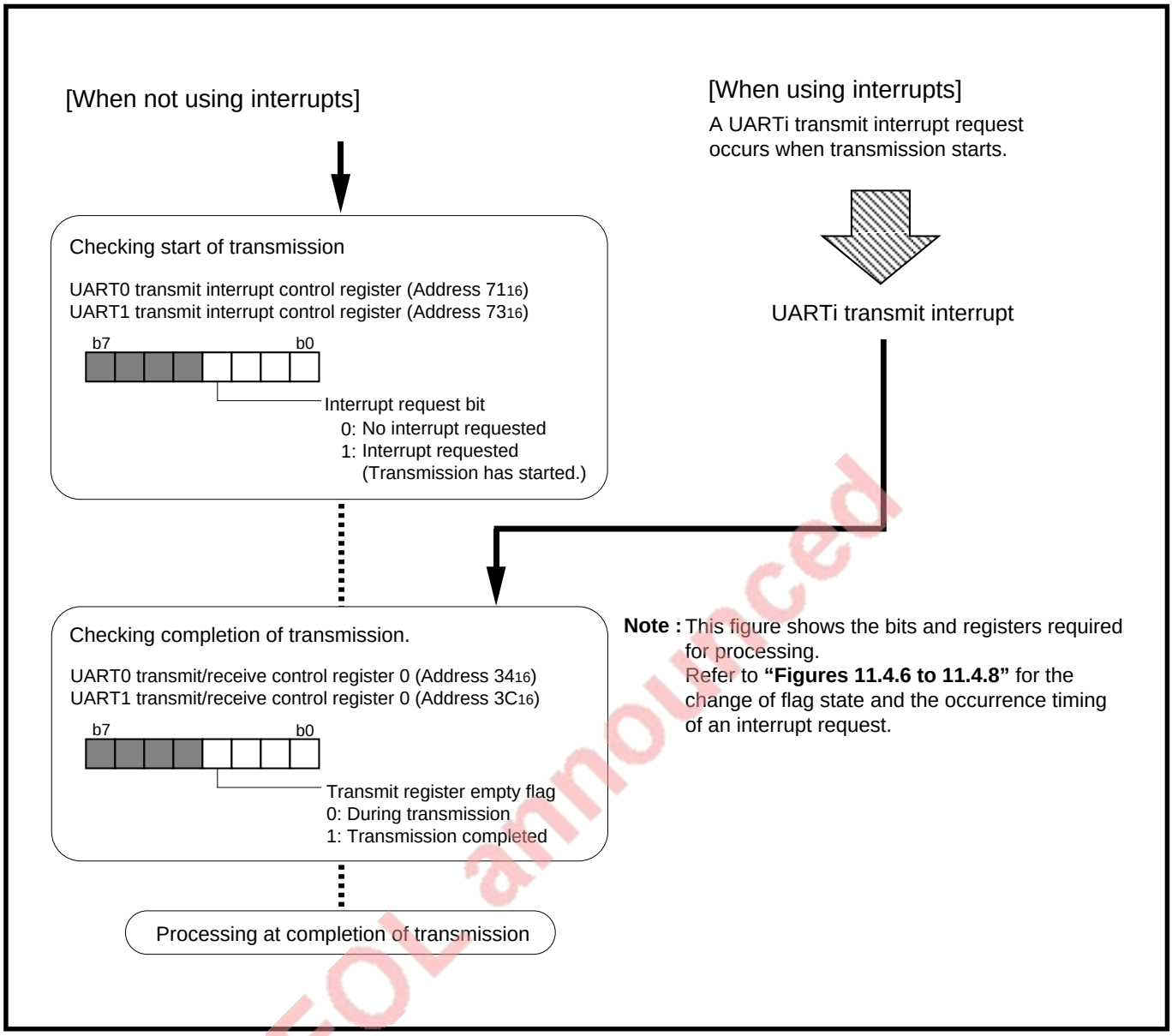


Fig. 11.4.5 Detection of transmit completion

EOL announced

11.4 Clock asynchronous serial I/O (UART) mode

11.4.4 Transmit operation

When the receive conditions described in section **"11.4.3 Method of transmission"** are satisfied, a transfer clock is generated and the following operations are automatically performed after 1 cycle of the transfer clock has passed.

- The UARTi transmit buffer register's contents are transferred to the UARTi transmit register.
- The transmit buffer empty flag is set to "1."
- The transmit register empty flag is cleared to "0."
- A UARTi transmit interrupt request occurs and the interrupt request bit is set to "1."

The transmit operations are described below:

① Data in the UARTi transmit register is transmitted from the TxD_i pin.
② This data is transmitted bit by bit sequentially in order of ST→DATA (LSB)→•••→DATA (MSB)→PAR →SP according to the transfer data format.
③ The transmit register empty flag is set to "1" at the center of the stop bit (or the second stop bit when selecting 2-stop bits), indicating completion of transmission. Additionally, whether the transmit conditions for the next data are satisfied or not is examined.

When the transmit conditions for the next data are satisfied in step ③, the start bit is generated following the stop bit, and the next data is transmitted. When performing transmission continuously, set the next transmit data in the UARTi transmit buffer register during transmission (when the transmit register empty flag = "0"). When the transmit conditions for the next data are not satisfied, the TxD_i pin outputs "H" level and the transfer clock stops.
Figures 11.4.6 and 11.4.7 show examples of transmit timing when the transfer data length = 8 bits, and Figure 11.4.8 shows an example of transmit timing when the transfer data length = 9 bits.

EOL announced

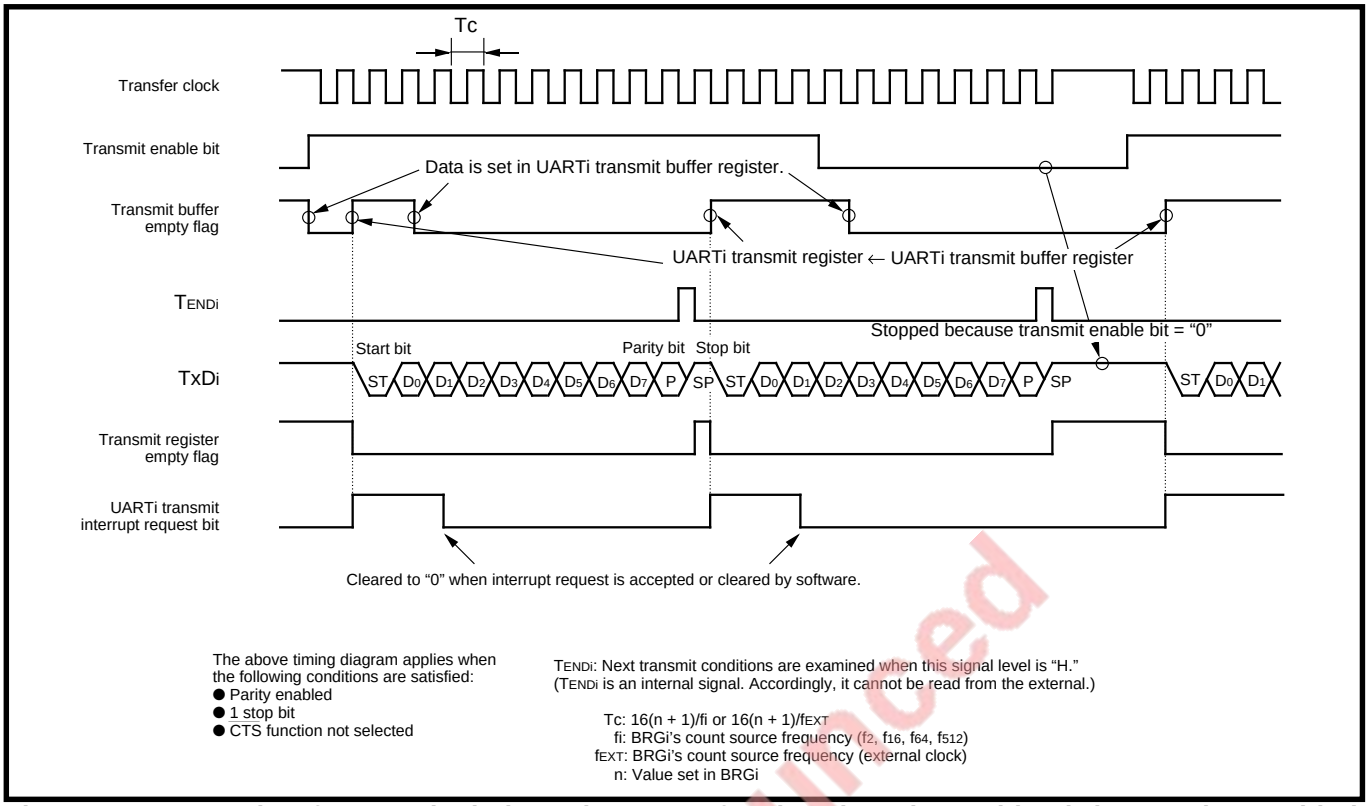


Fig. 11.4.6 Example of transmit timing when transfer data length = 8 bits (when parity enabled, selecting 1 stop bit, not selecting $\overline{\text{CTS}}$ function)

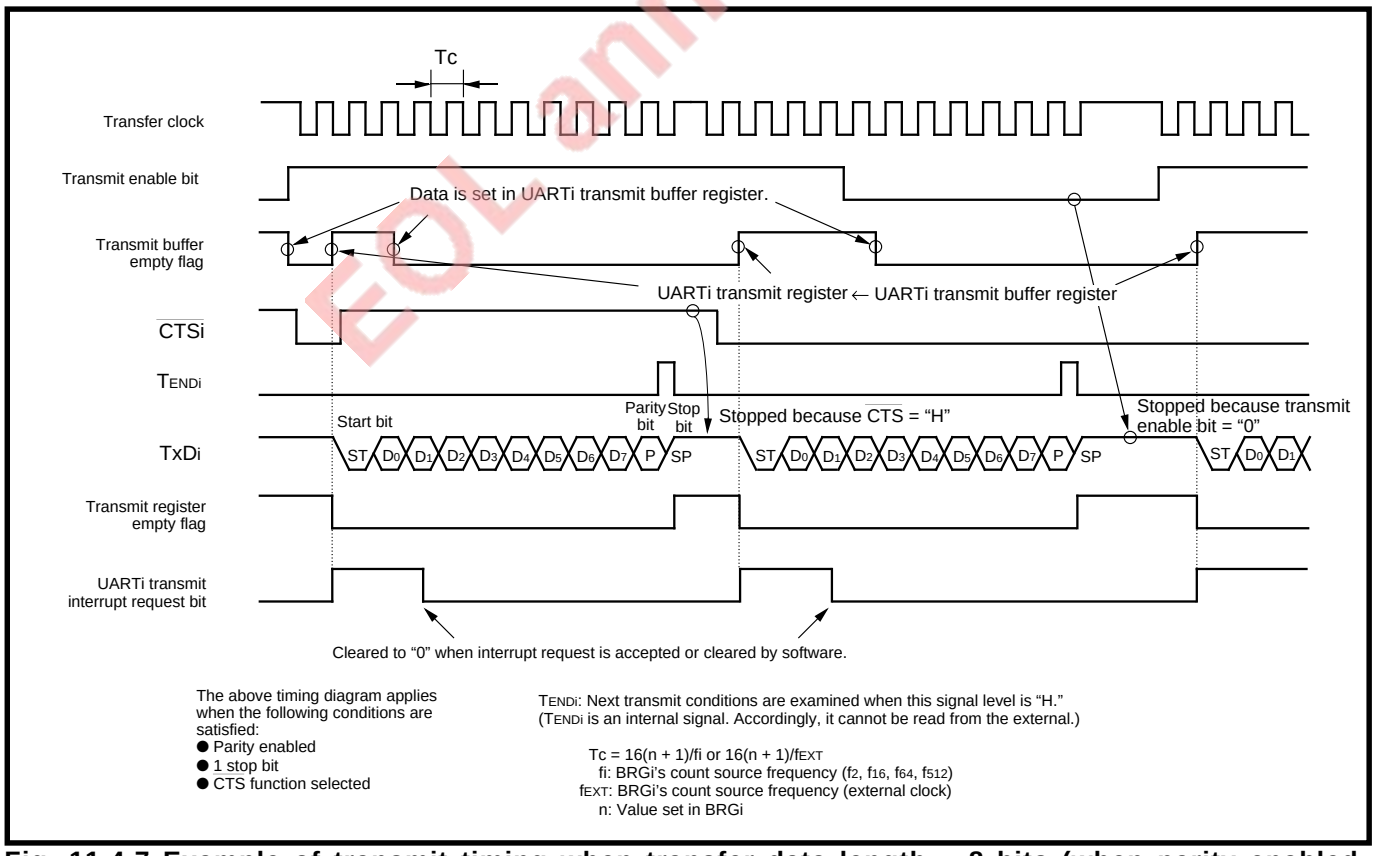
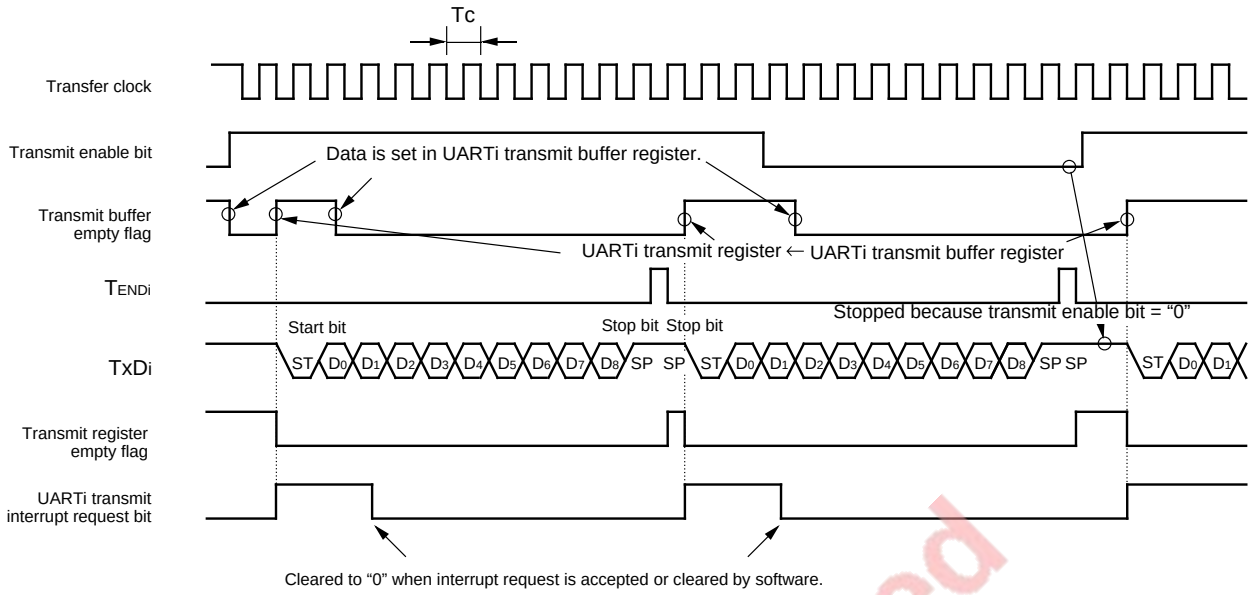


Fig. 11.4.7 Example of transmit timing when transfer data length = 8 bits (when parity enabled, selecting 1 stop bit, selecting $\overline{\text{CTS}}$ function)

EOL announced

11.4 Clock asynchronous serial I/O (UART) mode



The above timing diagram applies when the following conditions are satisfied:
- Parity disabled
- 2 stop bits
- CTS function disabled

TENDi: Next transmit conditions are examined when this signal level is "H."
(TENDi is an internal signal. Accordingly, it cannot be read from the external.)

Tc = 16(n + 1)/fi or 16(n + 1)/fEXT
fi: BRGi count source frequency (f_2, f_{16}, f_{64}, f_{512})
fEXT: BRGi count source frequency (external clock)
n: Value set in BRGi

Fig. 11.4.8 Example of transmit timing when transfer data length = 9 bits (when parity disabled, selecting 2 stop bits, not selecting CTS function)

EOL announced

11.4.5 Method of reception

Figure 11.4.9 shows an initial setting example for relevant registers when receiving. Reception is started when all of the following conditions (① and ②) are satisfied:

① Reception is enabled (receive enable bit = “1”).
② The start bit is detected.

By connecting the $\overline{RTS_i}$ pin (receiver side) and $\overline{CTS_i}$ pin (transmitter side), the timing of transmission and that of reception can be matched. For details, refer to section **“11.4.6 Receive operation.”**
When using interrupts, it is necessary to set the relevant registers to enable interrupts. For details, refer to **“CHAPTER 7. INTERRUPTS.”**
Figure 11.4.10 shows processing after receive completion.

EOL announced

11.4 Clock asynchronous serial I/O (UART) mode

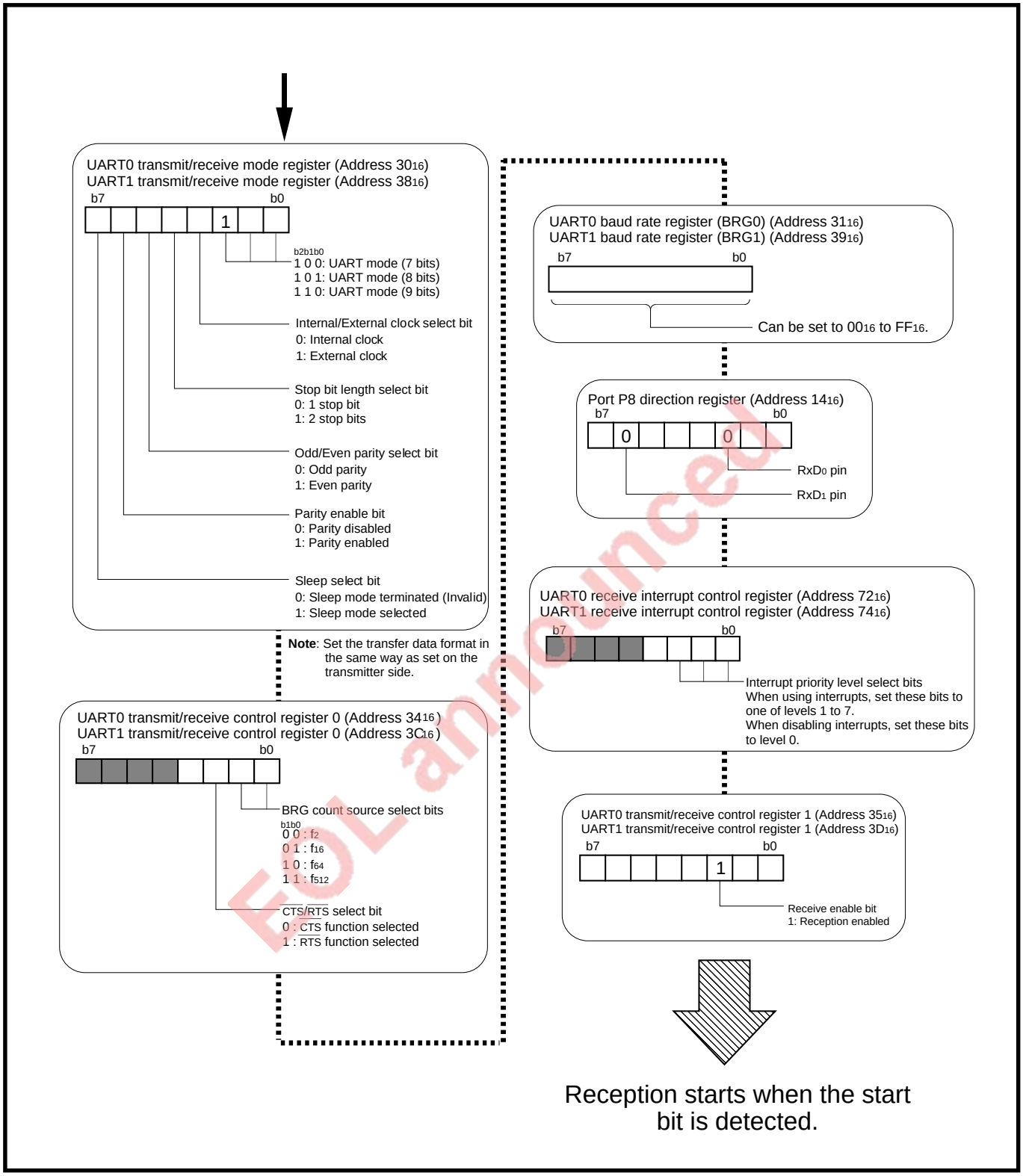
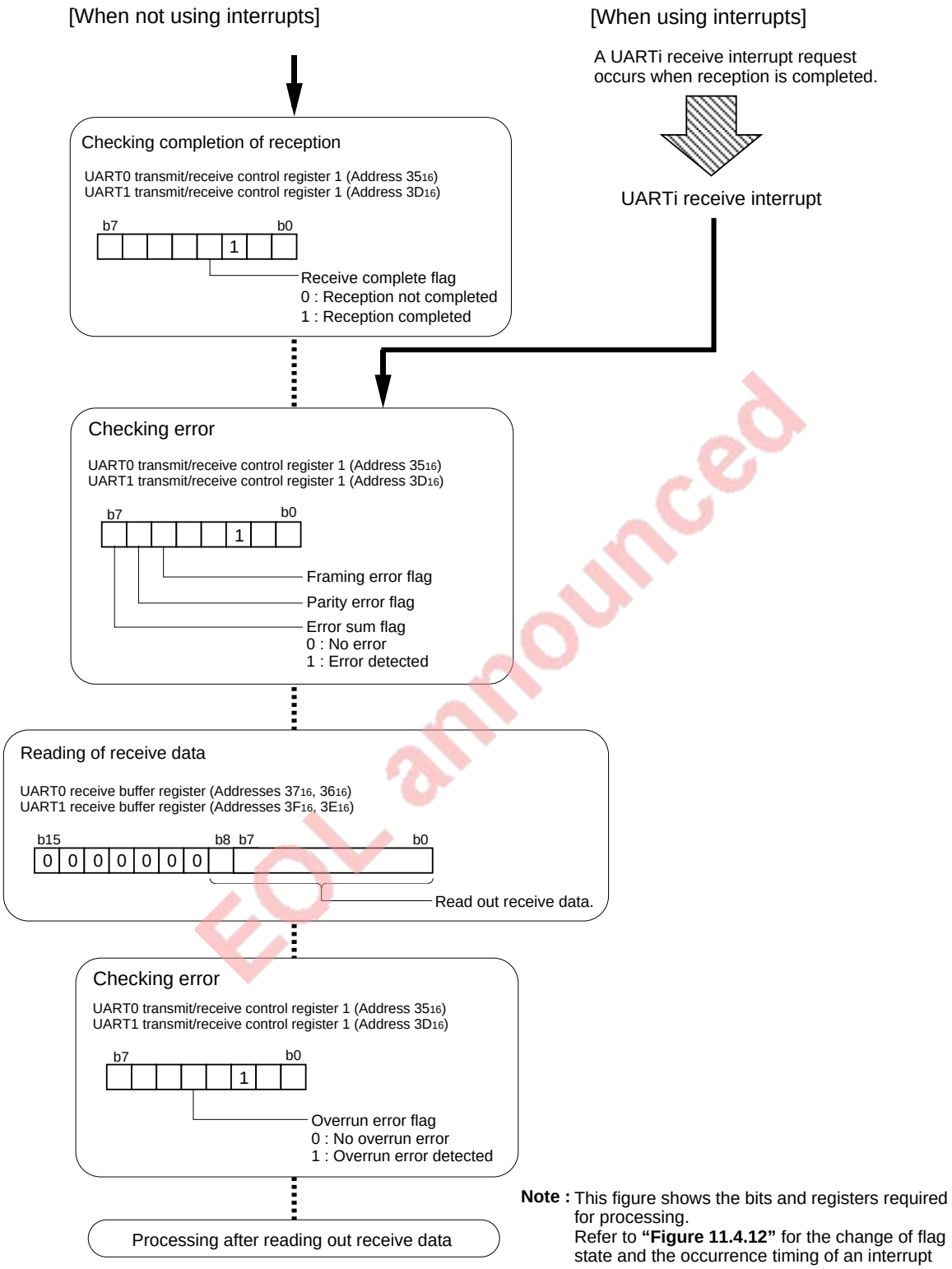


Fig. 11.4.9 Initial setting example for relevant registers when receiving



EOL announced

Note : This figure shows the bits and registers required for processing.
Refer to **"Figure 11.4.12"** for the change of flag state and the occurrence timing of an interrupt request.

Fig. 11.4.10 Processing after receive completion

11.4 Clock asynchronous serial I/O (UART) mode

11.4.6 Receive operation

When the receive enable bit is set to “1,” the UARTi enters the receive enable state. After this, reception starts when ST is detected and a transfer clock is generated.

In the case of selecting the $\overline{\text{RTS}}$ function, when the reception is enabled, the $\overline{\text{RTS}_i}$ pin’s output level becomes “L” to inform the transmitter side that reception is enabled. When reception is started, the $\overline{\text{RTS}_i}$ pin’s output level becomes “H.” Accordingly, by connecting the $\overline{\text{RTS}_i}$ pin to the $\overline{\text{CTS}_i}$ pin of the transmitter side, the timing of transmission and that of reception can be matched. Figure 11.4.11 shows an connection example.

The receive operation is described below.

① The input signal of the RxD_i pin is taken into the most significant bit of the UARTi receive register synchronously with the transfer clock’s rising edge.
② The contents of the UARTi receive register are shifted by 1 bit to the right.
③ Steps ① and ② are repeated at each rising edge of the transfer clock.
④ When one set of data has been prepared, in other words, when the shift has been performed several times according to the selected data format, the UARTi receive register’s contents are transferred to the UARTi receive buffer register.
⑤ Simultaneously with step ④, the receive complete flag is set to “1.” Additionally, a UARTi receive interrupt request occurs and its interrupt request bit is set to “1.”

The receive complete flag is cleared to “0” when the low-order byte of the UARTi receive buffer register is read out. The $\overline{\text{RTS}_i}$ pin’s output level becomes “L” simultaneously with step ⑤ (when selecting the $\overline{\text{RTS}}$ function). Figure 11.4.12 shows an example of receive timing when the transfer data length = 8 bits.

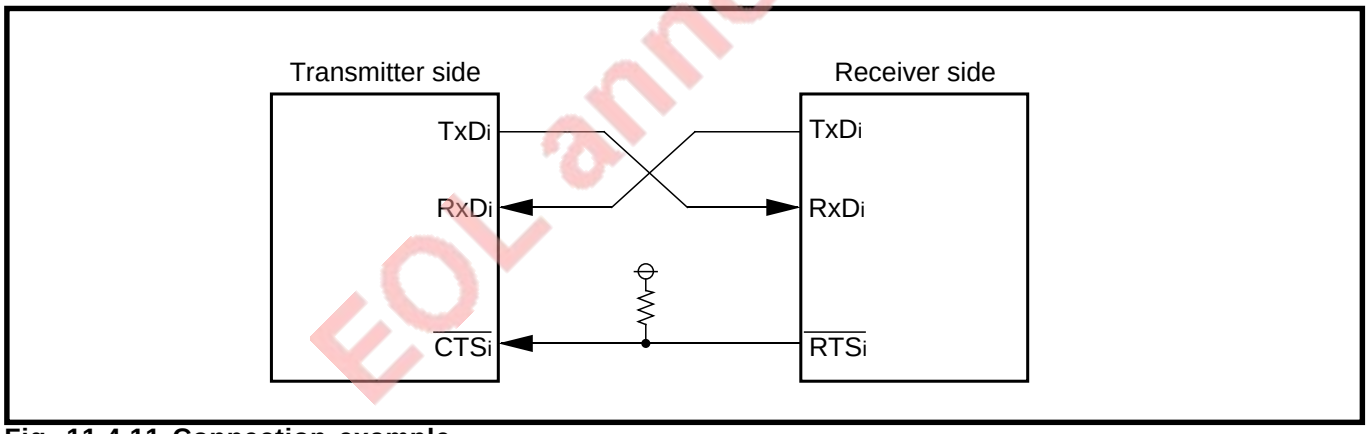


Fig. 11.4.11 Connection example

EOL announced

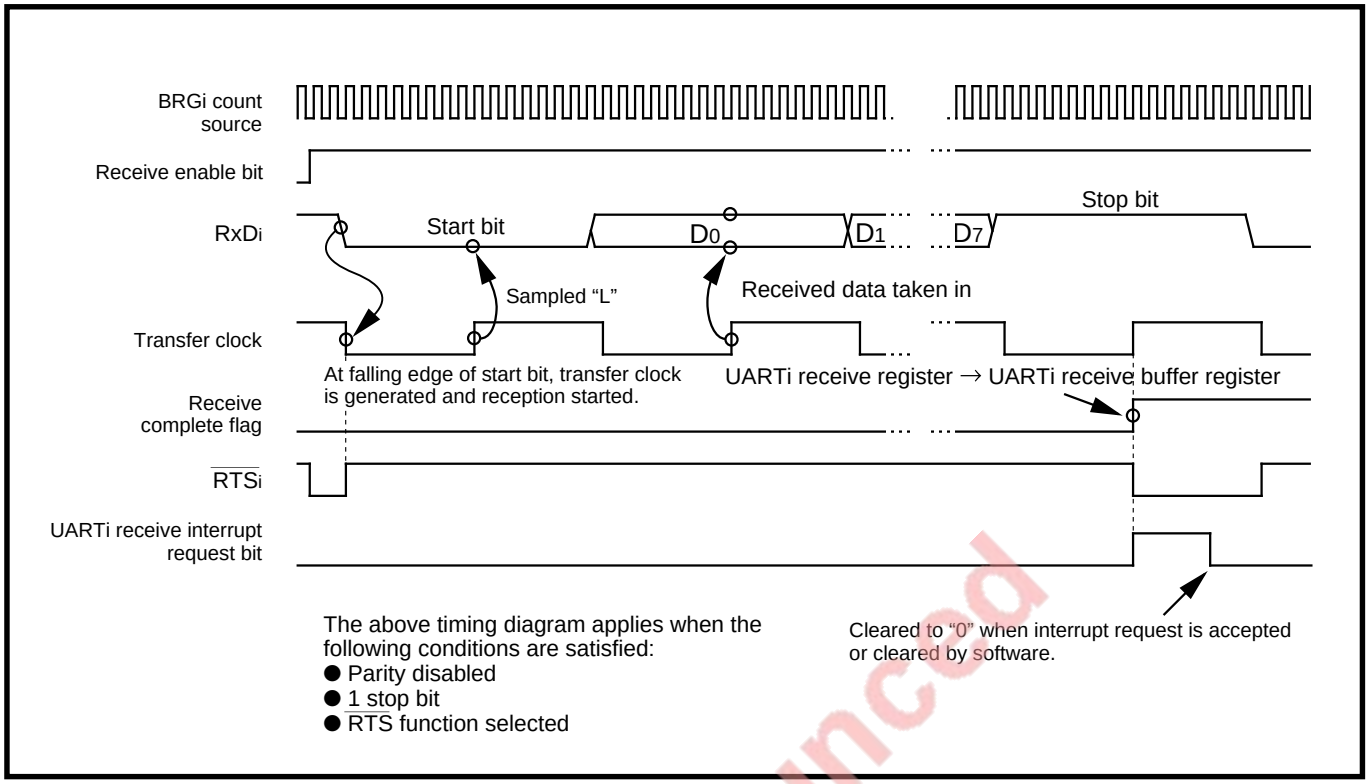


Fig. 11.4.12 Example of receive timing when transfer data length = 8 bits (when parity disabled, selecting 1 stop bit, selecting RTS function)

EOL announced

11.4 Clock asynchronous serial I/O (UART) mode

11.4.7 Processing on detecting error

In the UART mode, 3 types of errors can be detected. Each error can be detected when the data in the UARTi receive register is transferred to the UARTi receive buffer register, and the corresponding error flag is set to "1." When any error occurs, the error sum flag is set to "1." Accordingly, presence of errors can be judged by using the error sum flag.
Table 11.4.6 lists conditions for setting each error flag to "1" and method for clearing it to "0."

Table 11.4.6 Conditions set to "1" and method cleared to "0" for each error flag

Error flag	Conditions for being set to "1"	Method for being cleared to "0"
Overrun error flag	When the next data is prepared in the UARTi receive register with the receive complete flag = "1" (i.e., data is present in the UARTi receive buffer register). In other words, when the next data is prepared before the contents of the UARTi receive buffer register are read out. **(Note)** [UARTi receive interrupt request bit is not changed.]	•Clear the serial I/O mode select bits to "000_2." •Clear the receive enable bit to "0."
Framing error flag	When the number of detected stop bits does not match the set number of stop bits. [UARTi receive interrupt request bit is set to "1."]	•Clear the serial I/O mode select bits to "000_2." •Clear the receive enable bit to "0." •Read out the low-order byte of the UARTi receive buffer register.
Parity error flag	When the sum of "1"s in the parity bit and character bits does not match the set number of "1"s. [UARTi receive interrupt request bit is set to "1."]	•Clear the serial I/O mode select bits to "000_2." •Clear the receive enable bit to "0." •Read out the low-order byte of the UARTi receive buffer register.
Error sum flag	When 1 or more errors listed above occur.	•Clear the all error flags, which are overrun, framing and parity error flags.

Note: The next data is written into the UARTi receive buffer register.

EOL announced

When an error occurs during reception, initialize the error flag and the UARTi receive buffer register, and then perform reception again. When it is necessary to perform retransmission owing to an error which occurs in the receiver side during transmission, set the UARTi transmit buffer register again, and then restarts transmission.
The method of initializing the UARTi receive buffer register and that of setting the UARTi transmit buffer register again are described below.

(1) Method of initializing UARTi receive buffer register

① Clear the receive enable bit to "0" (reception disabled).
② Set the receive enable bit to "1" again (reception enabled).

(2) Method of setting UARTi transmit buffer register again

① Clear the serial I/O mode select bits to "000_2" (serial I/O invalid).
② Set the serial I/O mode select bits again.
③ Set the transmit enable bit to "1" (transmission enabled), and set the transmit data to the UARTi transmit buffer register.

11.4.8 Sleep mode

This mode is used to transfer data between the specified microcomputers, which are connected by using UARTi. The sleep mode is selected by setting the sleep select bit (bit 7 at addresses 30_{16}, 38_{16}) to "1" when receiving.

In the sleep mode, receive operation is performed when the MSB (D_8 when the transfer data is 9 bits length, D_7 when it is 8 bits length, D_6 when it is 7 bits length) of the receive data is "1." Receive operation is not performed when the MSB is "0." (The UARTi receive register's contents are not transferred to the UARTi receive buffer register. Additionally, the receive complete flag and error flags do not change and a UARTi receive interrupt request does not occur.)

The following shows an usage example of the sleep mode when the transfer data is 8 bits length.

① Set the same transfer data format for the master and slave microcomputers. Select the sleep mode for the slave microcomputers.
② Transmit data, which has "1" in bit 7 and the address of the slave microcomputer to be communicated in bits 0 to 6, from the master microcomputer to all slave microcomputers.
③ All slave microcomputers receive data of step ②. (At this time, a UARTi receive interrupt request occurs.)
④ For all slave microcomputers, check in the interrupt routine whether bits 0 to 6 in the receive data match their own addresses.
⑤ For the slave microcomputer of which address matches bits 0 to 6 in the receive data, terminate the sleep mode. (Do not terminate the sleep mode for the other slave microcomputers.)
By performing steps ② to ⑤, " the microcomputer which performs transfer" is specified.
⑥ Transmit data, which has "0" in bit 7, from the master microcomputer. (Only the microcomputer specified in steps ② to ⑤ can receive this data. The other microcomputers do not receive this data.)
⑦ By repeating step ⑥, transfer can be performed between two specific microcomputers continuously. When communicating with another microcomputer, perform steps ② to ⑤ in order to specify the new slave microcomputer.

EOL announced

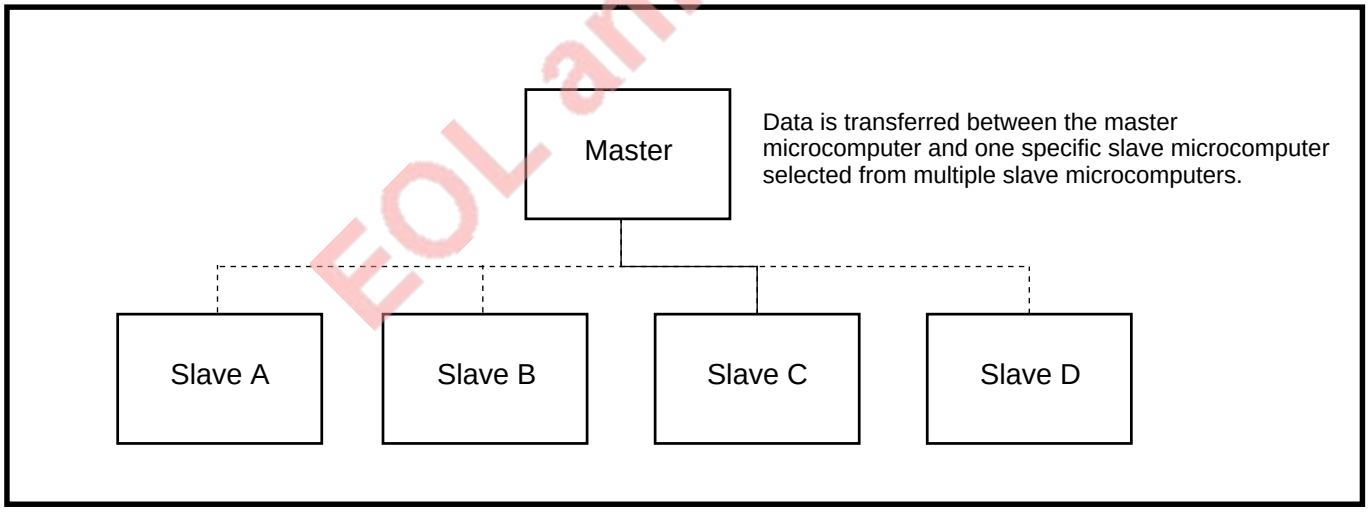


Fig. 11.4.13 Sleep mode

MEMORANDUM

EOL announced

CHAPTER 12
A-D CONVERTER

12.1 Overview
12.2 Block description
12.3 A-D conversion method
12.4 Absolute accuracy and differential non-linearity error
12.5 One-shot mode
12.6 Repeat mode
12.7 Single sweep mode
12.8 Repeat sweep mode
12.9 Precautions for A-D converter

EOL announced

12.1 Overview

Table 12.1.1 lists the performance specifications of the A-D converter.

Table 12.1.1 Performance specifications of A-D converter

Item	Performance specifications
A-D conversion method	Successive approximation conversion method
Resolution	8 bits
Absolute accuracy	±3 LSB
Analog input pin	8 pins (AN_0 to AN_7) (**Note**)
Conversion rate per analog input pin	57 ϕ_{AD}* cycles

ϕ_{AD}*: A-D converter's operation clock

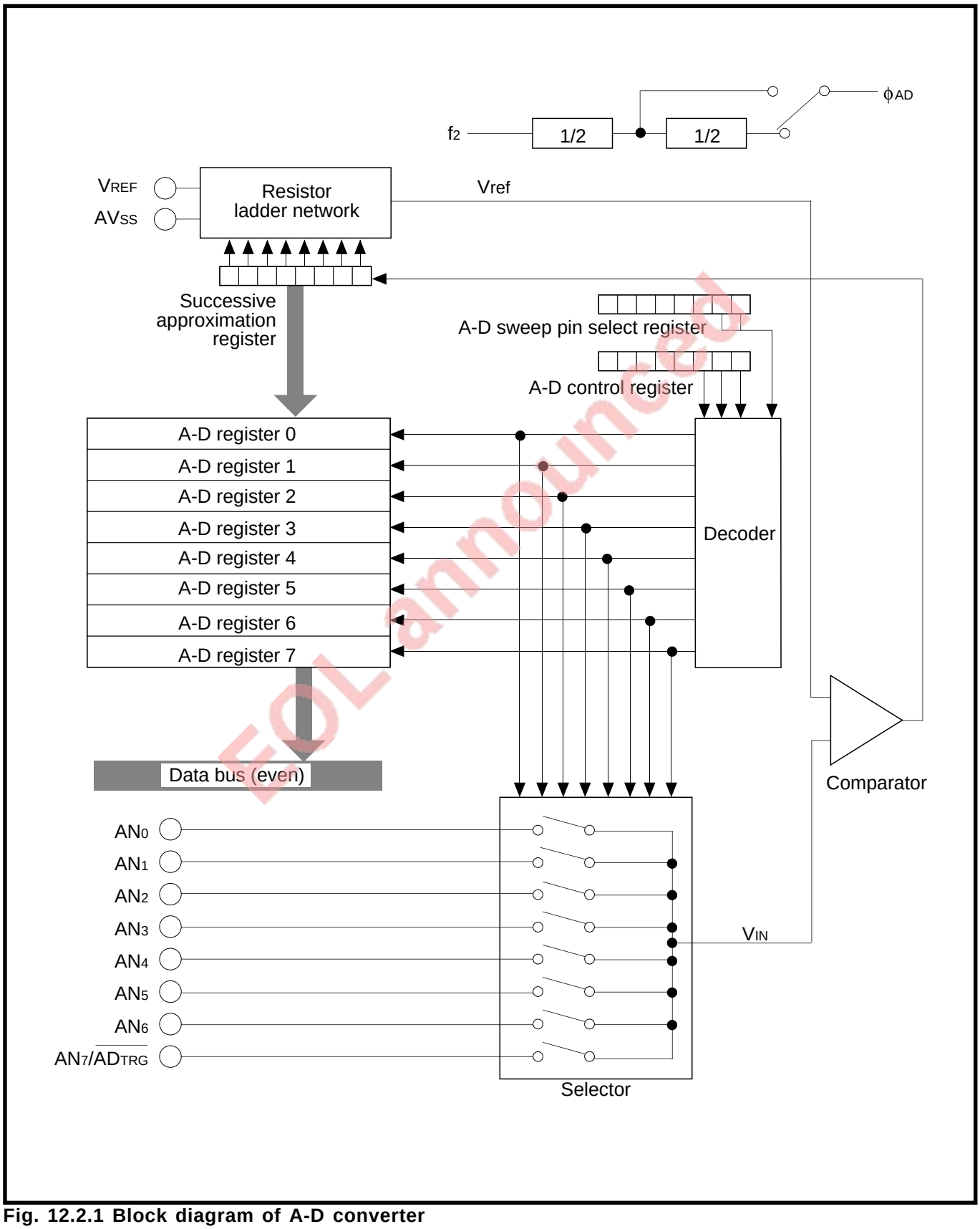
The A-D converter has the 4 operation modes listed below.

•One-shot mode
This mode is used to perform the operation once for a voltage input from one selected analog input pin.
•Repeat mode
This mode is used to perform the operation repeatedly for a voltage input from one selected analog input pin.
•Single sweep mode
This mode is used to perform the operation for voltages input from multiple selected analog input pins, one at a time.
•Repeat sweep mode
This mode is used to perform the operation repeatedly for voltages input from multiple selected analog input pins.

EOL announced

12.2 Block description

Figure 12.2.1 shows the block diagram of the A-D converter. Registers relevant to the A-D converter are described below.



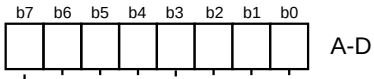
EOL announced

Fig. 12.2.1 Block diagram of A-D converter

12.2 Block description

12.2.1 A-D control register

Figure 12.2.2 shows the structure of the A-D control register. The A-D operation mode select bit selects the operation mode of the A-D converter. The other bits are described below.



A-D control register (Address $1E_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Analog input select bits (Valid in one-shot and repeat modes) **(Note 1)**	b2 b1 b0 0 0 0 : AN_0 selected 0 0 1 : AN_1 selected 0 1 0 : AN_2 selected 0 1 1 : AN_3 selected 1 0 0 : AN_4 selected 1 0 1 : AN_5 selected 1 1 0 : AN_6 selected 1 1 1 : AN_7 selected **(Note 2)**	Undefined	RW
1			Undefined	RW
2			Undefined	RW
3	A-D operation mode select bits	b4 b3 0 0 : One-shot mode 0 1 : Repeat mode 1 0 : Single sweep mode 1 1 : Repeat sweep mode	0	RW
4			0	RW
5	Trigger select bit	0 : Internal trigger 1 : External trigger	0	RW
6	A-D conversion start bit	0 : Stop A-D conversion 1 : Start A-D conversion	0	RW
7	A-D conversion frequency (ϕ_{AD}) select bit	0 : f_2 divided by 4 1 : f_2 divided by 2	0	RW

Notes 1: These bits are ignored in the single sweep and repeat sweep mode. (They may be either "0" or "1.")
2: When an external trigger is selected, the AN_7 pin cannot be used as an analog input pin.
3: Writing to each bit (except bit 6) of the A-D control register must be performed while the A-D converter halts.

EOL announced

Fig. 12.2.2 Structure of A-D control register

(1) Analog input select bits (bits 2 to 0)

These bits are used to select an analog input pin in the one-shot mode and repeat mode. Pins which are not selected as analog input pins function as programmable I/O ports.

These bits must be set again when the user switches the A-D operation mode to the one-shot mode or repeat mode after A-D conversion is performed in the single sweep mode or repeat sweep mode.

(2) Trigger select bit (bit 5)

This bit is used to select the source of trigger occurrence. (Refer to section **"(3) A-D conversion start bit."**)

(3) A-D conversion start bit (bit 6)

● **When internal trigger is selected**

Setting this bit to "1" generates a trigger, causing the A-D converter to start operating. Clearing this bit to "0" causes the A-D converter to stop operating.

In the one-shot mode or single sweep mode, this bit is cleared to "0" after the operation is completed. In the repeat mode or repeat sweep mode, the A-D converter continues operating until this bit is cleared to "0" by software.

● **When external trigger is selected**

When the $\overline{\text{ADTRG}}$ pin level goes from "H" to "L" with this bit = "1," a trigger occurs, causing the A-D converter to start operating. The A-D converter stops when this bit is cleared to "0."

In the one-shot mode or single sweep mode, this bit remains set to "1" even after the operation is completed. In the repeat mode or repeat sweep mode, the A-D converter continues operating until this bit is cleared to "0" by software.

(4) A-D conversion frequency (ϕ_{AD}) select bit (bit 7)

As listed in Table 12.2.1, the conversion time of the A-D converter varies depending on the operating clock (ϕ_{AD}) selected by this bit.

Since the A-D converter's comparator consists of capacity coupling amplifiers, keep that $\phi_{AD} \geq 250$ kHz during A-D conversion**.**

Table 12.2.1 Conversion time per one analog input pin (unit: μs)

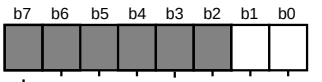
A-D conversion frequency (ϕ_{AD}) select bit		0	1
ϕ_{AD}		$f_2/4$	$f_2/2$
Conversion time	$f(X_{IN})$ = 8 MHz	57.0	28.5
	$f(X_{IN})$ = 16 MHz	28.5	14.25
	$f(X_{IN})$ = 25 MHz	18.24	9.12

EOL announced

12.2 Block description

12.2.2 A-D sweep pin select register

Figure 12.2.3 shows the structure of the A-D sweep pin select register.



A-D sweep pin select register (Address $1F_{16}$)

Bit	Bit name	Functions	At reset	RW
0	A-D sweep pin select bits (Valid in single sweep and repeat sweep mode) **(Note 1)**	b1 b0 0 0 : AN_0, AN_1 (2 pins) 0 1 : AN_0 to AN_3 (4 pins) 1 0 : AN_0 to AN_5 (6 pins) 1 1 : AN_0 to AN_7 (8 pins) **(Note 2)**	1	RW
1			1	RW
7 to 2	Nothing is assigned.		Undefined	—

Notes 1: These bits are invalid in the one-shot and repeat modes. (They may be either "0" or "1.")
2: When selecting an external trigger, the AN_7 pin cannot be used as an analog input pin.
3: Writing to each bit of the A-D sweep pin select register must be performed while the A-D converter halts.

Fig. 12.2.3 Structure of A-D control register 1

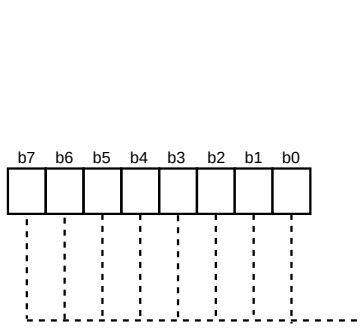
(1) A-D sweep pin select bits (bits 1 and 0)

These bits are used to select analog input pins in the single sweep mode or repeat sweep mode. In the single sweep mode and repeat sweep mode, pins which are not selected as analog input pins function as programmable I/O ports.

EOL announced

12.2.3 A-D register i (i = 0 to 7)

Figure 12.2.4 shows the structure of the A-D register i. When the A-D conversion is completed, the conversion result (contents of the successive approximation register) is stored into this register. Each A-D register i corresponds to an analog input pin (AN_i).



A-D register 0 (Addresses 20_{16})
A-D register 1 (Addresses 22_{16})
A-D register 2 (Addresses 24_{16})
A-D register 3 (Addresses 26_{16})
A-D register 4 (Addresses 28_{16})
A-D register 5 (Addresses $2A_{16}$)
A-D register 6 (Addresses $2C_{16}$)
A-D register 7 (Addresses $2E_{16}$)

Bit	Functions	At reset	RW
7 to 0	Reads an A-D conversion result.	Undefined	RO

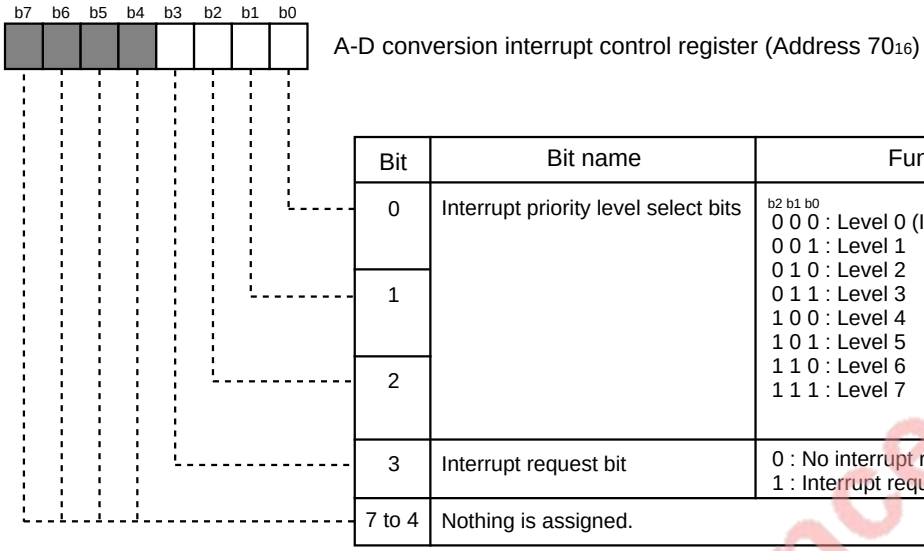
Fig. 12.2.4 Structure of A-D register i

EOL announced

12.2 Block description

12.2.4 A-D conversion interrupt control register

Figure 12.2.5 shows the structure of the A-D conversion interrupt control register. For details about interrupts, refer to **"CHAPTER 7. INTERRUPTS."**



Bit	Bit name	Functions	At reset	RW
0	Interrupt priority level select bits	b2 b1 b0 0 0 0 : Level 0 (Interrupt disabled) 0 0 1 : Level 1 Low level 0 1 0 : Level 2 0 1 1 : Level 3 1 0 0 : Level 4 1 0 1 : Level 5 1 1 0 : Level 6 1 1 1 : Level 7 High level	0	RW
1			0	RW
2			0	RW
3	Interrupt request bit	0 : No interrupt requested 1 : Interrupt requested	0	RW
7 to 4	Nothing is assigned.		Undefined	—

Fig. 12.2.5 Structure of A-D conversion interrupt control register

(1) Interrupt priority level select bits (bits 2 to 0)

These bits select an A-D conversion interrupt's priority level. When using A-D conversion interrupts, select one of the priority levels (1 to 7). When an A-D conversion interrupt request occurs, its priority level is compared with the processor interrupt priority level (IPL). The requested interrupt is enabled only when its priority level is higher than the IPL. (However, this applies when the interrupt disable flag (I) = "0.")

To disable A-D conversion interrupts, set these bits to "000$_2$" (level 0).

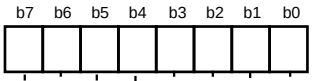
(2) Interrupt request bit (bit 3)

This bit is set to "1" when an A-D conversion interrupt request occurs. This bit is automatically cleared to "0" when the A-D conversion interrupt request is accepted. This bit can be set to "1" or cleared to "0" by software.

EOL announced

12.2.5 Port P7 direction register

Input pins of the A-D converter are multiplexed with port P7. When using these pins as A-D converter's input pins, set the corresponding bits of the port P7 direction register to "0" to set these port pins for the input mode. Figure 12.2.6 shows the relationship between the port P7 direction register and A-D converter's input pins.



Port P7 direction register (Address 11_{16})

Bit	Corresponding pin	Functions	At reset	RW
0	AN_0 pin	0 : Input mode 1 : Output mode When using these pins as A-D converter's input pins, set the corresponding bits to "0."	0	RW
1	AN_1 pin		0	RW
2	AN_2 pin		0	RW
3	AN_3 pin		0	RW
4	AN_4 pin		0	RW
5	AN_5 pin		0	RW
6	AN_6 pin		0	RW
7	$AN_7/\overline{ADTRG}$ pin		0	RW

Fig. 12.2.6 Relationship between port P7 direction register and A-D converter's input pins

EOL announced

12.3 A-D conversion method

The A-D converter compares the comparison voltage (V_{ref}), which is internally generated according to the contents of the successive approximation register, with the analog input voltage (V_{IN}), which is input from the analog input pin (AN_i). By reflecting the comparison result on the successive approximation register, V_{IN} is converted into a digital value. When a trigger is generated, the A-D converter performs the following processing:

① **Determining bit 7 of the successive approximation register**
The A-D converter compares V_{ref} with V_{IN}. At this time, the contents of the successive approximation register is "10000000_2" (initial value).
Bit 7 of the successive approximation register changes according to the comparison result as follows:
When $V_{ref} < V_{IN}$, bit 7 = "1"
When $V_{ref} > V_{IN}$, bit 7 = "0"

② **Determining bit 6 of the successive approximation register**
After setting bit 6 of the successive approximation register to "1," the A-D converter compares V_{ref} with V_{IN}. Bit 6 changes according to the comparison result as follows:
When $V_{ref} < V_{IN}$, bit 6 = "1"
When $V_{ref} > V_{IN}$, bit 6 = "0"

③ **Determining bits 5 to 0 of the successive approximation register**
Operations in ② are performed for bits 5 to 0.
When bit 0 is determined, the contents (conversion result) of the successive approximation register is transferred to the A-D register i.

The comparison voltage (V_{ref}) is generated according to the latest contents of the successive approximation register. Table 12.3.1 lists the relationship between the successive approximation register's contents and V_{ref}. Table 12.3.2 lists changes of the successive approximation register and V_{ref} during the A-D conversion. Figure 12.3.1 shows the ideal A-D conversion characteristics.

Table 12.3.1 Relationship between successive approximation register's contents and V_{ref}

Successive approximation register's contents: n	V_{ref} (V)
0	0
1 to 255	$\frac{V_{REF}*}{256} \times (n - 0.5)$

V_{REF}*: Reference voltage

EOL announced

Table 12.3.2 Change in successive approximation register and V_{ref} during A-D conversion

	Successive approximation register	Change of Vref
A-D converter halt	b7 b0 1 0 0 0 0 0 0 0	$\frac{V_{REF}}{2}$ [V]
1st comparison	1 0 0 0 0 0 0 0	$\frac{V_{REF}}{2} - \frac{V_{REF}}{512}$ [V]
2nd comparison	n_7 1 0 0 0 0 0 0 1st comparison result	$\frac{V_{REF}}{2} \pm \frac{V_{REF}}{4} - \frac{V_{REF}}{512}$ [V] $\begin{pmatrix} \bullet n_7=1 & + \frac{V_{REF}}{4} \\ \bullet n_7=0 & - \frac{V_{REF}}{4} \end{pmatrix}$
3rd comparison	n_7 n_6 1 0 0 0 0 0 2nd comparison result	$\frac{V_{REF}}{2} \pm \frac{V_{REF}}{4} \pm \frac{V_{REF}}{8} - \frac{V_{REF}}{512}$ [V] $\begin{pmatrix} \bullet n_6=1 & + \frac{V_{REF}}{8} \\ \bullet n_6=0 & - \frac{V_{REF}}{8} \end{pmatrix}$
⋮	⋮	⋮
8th comparison	n_7 n_6 n_5 n_4 n_3 n_2 n_1 1	$\frac{V_{REF}}{2} \pm \frac{V_{REF}}{4} \pm \frac{V_{REF}}{8} \pm \cdots\cdots \pm \frac{V_{REF}}{256} - \frac{V_{REF}}{512}$ [V]
Conversion complete	n_7 n_6 n_5 n_4 n_3 n_2 n_1 n_0	

EOL announced

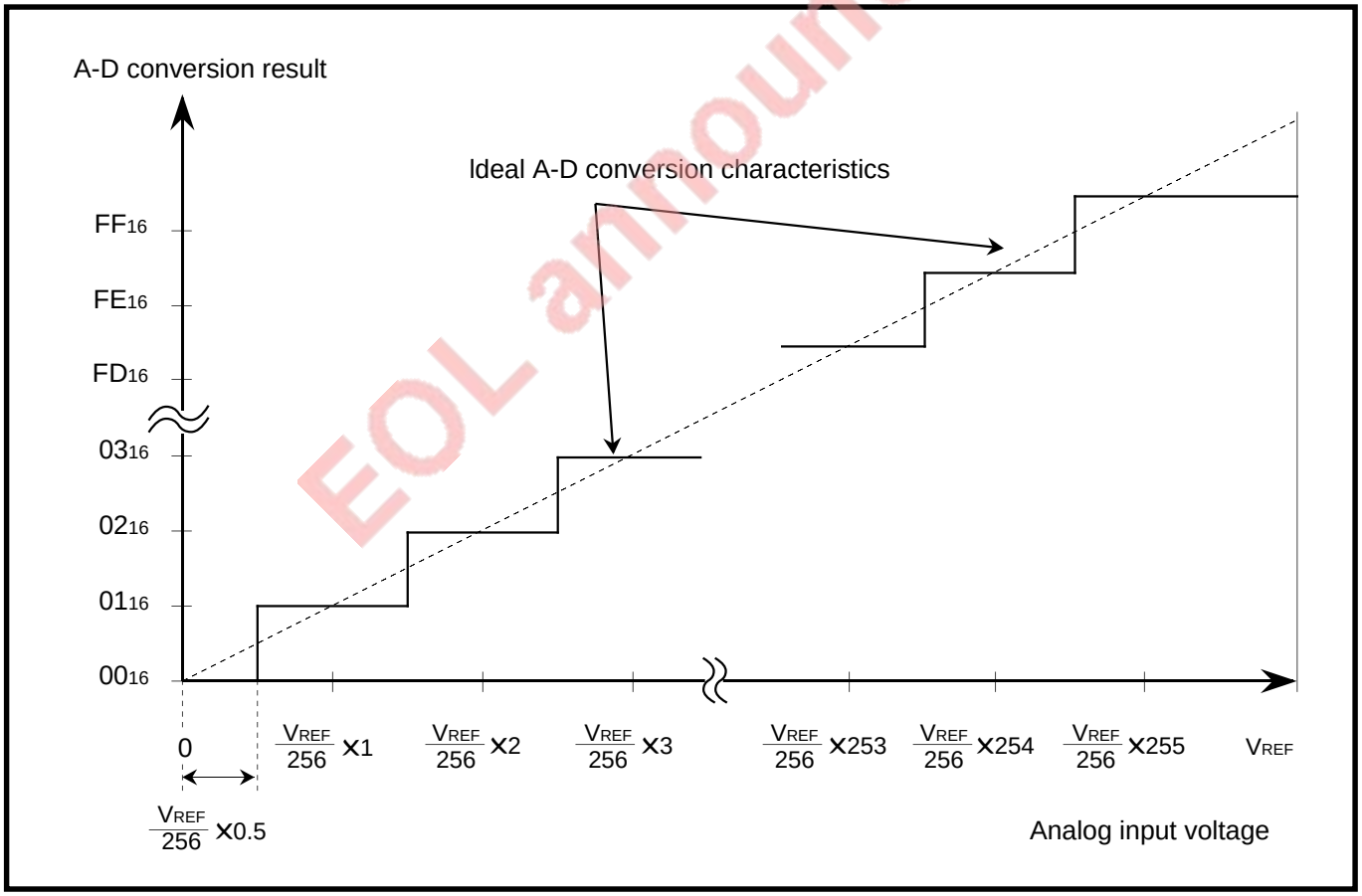


Fig. 12.3.1 Ideal A-D conversion characteristics

12.4 Absolute accuracy and differential non-linearity error

The A-D converter's accuracy is described below. Refer to section "**Appendix 12.3 A-D converter standard characteristics**," also.

12.4.1 Absolute accuracy

The absolute accuracy is the difference expressed in the LSB between the actual A-D conversion result and the output code of an A-D converter with ideal characteristics. The analog input voltage when measuring the accuracy is assumed to be the mid point of the input voltage width that outputs the same output code from an A-D converter with ideal characteristics. For example, when V_{REF} = 5.12 V, 1 LSB width is 20 mV, and 0 mV, 20 mV, 40 mV, 60 mV, 80 mV, ... are selected as the analog input voltages.

The absolute accuracy = ±3 LSB indicates that when the analog input voltage is 100 mV, the output code expected from an ideal A-D conversion characteristics is "005_{16}," however the actual A-D conversion result is between "002_{16}" to "008_{16}."

The absolute accuracy includes the zero error and the full-scale error.

The absolute accuracy is degraded when V_{REF} is lowered. Any of the output codes for analog input voltages from V_{REF} to AV_{CC} is "FF_{16}."

EOL announced

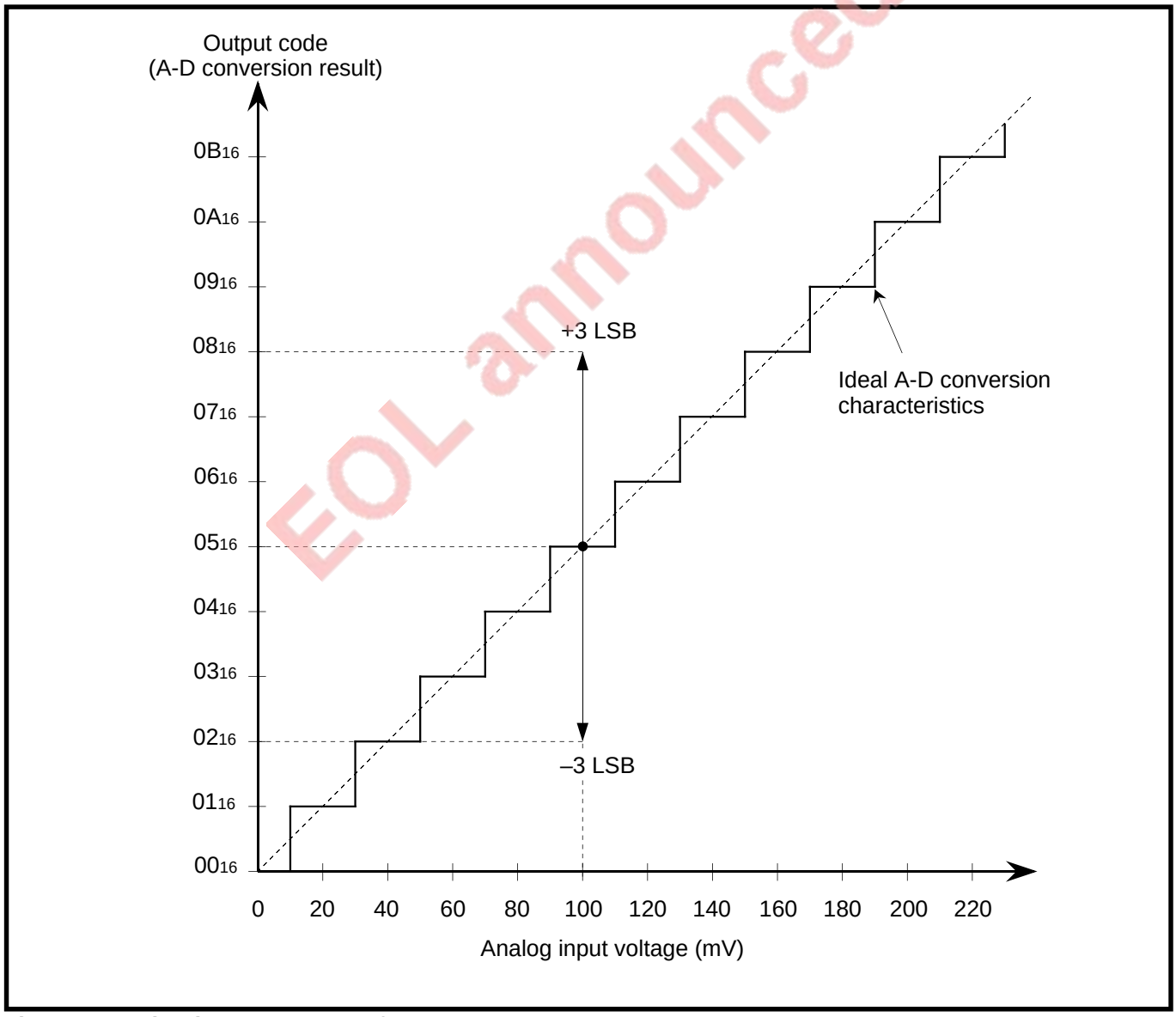
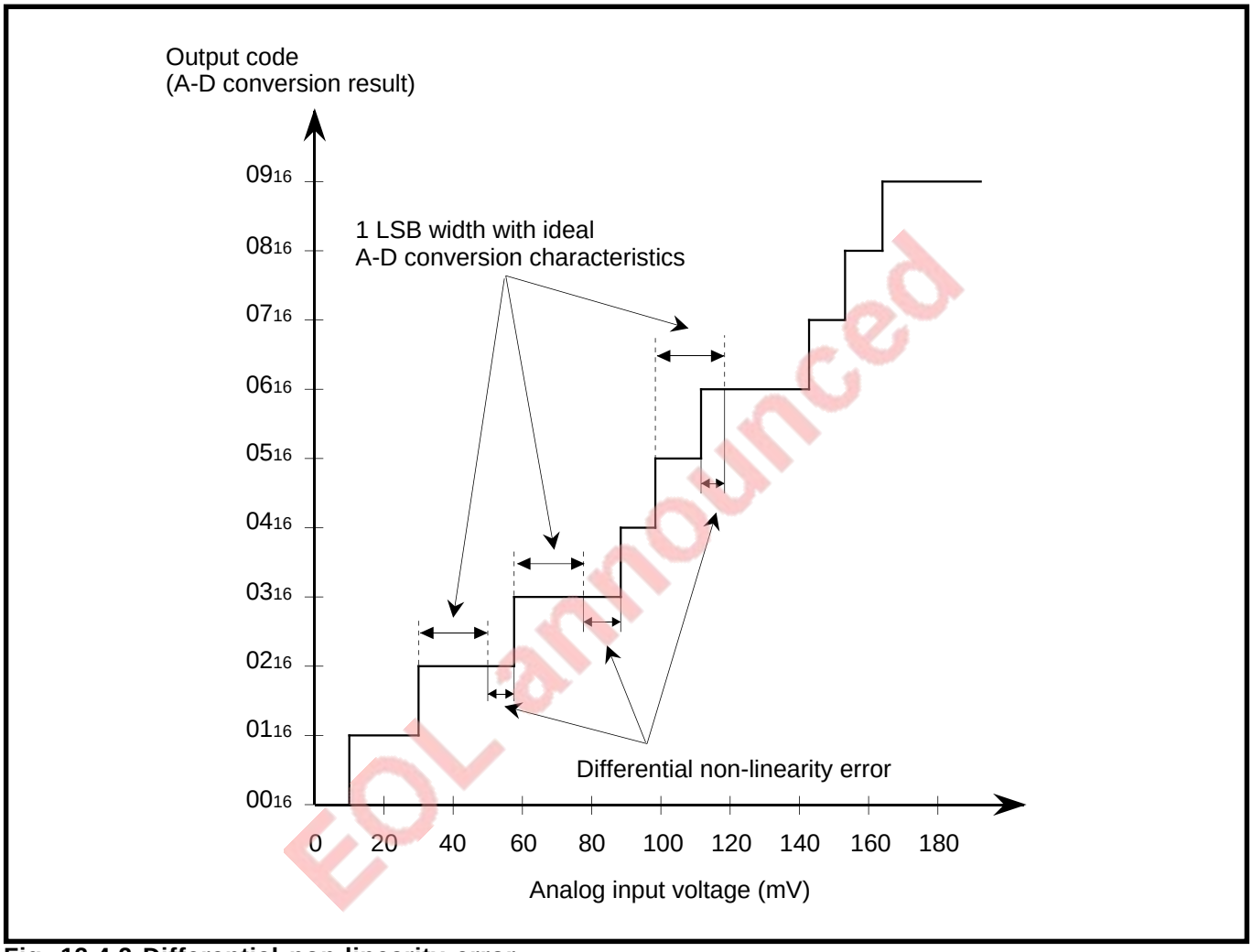


Fig. 12.4.1 Absolute accuracy of A-D converter

12.4.2 Differential non-linearity error

The differential non-linearity error indicates the difference between the 1 LSB step width (the ideal analog input voltage width while the same output code is expected to output) of an A-D converter with ideal characteristics and the actual measured step width (the actual analog input voltage width while the same output code is output). For example, when V_{REF} = 5.12 V, the 1 LSB width of an A-D converter with ideal characteristics is 20 mV, however when the differential non-linearity error is ±1 LSB, the actual measured 1 LSB width is 0 to 40 mV.



EOL announced

Fig. 12.4.2 Differential non-linearity error

12.5 One-shot mode

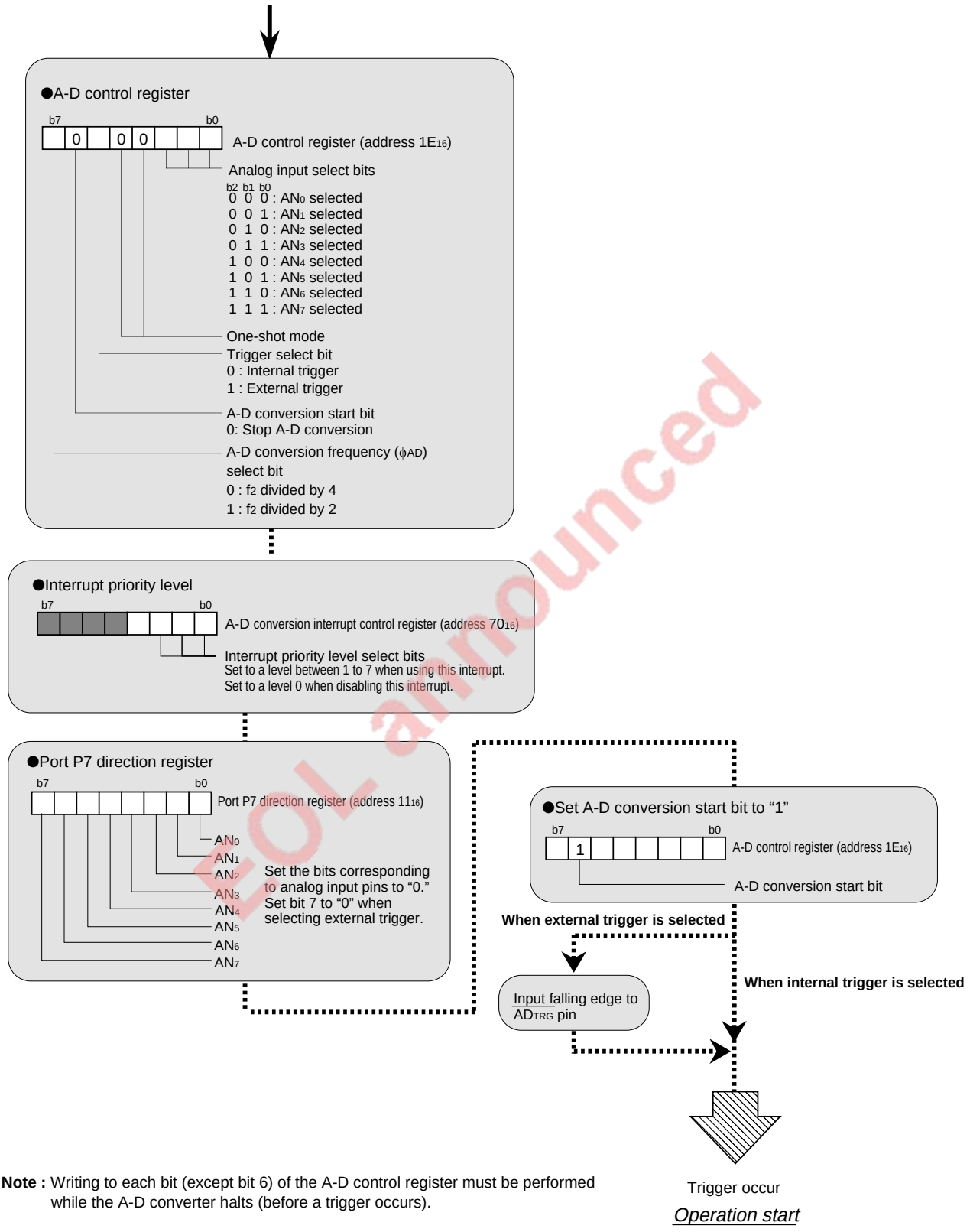
In the one-shot mode, the operation for the input voltage from the one selected analog input pin is performed once, and the A-D conversion interrupt request occurs when the operation is completed.

12.5.1 Settings for one-shot mode

Figure 12.5.1 shows an initial setting example for registers relevant to the one-shot mode.

When using an interrupt, it is necessary to set the relevant registers to enable the interrupt. Refer to **"CHAPTER 7. INTERRUPTS"** for more descriptions.

EOL announced



EOL announced

Note : Writing to each bit (except bit 6) of the A-D control register must be performed while the A-D converter halts (before a trigger occurs).

Fig. 12.5.1 Initial setting example for registers relevant to one-shot mode

12.5 One-shot mode

12.5.2 One-shot mode operation description

(1) When an internal trigger is selected

① The A-D converter starts operation when the A-D conversion start bit is set to “1.”

② The A-D conversion is completed after 57 cycles of ϕ_{AD}. Then, the contents of the successive approximation register (conversion result) are transferred to the A-D register i.

③ At the same time as step ②, the A-D conversion interrupt request bit is set to “1.”

④ The A-D conversion start bit is cleared to “0” and the A-D converter stops operation.

(2) When an external trigger is selected

① The A-D converter starts operation when the input level to the $\overline{AD_{TRG}}$ pin changes from “H” to “L” while the A-D conversion start bit is “1.”

② The A-D conversion is completed after 57 cycles of ϕ_{AD}. Then, the contents of the successive approximation register (conversion result) are transferred to the A-D register i.

③ At the same time as step ②, the A-D conversion interrupt request bit is set to “1.”

④ The A-D conversion stops.

The A-D conversion start bit remains set to “1” after the operation is completed. Accordingly, the operation of the A-D converter can be performed again from step ① when the level of the $\overline{AD_{TRG}}$ pin changes from “H” to “L.”
When the level of the $\overline{AD_{TRG}}$ pin changes from “H” to “L” during operation, the operation at that point is cancelled and is restarted from step ①.

Figure 12.5.2 shows the conversion operation in the one-shot mode.

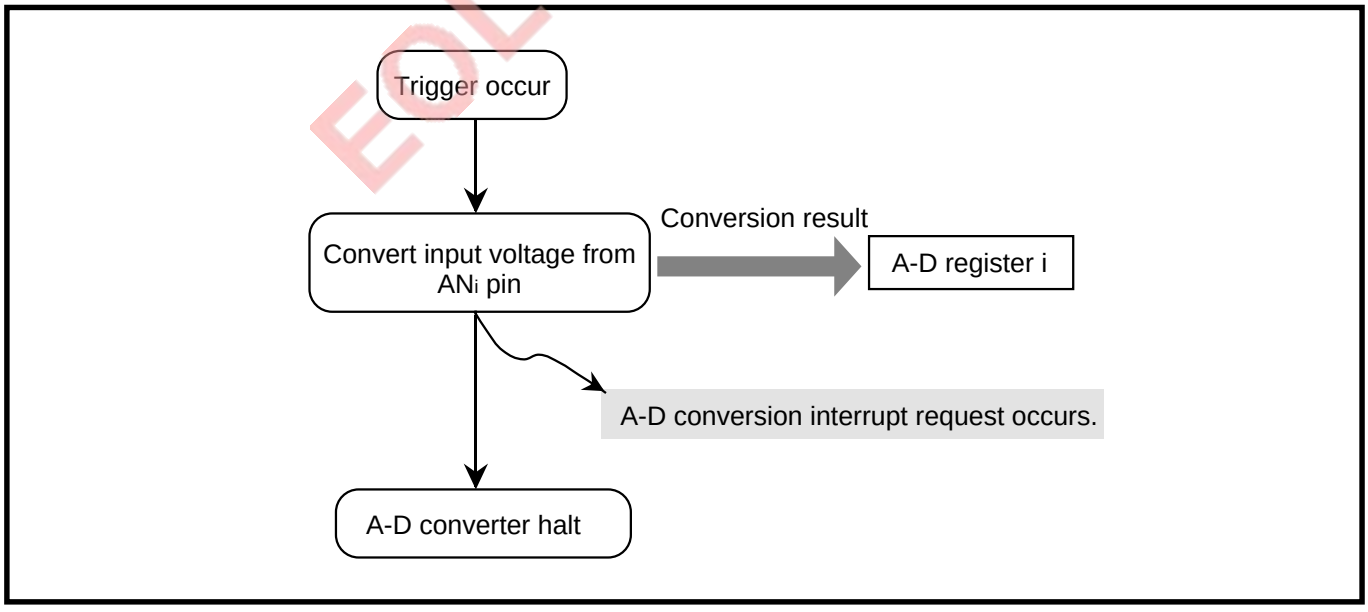


Fig. 12.5.2 Conversion operation in one-shot mode

EOL announced

12.6 Repeat mode

In the repeat mode, the operation for the input voltage from the one selected analog input pin is performed repeatedly.

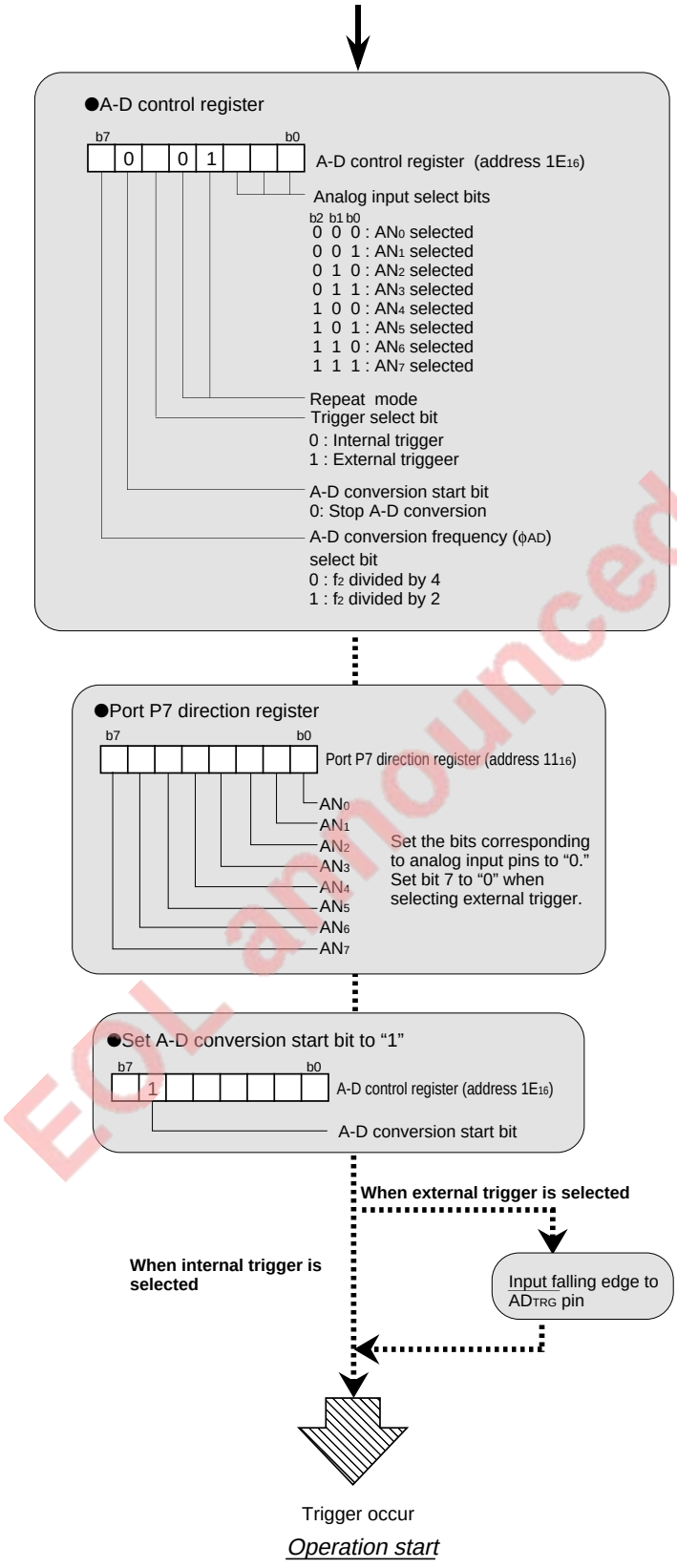
In this mode, no A-D conversion interrupt request occurs. Additionally, the A-D conversion start bit (bit 6 at address $1E_{16}$) remains set to “1” until it is cleared to “0” by software, and the operation is performed repeatedly while the A-D conversion start bit is “1.”

12.6.1 Settings for repeat mode

Figure 12.6.1 shows an initial setting example for registers relevant to the repeat mode.

EOL announced

12.6 Repeat mode



EOL announced

Note : Writing to each bit (except bit 6) of the A-D control register must be performed while the A-D converter halts (before a trigger occurs).

Fig. 12.6.1 Initial setting example for registers relevant to repeat mode

12.6.2 Repeat mode operation description

(1) When an internal trigger is selected

① The A-D converter starts operation when the A-D conversion start bit is set to “1.”

② The first A-D conversion is completed after 57 cycles of ϕ_{AD}. Then, the contents of the successive approximation register (conversion result) are transferred to the A-D register i.

③ The A-D converter repeats operation until the A-D conversion start bit is cleared to “0” by software. The conversion result is transferred to the A-D register i each time the conversion is completed.

(2) When an external trigger is selected

① The A-D converter starts operation when the input level to the $\overline{AD_{TRG}}$ pin changes from “H” to “L” while the A-D conversion start bit is “1.”

② The first A-D conversion is completed after 57 cycles of ϕ_{AD}. Then, the contents of the successive approximation register (conversion result) are transferred to the A-D register i.

③ The A-D converter repeats operation until the A-D conversion start bit is cleared to “0” by software. The conversion result is transferred to the A-D register i each time the conversion is completed.

When the level of the $\overline{AD_{TRG}}$ pin changes from “H” to “L” during operation, the operation at that point is cancelled and is restarted from step ①.

Figure 12.6.2 shows the conversion operation in the repeat mode.

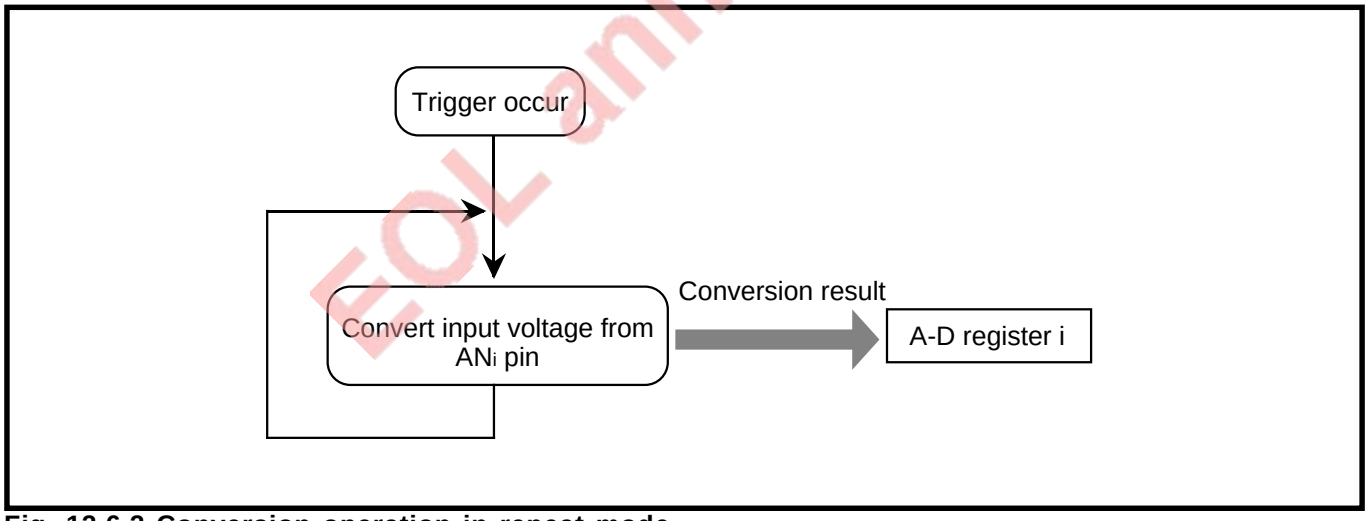


Fig. 12.6.2 Conversion operation in repeat mode

EOL announced

12.7 Single sweep mode

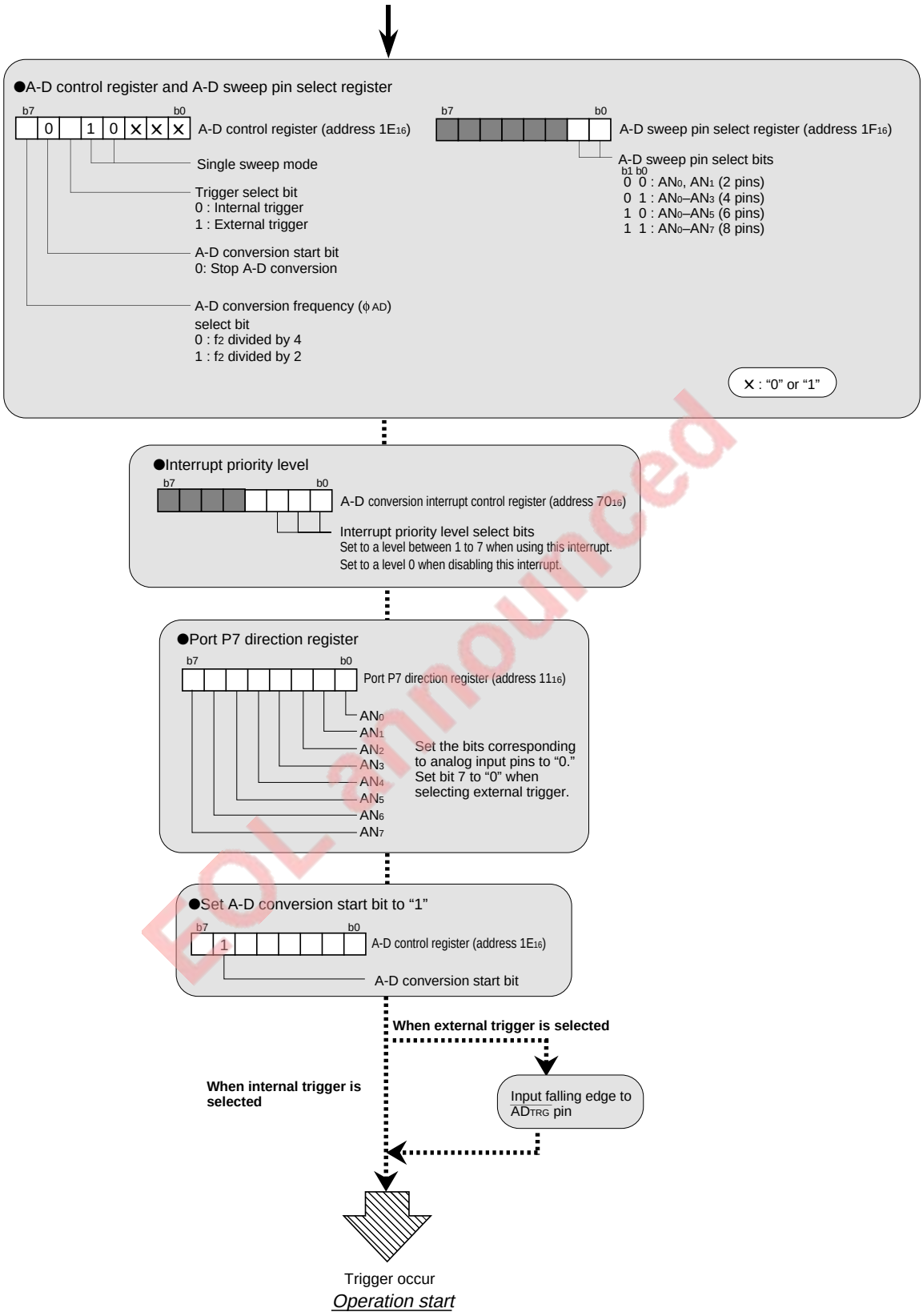
In the single sweep mode, the operation for the input voltage from multiple selected analog input pins is performed, one at a time. The A-D converter is operated in ascending sequence from the AN_0 pin. The A-D conversion interrupt request occurs when the operation for all selected input pins are completed.

12.7.1 Settings for single sweep mode

Figure 12.7.1 shows an initial setting example for registers relevant to the single sweep mode.

When using an interrupt, it is necessary to set the relevant registers to enable the interrupt. Refer to **"CHAPTER 7. INTERRUPTS"** for more information.

EOL announced



EOL announced

Note : Writing to each bit (except bit 6) of the A-D control register and each bit of the A-D sweep pin select register must be performed while the A-D converter halts (before a trigger occurs).

Fig. 12.7.1 Initial setting example for registers relevant to single sweep mode

12.7 Single sweep mode

12.7.2 Single sweep mode operation description

(1) When an internal trigger is selected

① The operation for the input voltage from the AN_0 pin starts when the A-D conversion start bit is set to "1."

② The A-D conversion of the input voltage from the AN_0 pin is completed after 57 cycles of ϕ_{AD}. Then, the contents of the successive approximation register (conversion result) are transferred to the A-D register 0.

③ For all of the selected analog input pins, the A-D conversion is performed.
The conversion result is transferred to the A-D register i each time each pin is converted.

④ When step ③ is completed, the A-D conversion interrupt request bit is set to "1."

⑤ The A-D conversion start bit is cleared to "0" and the A-D converter stops operation.

(2) When an external trigger is selected

① The A-D converter starts operation for the input voltage from the AN_0 pin when the input level to the $\overline{AD_{TRG}}$ pin changes from "H" to "L" while the A-D conversion start bit is "1."

② The A-D conversion of the input voltage from the AN_0 pin is completed after 57 cycles of ϕ_{AD}. Then, the contents of the successive approximation register (conversion result) are transferred to the A-D register 0.

③ For all of the selected analog input pins, the A-D conversion is performed.
The conversion result is transferred to the A-D register i each time each pin is converted.

④ When step ③ is completed, the A-D conversion interrupt request bit is set to "1."

⑤ The A-D conversion stops.

The A-D conversion start bit remains set to "1" after the operation is completed. Accordingly, the operation of the A-D converter can be performed again from step ① when the level of the $\overline{AD_{TRG}}$ pin changes from "H" to "L."
When the level of the $\overline{AD_{TRG}}$ pin changes from "H" to "L" during operation, the operation at that point is cancelled and is restarted from step ①.

Figure 12.7.2 shows the conversion operation in the single sweep mode.

FOL announced

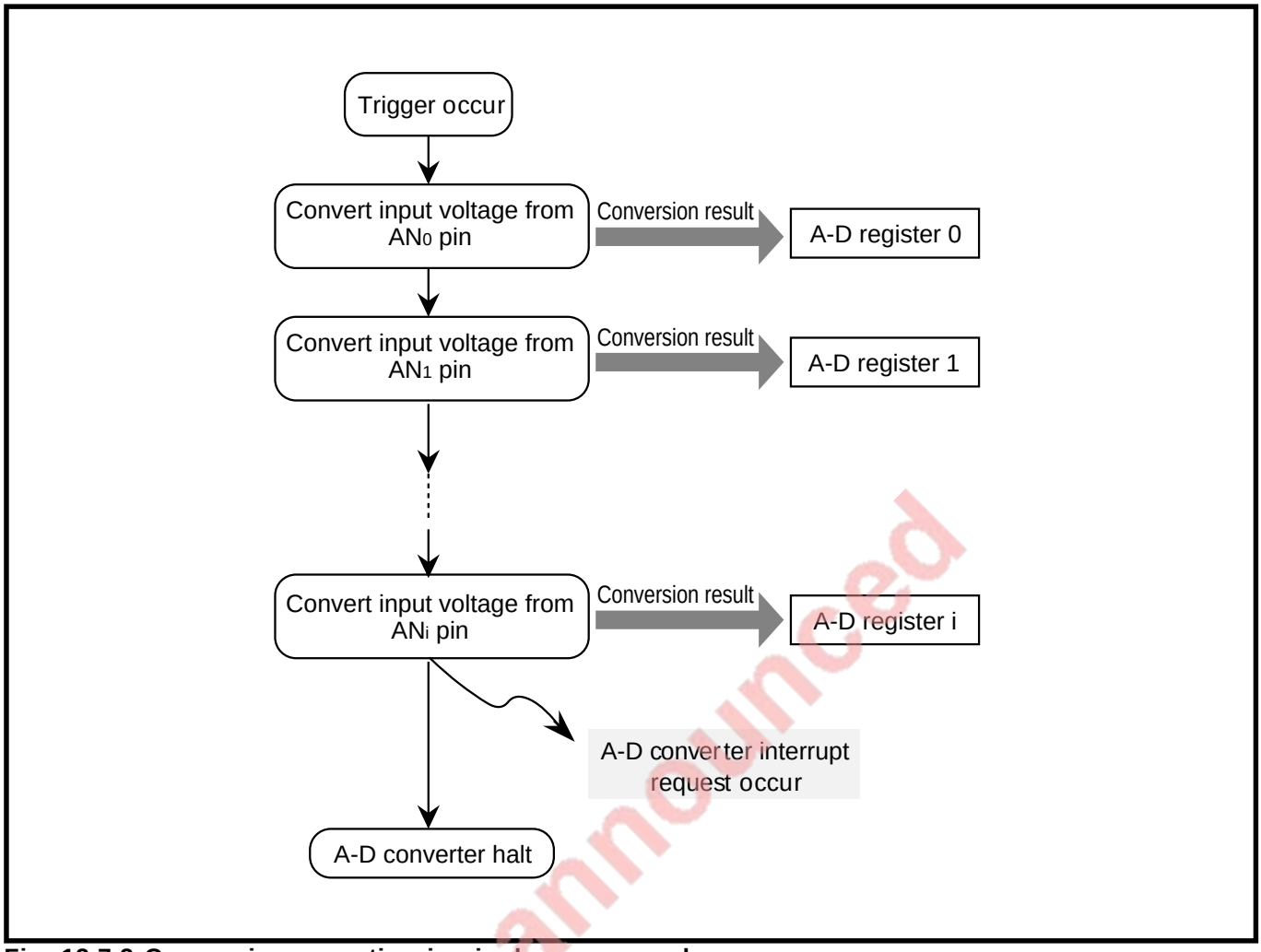


Fig. 12.7.2 Conversion operation in single sweep mode

EOL announced

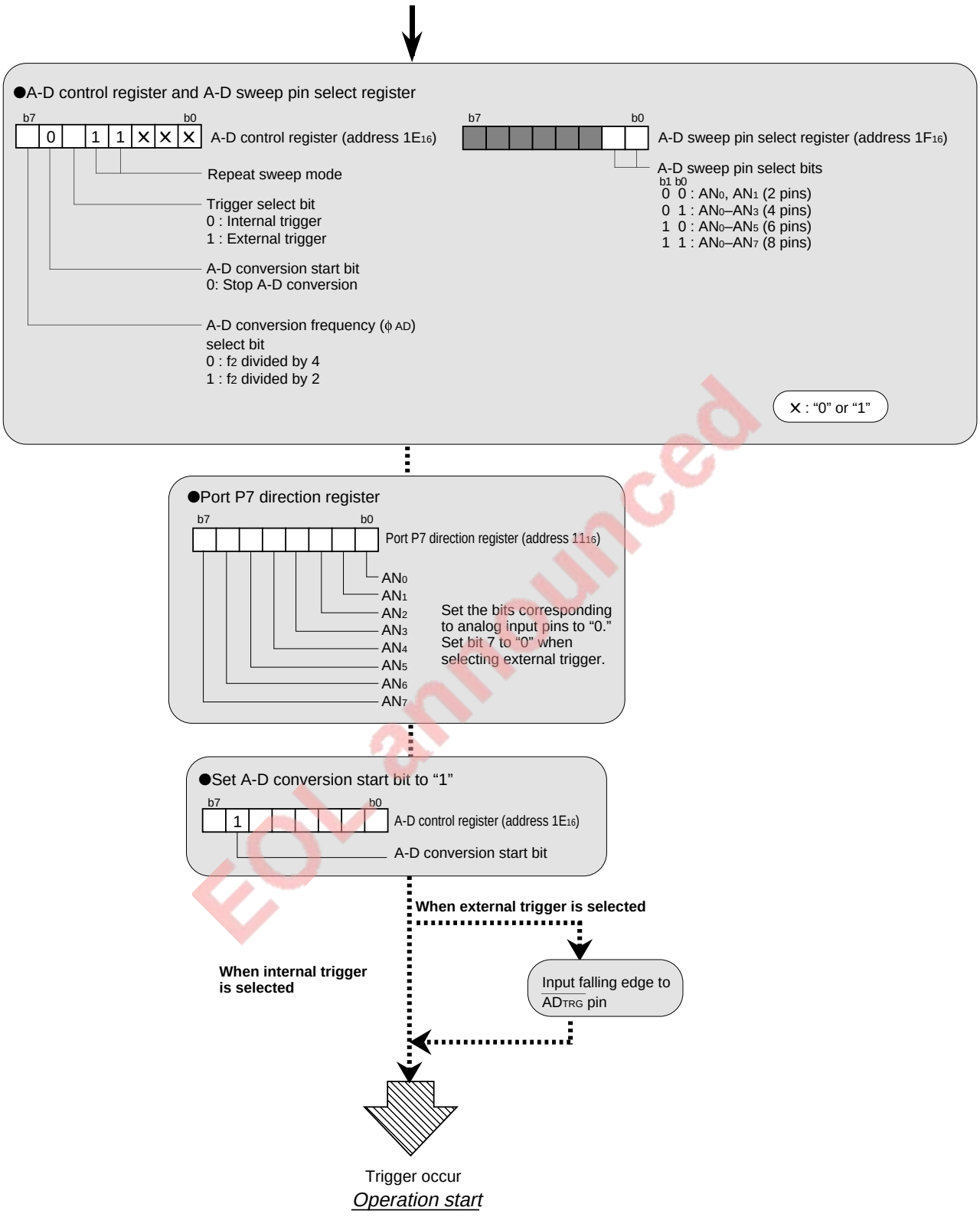
12.8 Repeat sweep mode

In the repeat sweep mode, the operation for the input voltages from the multiple selected analog input pins is performed repeatedly. The A-D converter is operated in ascending sequence from the AN_0 pin.
In this mode, no A-D conversion interrupt request occurs. Additionally, the A-D conversion start bit (bit 6 at address $1E_{16}$) remains set to "1" until it is cleared to "0" by software, and the operation is performed repeatedly while the A-D conversion start bit is "1."

12.8.1 Settings for repeat sweep mode

Figure 12.8.1 shows an initial setting example for registers relevant to the repeat sweep mode.

EOL announced



Note : Writing to each bit (except bit 6) of the A-D control register and each bit of the A-D sweep pin select register must be performed while the A-D converter halts (before a trigger occurs).

Fig. 12.8.1 Initial setting example for registers relevant to repeat sweep mode

EOL announced

12.8.2 Repeat sweep mode operation description

(1) When an internal trigger is selected

① The operation for the input voltage from the AN_0 pin starts when the A-D conversion start bit is set to "1."

② The A-D conversion of the input voltage from the AN_0 pin is completed after 57 cycles of ϕ_{AD}. Then, the contents of the successive approximation register (conversion result) are transferred to the A-D register 0.

③ For all of the selected analog input pins, the A-D conversion is performed.
The conversion result is transferred to the A-D register i each time each pin is converted.

④ For all of the selected analog input pins, the A-D conversion is performed again.

⑤ The operation is performed repeatedly until the A-D conversion start bit is cleared to "0" by software.

(2) When an external trigger is selected

① The A-D converter starts operation for the input voltage from the AN_0 pin when the input level to the $\overline{AD_{TRG}}$ pin changes from "H" to "L" while the A-D conversion start bit is "1."

② The A-D conversion of the input voltage from the AN_0 pin is completed after 57 cycles of ϕ_{AD}. Then, the contents of the successive approximation register (conversion result) are transferred to the A-D register 0.

③ For all of the selected analog input pins, the A-D conversion is performed.
The conversion result is transferred to the A-D register i each time each pin is converted.

④ For all of the selected analog input pins, the A-D conversion is performed again.

⑤ The operation is performed repeatedly until the A-D conversion start bit is cleared to "0" by software.

When the level of the $\overline{AD_{TRG}}$ pin changes from "H" to "L" during operation, the operation at that point is cancelled and is restarted from step ①.

Figure 12.8.2 shows the conversion operation in the repeat sweep mode.

EOL announced

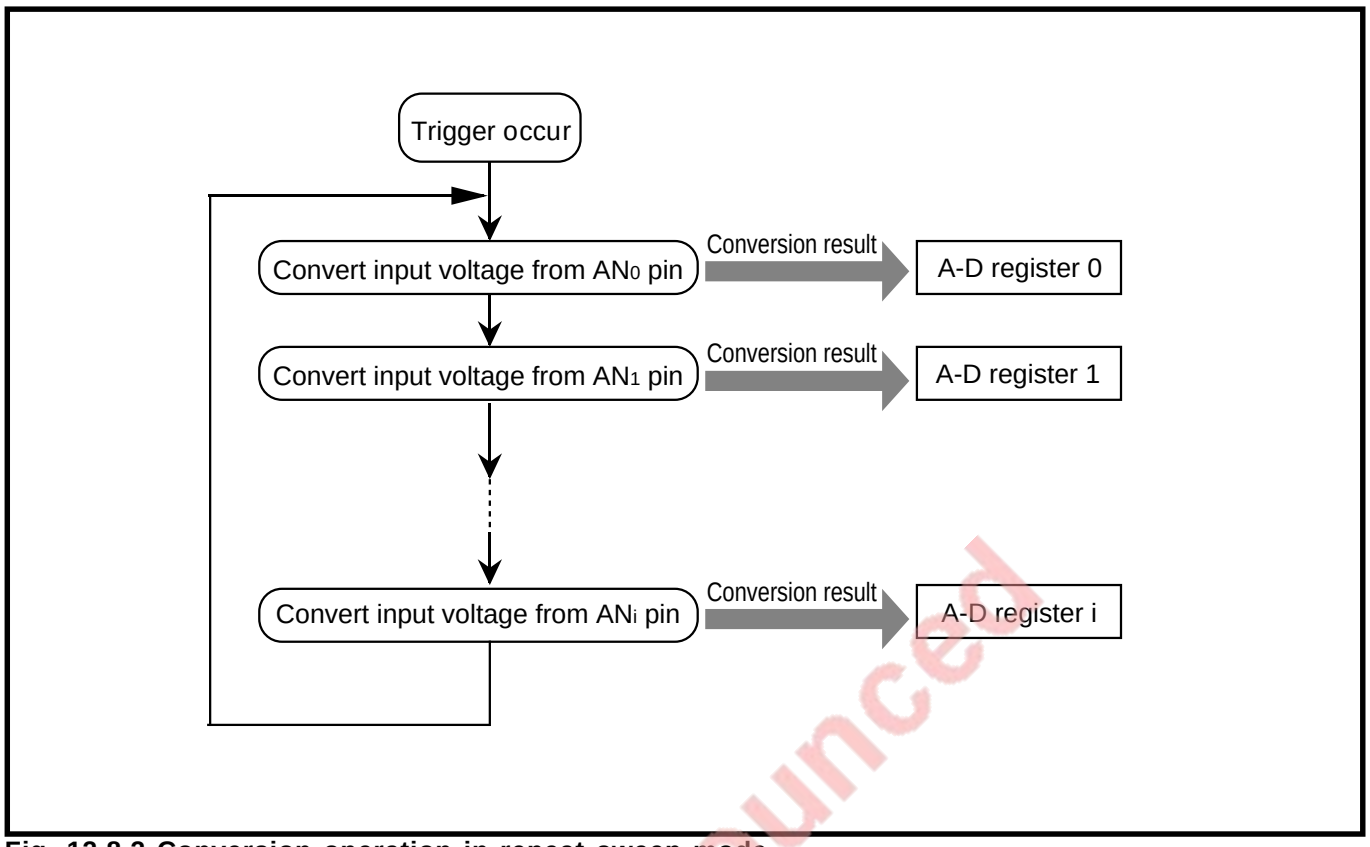


Fig. 12.8.2 Conversion operation in repeat sweep mode

EOL announced

12.9 Precautions for A-D converter

1. Writing to the following must be performed before a trigger occurs (while the A-D converter halts).
- •Each bit (except bit 6) of the A-D control register
- •Each bit of the A-D sweep pin select register

2. When an external trigger is selected, the $AN_7/\overline{AD_{TRG}}$ pin is disconnected from the comparator. Therefore, this pin cannot be used as an analog input pin.
When the AN_7 pin is selected as an analog input pin while an external trigger is selected, the A-D converter operates, however, an undefined value is stored into the A-D register 7.

3. Refer to **"Appendix. 8 Countermeasure against noise"** when using the A-D converter.

EOL announced

CHAPTER 13

DMA CONTROLLER

EOL announced

13.1 Overview
13.2 Block description
[Precautions for DMAC]
13.3 Control
13.4 Operation
[Precautions for 2-bus cycle transfer]
[Precautions for 1-bus cycle transfer]
[Precautions for burst transfer mode]
[Precautions for cycle-steal transfer mode]
13.5 Single transfer mode
13.6 Repeat transfer mode
13.7 Array chain transfer mode
[Precautions for array chain transfer mode]
13.8 Link array chain transfer mode
[Precautions for link array chain transfer mode]
13.9 DMA transfer time

13.1 Overview

The DMA controller (hereafter called DMAC) transfers data using the bus and bypassing the CPU. DMAC of the M37721 provides four independent channels of DMA0–DMA3, which have the same function each. In this chapter, the source and destination of each DMA transfer are represented as follows.

- Memory
  A device which needs its own address to be specified
  Examples: Internal RAM and SFRs, external memory, and memory-mapped I/Os
- I/O
  A device which does not need its own address to be specified
  Example: External I/O devices

13.1.1 Performance overview

Table 13.1.1 lists the performance overview.

Table 13.1.1 DMAC performance overview

Item	Performance specifications
Number of channels	4 channels
Transfer space	16 Mbytes (between arbitrary spaces)
Number of transfer bytes	Maximum of 16 Mbytes
DMA request source	Internal 14 sources and External 1 source
Channel priority	Fixed or Rotating
Transfer rate	Maximum of 12.5 Mbytes/sec (at $f(X_{IN})$ = 25 MHz, 1-bus cycle transfer) Maximum of 6.25 Mbytes/sec (at $f(X_{IN})$ = 25 MHz, 2-bus cycle transfer)
Data transfer method	1-bus cycle or 2-bus cycle transfer
Transfer unit	8 or 16 bits
Address direction of transfer	Fixed, Forward, or Backward (Directions of source and destination are independently selectable.)
Transfer mode	Burst transfer or Cycle-steal transfer mode
Continuous transfer mode	Single transfer, Repeat transfer, Array chain transfer, or Link array chain transfer mode

EOL announced

13.1.2 Bus use priority levels

The bus use priority levels are fixed by hardware as follows:

DRAMC > Hold function > DMAC > CPU
(DRAM refresh)

Because DMAC has the third priority, it actually operates as follows:

- **When DRAM refresh request or Hold request is generated during DMA transfer**
  After the transfer of one transfer unit (8-bit or 16-bit data), which is being performed at that time, is complete, DMAC relinquishes the bus to a DRAM refresh or a Hold function.
  When DMAC regains the right to use bus after the DRAM refresh ends or the Hold state is removed, DMA transfer is restarted at the following address.
- **When DMA request is generated during DRAM refresh or in Hold state**
  DMAC gains the right to use bus after the DRAM refresh ends or the Hold state is removed.
- **When DMA request is generated while CPU uses bus**
  Upon end of the bus cycle, DMAC gains the right to use bus if any DRAM refresh request or Hold request is not generated at that time.
  If a DRAM refresh request or a Hold request is generated when the bus cycle ends, DMAC gains the right to use bus after the DRAM refresh ends or the Hold state is removed.

For details, refer to section **"13.2.1 Bus access control circuit"** and bus request sampling signals in timing diagrams.

13.1.3 Modes

DMAC has the following transfer methods and modes. Because these methods and modes are independent each other, any combination between them is selectable.

(1) Data transfer method

■ **2-bus cycle transfer**

This is a method used to transfer data between memories. A DMA transfer consumes 2 cycles: a read and a write cycle of data.
For details, refer to section **"13.4.1 2-bus cycle transfer."**

■ **1-bus cycle transfer**

This is a method used to transfer data between a memory and an I/O. A read and write of data is carried out at the same time (in 1-bus cycle), so that high-speed transfer can be accomplished.
For details, refer to section **"13.4.2 1-bus cycle transfer."**

(2) Transfer unit

■ **8-bit transfer**

A minimum unit of DMA transfer is 8 bits; that is, an 8-bit data is transferred for one DMA request in the cycle-steal transfer mode.
In the burst transfer mode, if a DRAM refresh request or a Hold request is generated during DMA transfer, or if $\overline{TC}$ input is driven from "H" to "L" to force DMA transfer into termination, DMAC relinquishes the bus after completion of 8-bit data transfer which is being performed at that time.

■ **16-bit transfer**

A minimum unit of DMA transfer is 16 bits; that is, a 16-bit data is transferred for one DMA request in the cycle-steal transfer mode.
In the burst transfer mode, if a DRAM refresh request or a Hold request is generated during DMA transfer, or if $\overline{TC}$ input is driven from "H" to "L" to force DMA transfer into termination, DMAC relinquishes the bus after completion of 16-bit data transfer which is being performed at that time.

EOL announced

(3) Transfer modes

■ **Burst transfer mode**

When once a DMA request is accepted in this mode, an entire batch of data is transferred. Neither is the right to use bus returned to the CPU, nor the DMA request of the channel with the higher priority is accepted until the transfer is complete. However, if an external source ($\overline{\text{DMAREQ}}$i) is selected as a DMA request source with the level sense selected, DMA transfer is performed when the $\overline{\text{DMAREQ}}$i pin's input level is "L," and the right to use bus is returned to the CPU when the $\overline{\text{DMAREQ}}$i pin's input level is "H." Even in this case, any DMA request of the other channels is not accepted until the entire batch of data has been transferred.

For details, refer to section **"13.4.3 Burst transfer mode."**

■ **Cycle-steal transfer mode**

For each DMA request, 1 transfer unit of data is transferred. (Hereafter, transferring 1-transfer-unit data, which is 8-bit or 16-bit data in the M37721, is called "1-unit transfer.")

When 1-unit transfer is complete and another DMA request (including that of other channels) is not generated, the DMAC relinquishes the right to use bus to the CPU.

In the cycle-steal transfer mode, all of the DMA request sources are available.

For details, refer to section **"13.4.4 Cycle-steal transfer mode."**

Figure 13.1.1 shows the outline of the DMA transfer modes.

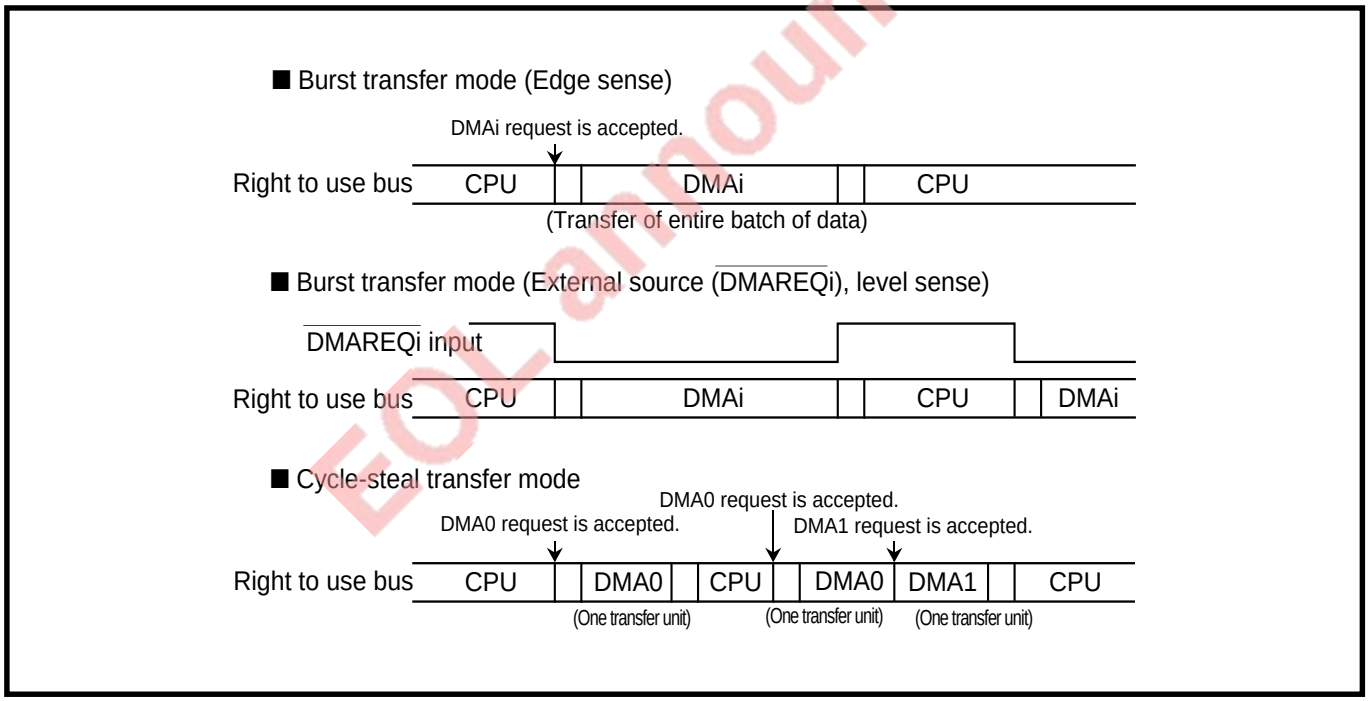


Fig. 13.1.1 Outline of DMA transfer modes

EOL announced

(4) Continuous transfer mode

■ **Single transfer mode**

1 block of data is transferred once.
For details, refer to section **"13.5 Single transfer mode."**

■ **Repeat transfer mode**

1 block of data is transferred repeatedly.
For details, refer to section **"13.6 Repeat transfer mode."**

■ **Array chain transfer mode**

Several blocks of data are transferred.
The transfer parameters (transfer source and destination addresses, the number of transfer bytes) of each block must be located on the memory in series.
For details, refer to section **"13.7 Array chain transfer mode."**

■ **Link array chain transfer mode**

Several blocks of data are transferred.
Transfer parameters for each block can be located on the memory in separate, in a unit of 1-block's parameters.
For details, refer to section **"13.8 Link array chain transfer mode."**

EOL announced

13.2 Block description

Figures 13.2.1 and 13.2.2 show the DMAC block diagrams, and relevant registers are described below.

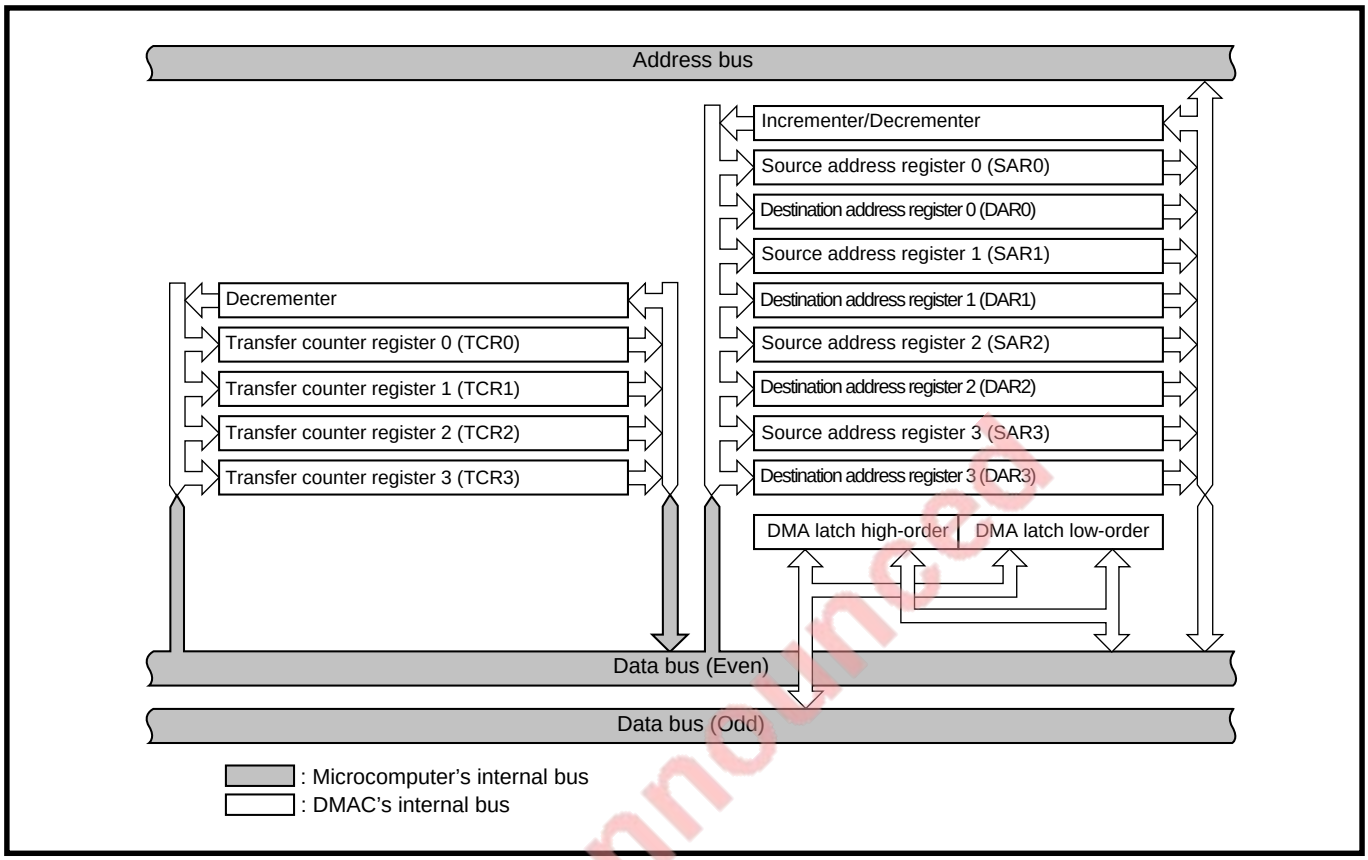


Fig. 13.2.1 DMAC block diagram (1)

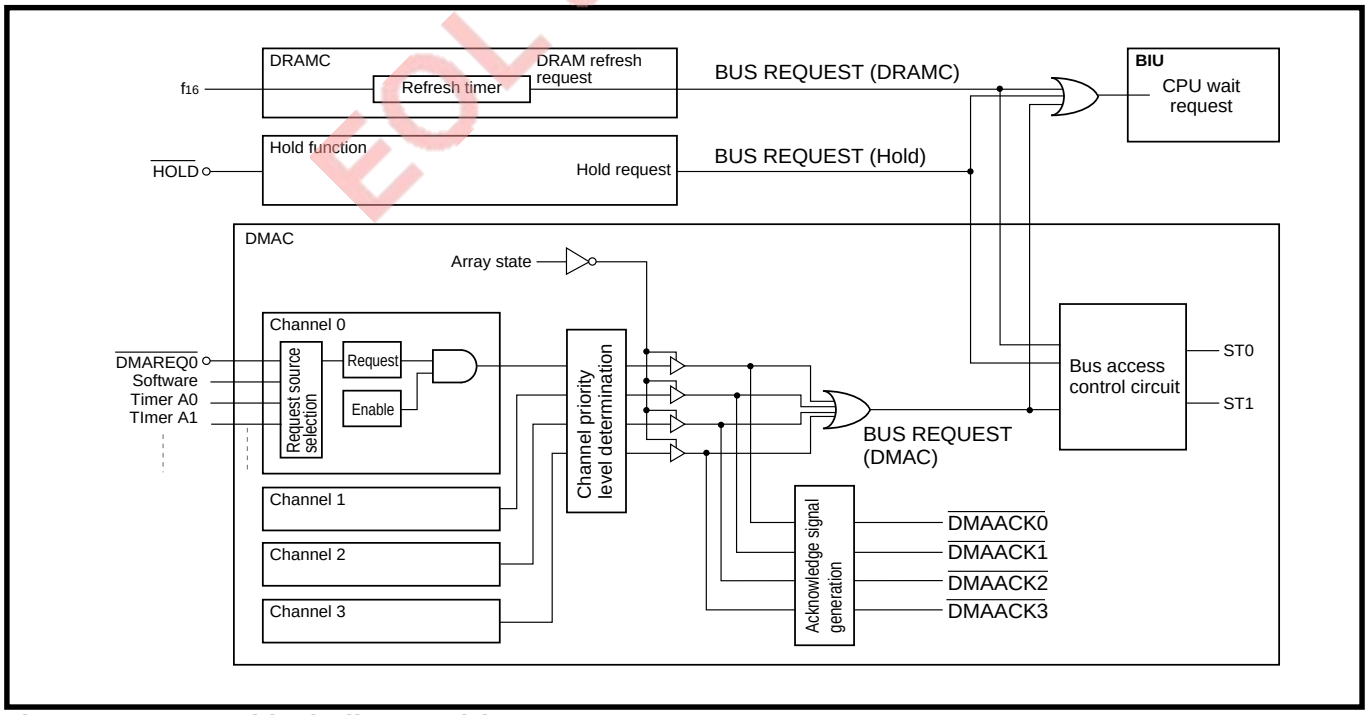


Fig. 13.2.2 DMAC block diagram (2)

EOL announced

13.2.1 Bus access control circuit

In the M37721, the bus is used by DRAMC, Hold function, DMAC, and CPU.
When each request of DRAM refresh, Hold, and DMA is generated, each of DRAMC, Hold function, and DMAC issues its bus request to the bus access control circuit in DMAC. (Refer to **"Figure 13.2.2."**)
Table 13.2.1 lists the bus request generating sources.

Table 13.2.1 Bus request generating sources

Bus request	Bus request generating source
BUS REQUEST (DRAMC)	DRAM refresh request (Generated by an underflow of the refresh timer.)
BUS REQUEST (Hold)	Hold request (Generated by "L"-level input to the $\overline{\text{HOLD}}$ pin.)
BUS REQUEST (DMAC)	DMA request (Generated by a DMA request source.)

The bus access control circuit relinquishes the right to use bus to the function with the highest priority among functions, which issue bus requests when the BUS REQUEST signal is sampled. This is the bus request acceptance. If any bus request is not generated at bus request sampling, the CPU gains the right to use bus.
The bus use priority levels are fixed by hardware, and the bus status is reported by status signal outputs ST0 and ST1. Table 13.2.2 lists the relationship between the bus use priority level, bus status, and status signals.

Table 13.2.2 Relationship between bus use priority level, bus status, and status signals

Bus use priority level	Bus status	Status signals	
		ST1	ST0
1 (Highest)	DRAM refresh	0	0
2	Hold	0	1
3	DMAC	1	0
4 (Lowest)	CPU (Including the term while the CPU does not use the bus; for example, the term when the CPU is calculating and does not use the bus)	1	1

EOL announced

13.2 Block description

The BUS REQUEST signal is sampled at a break in bus use. Table 13.2.3 and Figure 13.2.3 shows the timings of bus request sampling. Also, bus request sampling signals are shown in them.

Table 13.2.3 Bus request sampling timing

Bus user		Bus request sampling timing	
DRAM refresh		After completion of a DRAM refresh cycle	
Hold		Every 1 cycle of ϕ	
DMAC	S/R **(Note 1)**	All except the following	After completion of 1-unit transfer
		At the end of each block	After 1-unit transfer and terminate-processing (3 cycles of ϕ) etc. are performed sequentially
	Array/ Link **(Note 1)**	All except the following	After completion of 1-unit transfer
		At the end of each block (except the last block)	■ **During transfer in burst transfer mode** After the last 1-unit transfer of 1 block, the subsequent 3 cycles of ϕ, and a read of the first 2 bytes in the array state of the next block are performed sequentially ■ **During transfer in cycle-steal transfer mode** After the last 1-unit transfer of 1 block and the subsequent 3 cycles of ϕ are performed sequentially
		At the end of the last block	After 1-unit transfer and terminate-processing (3 cycles of ϕ) are performed sequentially
		At an array state	After a read of 2 bytes of a transfer parameter
CPU		When an instruction is fetched into queue buffer	After completion of 1 bus cycle
		At a read from or a write into memory	After completion of 1 bus cycle, or after completion of the second bus cycle if a 16-bit data is accessed in a unit of 8 bits **(Note 2)**.
		While CPU does not use bus	Every 1 cycle of ϕ

Notes 1: S = Single transfer mode, R = Repeat transfer mode, Array = Array chain transfer mode, Link = Link array chain transfer mode

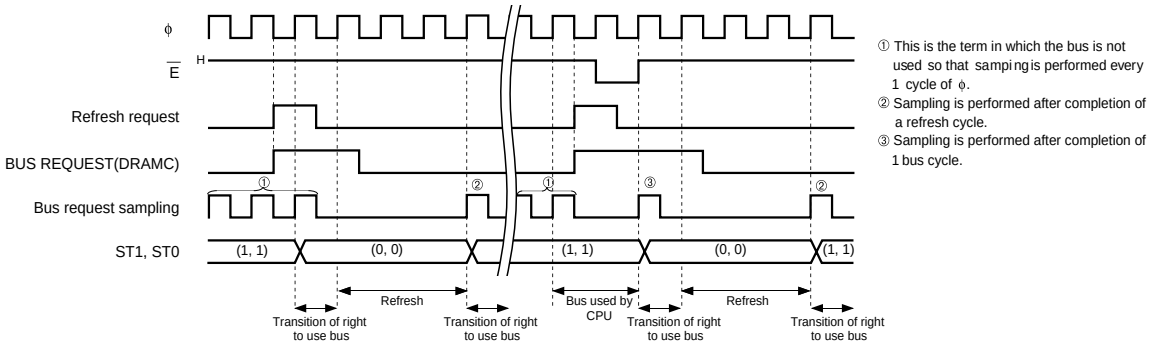
2: This applies when the data bus width is 8 bits or when memory is accessed starting at an odd address.

EOL announced

If a DRAM refresh request or a Hold request is generated during a data transfer in the burst transfer mode, the request is accepted at the above-mentioned bus request sampling. Another DMA request (including that of other channels) cannot be accepted until the DMA transfer which is in progress normally terminates or is forced into termination.

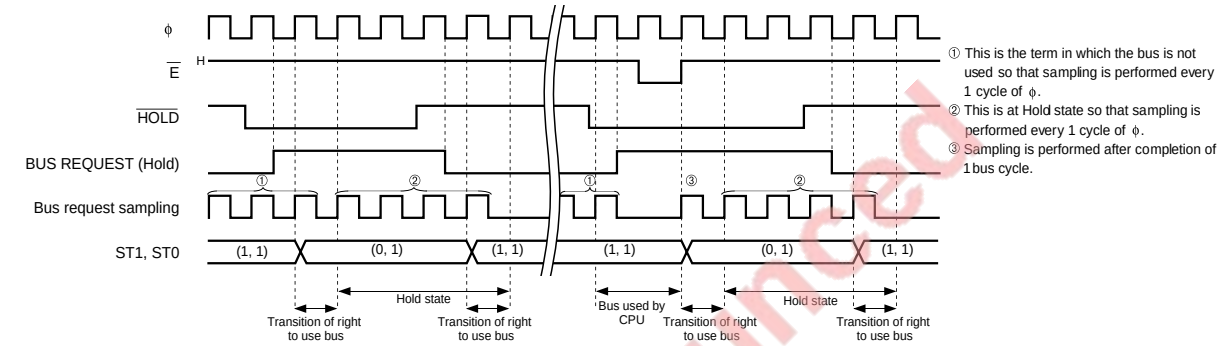
If a DRAM refresh request, a Hold request or another DMA request (including that of other channels) is generated during a data transfer in the cycle-steal transfer mode, the bus request with the highest priority is accepted at the above-mentioned bus request sampling. (If only several DMA requests are generated, the request of the channel whose priority is highest is accepted.)
If any bus request is not generated at the above-mentioned bus sampling, the right to use bus is relinquished to the CPU.
Note that no DMA request is accepted in array states.

■ DRAM refresh



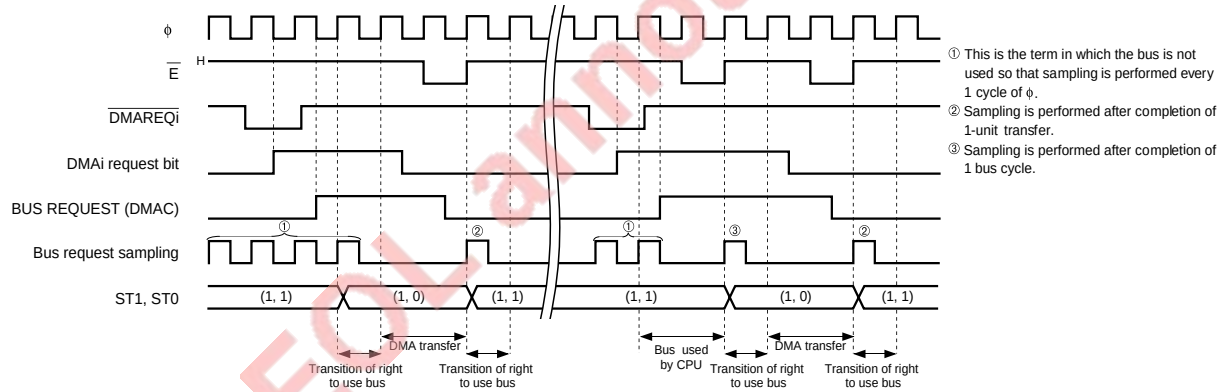
① This is the term in which the bus is not used so that samping is performed every 1 cycle of ϕ.
② Sampling is performed after completion of a refresh cycle.
③ Sampling is performed after completion of 1 bus cycle.

■ Hold



① This is the term in which the bus is not used so that sampling is performed every 1 cycle of ϕ.
② This is at Hold state so that sampling is performed every 1 cycle of ϕ.
③ Sampling is performed after completion of 1 bus cycle.

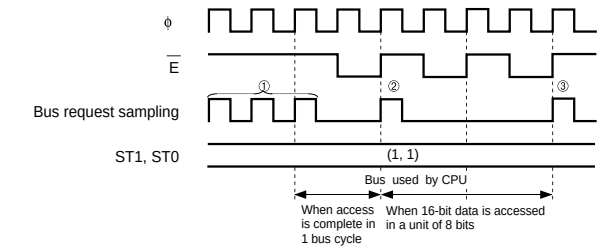
■ DMA transfer



① This is the term in which the bus is not used so that sampling is performed every 1 cycle of ϕ.
② Sampling is performed after completion of 1-unit transfer.
③ Sampling is performed after completion of 1 bus cycle.

The above applies on the following conditions:
•Cycle-steal transfer mode
•DMA request source = external request ($\overline{\text{DMAREQi}}$)
•1-bus cycle transfer
•No Wait

■ CPU



① This is the term in which the bus is not used so that sampling is performed every 1 cycle of ϕ.
② Sampling is performed after completion of 1 bus cycle.
③ 16-bit data is accessed in a unit of 8 bits, so that sampling is performed after completion of the second bus cycle.

EOL announced

Fig. 13.2.3 Timing of bus request sampling

13.2.2 DMAC control register L

Figure 13.2.4 shows the structure of DMAC control register L. Bit 0 is described in section **"13.3.3 Channel priority levels,"** and bits 4–7 are also in section **"13.3.2 DMA requests."**

(1) $\overline{TC}$ pin validity bit (Bit 1)

When this bit is set to "1," port $P10_3$ functions as the $\overline{TC}$ pin. The $\overline{TC}$ pin is of an N-channel open-drain type and provides the following functions:

- **Terminal count signal output**
  When the transfer of an entire batch of data is normally terminated, the pin outputs "L" for 1 cycle of ϕ. (Refer to section **"13.3.5 (1) Normal termination."**)

- **Forced termination signal input**
  When the $\overline{TC}$ pin's input level goes from "H" to "L" during DMA transfer, this DMA transfer is forced into termination. (Refer to section **"13.3.5 (2) Forced termination."**)



DMAC control register L (Address 68_{16})

Bit	Bit name	Functions	At reset	RW
0	Priority select bit	0 : Fixed 1 : Rotating	0	RW
1	$\overline{TC}$ pin validity bit	0 : Invalid ($P10_3$ pin functions as a programmable I/O port (CMOS).) 1 : Valid ($P10_3$ pin functions as $\overline{TC}$ pin (N-channel open-drain).)	0	RW
3, 2	Nothing is assigned.		Undefined	–
4	DMA0 request bit	0 : No request 1 : Requested **(Note 1)**	0	RW
5	DMA1 request bit		0	RW
6	DMA2 request bit		0	RW
7	DMA3 request bit		0	RW

Notes 1: The state of bits 4 to 7 is not changed when writing "1" to these bits.

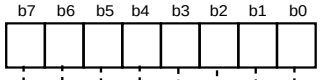
2: • When writing to this register while any of DMAi enable bits (bits 4 to 7 at address 69_{16}) is "1," use the **LDM** or **STA** instruction in m flag = "1." When DMAi request bit (bits 4 to 7 at address 68_{16}) must not be changed, set DMAi request bit to "1."
• When writing to this register while all of DMAi enable bits (bits 4 to 7 at address 69_{16}) are "0," m flag may be "0" or "1." Use the **LDM** or **STA** instruction for writing to this register. When DMAi request bit (bits 4 to 7 at address 68_{16}) must not be changed, set DMAi request bit to "1."

EOL announced

Fig. 13.2.4 Structure of DMAC control register L

13.2.3 DMAC control register H

Figure 13.2.5 shows the structure of DMAC control register H. Each of bits 0–3 is a software DMAi (i = 0 to 3) request bit, which corresponds to each channel. When a software DMA source is selected as a DMA request source, each of these bits is valid. (Refer to **"13.3.2 DMA requests.")** Bits 4–7 are described in section **"13.3.1 DMA enabling."**



DMAC control register H (Address 69_{16})

Bit	Bit name	Functions	At reset	RW
0	Software DMA0 request bit	1 : DMA request (Valid when software DMA source is selected.) The value is "0" at reading.	0	WO
1	Software DMA1 request bit		0	WO
2	Software DMA2 request bit		0	WO
3	Software DMA3 request bit		0	WO
4	DMA0 enable bit	0 : Disabled 1 : Enabled	0	RW
5	DMA1 enable bit		0	RW
6	DMA2 enable bit		0	RW
7	DMA3 enable bit		0	RW

Note: When any of bits 4 to 7 is set to "1," use the **CLB** or **SEB** instruction for writing to this register.

Fig. 13.2.5 Structure of DMAC control register H

EOL announced

13.2.4 Source address register i (SARi)

Source address register i (hereafter called SARi) is a 24-bit register with a latch.
SARi indicates the transfer source address of the data to be transferred next.
The SARi latch has the following functions:

- Maintains the value written to the address of SARi (in the single transfer and repeat transfer modes).
- Indicates the start address of the transfer parameter memory of the next block (in the array chain transfer and link array chain transfer modes).

When a value is written into the address of SARi, the same value is written into SARi and the SARi latch.
When writing a value to the address of SARi, all 24 bits must be written.
The contents of SARi can be read by reading the address of SARi; however, the value of the SARi latch cannot be read. (Refer to **"Tables 13.2.4 and 13.2.5."**)

13.2.5 Destination address register i (DARi)

Destination address register i (hereafter called DARi) is a 24-bit register with a latch.
DARi indicates the transfer destination address of the data to be transferred next.
The DARi latch maintains the value written to the address of DARi.
When a value is written into the address of DARi, the same value is written into DARi and the DARi latch.
When writing a value to the address of DARi, all 24 bits must be written.
The contents of DARi can be read by reading the address of DARi; however, the value of the DARi latch cannot be read. (Refer to **"Tables 13.2.4 and 13.2.5."**)

13.2.6 Transfer counter register i (TCRi)

Transfer counter register i (hereafter called TCRi) is a 24-bit register with a latch.
TCRi indicates the number of remaining bytes of the block under transfer.
The TCRi latch has the following functions:

- Maintains the value written to the address of TCRi (in the single transfer and repeat transfer modes).
- Indicates the number of remaining blocks (in the array chain transfer mode).

When a value is written into the address of TCRi, the same value is written into TCRi and the TCRi latch.
When writing a value to the address of TCRi, all 24 bits must be written.
The contents of TCRi can be read by reading the address of TCRi; however, the value of the TCRi latch cannot be read. (Refer to **"Tables 13.2.4 and 13.2.5."**)

Table 13.2.4 Addresses of SARi, DARi, and TCRi

Channel	Source address register i (SARi)	Destination address register i (DARi)	Transfer counter register i (TCRi)
0	$1FC2_{16}$–$1FC0_{16}$	$1FC6_{16}$–$1FC4_{16}$	$1FCA_{16}$–$1FC8_{16}$
1	$1FD2_{16}$–$1FD0_{16}$	$1FD6_{16}$–$1FD4_{16}$	$1FDA_{16}$–$1FD8_{16}$
2	$1FE2_{16}$–$1FE0_{16}$	$1FE6_{16}$–$1FE4_{16}$	$1FEA_{16}$–$1FE8_{16}$
3	$1FF2_{16}$–$1FF0_{16}$	$1FF6_{16}$–$1FF4_{16}$	$1FFA_{16}$–$1FF8_{16}$

EOL announced

Table 13.2.5 Functions of SARi, DARi, and TCRi

Register \ Mode		Single transfer mode	Repeat transfer mode	Array chain transfer mode	Link array chain transfer mode
SARi	SARi	Indicates the transfer source address of the data to be transferred next.			
	SARi latch	Maintains the transfer start address of the source.		Indicates the start address of the transfer parameter memory of the next block.	
DARi	DARi	Indicates the transfer destination address of the data next to be transferred			
	DARi latch	Maintains the transfer start address of the destination.		(Not used)	(Not used)
TCRi	TCRi	Indicates the number of remaining bytes of the block under transfer.			
	TCRi latch	Maintains the byte number of the transfer data.		Indicates the number of remaining blocks.	(Not used) **(Note)**

Note: Any value other than 0 (000001_{16}–$FFFFFF_{16}$) must be written before DAM transfer.

13.2.7 Incrementer/Decrementer

The incrementer/decrementer is a 24-bit register. After every 1-unit transfer, that increments (adds) or decrements (subtract) the contents of SARi and DARi.
Table 13.2.6 lists the increment/decrement values.

Table 13.2.6 Increment/Decrement values

Transfer unit	Address directions	
	Forward	Backward
8 bits	+ 1	– 1
16 bits	+ 2	– 2

13.2.8 Decrementer

The decrementer is a 24-bit register. After every 1-unit transfer, that decrements the contents of TCRi by 1 when the transfer unit is 8 bits, and by 2 when 16 bits.
In the array chain transfer mode, every time a transfer parameter is read, the contents of the TCRi latch are also decremented by 1.

13.2.9 DMA latch

The DMA latch is a 16-bit latch.
In 2-bus cycle transfer mode, the DMA latch maintains the value read from the transfer source memory with a read cycle until this value is written into the transfer destination memory.
In 1-bus cycle transfer mode, the DMA latch is used to copy data. For copy, refer to section **"13.4.2 1-bus cycle transfer."**

EOL announced

13.2.10 DMAi mode register L

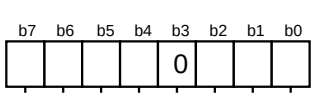
Figure 13.2.6 shows the structure of DMAi mode register L. For bit 0, refer to section **"13.1.3 (2) Transfer unit."** For bit 1, refer to section **"13.4.1 2-bus cycle transfer"** and section **"13.4.2 1-bus cycle transfer"**; for bit 2, refer to section **"13.4.3 Burst transfer mode"** and section **"13.4.4 Cycle-steal transfer mode."**

(1) Transfer source address direction select bits (bits 4 and 5) and Transfer destination address direction select bits (bits 6 and 7)

Address direction means an order of accessing memory in DMA transfer and is defined as follows:

- Fixed direction: an address does not move.
- Forward direction: an address moves upward from the specified start address.
- Backward direction: an address moves downward from the specified start address.

For details, refer to section **"13.4.1 (3) Address directions in 2-bus cycle transfer"** and section **"13.4.2 (3) Address directions in 1-bus cycle transfer."**



DMA0 mode register L (Address $1FCC_{16}$)
DMA1 mode register L (Address $1FDC_{16}$)
DMA2 mode register L (Address $1FEC_{16}$)
DMA3 mode register L (Address $1FFC_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Number-of-unit-transfer-bits select bit **(Note)**	0 : 16 bits 1 : 8 bits	0	RW
1	Transfer method select bit	0 : 2-bus cycle transfer 1 : 1-bus cycle transfer	0	RW
2	Transfer mode select bit	0 : Burst transfer mode 1 : Cycle-steal transfer mode	0	RW
3	Fix this bit to "0."		0	RW
4	Transfer source address direction select bits	b5b4 0 0 : Fixed 0 1 : Forward 1 0 : Backward 1 1 : Do not select.	0	RW
5			0	RW
6	Transfer destination address direction select bits	b7b6 0 0 : Fixed 0 1 : Forward 1 0 : Backward 1 1 : Do not select.	0	RW
7			0	RW

Note: When the external data bus has a width of 8 bits and 1-bus cycle transfer is selected, set bit 0 to "1."

EOL announced

Fig. 13.2.6 Structure of DMAi mode register L

13.2.11 DMAi mode register H

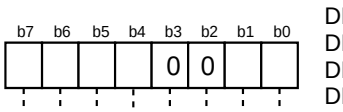
Figure 13.2.7 shows the structure of DMAi mode register H. Bits 0 and 1 are used in 1-bus cycle transfer. For details, refer to section **"13.4.2 1-bus cycle transfer."** Bits 6 and 7 are the bits for selecting the continuous transfer mode. For details, refer to section **"13.5 Single transfer mode"** through section **"13.8 Link array chain transfer mode."**

(1) Transfer source wait bit and Transfer destination wait bit (bits 4 and 5)

When each of these bits is set to "1," 1-bus cycle in a DMA transfer consumes 3 cycles of ϕ, and when cleared to "0," 2 cycles of ϕ.

These bits are valid for the internal and external areas. In the DRAM area, however, 1-bus cycle consumes 3 cycles of ϕ regardless of the states of these bits. (Refer to **"CHAPTER 14. DRAM CONTROLLER."**)

The wait bit (bit 2 at address $5E_{16}$) is invalid in DMA transfer. However, Ready function is still valid in DMA transfer.



DMA0 mode register H (Address $1FCD_{16}$)
DMA1 mode register H (Address $1FDD_{16}$)
DMA2 mode register H (Address $1FED_{16}$)
DMA3 mode register H (Address $1FFD_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Transfer direction select bit (Used in 1-bus cycle transfer)**(Note 1)**	0 : From memory to I/O 1 : From I/O to memory	0	RW
1	I/O connection select bit (Valid in 1-bus cycle transfer)	Refer to below.	0	RW
2	Fix these bits to "0."		0	RW
3			0	RW
4	Transfer source wait bit **(Note 2)**	0 : Wait 1 : No Wait	0	RW
5	Transfer destination wait bit **(Note 2)**		0	RW
6	Continuous transfer mode select bits	b7b6 0 0 : Single transfer 0 1 : Repeat transfer 1 0 : Array chain transfer 1 1 : Link array chain transfer	0	RW
7			0	RW

Notes 1: Set bit 0 to "0" in 2-bus cycle transfer.
2: Bits 4 and 5 are valid to the external and internal areas. However, DRAM area is always handled with "Wait" regardless of the contents of these bits. The wait bit (bit 2 at address $5E_{16}$) is invalid in DMA transfer.

Setting for I/O connection select bit

Transfer method	External data bus width	I/O connection	Setting for I/O connection select bit
2-bus cycle transfer	——	——	It may be either "0" or "1."
1-bus cycle transfer	8 bits	D_0–D_7	0
	16 bits	D_0–D_{15} (16-bit I/0 × 1 or 8-bit I/O × 2)	0
		D_0–D_7 (8-bit I/O)	0
		D_8–D_{15} (8-bit I/O)	1

EOL announced

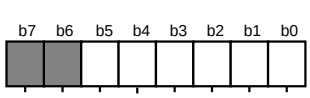
Fig. 13.2.7 Structure of DMAi mode register H

13.2.12 DMAi control register

Figure 13.2.8 shows the structure of the DMAi control register. For bits 0–4, refer to section **"13.3.2 (1) DMA request sources."**

(1) $\overline{\text{DMAACK}}$i validity bit (bit 5)

When this bit is set to "1," the corresponding pin of port P9 serves as the $\overline{\text{DMAACK}}$i pin and outputs "L" during a DMA transfer. For details, refer to each timing diagram of section **"13.5 Single transfer mode"** through section **"13.8 Link array chain transfer mode."**



DMA0 control register (Address $1FCE_{16}$)
DMA1 control register (Address $1FDE_{16}$)
DMA2 control register (Address $1FEE_{16}$)
DMA3 control register (Address $1FFE_{16}$)

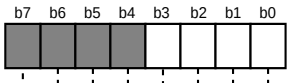
Bit	Bit name	Functions	At reset	RW
0	DMA request source select bits **(Note)**	b3b2b1b0 0 0 0 0 : Do not select. 0 0 0 1 : External source ($\overline{\text{DMAREQ}}$i) 0 0 1 0 : Software DMA source 0 0 1 1 : Timer A0 0 1 0 0 : Timer A1 0 1 0 1 : Timer A2 0 1 1 0 : Timer A3 0 1 1 1 : Timer A4 1 0 0 0 : Timer B0 1 0 0 1 : Timer B1 1 0 1 0 : Timer B2 1 0 1 1 : UART0 receive 1 1 0 0 : UART0 transmit 1 1 0 1 : UART1 receive 1 1 1 0 : UART1 transmit 1 1 1 1 : A-D conversion	0	RW
1			0	RW
2			0	RW
3			0	RW
4	Edge sense/Level sense select bit (Used when external source and burst transfer mode are selected) **(Note)**	0 : Edge sense (Falling edge) 1 : Level sense ("L" level)	0	RW
5	$\overline{\text{DMAACK}}$i validity bit	0 : Invalid (The pin functions as a programmable I/O port.) 1 : Valid (The pin functions as $\overline{\text{DMAACK}}$i.)	0	RW
7, 6	Nothing is assigned.		Undefined	–

Note: When a certain source other than an external source is selected by bits 0 to 3 or when the cycle-steal transfer mode is selected, set bit 4 to "0."
Level sense can be selected only when both of the external source and the burst transfer mode are selected.

Fig. 13.2.8 Structure of DMAi control register

13.2.13 DMAi interrupt control register

Figure 13.2.9 shows the structure of the DMAi interrupt control register. For details about interrupts, refer to **"CHAPTER 7. INTERRUPTS."**



DMAi interrupt control register (i = 0 to 3) (Addresses $6C_{16}$ to $6F_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Interrupt priority level select bits	b2 b1 b0 0 0 0 : Level 0 (Interrupt disabled) 0 0 1 : Level 1 Low level 0 1 0 : Level 2 0 1 1 : Level 3 1 0 0 : Level 4 1 0 1 : Level 5 1 1 0 : Level 6 1 1 1 : Level 7 High level	0	RW
1			0	RW
2			0	RW
3	Interrupt request bit	0 : No interrupt requested 1 : Interrupt requested	0	RW
7 to 4	Nothing is assigned.		Undefined	–

Fig. 13.2.9 Structure of DMAi interrupt control register

(1) Interrupt priority level select bits (bits 2 to 0)

These bits select a DMAi interrupt's priority level. When using DMAi interrupts, select one of the priority levels (1 to 7). When a DMAi interrupt request occurs, its priority level is compared with the processor interrupt priority level (IPL). The requested interrupt is enabled only when its priority level is higher than the IPL. (However, this applies when the interrupt disable flag (I) = "0.")
To disable DMAi interrupts, set these bits to "000_2" (level 0).

(2) Interrupt request bit (bit 3)

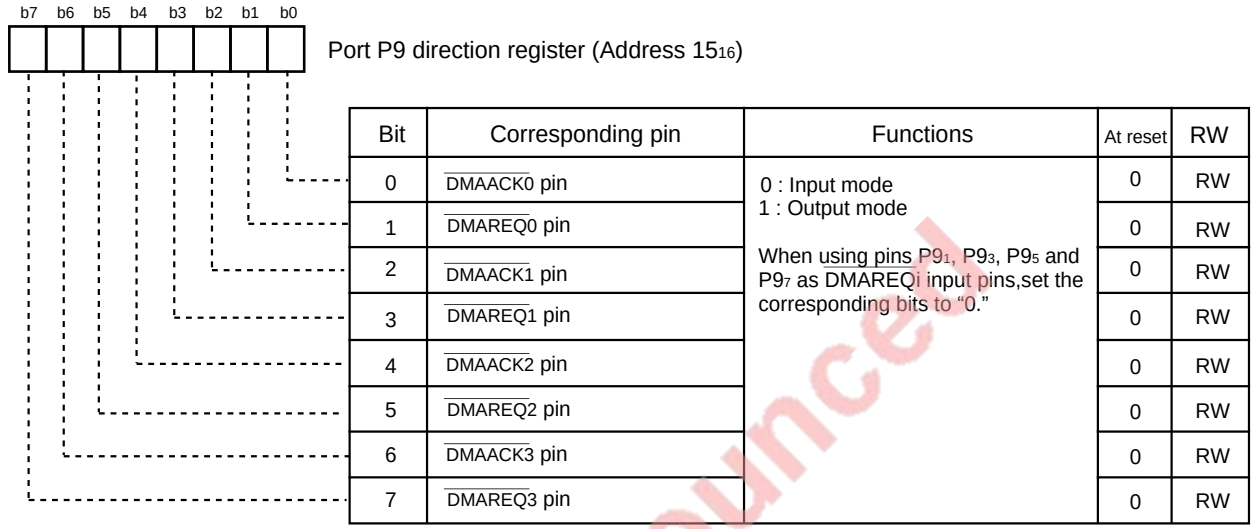
This bit is set to "1" when a DMAi interrupt request occurs after the DMA transfer is complete. This bit is automatically cleared to "0" when the DMAi interrupt request is accepted. This bit can be set to "1" or cleared to "0" by software.

Not announced

13.2 Block description

13.2.14 Port P9 direction register

I/O pins of DMAi are multiplexed with port P9. When using these pins as the $\overline{\text{DMAREQi}}$ input pins, set the corresponding bits of the port P9 direction register to "0" to set these port pins for the input mode. When using these pins as the $\overline{\text{DMAACKi}}$ output pins, these pins are forcibly set to the $\overline{\text{DMAACKi}}$ output pins regardless of the direction register's contents. Figure 13.2.10 shows the relationship between the port P9 direction register and DMAi's I/O pins.



Bit	Corresponding pin	Functions	At reset	RW
0	$\overline{\text{DMAACK0}}$ pin	0 : Input mode 1 : Output mode When using pins $P9_1$, $P9_3$, $P9_5$ and $P9_7$ as $\overline{\text{DMAREQi}}$ input pins,set the corresponding bits to "0."	0	RW
1	$\overline{\text{DMAREQ0}}$ pin		0	RW
2	$\overline{\text{DMAACK1}}$ pin		0	RW
3	$\overline{\text{DMAREQ1}}$ pin		0	RW
4	$\overline{\text{DMAACK2}}$ pin		0	RW
5	$\overline{\text{DMAREQ2}}$ pin		0	RW
6	$\overline{\text{DMAACK3}}$ pin		0	RW
7	$\overline{\text{DMAREQ3}}$ pin		0	RW

Fig. 13.2.10 Relationship between port P9 direction register and DMAi's I/O pins

[Precautions for DMAC]

Do not access the registers relevant to DMAC by using DMA transfers; the address of the accessing register collides with that of the accessed one on the DMAC internal bus.

EOL announced

13.3 Control

The conditions for performing DMA transfer of DMAi (i = 0–3) are as follows:

- Neither a DRAM refresh request nor a Hold request is generated.
- A request of the channel with a higher priority than that of DMAi is not generated; or the request is disabled though it has been generated.
- DMAi is enabled (DMAi enable bit = “1”).
- A DMAi request is generated (DMAi request bit = “1”).

The control method for each channel is described below.

13.3.1 DMA enabling

Each of DMA channels 0–3 has a DMAi enable bit (bits 4–7 at address 69_{16}). Table 13.3.1 lists the conditions for changing each DMAi enable bit.

Table 13.3.1 Conditions for changing DMAi enable bit

DMAi enable bit	Conditions for bit change
Is set to “1.”	A write of “1” to the DMAi enable bit
Is cleared to “0.”	• A write of “0” to the DMAi enable bit • Transfer of an entire batch of data is complete (normal termination). • A change of $\overline{TC}$ input level from “H” to “L” during a DMA transfer of DMAi **(Note)** (Forced termination, when $\overline{TC}$ pin is valid.)

Note: In the burst transfer mode (level sense), however, the term from the DMA transfer start until the transfer completion of an entire batch of data is applied. (It is also valid while the CPU has the right to use bus.)

EOL announced

13.3.2 DMA requests

(1) DMA request sources

DMA request sources are specified by the DMA request source select bits and the edge sense/level sense select bit. (Refer to **"Figure 13.2.8."**)
Table 13.3.2 lists the conditions for generating a DMA request.

Table 13.3.2 Conditions for generating DMA request

DMA request sources		Condition for generating DMA request
External source $\overline{\text{DMAREQi}}$	Level sense	"L"-level input to the $\overline{\text{DMAREQi}}$ pin (only in the burst transfer mode)
	Edge sense	Change of the $\overline{\text{DMAREQi}}$ input pin's level from "H" to "L"
Software DMAi request		A write of "1" to the software DMAi request bit (each of bits 0–3 at address 69_{16}; refer to **"Figure 13.2.5."**)
Timers A0–A4, Timers B0–B2, UART0, UART1, A-D converter		When the interrupt request bit of each peripheral is set to "1" by the activity of peripherals (If "1" is written to any of these interrupt request bits by software, the DMAi request bit does not change. Also, whatever value within 0–7 an interrupt priority level takes, this does not affect DMA requests.)

(2) Change of DMAi request bit

A read of the DMAi request bits (each of bits 4–7 at address 68_{16}) indicates whether the corresponding channel (0–3) is generating its DMA request or not. The DMAi request bit changes synchronized with the falling edge of ϕ_1. Table 13.3.3 lists the conditions for changing the DMAi request bit.
For the timing of changing the DMAi request bit, refer to **"Figures 13.3.2 and 13.3.3."**

Table 13.3.3 Conditions for changing DMAi request bit

Mode / DMAi request bit	Burst transfer mode		Cycle-steal transfer mode
	Edge sense	Level sense	
Is set to "1." **(Note)**	Generation of DMAi request (Refer to **"Table 13.3.2."**)	Generation of DMAi request ("L"-level input to the $\overline{\text{DMAREQi}}$ pin)	Generation of DMAi request (Refer to **"Table 13.3.2."**)
Is cleared to "0."	• Normal termination • Change of the $\overline{\text{TC}}$ pin's input level from "H "to "L" during DMA transfer (when the $\overline{\text{TC}}$ pin is valid)	• "H"-level input to the $\overline{\text{DMAREQi}}$ pin • Change of the $\overline{\text{TC}}$ pin's input level from "H" to "L" (when the $\overline{\text{TC}}$ pin is valid) • A write of "0" to the DMAi request bit • A write of "0" to the DMAi enable bit	• Start of 1-unit transfer • Change of the $\overline{\text{TC}}$ pin's input level from "H" to "L" during DMA transfer (when the $\overline{\text{TC}}$ pin is valid) • A write of "0" to the DMAi request bit • A write of "0" to the DMAi enable bit

Note: While the DMAi enable bit is "0," the DMAi request bit is not set to "1" even if a DMA request is generated. When the DMAi enable bit is cleared to "0," also the DMAi request bit is cleared to "0." However, the DMA request generated while the DMAi enable bit = "0" is maintained; and when the DMAi enable bit is set to "1," the DMAi request bit is also set to "1," except for the burst transfer mode (level sense).

EOL announced

13.3.3 Channel priority levels

When the DMA enable bits of several channels are “1” and their DMA request bits are set to “1,” the request of the channel with the highest priority is accepted first.
The fixed or rotating channel priority can be selected by the priority select bit (bit 0 at address 68_{16}). The priority levels themselves cannot be specified arbitrary.
The channel priority levels are determined after the DMA requests are determined.

(1) Fixed priority

The fixed priority is selected when the priority select bit (bit 0 at address 68_{16}) = “0.”
In the fixed priority, the channel priority levels are as follows:
channel 0 > channel 1 > channel 2 > channel 3.

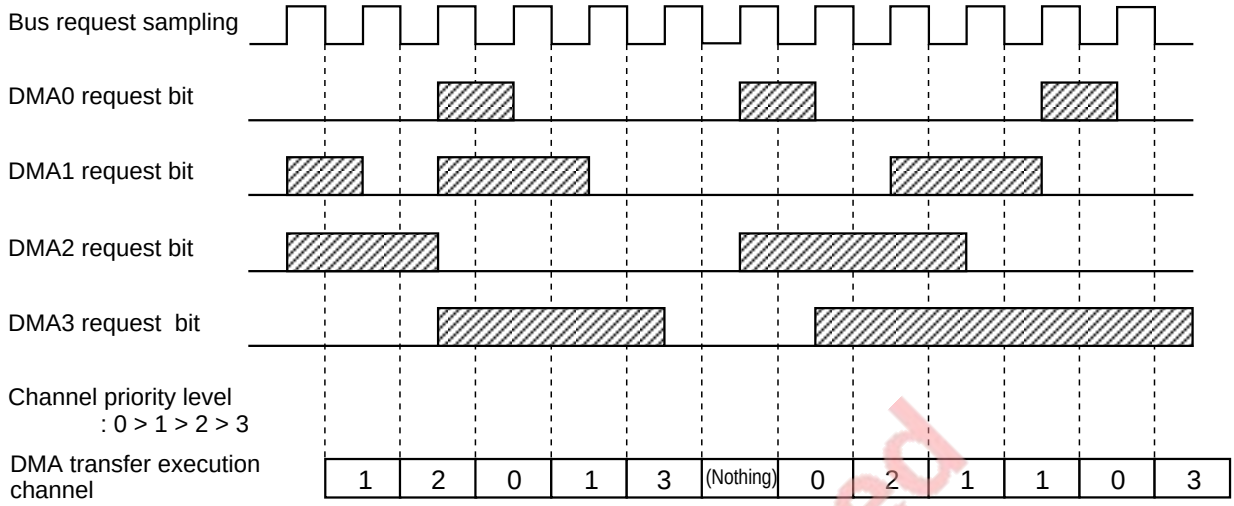
(2) Rotating priority

The rotating priority is selected when the priority select bit = “1.”
After reset, the priority levels are the same descending order as in the fixed priority:
channel 0 > channel 1 > channel 2 > channel 3.
Then, after every normal termination of a DMA transfer, the priority levels rotate in such a way that the lowest priority is given to the channel having been performed.
When DMA transfer is forced into termination, the channel priority levels does not rotate.

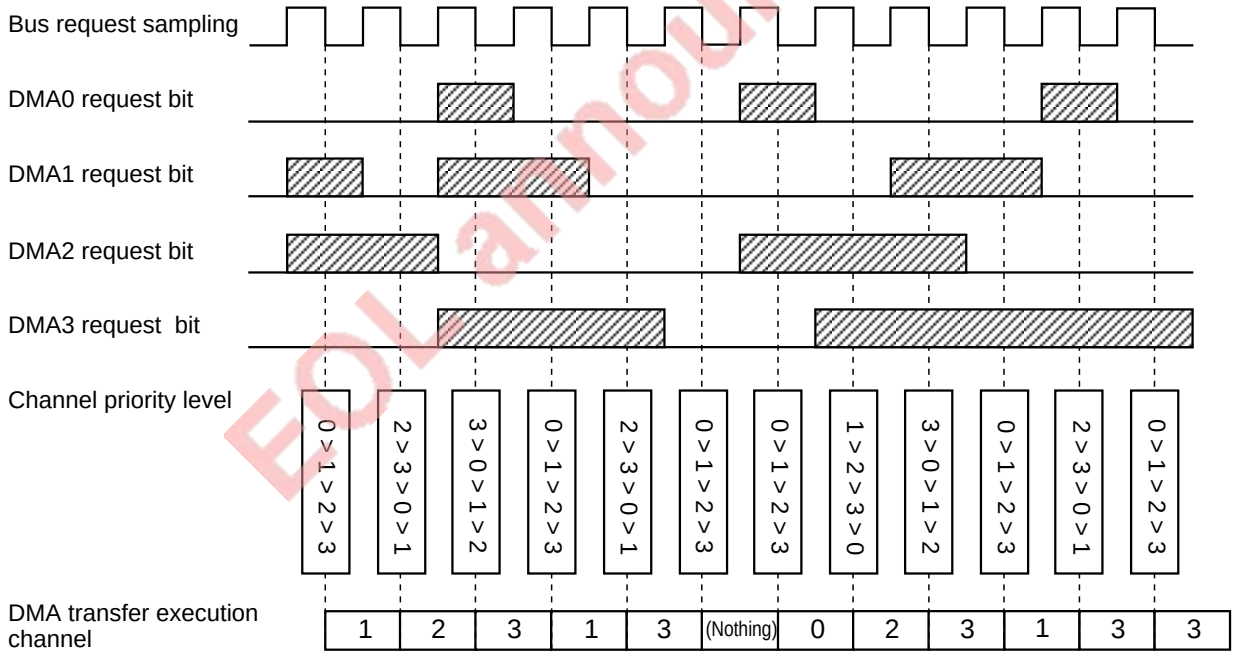
Figure 13.3.1 shows an example of determining the channel priority levels.

EOL announced

● Priority level: Fixed priority



● Priority level: Rotating priority



EOL announced

The above timing diagram applies on the following conditions:
- No DRAM refresh request, no Hold request
- All of DMAi enable bits are “1.”

Fig. 13.3.1 Example of determining channel priority levels

13.3.4 Processing from DMA request until DMA transfer execution

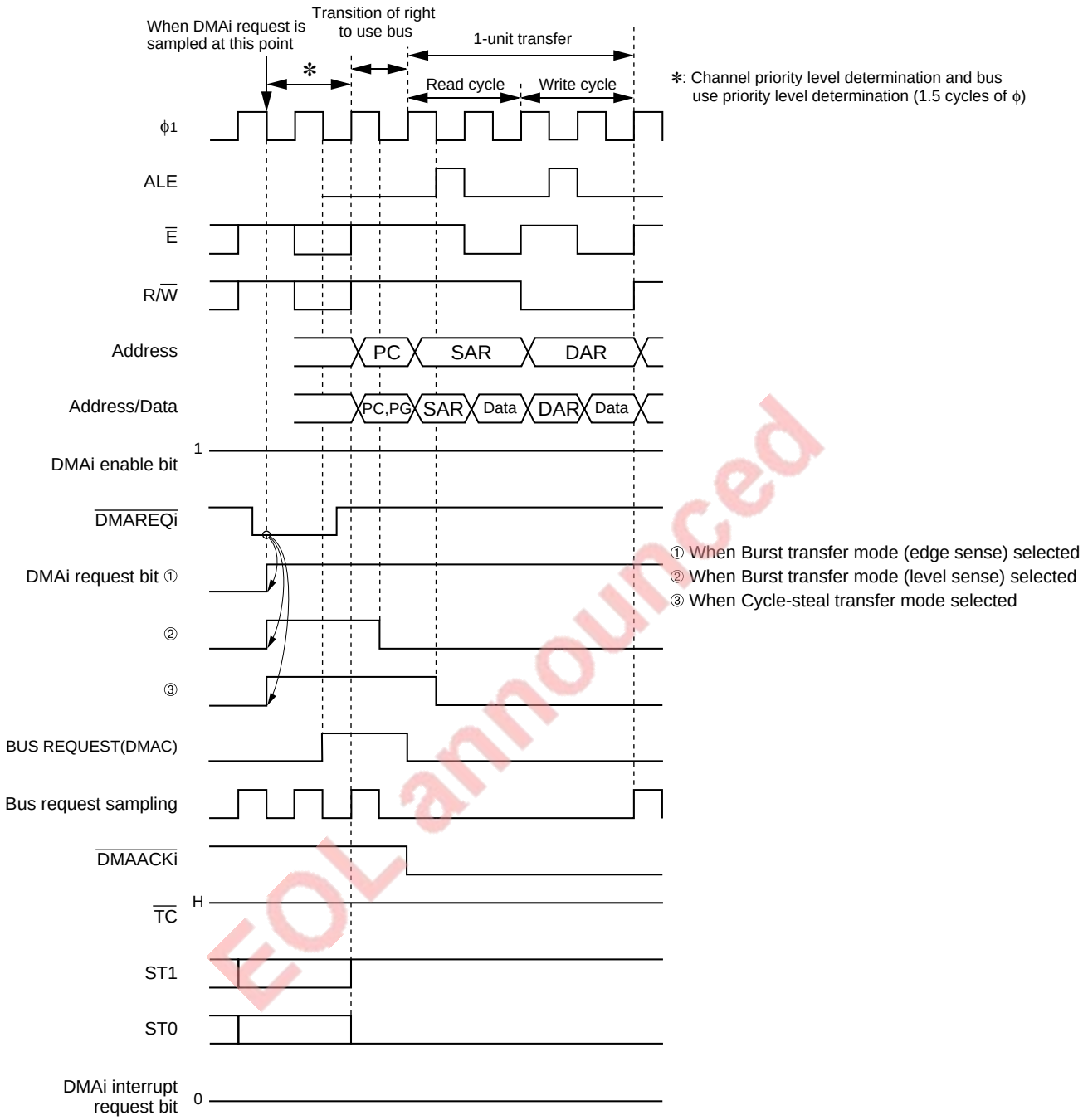
DMA requests are sampled at every falling edge of ϕ_1; when requested, the DMAi request bit is set to "1." Then, the channel priority levels and bus use priority levels are determined, and BUS REQUEST (DMAC) goes "1" if any DRAM refresh request or Hold request is not generated **(Note).**
BUS REQUEST (DMAC) signal is sampled while the bus request sampling signal is "1" and is accepted (DMA request acceptance).

Figure 13.3.2 shows an example of timing from the determination of a DMA request until the DMA transfer execution.
Refer to section **"13.9 DMA transfer time"** for the time from DMA request generation until the CPU's regaining the right to use bus via DMA transfer.

Note: In the following cases, BUS REQUEST (DMAC) does not go "1." However, the DMAi request bit remains set to "1." Accordingly, after completion of each state, the channel priority levels and bus use priority levels are determined, and BUS REQUEST (DMAC) goes "1" if any DRAM refresh request or Hold request is not generated.

- When a DMA request is generated during a burst transfer or in an array state
  (However, if a DRAM refresh request or Hold request is generated during this term, its BUS REQUEST goes "1.")
- When a DMA request is not accepted with a DRAM refresh request or Hold request generated

EOL announced



EOL announced

The above timing diagram applies on the following conditions:
- Single transfer mode, or Repeat transfer mode
- 2-bus cycle transfer
- No Wait
- DMAACKi valid, TC valid
- External source (DMAREQi)
- After DMAi request occurs ("L" is input to the DMAREQi pin.), the right to use bus is relinquished to DMAC at the shortest time.

Fig. 13.3.2 Example of timing from determination of DMA request until DMA transfer execution

13.3.5 Termination of DMA transfer

As the methods of terminating DMA transfer, normal and forced termination are used.

(1) Normal termination

All of the DMAi transfers terminate and DMAC stops. This method is used in the single transfer, array chain transfer, and link array chain transfer modes. In the repeat transfer mode, however, normal termination cannot be applied to terminating transfer; then, forced termination must be used. (Refer to **"(2) Forced termination"** of this section.)

Table 13.3.4 lists the states of DMAC at normal termination.

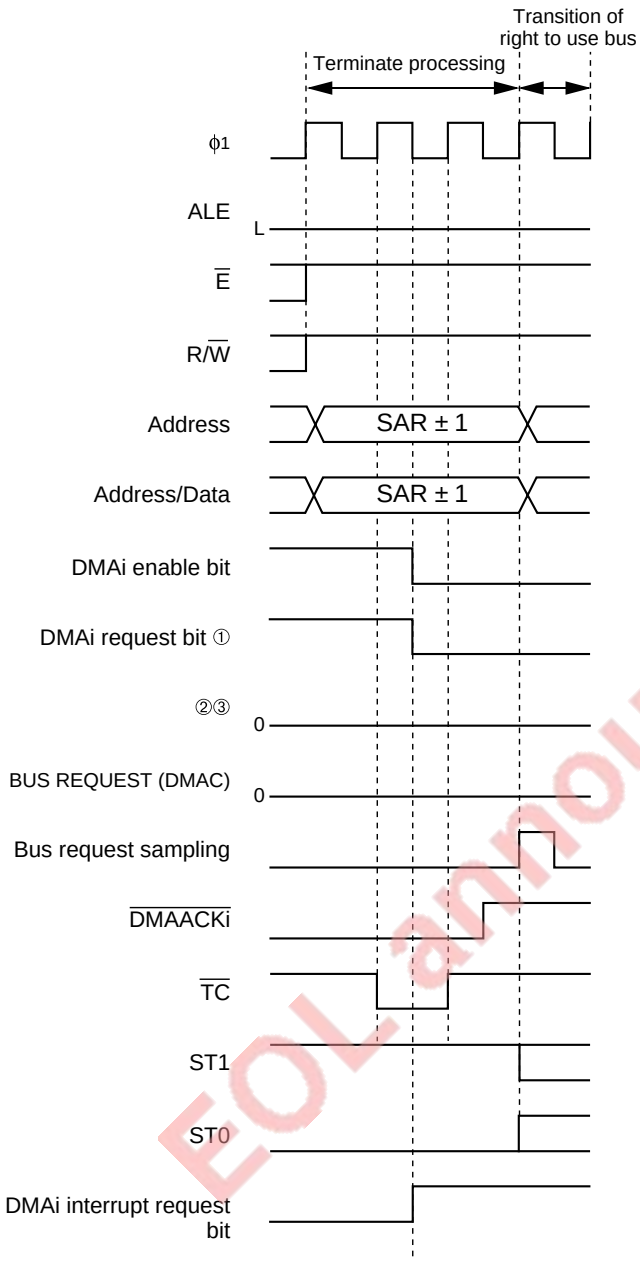
Table 13.3.4 States of DMAC at normal termination

Item	State
DMAi interrupt request bit	1
DMAi request bit	In the burst transfer mode (edge sense): 0 In the burst transfer mode (level sense): not changed In the cycle-steal transfer mode: not changed **(Note)**
DMAi enable bit	0
$\overline{TC}$ output	Outputs "L" (when the $\overline{TC}$ pin is valid)
Channel priority levels	Rotating (when the rotating priority is selected)

Note: In the cycle-steal transfer mode, the DMAi request bit is cleared to "0" when a DMA request is accepted. This bit is does not change at normal termination.

At normal termination, the CPU regains the right to use bus after the terminate processing (3 cycles of ϕ) via the transition of the right to use bus (1 cycle of ϕ). Figure 13.3.3 shows a timing example at normal termination.

EOL announced



① When Burst transfer mode (edge sense) selected
② When Burst transfer mode (level sense) selected
③ When Cycle-steal transfer mode selected

EOL announced

The above timing diagram applies on the following conditions:
- DMAACKi valid, TC valid
- External source (DMAREQi)

Fig. 13.3.3 Timing example at normal termination

(2) Forced termination

The methods of terminating DMAC other than normal termination are as follows:

- Drives the $\overline{TC}$ pin's input level from "H" to "L" during a DMA transfer (when the $\overline{TC}$ pin is valid).
- Writes "0" to the DMAi enable bit.

Table 13.3.5 lists the states of DMAC at forced termination.

Table 13.3.5 States of DMAC at forced termination

Item	State
DMAi interrupt request bit	Not changed.
DMAi request bit	0
DMAi enable bit	0
$\overline{TC}$ output	Not changed.
Channel priority levels	Not changed.

When the $\overline{TC}$ pin is used for forced termination, select "$\overline{TC}$ pin valid" (bit 1 at address 68_{16} = "1"). Forced termination by the $\overline{TC}$ input is valid in the following cases:

- During a DMA transfer in the burst transfer mode (edge sense)
- During the term from the DMA transfer start until the transfer completion of an entire batch of data in the burst transfer mode with the level sense selected. (It is also valid while the CPU has the right to use bus).
- During a DMA transfer in the cycle-steal transfer mode (Forced termination by the $\overline{TC}$ input is invalid while the CPU has the right to use bus.)

The $\overline{TC}$ pin's input is determined at the falling edge of ϕ_1, and DMAC will relinquish the right to use bus to the CPU upon completion of the 1-unit transfer under execution at that time.

At the forced termination by the DMAi enable bit, "0" is written to this bit at the rising edge of $\overline{E}$ of a write cycle to the DMAi enable bit. Accordingly, DMAi is disabled after this write.

EOL announced

13.3.6 DMA transfer restart after termination

(1) Restarting the same DMA transfer as the previous one from the beginning

At normal and forced termination, the latches of SARi, DARi, and TCRi maintain their values written before the transfer start. (Refer to **"Figure 13.3.4-a."**) Therefore, DMA transfer must be restarted according to the following procedures:

- **In single or repeat transfer mode**
  Set the DMAi enable bit to "1." It is not necessary to re-set the values of SARi, DARi, and TCRi by software. (Refer to **"Figure 13.3.4-b."**)

- **In array chain or link array chain transfer mode**
  ① Re-set the values of SARi and TCRi.
  ② Set the DMAi enable bit to "1."

(2) Restarting transfer of data subsequent to one which has been transferred just before forced termination

When reading values at the addresses of SARi, DARi, and TCRi after forced termination, the values of these registers (counters) can be read. These read values are the transfer source address, the transfer destination address which were to be transferred subsequently, and the number of remaining bytes.
When writing these read values to the addresses of SARi, DARi, and TCRi respectively, the same values are also written to their latches. When setting the DMAi enable bit to "1" under this condition, transfer of data subsequent to one which has been transferred just before forced termination is restarted. (Refer to **"Figure 13.3.4-c."**)

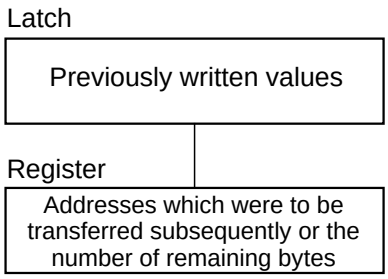
- **In single transfer mode**
  The remaining data can be transferred by the following procedure:
  ① Read the values at addresses of SARi, DARi and TCRi. Then, rewrite these values into these addresses.
  ② Set the DMAi enable bit to "1."

- **In repeat transfer, array chain transfer, and link array chain transfer modes**
  The remaining data of the block that was interrupted by forced termination can be transferred by the following procedure:
  ① Switch over the current mode to the single transfer mode.
  ② Read the values at addresses of SARi, DARi and TCRi. Then, rewrite these values into these addresses.
  ③ Set the DMAi enable bit to "1." (Refer to **"Figure 13.3.4-c."**)
  ④ In order to transfer the next block, switch over the current mode to the previous mode after the above-mentioned transfer is normally terminated. Then, re-set the values of SARi, DARi, and TCRi.

In the array chain or the link array chain transfer mode, information such as the next transfer parameters etc. cannot be read from each latch.

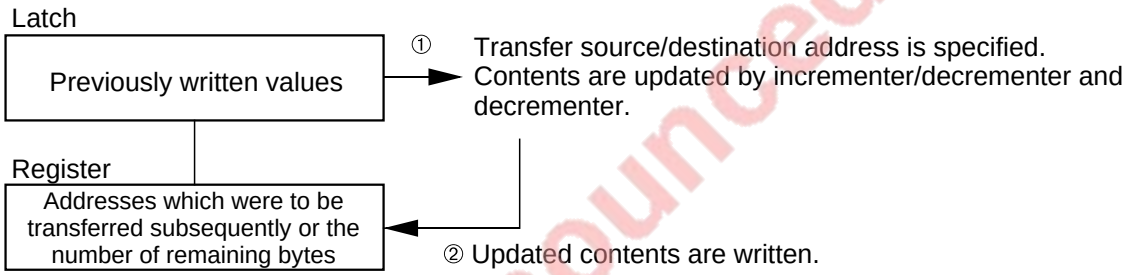
EOL announced

a. State at forced termination

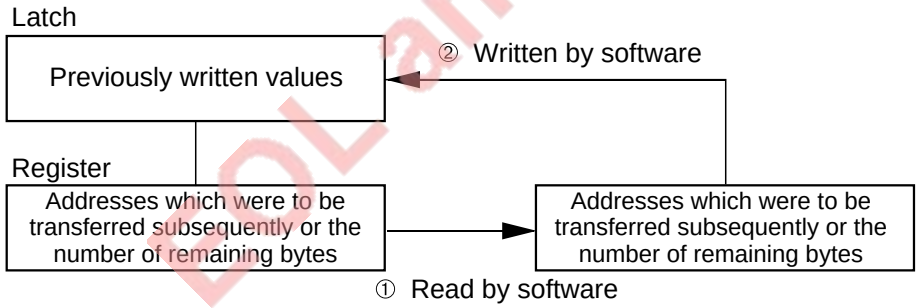


b. When setting DMAi enable bit to “1” without rewriting values at addresses of SARi, DARi, and TCRi

A value read from each latch is used by hardware only at the first 1-unit transfer.
The contents updated by the incrementer/decrementer and the decrementer are loaded in each register.
Values are used by reading them from registers at the second and the following 1-unit transfers.



c. When reading values at addresses of SARi, DARi, and TCRi after forced termination and rewriting them



DO Announced

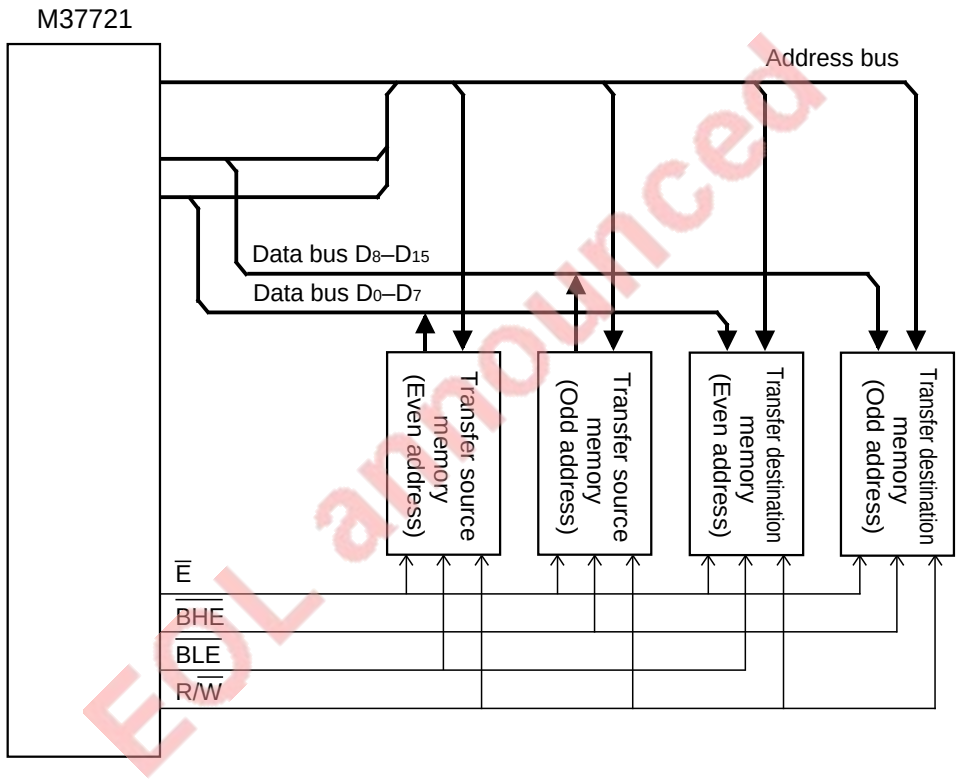
Fig.13.3.4 States of SARi, DARi, TCRi after forced termination

13.4 Operation

Operation of 1-unit transfer varies according to the data transfer method (2-bus cycle or 1-bus cycle transfer). In addition, how many units of transfer data are transferred for a DMA request varies according to the transfer mode (burst transfer or cycle-steal transfer mode).
These data transfer methods and modes are described below.

13.4.1 2-bus cycle transfer

When the transfer method select bit (Refer to **"Figure 13.2.6."**) = "0," 2-bus cycle transfer is selected. 2-bus cycle transfer is the method used to transfer data between memories. Since this method has a read and a write cycle, it consumes a minimum of 2 bus cycles for 1-unit transfer.
Figure 13.4.1 shows an example of connecting external memories in 2-bus cycle transfer.



Note. The external circuit such as an address latch is disregarded.

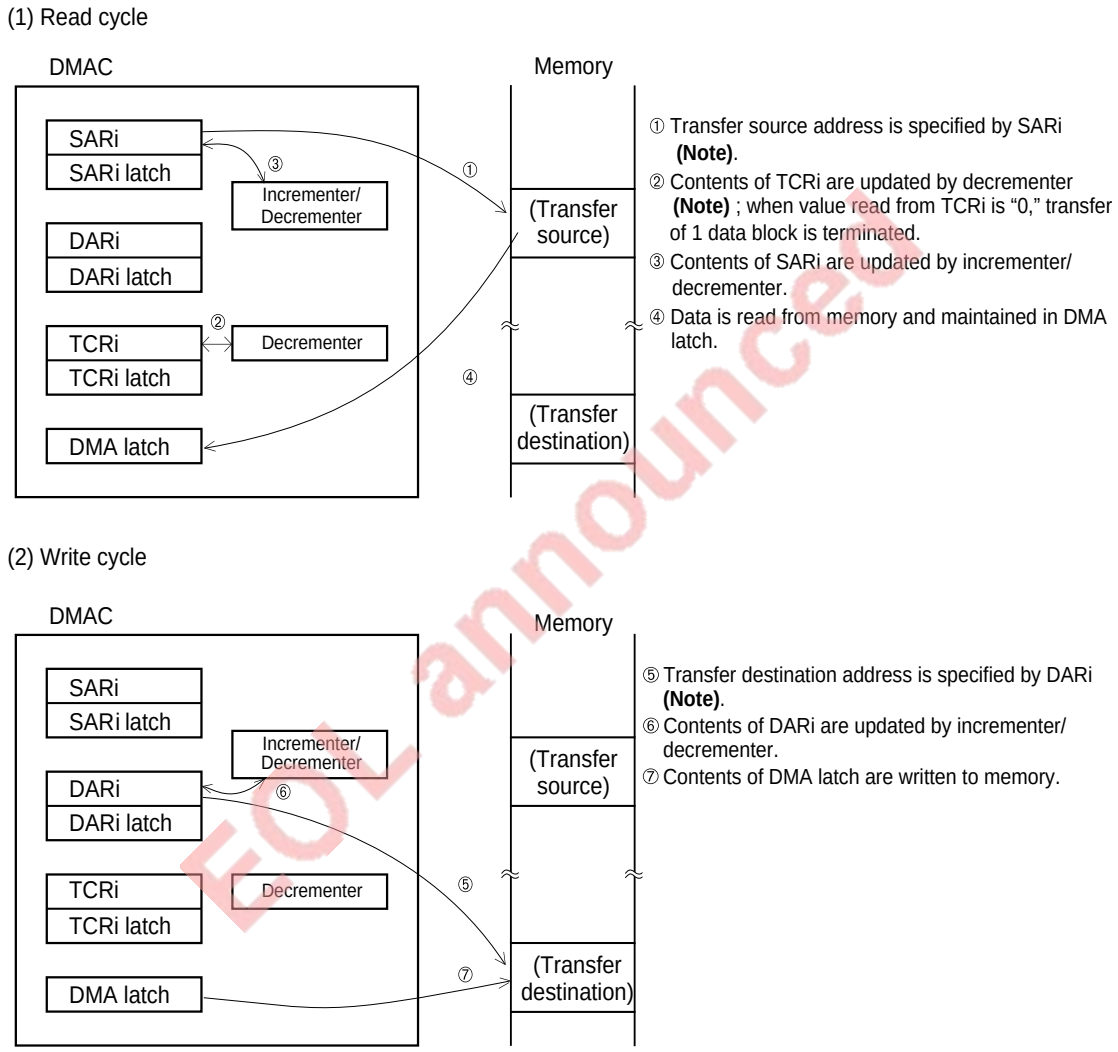
Fig. 13.4.1 Example of connecting external memories in 2-bus cycle transfer

EOL announced

(1) Register operation in 2-bus cycle transfer

Figure 13.4.2 shows a basic operation of registers for 1-unit transfer in 2-bus cycle transfer. For register values to be specified, refer to section **"13.5 Single transfer mode"** through section **"13.8 Link array chain transfer mode."** It is because that these values vary according to continuous transfer modes.

In 2-bus cycle transfer, the data read at a read cycle is maintained temporarily in the DMA latch, and the contents of this latch are written to a memory at a write cycle.



✻ When the transfer unit is 16 bits

- When an even address is accessed with 16-bit external data bus width, data can be read or written at 1-bus cycle. Accordingly, the incrementer/decrementer and the decrementer increment or decrement by 2, and sequences ① through ⑦ are performed once.
- When an odd address is accessed with 16-bit external data bus width or when 8 bits is used as external data bus width, data is read or written at 2-bus cycles, and sequences ① through ④ or ⑤ through ⑦ are repeated twice. The incrementer/decrementer and the decrementer increment or decrement by 1 every time sequences ① through ④ or ⑤ through ⑦ are performed once.

Note: In the single transfer mode and repeat transfer mode, only at the first transfer of the block, the values read from SARi latch, DARi latch, and TCRi latch are used. (The results obtained by increment or decrement are written to SARi, DARi, and TCRi. Except for the first transfer of the block, the values read from SARi, DARi, and TCRi are used.)

Fig. 13.4.2 Basic operation of registers for 1-unit transfer in 2-bus cycle transfer

EOL announced

(2) Bus operation in 2-bus cycle transfer

The time required for 1-unit transfer in 2-bus cycle transfer is given by the following formula:

Transfer time per 1-unit transfer = (Read cycle) + (Write cycle)

Since any area can be specified as a transfer source or a transfer destination, a read cycle varies with the conditions of a transfer source, and a write cycle with that of a transfer destination.

Table 13.4.1 lists the time required for a read or write cycle per 1-unit transfer in 2-bus cycle transfer, and Figure 13.4.3 shows the bus-cycle operation waveforms in 2-bus cycle transfer.

Table 13.4.1 Time required for a read or write cycle per 1-unit transfer in 2-bus cycle transfer

External bus width	Transfer unit	Address directions	Data's start address	Read/Write cycle (Unit: ϕ cycle)			
				Formula	No Wait	With Wait	DRAM area
16 bits (including internal bus)	16 bits	Fixed/Forward	Even	1 + i	2 (a)	3 (d)	
			Odd	2 + 2i	4 (c)	6 (f)	
		Backward	Even	2 + 2i	4 (c)	6 (f)	
			Odd	2 + i	3 (b)	4 (e)	4 (e) **(Note)**
	8 bits	Fixed/Forward/Backward	Even/Odd	1 + i	2 (a)	3 (d)	
8 bits	16 bits	Fixed/Forward/Backward	Even/Odd	2 + 2i	4 (c)	6 (f)	
	8 bits	Fixed/Forward/Backward	Even/Odd	1 + i	2 (a)	3 (d)	

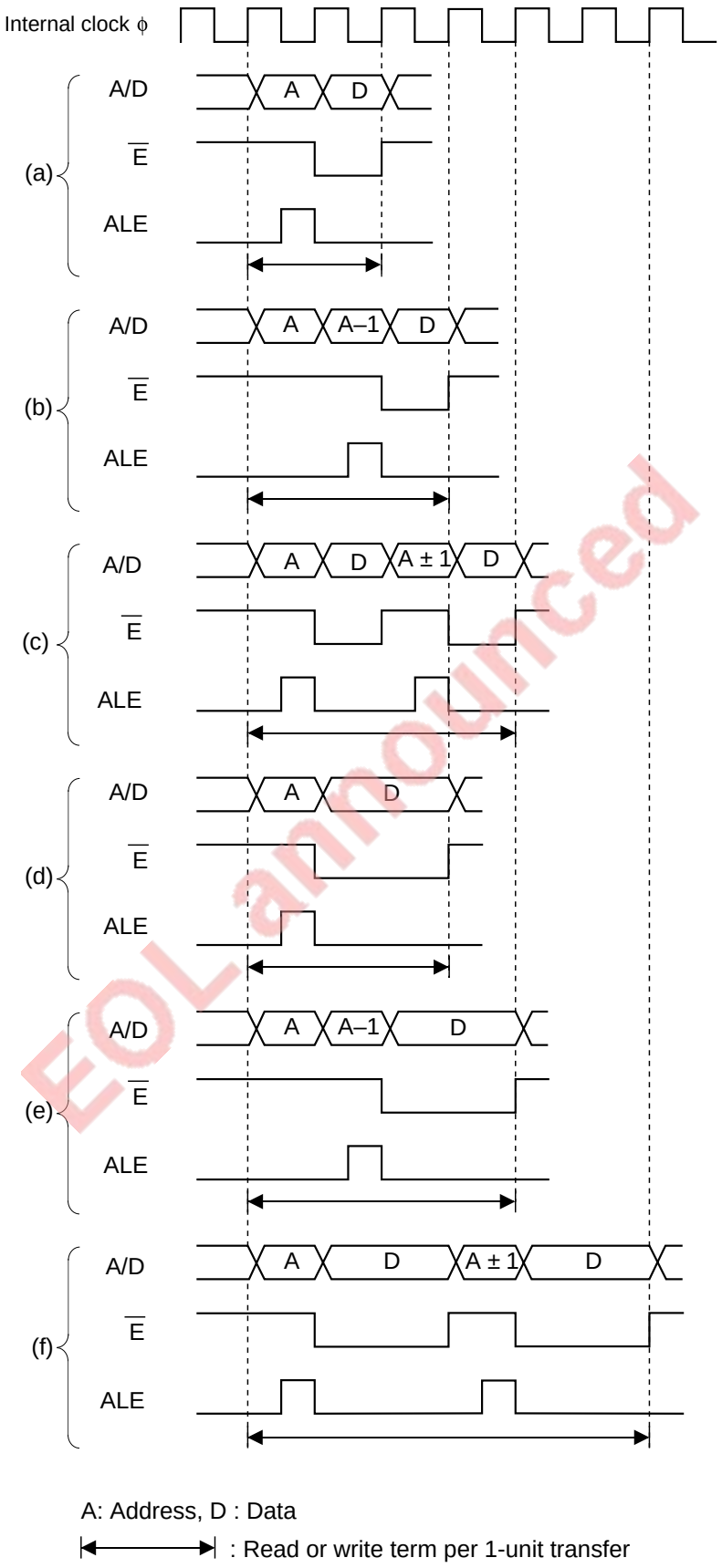
Address directions: Refer to section **"13.4.1 (3) Address directions in 2-bus cycle transfer."**

i: A term of $\overline{E}$ = "L" in 1-bus cycle; i = 1 at "No Wait", and i = 2 at "With Wait" or "DRAM area".
When Ready function is used (Refer to section **"3.3 Ready function."**), the number of cycles extended by Ready must be added.

( ): Indicates the corresponding waveform in Figure 13.4.3.

Note: When a transfer destination applies to this condition, 2-bus cycle transfer cannot be performed. When a transfer source applies to this condition and a transfer destination is in the DRAM area, 2-bus cycle transfer cannot also be performed.

EOL announced



EOL announced

Fig. 13.4.3 Bus-cycle operation waveforms in 2-bus cycle transfer

(3) Address directions in 2-bus cycle transfer

In 2-bus cycle transfer, the address direction of a transfer source and that of a transfer destination each can be selected independently. (Refer to **"Figure 13.2.6."**) Addresses move in the specified direction by the transfer unit. Tables 13.4.2 through 13.4.4 list address directions in 2-bus cycle transfer and examples of transfer result.

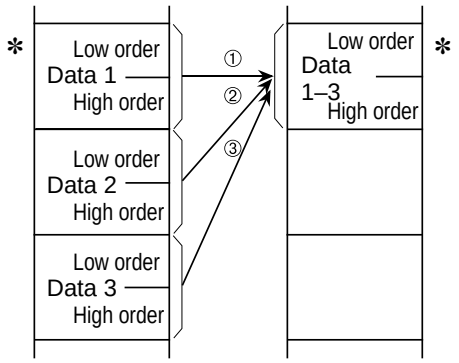
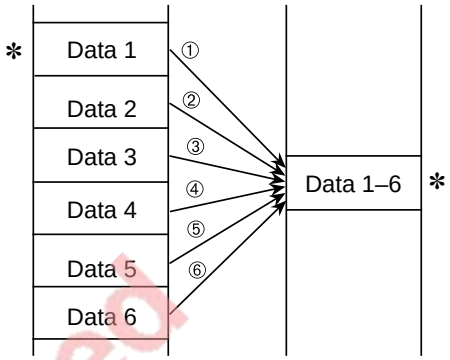
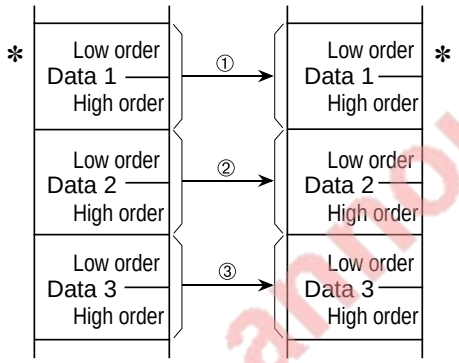
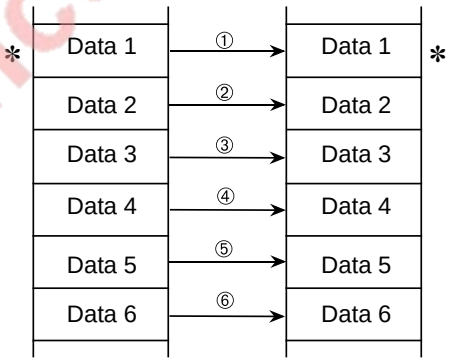
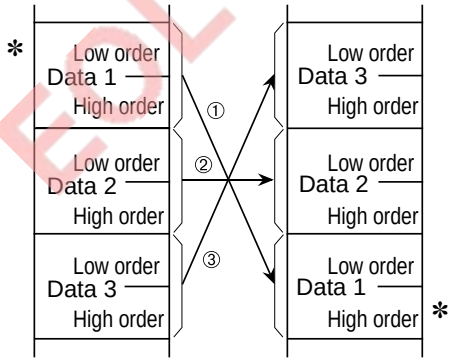
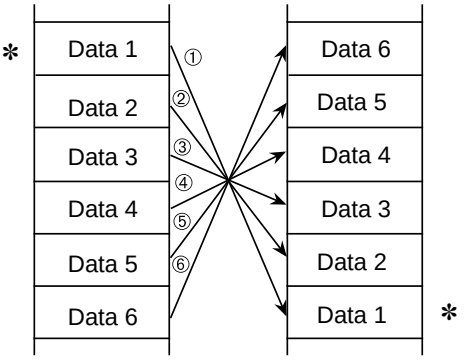
Tables 13.4.2 Address directions in 2-bus cycle transfer and examples of transfer result (1)

Address direction		External data bus width : 16 bits or 8 bits					
		Transfer unit : 16 bits			Transfer unit : 8 bits		
Transfer source	Transfer destination	Data arrangement on transfer source memory	Transfer sequence	Data arrangement on transfer destination memory (transfer result)	Data arrangement on transfer source memory	Transfer sequence	Data arrangement on transfer destination memory (transfer result)
Fixed	Fixed	✻ Low order Data High order		Low order Data High order ✻	✻ Data		Data ✻
Fixed	Forward	✻ Low order Data High order	① ② ③	Low order Data High order Low order Data High order Low order Data High order	✻ Data	① ② ③ ④ ⑤ ⑥	Data ✻ Data Data Data Data Data Data
Fixed	Backward	✻ Low order Data High order	③ ② ①	Low order Data High order Low order Data High order Low order Data High order ✻	✻ Data	⑥ ⑤ ④ ③ ② ①	Data Data Data Data Data Data ✻

✻ : Transfer start address

EOL announced

Tables 13.4.3 Address directions in 2-bus cycle transfer and examples of transfer result (2)

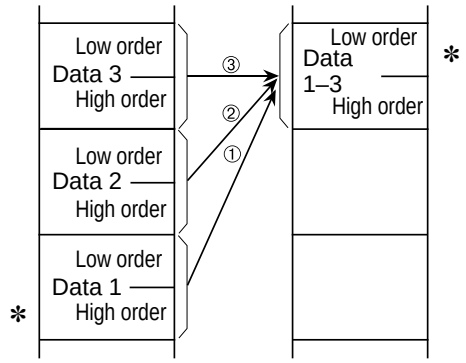
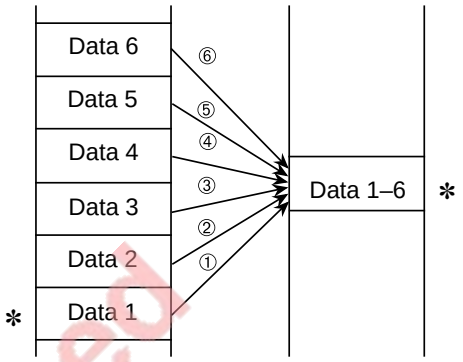
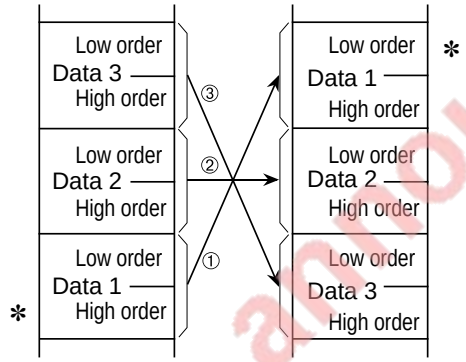
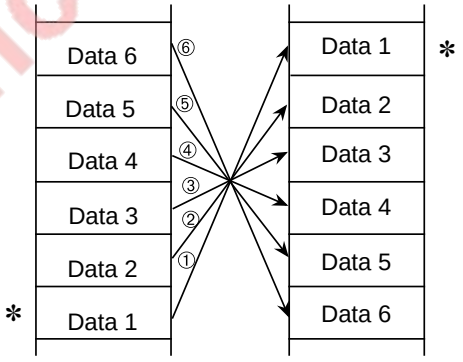
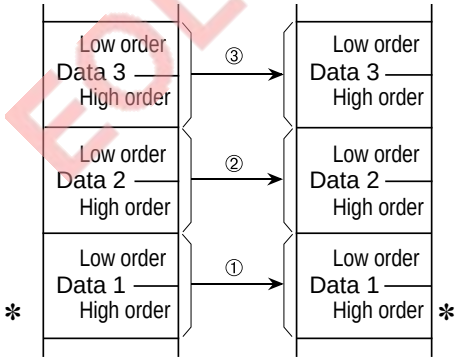
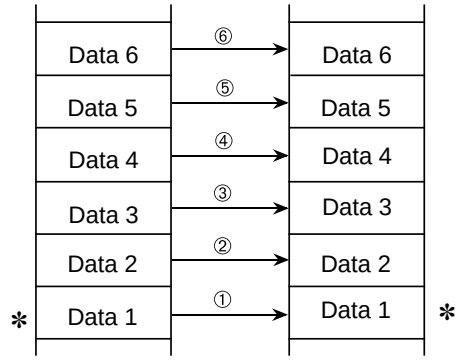
Address direction		External data bus width : 16 bits or 8 bits					
		Transfer unit : 16 bits			Transfer unit : 8 bits		
Transfer source	Transfer destination	Data arrangement on transfer source memory	Transfer sequence	Data arrangement on transfer destination memory (transfer result)	Data arrangement on transfer source memory	Transfer sequence	Data arrangement on transfer destination memory (transfer result)
Forward	Fixed						
Forward	Forward						
Forward	Backward						

Note: The position relationship between low-order byte and high-order byte is not reversed.

✻ : Transfer start address

EOL announced

Tables 13.4.4 Address directions in 2-bus cycle transfer and examples of transfer result (3)

Address direction		External data bus width : 16 bits or 8 bits					
		Transfer unit : 16 bits			Transfer unit : 8 bits		
Transfer source	Transfer destination	Data arrangement on transfer source memory	Transfer sequence	Data arrangement on transfer destination memory (transfer result)	Data arrangement on transfer source memory	Transfer sequence	Data arrangement on transfer destination memory (transfer result)
Backward	Fixed						
Backward	Forward						
Backward	Backward						

✻ : Transfer start address

EOL announced

[Precautions for 2-bus cycle transfer]

When the 16-bit external data bus width = 16 bits and the transfer unit = 16 bits under the following conditions, 2-bus cycle transfer cannot be performed. (Refer to **"Table 13.4.1."**)

- **Conditions for transfer destination**
  Transfer destination = DRAM area, Address direction = Backward, Data's start address = Odd address

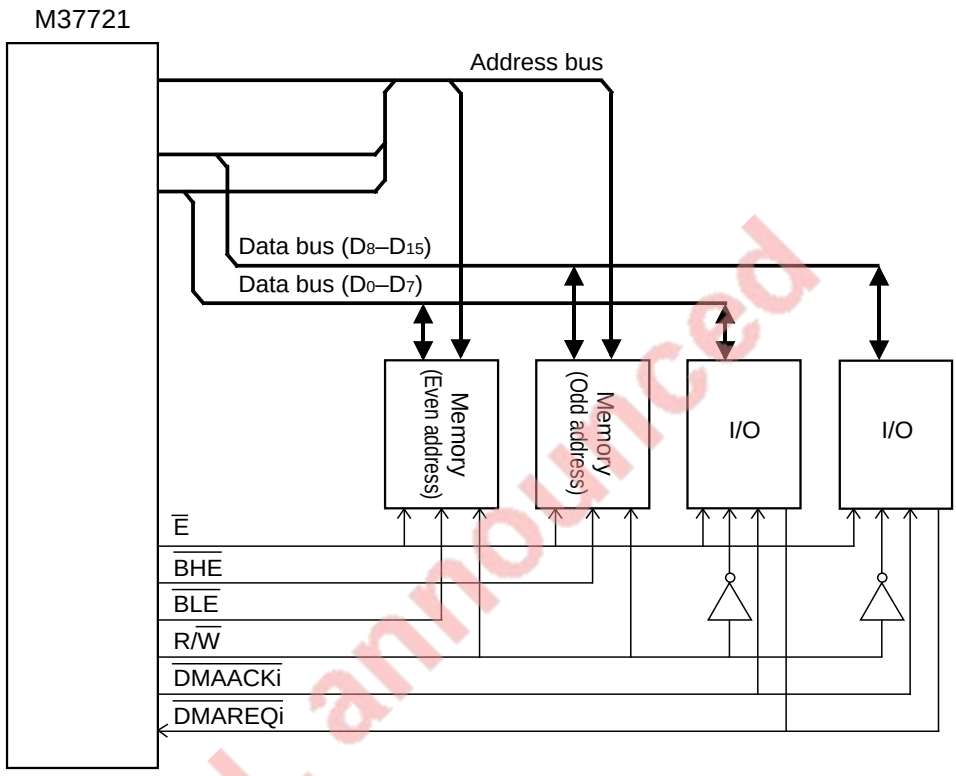
- **Conditions for transfer source and destination**
  Transfer source = DRAM area, Address direction = Backward, Data's start address = Odd address
  Transfer destination = DRAM area

EOL announced

13.4.2 1-bus cycle transfer

When the transfer method select bit (Refer to **"Figure 13.2.6."**) = "1," 1-bus cycle transfer is selected. 1-bus cycle transfer is the method used to transfer data between a memory and an I/O. In this method, a read and write of 1-tansfer-unit data are simultaneously performed during 1-bus cycle. The address bus, $\overline{BHE}$, $\overline{BLE}$, and R/$\overline{W}$ indicate the states of memory.

Figure 13.4.4 shows an example of connecting external memories and I/Os in 1-bus cycle transfer.



Note. The external circuit such as an address latch is disregarded.

Fig. 13.4.4 Example of connecting external memories and I/Os in 1-bus cycle transfer

EOL announced

In 1-bus cycle transfer, the following considerations must be taken in designing the system:

- Achieve the condition that 1-transfer-unit data can be accessed in 1-bus cycle. (Refer to **"Table 13.4.5."**)
- Specify the transfer address direction and I/O connections. (Refer to **"Figure 13.2.7."**)
- Compose the read and write signal generating circuit externally. (These signals are for I/Os.)
  The M37721 outputs signals to the memory. Accordingly, make sure to compose the circuit which generates write signals for I/Os when the M37721 outputs read signals; which generates read signals for I/Os when the M37721 outputs write signals. Figure 13.4.5 shows an example of the circuit generating a write signal and a read signal for I/Os.

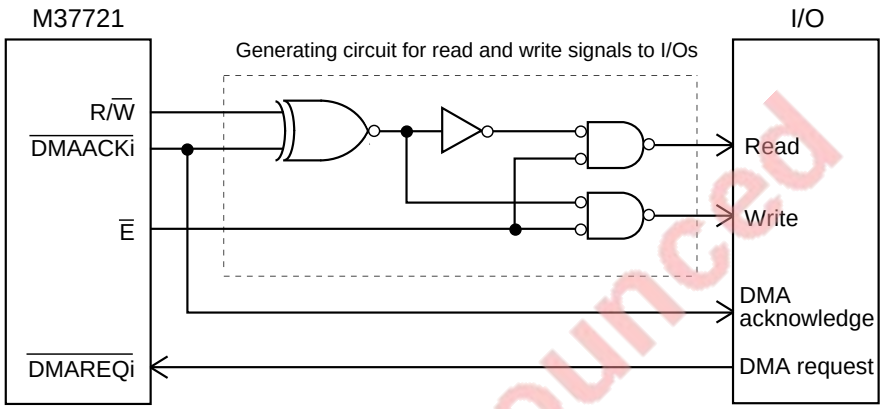


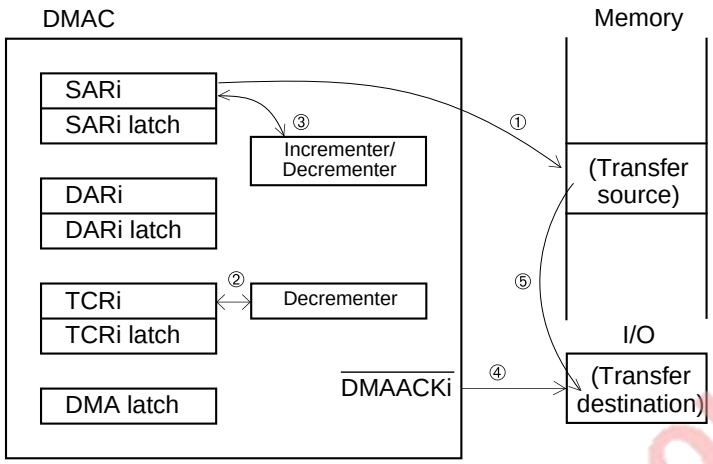
Fig. 13.4.5 Example of circuit generating write signal and read signal for I/Os

EOL announced

(1) Register operation in 1-bus cycle transfer

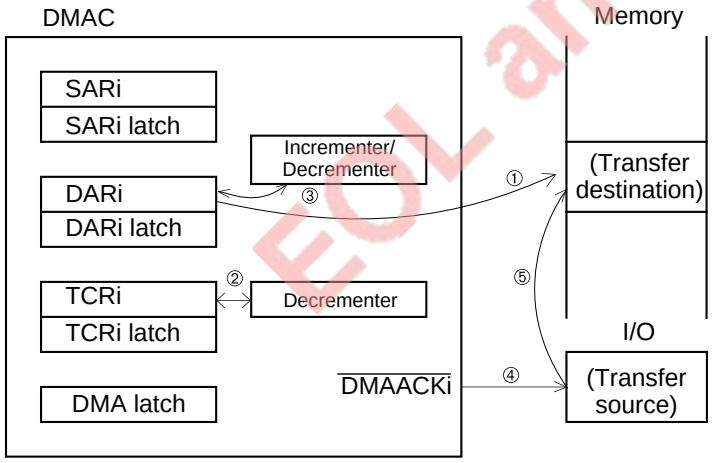
Figure 13.4.6 shows a basic operation of registers for 1-unit transfer in 1-bus cycle transfer. For register values to be specified, refer to section **"13.5 Single transfer mode"** through section **"13.8 Link array chain transfer mode."** It is because these values vary depending on each continuous transfer mode. In 1-bus cycle transfer, a read and write of 1-transfer-unit data are simultaneously performed during 1-bus cycle.

●When transferring from memory to I/O



① Transfer source address is specified by SARi **(Note)**.
② Contents of TCRi are updated by decrementer **(Note)**; when value read from TCRi is "0," transfer of 1 data block is terminated.
③ Contents of SARi are updated by incrementer/ decrementer.
④ I/O is specified by $\overline{\text{DMAACKi}}$.
⑤ Data is output from memory and is written to I/O simultaneously (R/$\overline{\text{W}}$ = H level).

●When transferring from I/O to memory



① Transfer source address is specified by DARi **(Note)**.
② Contents of TCRi are updated by decrementer **(Note)**; when value read from TCRi is "0," transfer of 1 data block is terminated.
③ Contents of DARi are updated by incrementer/ decrementer.
④ I/O is specified by $\overline{\text{DMAACKi}}$.
⑤ Data is output from I/O and is written to memory simultaneously (R/$\overline{\text{W}}$ = L level).

✻ When the transfer unit is 16 bits, the incrementer/decrementer and the decrementer increment or decrement by 2.

Note: In the single transfer mode and repeat transfer mode, only at the first transfer of the block, the values read from SARi latch, DARi latch, and TCRi latch are used. (The results obtained by increment or decrement are written to SARi, DARi, and TCRi. Except for the first transfer of the block, the values read from SARi, DARi, and TCRi are used.)

Fig. 13.4.6 Basic operation of registers for 1-unit transfer in 1-bus cycle transfer

EOL announced

(2) Bus operation in 1-bus cycle transfer

The time required for 1-unit transfer in 1-bus cycle transfer is given by the following formulas:

- Transfer from memory to I/O: Transfer time per 1-unit transfer = (Read cycle of memory)
- Transfer from I/O to memory: Transfer time per 1-unit transfer = (Write cycle of memory)

In 1-bus cycle transfer, 1-transfer-unit data is accessed in 1-bus cycle, so that limitations are imposed on the transfer conditions to be applied. Table 13.4.5 lists the conditions of 1-bus cycle transfer and the transfer time per 1-unit transfer, and Figure 13.4.7 shows the bus-cycle operation waveforms in 1-bus cycle transfer.

Table 13.4.5 Conditions of 1-bus cycle transfer and Transfer time per 1-unit transfer

External bus width	Transfer unit	Address directions	Data's start address	Read/Write cycle (Unit: ϕ cycle)			
				Formula	No Wait	With Wait	DRAM area
16 bits (including internal bus)	16 bits	Fixed/Forward	Even	1 + i	2 (a)	3 (c)	
			Odd				
		Backward	Even				
			Odd	2 + i	3 (b)	4 (d)	
	8 bits	Fixed/Forward/ Backward	Even/Odd	1 + i	2 (a)	3 (c)	
8 bits	16 bits	Fixed/Forward/ Backward	Even/Odd				
	8 bits	Fixed/Forward/ Backward	Even/Odd	1 + i	2 (a)	3 (c)	

Address directions: Refer to section **"13.4.2 (3) Address directions in 1-bus cycle transfer."**
There is no address direction on the I/O side.

i: A term of $\overline{E}$ = 'L" in 1-bus cycle; i = 1 at "No Wait", and i = 2 at "With Wait" or "DRAM area".
When Ready function is used (Refer to section **"3.3 Ready function."**), the number of cycles extended by Ready must be added.

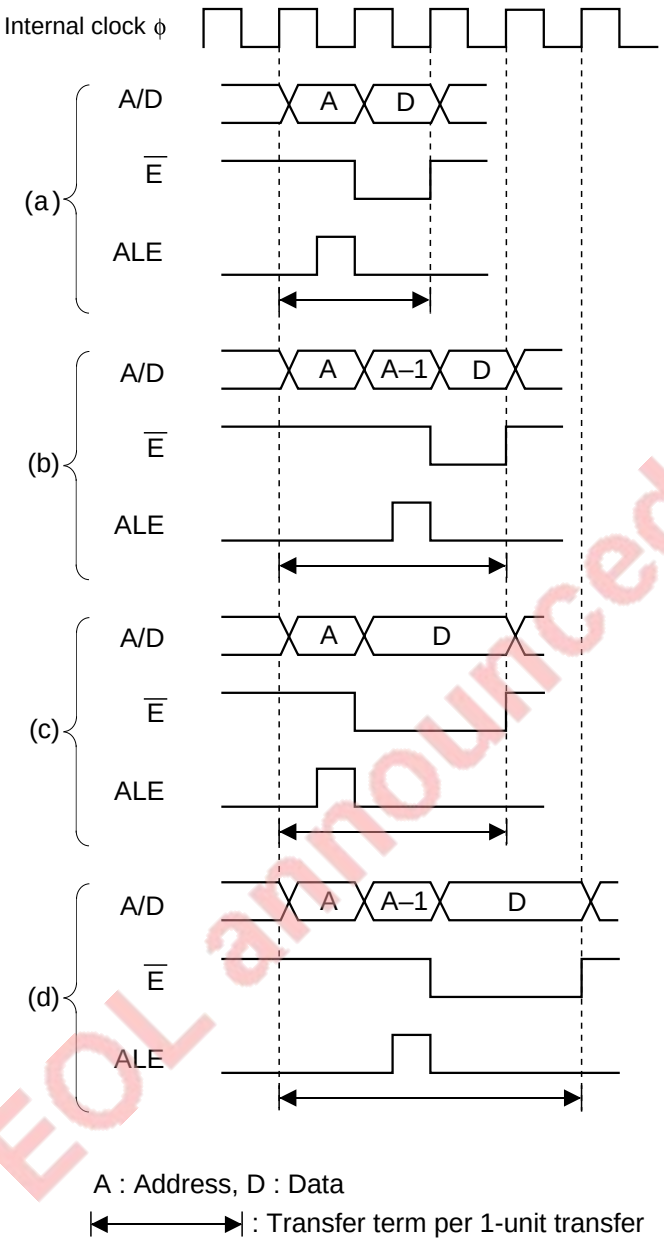
( ): Indicates the corresponding waveform in Figure 13.4.7.

/: 1-bus cycle transfer cannot be performed.

EOL announced

When the external data bus width = 16 bits and the transfer unit = 8 bits are selected, the data bus which the memory uses and the data bus to which I/O is connected may be different. In such a case, data is copied from the data bus of a transfer source to that of a transfer destination by using the DMA latch. For the combination that data copy may occur, data copy delay time $t_{d(data)}$ must be taken into consideration.

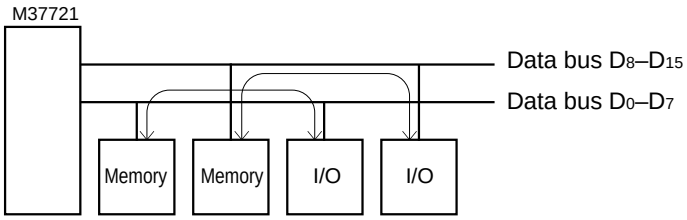
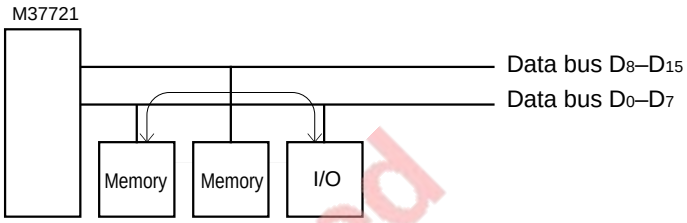
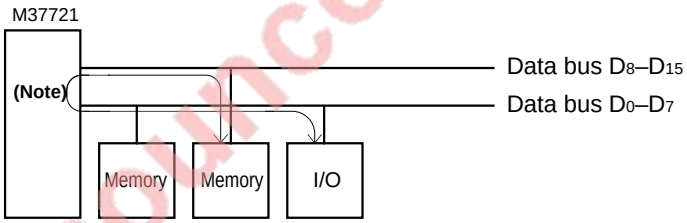
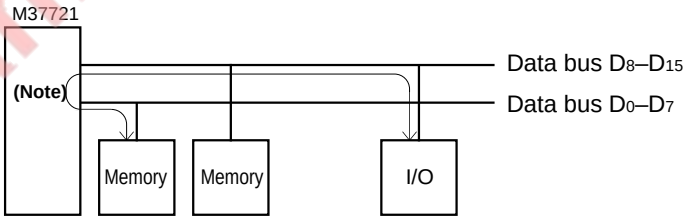
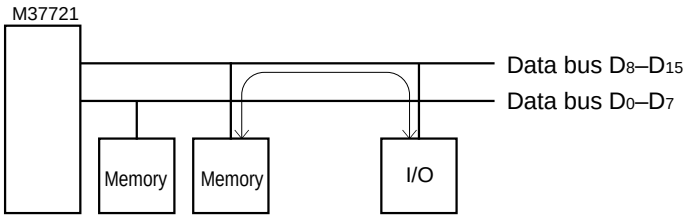
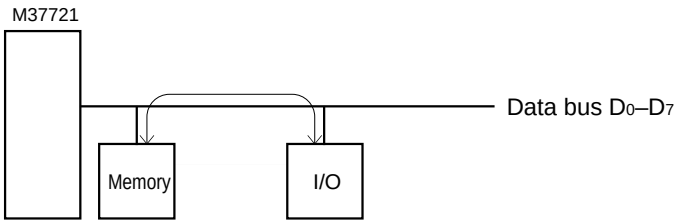
Table 13.4.6 lists the data flows on the data bus in 1-bus cycle transfer, and Table 13.4.7 lists the outputs of the address bus, the data bus, and the bus control signals in 1-bus cycle transfer.



EOL announced

Fig. 13.4.7 Bus-cycle operation waveforms in 1-bus cycle transfer

Table 13.4.6 Data flows on data bus in 1-bus cycle transfer

External data bus width	I/O connection	Transfer unit	Read/Write address of memory	Data flow
16 bits	Data bus D_0–D_7 and D_8–D_{15}	16 bits	Even address and Odd address	
16 bits	Data bus D_0–D_7	8 bits	Even address	
			Odd address	 **Note:** Data is copied from data bus D_0–D_7 to D_8–D_{15} or from data bus D_8–D_{15} to D_0–D_7 in the M37721's DMAC. Note the data copy delay time $t_{d(data)}$.
16 bits	Data bus D_8–D_{15}	8 bits	Even address	 **Note:** Data is copied from data bus D_0–D_7 to D_8–D_{15} or from data bus D_8–D_{15} to D_0–D_7 in the M37721's DMAC. Note the data copy delay time $t_{d(data)}$.
			Odd address	
8 bits	Data bus D_0–D_7	8 bits	Even address and Odd address	

Note: When the memory is the internal memory or SFR, the above case for external data bus width = 16 bits applies.

EOL announced

Table 13.4.7 Outputs of address bus, data bus, and bus control signals in 1-bus cycle transfer

External data bus width	I/O connection	Transfer unit	Read/Write address of memory	Output of address bus, data bus, and bus control signals	
				Transferred from memory to I/O	Transferred from I/O to memory
16 bits	Data bus D_0–D_7 and D_8–D_{15}	16 bits	Even address and Odd address	A_8/D_8–A_{15}/D_{15}: Transfer source address / Odd address data A_{16}/D_0–A_{23}/D_7: Transfer source address / Even address data $\overline{BHE}$: L $\overline{BLE}$: L	Transfer destination address Transfer destination address L L
16 bits	Data bus D_0–D_7	8 bits	Even address	A_8/D_8–A_{15}/D_{15}: Transfer source address / Invalid data A_{16}/D_0–A_{23}/D_7: Transfer source address / Even address data $\overline{BHE}$: H $\overline{BLE}$: L	Transfer destination address / Invalid data Transfer destination address H L
			Odd address	A_8/D_8–A_{15}/D_{15}: Transfer source address / Copy A_{16}/D_0–A_{23}/D_7: Transfer source address / Odd address data $\overline{BHE}$: L $\overline{BLE}$: H	Transfer destination address / I/O data Transfer destination address / Copy L H
16 bits	Data bus D_8–D_{15}	8 bits	Even address	A_8/D_8–A_{15}/D_{15}: Transfer source address / Even address data A_{16}/D_0–A_{23}/D_7: Transfer source address / Copy $\overline{BHE}$: H $\overline{BLE}$: L	Transfer destination address / Copy Transfer destination address / I/O data H L
			Odd address	A_8/D_8–A_{15}/D_{15}: Transfer source address / Odd address data A_{16}/D_0–A_{23}/D_7: Transfer source address / Invalid data $\overline{BHE}$: L $\overline{BLE}$: H	Transfer destination address Transfer destination address / Invalid data L H
8 bits	Data bus D_0–D_7	8 bits	Even address and Odd address	A_8–A_{15}: Transfer source address A_{16}/D_0–A_{23}/D_7: Transfer source address / Data	Transfer destination address Transfer destination address

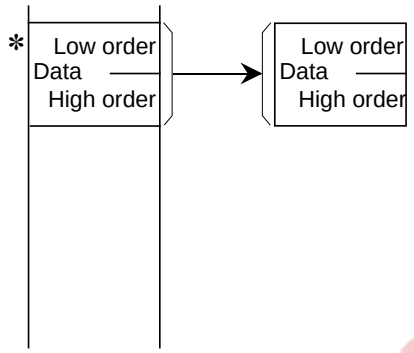
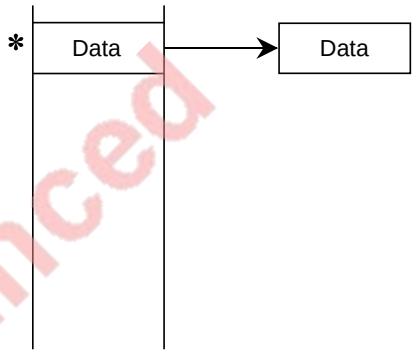
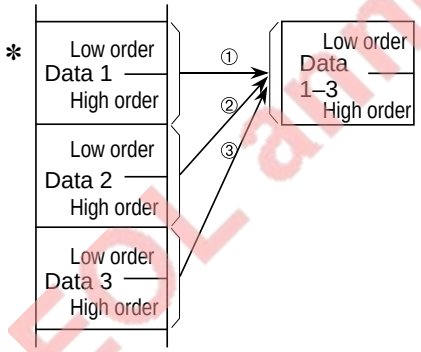
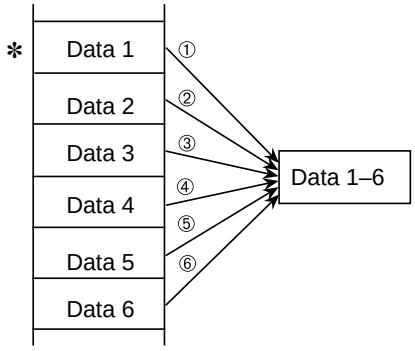
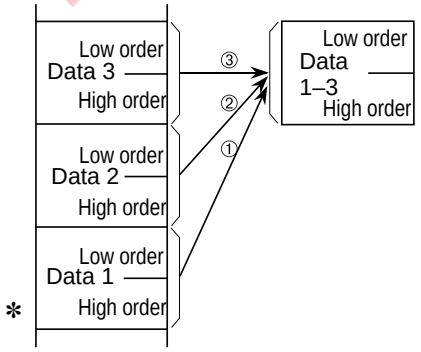
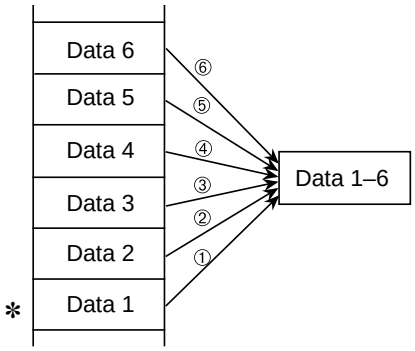
: When the memory is the internal memory or SFR, data is output. When the memory is the external memory, it enters a floating state.

EOL announced

(3) Address directions in 1-bus cycle transfer

In 1-bus cycle transfer, the transfer source and destination address directions of memory are selectable. (Refer to **"Figure 13.2.6."**) Addresses move in the specified direction by the transfer unit.
Tables 13.4.8 and 13.4.9 list address directions in 1-bus cycle transfer and examples of transfer results.

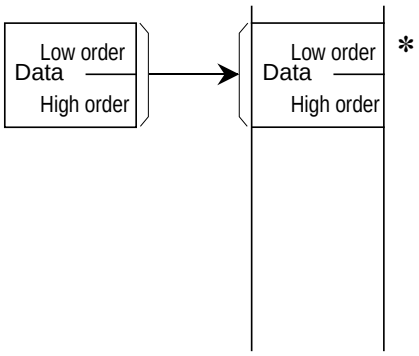
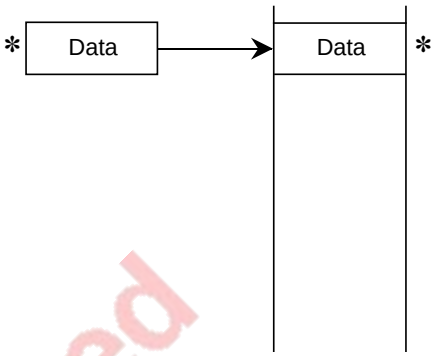
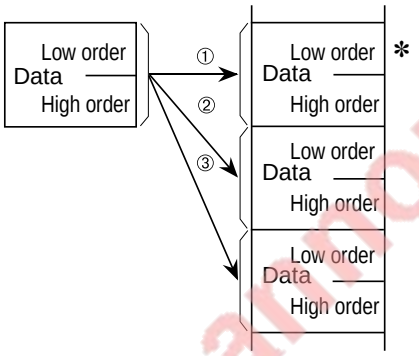
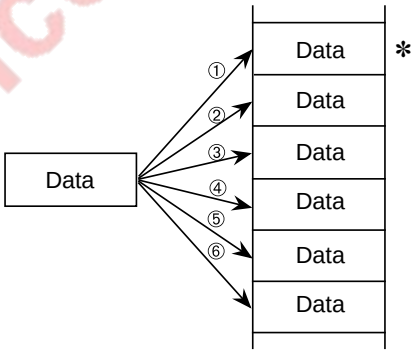
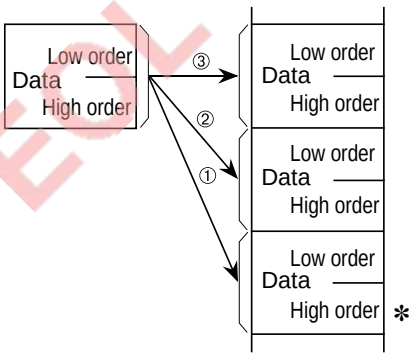
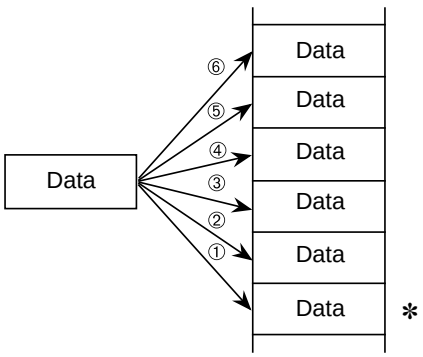
Table 13.4.8 Address directions in 1-bus cycle transfer and examples of transfer results (1)

Address direction		External data bus width : 16 bits			External data bus width : 16 bits or 8 bits		
		Transfer unit : 16 bits			Transfer unit : 8 bits		
Transfer source memory	Transfer destination I/O	Data arrangement on transfer source memory	Transfer sequence	Transfer destination I/O	Data arrangement on transfer source memory	Transfer sequence	Transfer destination I/O
Fixed	—						
Forward	—						
Backward	—						

✻ : Transfer start

EOL announced

Table 13.4.9 Address directions in 1-bus cycle transfer and examples of transfer results (2)

Address direction		External data bus width : 16 bits			External data bus width : 16 bits or 8 bits		
		Transfer unit : 16 bits			Transfer unit : 8 bits		
Transfer source I/O	Transfer destination memory	Transfer source I/O	Transfer sequence	Data arrangement on transfer destination memory (transfer result)	Transfer source I/O	Transfer sequence	Data arrangement on transfer destination memory (transfer result)
—	Fixed						
—	Forward						
—	Backward						

*: Transfer start

EOL announced

[Precautions for 1-bus cycle transfer]

1. The area that overlaps with internal RAM and SFRs must not be assigned to an external memory. When the contents in the overlapped area are read, the data of internal RAM or SFRs and that of external memory are simultaneously placed on the data bus; and they collide with each other.

2. For the system that transfers data with 16-bit external data bus width from an external memory to an 8-bit I/O, the external memory must be composed to be read in a unit of 8 bits.
If the external memory cannot be read in a unit of 8 bits, the data read from the external memory at data copy collides with the copied data on the data bus.

3. Under the following conditions, 1-bus cycle transfer cannot be performed. (Refer to **"Table 13.4.5."**):

- External data bus width = 16 bits, Transfer unit = 16 bits, Address direction of memory = Fixed or Forward, Data's start address of memory = Odd
- External data bus width = 16 bits, Transfer unit = 16 bits, Address direction of memory = Backward, Data's start address of memory = Even
- External data bus width = 16 bits, Transfer unit = 16 bits, Target memory of DMA transfer = DRAM area, Address direction of memory = Backward, Data's start address of memory = Odd
- External data bus width = 8 bits, Transfer unit = 16 bits

EOL announced

13.4.3 Burst transfer mode

The burst transfer mode can operate in either edge sense or level sense mode.

(1) Burst transfer mode (edge sense)

When the transfer mode select bit = "0" and the edge sense/level sense select bit = "0," this mode is selected. (Refer to **"Figures 13.2.6 and 13.2.8."**)

In this mode, all of the DMA request sources are available.

Figure 13.4.8 shows a transfer example in the burst transfer mode (edge sense).
When once a DMA request is accepted in this mode, an entire batch of data is transferred: the right to use bus is not returned to the CPU until the transfer is complete.
During a burst transfer, any DMA request (including that of other channels) cannot be accepted. However, the BUS REQUEST signal is sampled basically at every completion of 1-unit transfer. (Refer to **"Table 13.2.3."**) When a DRAM refresh request or Hold request is generated at this time, the right to use bus is not returned to the CPU, and the request is accepted.
When the transfer of an entire batch of data is complete, the DMAC relinquishes the right to use bus to the CPU. When the next DMA request is generated, the right is once returned to the CPU to sample the DMA request.

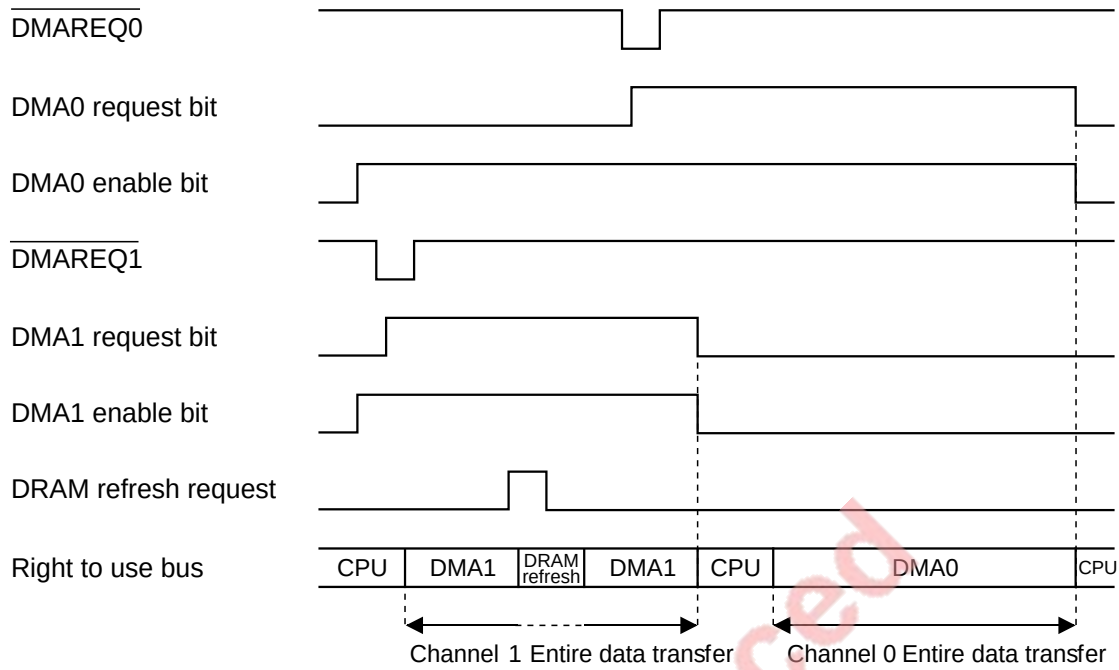
(2) Burst transfer mode (level sense)

When the transfer mode select bit = "0" and the edge sense/level sense select bit = "1 ," this mode is selected. (Refer to **"Figures 13.2.6 and 13.2.8."**)

In this mode, only the external source is used as a DMA request source. Set the DMA request source select bits to "0001_2." (Refer to **"Figure 13.2.8."**)

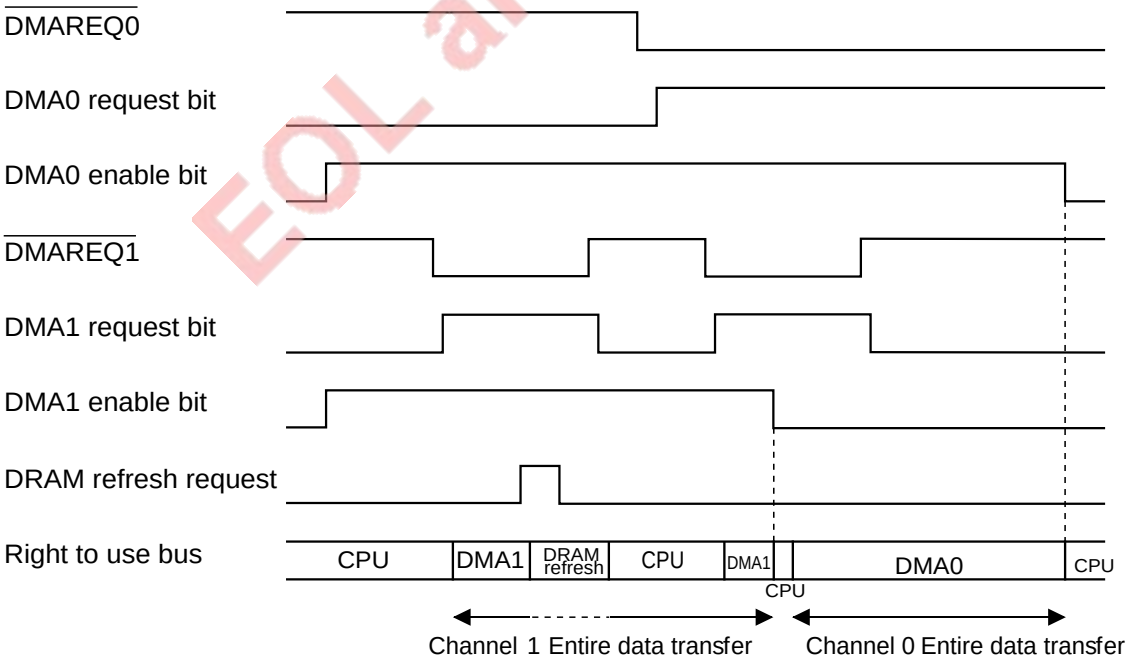
Figure 13.4.9 shows a transfer example in the burst transfer mode (level sense).
When the $\overline{\text{DMAREQi}}$ pin's input level = "L," the DMAi request bit is cleared to "0"; when this pin's input level = "L," the DMAi request bit is set to "1."
Therefore, when the $\overline{\text{DMAREQi}}$ pin's input level is "L" with the DMAi enable bit = "1," a DMA transfer starts. When the $\overline{\text{DMAREQi}}$ pin's input level goes from "L" to "H," the right to use bus will be returned to the CPU at completion of 1-unit transfer under execution at that time. When the $\overline{\text{DMAREQi}}$ pin's input level goes "L" again, the DMA transfer restarts at the next address.
Once a DMAi transfer starts, any DMA request (including that of other channels) cannot be accepted, even if the $\overline{\text{DMAREQi}}$ pin's input level is "H," until the transfer is terminated normally or forcibly. However, the BUS REQUEST signal is sampled basically at every completion of 1-unit transfer. (Refer to **"Table 13.2.3."**) When a DRAM refresh request or Hold request is generated at this time, the right to use bus is not returned to the CPU, and the request is accepted.
When the transfer of an entire batch of data is complete, the DMAC relinquishes the right to the CPU. If the next DMA request is generated, the right is once returned to the CPU to sample the DMA request.

Not announced



This example applies on the following conditions:
• Both of DMA0 and DMA1 request sources are external sources.
• Channel priority level: Fixed (Channel 0 > Channel 1)

Fig. 13.4.8 Transfer example in the burst transfer mode (edge sense)



This example applies on the following conditions:
• Channel priority level: Fixed (Channel 0 > Channel 1)

Fig. 13.4.9 Transfer example in the burst transfer mode (level sense)

EOL announced

[Precautions for burst transfer mode]

1. In the burst transfer mode (edge sense), the DMAi request bit is cleared to “0” when the transfer of an entire batch of data is complete or the transfer is forced into termination. Therefore, another DMA request of the same channel i is invalid if generated during DMAi transfer.

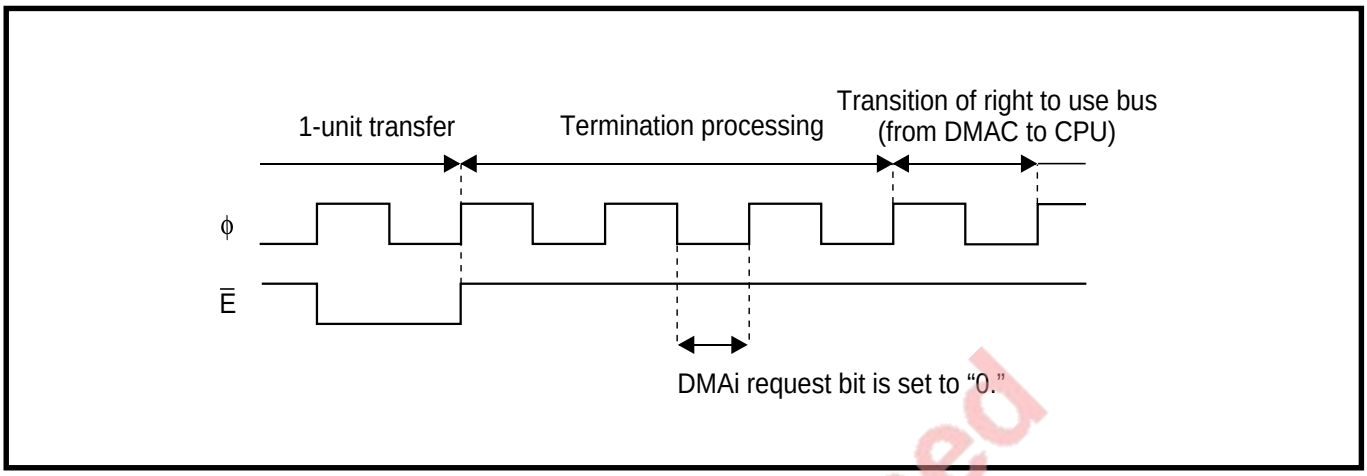


Fig. 13.4.10 Timing when clearing DMAi request bit to “0” in burst transfer mode

2. Because interrupt priority levels are determined while the CPU fetches an operation code, interrupt requests are not accepted during a DMA transfer. In the burst transfer mode (edge sense), therefore, interrupt requests cannot be accepted until the transfer of an entire batch of data is complete or the transfer is forced into termination.

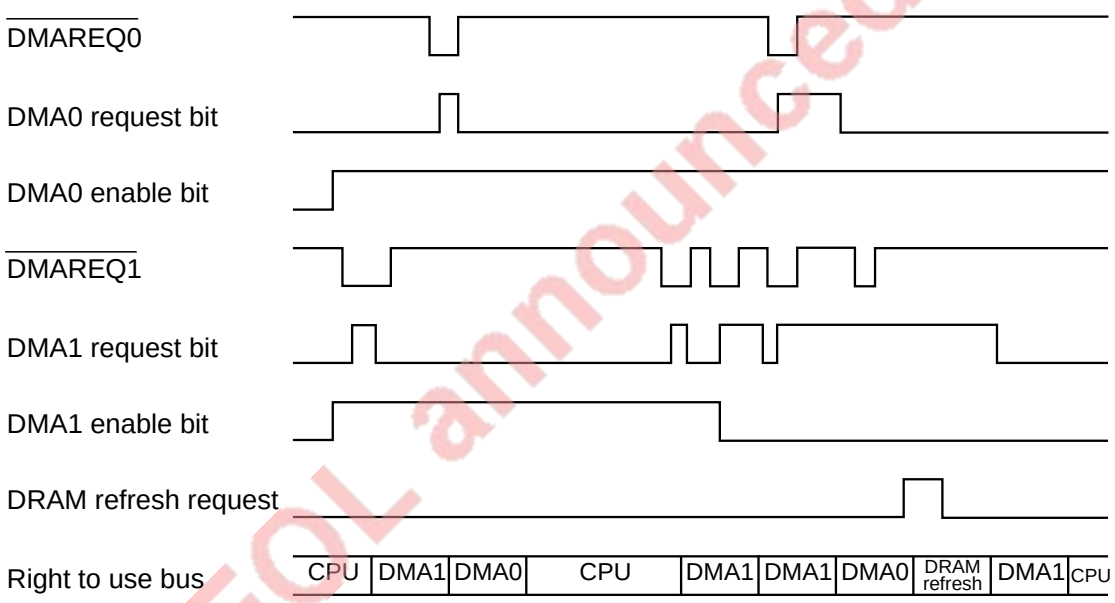
EOL announced

13.4.4 Cycle-steal transfer mode

When the transfer mode select bit = “1” and the edge sense/level sense select bit = “0,” this mode is selected. (Refer to **“Figures 13.2.6 and 13.2.8.”**)

In this mode, all of the DMA request sources are available.

Figure 13.4.11 shows a transfer example in the cycle-steal transfer mode
1-transfer-unit data is transferred for each DMA request.
The BUS REQUEST signal is sampled basically at every completion of 1-unit transfer. (Refer to **“Table 13.2.3.”**) When a DRAM refresh request or Hold request is generated at this time, the right to use bus is not returned to the CPU, and the request is accepted. When several DMA requests are generated, the request of the channel which has the highest priority among them is accepted, and DMA transfer is performed without returning the right to use bus to the CPU. When any request is not generated, the CPU gains the right.



This example applies on the following conditions:
• Both of DMA0 and DMA1 request sources are external sources
• Channel priority level: Fixed (Channel 0 > Channel 1)

Fig. 13.4.11 Transfer example in cycle-steal transfer mode

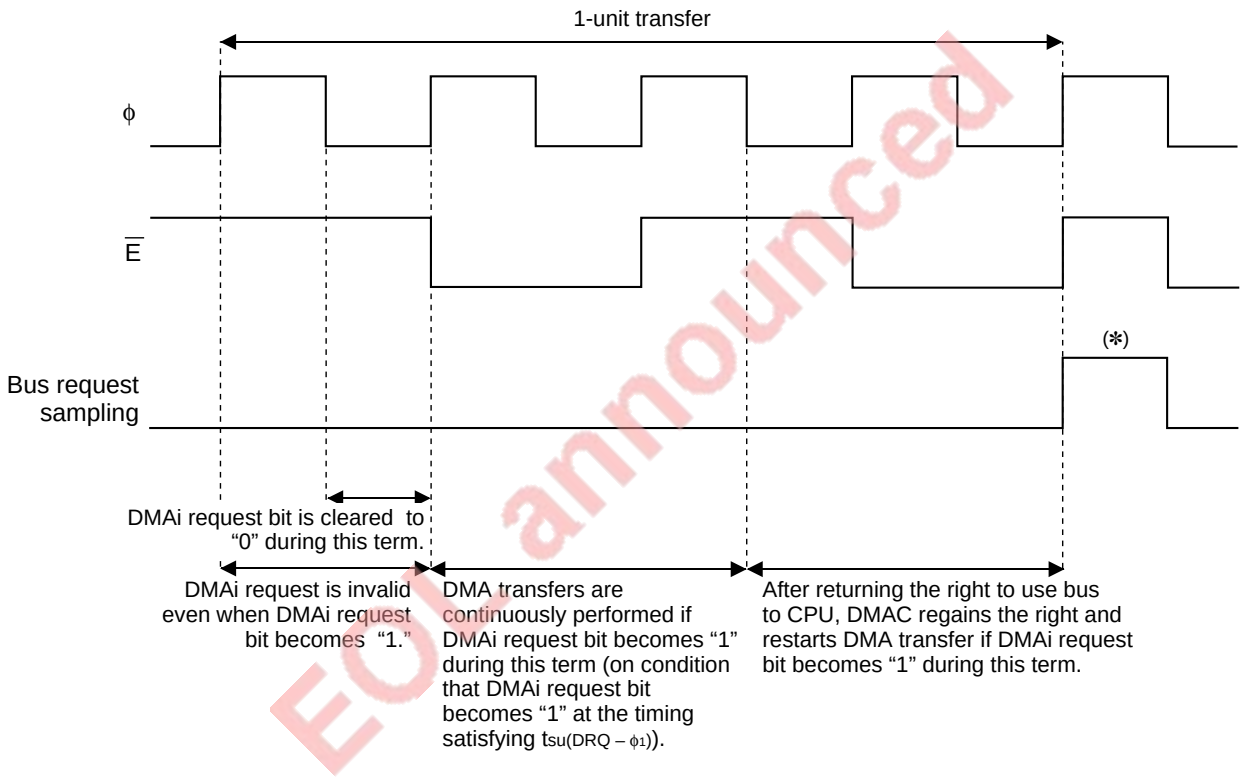
EOL announced

[Precautions for cycle-steal transfer mode]

1. When DMA transfers of the same channel are continuously performed

In the cycle-steal transfer mode, the DMAi request bit is cleared to "0" in every 1-unit transfer. Also, it takes 1.5 cycles of ϕ from the generation of a DMA request until that of a BUS REQUEST (DMAC). Therefore, if another DMA request of the same channel i is generated during a DMAi transfer in the cycle-steal transfer mode, any one of the following three cases occurs depending on the timing of request generation:

- The DMA request becomes invalid.
- The DMA transfer continues without returning the right to use bus.
- After returning the right to use bus to the CPU, the DMAC regains the right and restarts the DMA transfer.



When a DMAi request is generated at the following timings, it is not in time to the next bus request sampling (*). Therefore, DMAC returns the right to use bus to the CPU. Then, DMAC regains the right and restarts the DMA transfer.

- Except for the last 1-unit transfer, 1-bus cycle transfer is selected without Wait.
- Except for the last 1-unit transfer, 1-bus cycle transfer is selected with Wait. In addition, a time of 0.5 cycle of ϕ is less than $t_{su(DRQ-\phi 1)}$.

Fig. 13.4.12 Conditions for performing DMA transfers of the same channel continuously

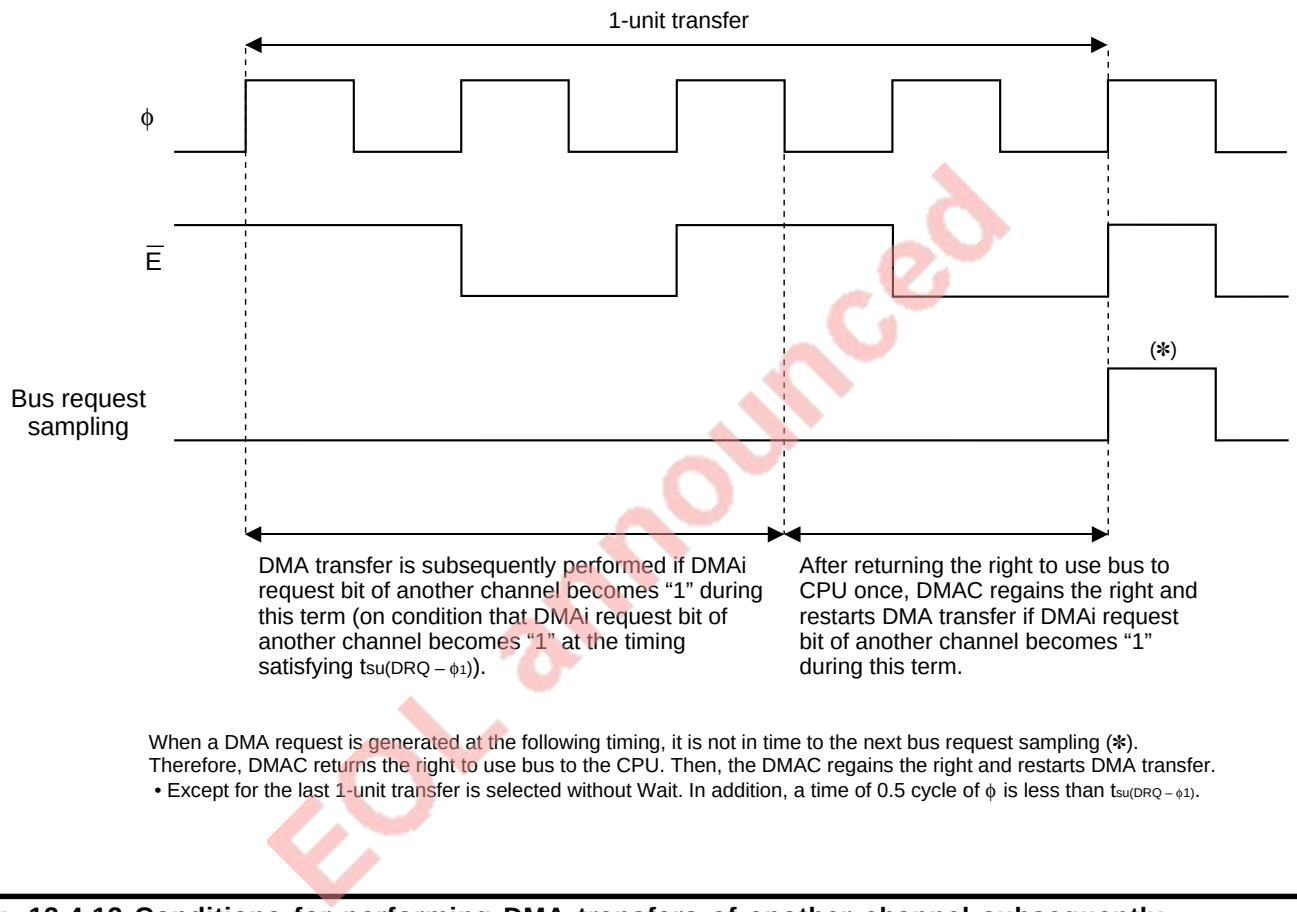
EOL announced

2. When a DMA transfer of another channel is subsequently performed

In the cycle-steal transfer mode, it takes 1.5 cycles of ϕ from the generation of a DMA request until that of a BUS REQUEST (DMAC).

Therefore, if a DMA request of another channel is generated during a DMAi transfer in the cycle-steal mode, either one of the following two cases occurs depending on its timing of request generation:

- The DMAC performs the DMA transfer subsequently without returning the right to use bus.
- After returning the right to use bus to the CPU once, the DMAC regains the right and performs the DMA transfer.



EOL announced

Fig. 13.4.13 Conditions for performing DMA transfers of another channel subsequently

13.5 Single transfer mode

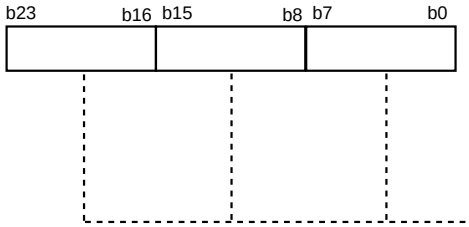
This mode is used to transfer a block of data once.
Table 13.5.1 lists the specifications of the single transfer mode, and Figure 13.5.1 shows the register structures of SARi, DARi, and TCRi in this mode.

Table 13.5.1 Specifications of single transfer mode

Item	Performance specifications
Transfer parameter memory	Not required.
Condition of normal termination	TCRi = 0
Conditions of forced termination	● Falling edge of the $\overline{TC}$ pin's input from "H" to "L" (when the $\overline{TC}$ pin validity bit ="1") ● Write "0" to the DMAi enable bit
Interrupt request generation timing	At normal termination
Functions of registers	SARi latch: Indicates the transfer start address of data block at the transfer source. SARi: Indicates the address of the next transfer source. DARi latch: Indicates the transfer start address of data block at the transfer destination. DARi: Indicates the address of the next transfer destination. TCRi latch: Indicates the number of transfer bytes. TCRi: Indicates the number of remaining transfer bytes.

$\overline{TC}$ pin validity bit: Bit 1 at address 68_{16}

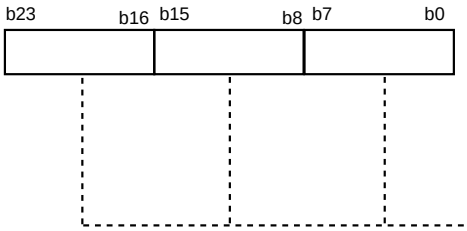
EOL announced



Source address register 0 (Addresses $1FC2_{16}$ to $1FC0_{16}$) (SAR0)
Source address register 1 (Addresses $1FD2_{16}$ to $1FD0_{16}$) (SAR1)
Source address register 2 (Addresses $1FE2_{16}$ to $1FE0_{16}$) (SAR2)
Source address register 3 (Addresses $1FF2_{16}$ to $1FF0_{16}$) (SAR3)

Bit	Functions	At reset	RW
23 to 0	[Write] Set the transfer start address of the source. These bits can be set to "000000_{16}" to "$FFFFFF_{16}$." [Read] The read value indicates the source address of data which is next transferred.	Undefined	RW

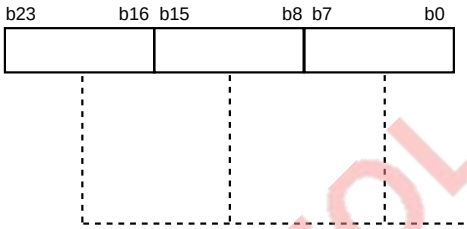
Note: When writing to this register, write to all 24 bits.



Destination address register 0 (Addresses $1FC6_{16}$ to $1FC4_{16}$) (DAR0)
Destination address register 1 (Addresses $1FD6_{16}$ to $1FD4_{16}$) (DAR1)
Destination address register 2 (Addresses $1FE6_{16}$ to $1FE4_{16}$) (DAR2)
Destination address register 3 (Addresses $1FF6_{16}$ to $1FF4_{16}$) (DAR3)

Bit	Functions	At reset	RW
23 to 0	[Write] Set the transfer start address of the destination. These bits can be set to "000000_{16}" to "$FFFFFF_{16}$." [Read] The read value indicates the destination address of data which is next transferred.	Undefined	RW

Note: When writing to this register, write to all 24 bits.



Transfer counter register 0 (Addresses $1FCA_{16}$ to $1FC8_{16}$) (TCR0)
Transfer counter register 1 (Addresses $1FDA_{16}$ to $1FD8_{16}$) (TCR1)
Transfer counter register 2 (Addresses $1FEA_{16}$ to $1FE8_{16}$) (TCR2)
Transfer counter register 3 (Addresses $1FFA_{16}$ to $1FF8_{16}$) (TCR3)

Bit	Functions	At reset	RW
23 to 0	[Write] Set the byte number of the transfer data. These bits can be set to "000001_{16}" to "$FFFFFF_{16}$." [Read] The read value indicates remaining byte number of the transfer data.	Undefined	RW

Note: When writing to this register, write to all 24 bits.
Do not set this register to "000000_{16}."

EOL announced

Fig. 13.5.1 Register structures of SARi, DARi, and TCRi in single transfer mode

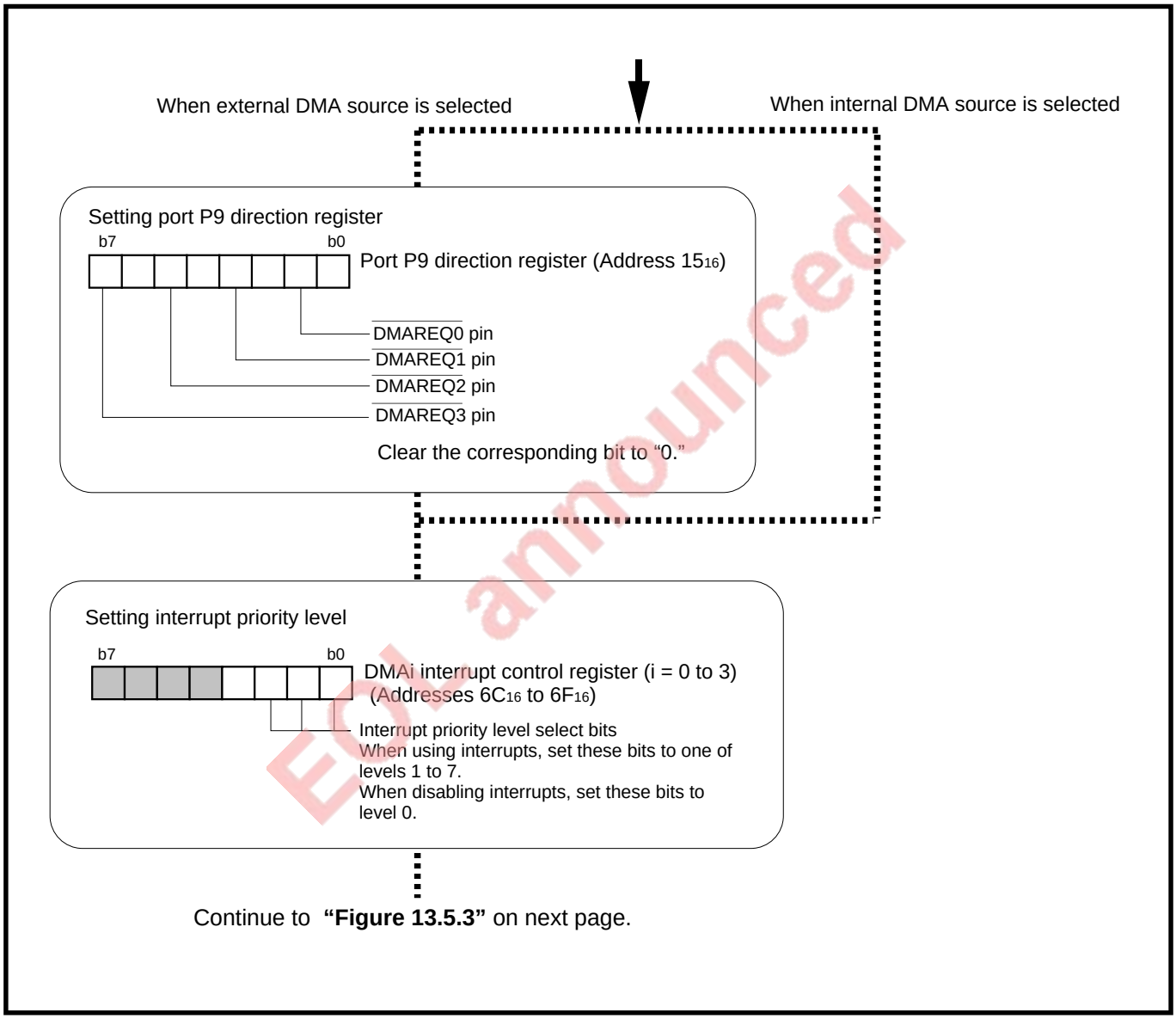
13.5 Single transfer mode

13.5.1 Setting of single transfer mode

Figures 13.5.2 through 13.5.4 show an initial setting example for registers relevant to the single transfer mode.

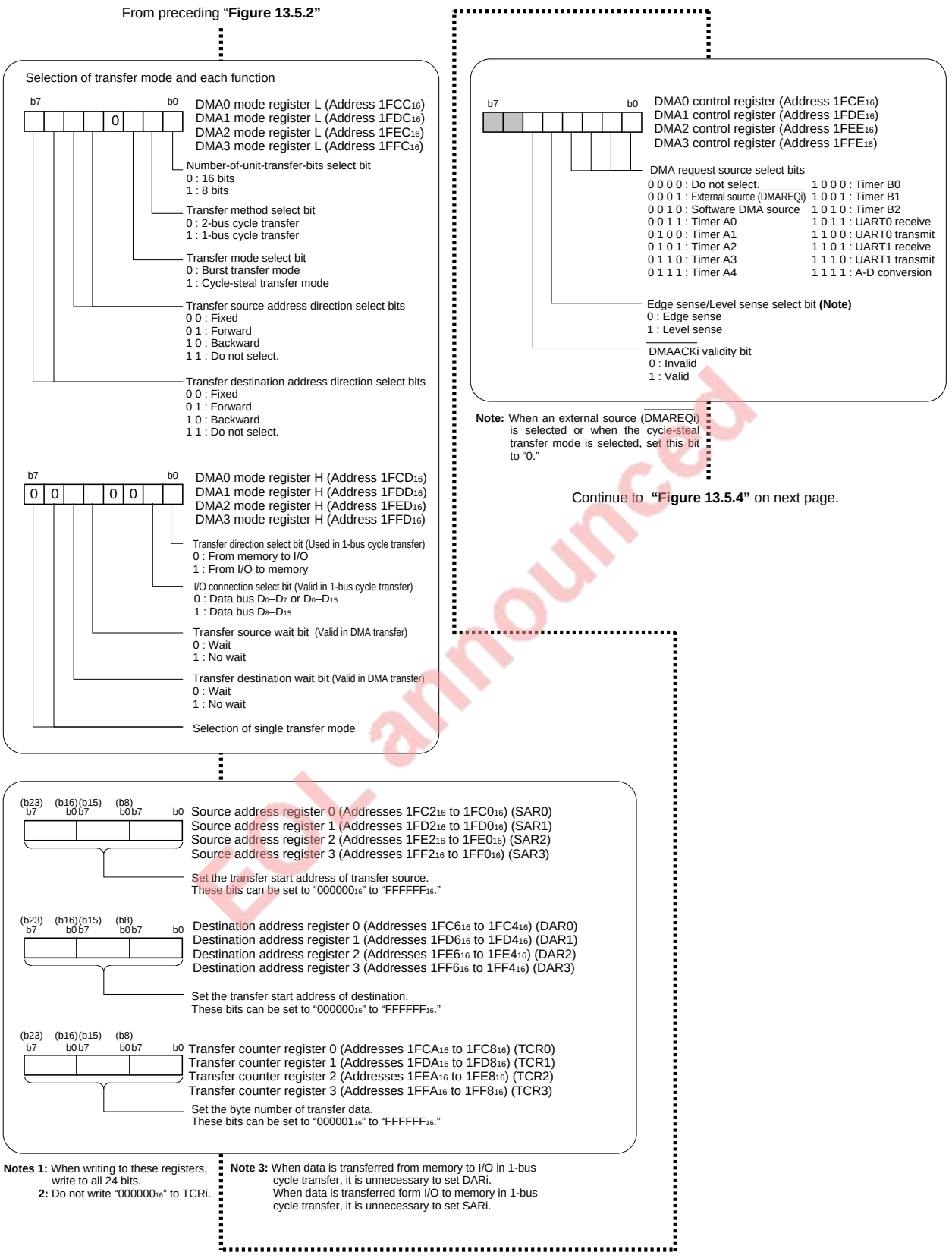
In addition, when timer A, timer B, UART, or the A-D converter is selected as a DMA request source, the setting for the peripheral is required. For details of the setting, refer to the chapter of each peripheral function.

When a DMAi interrupt is used, the setting for enabling the interrupt is also required. For details, refer to **"CHAPTER 7. INTERRUPTS."**



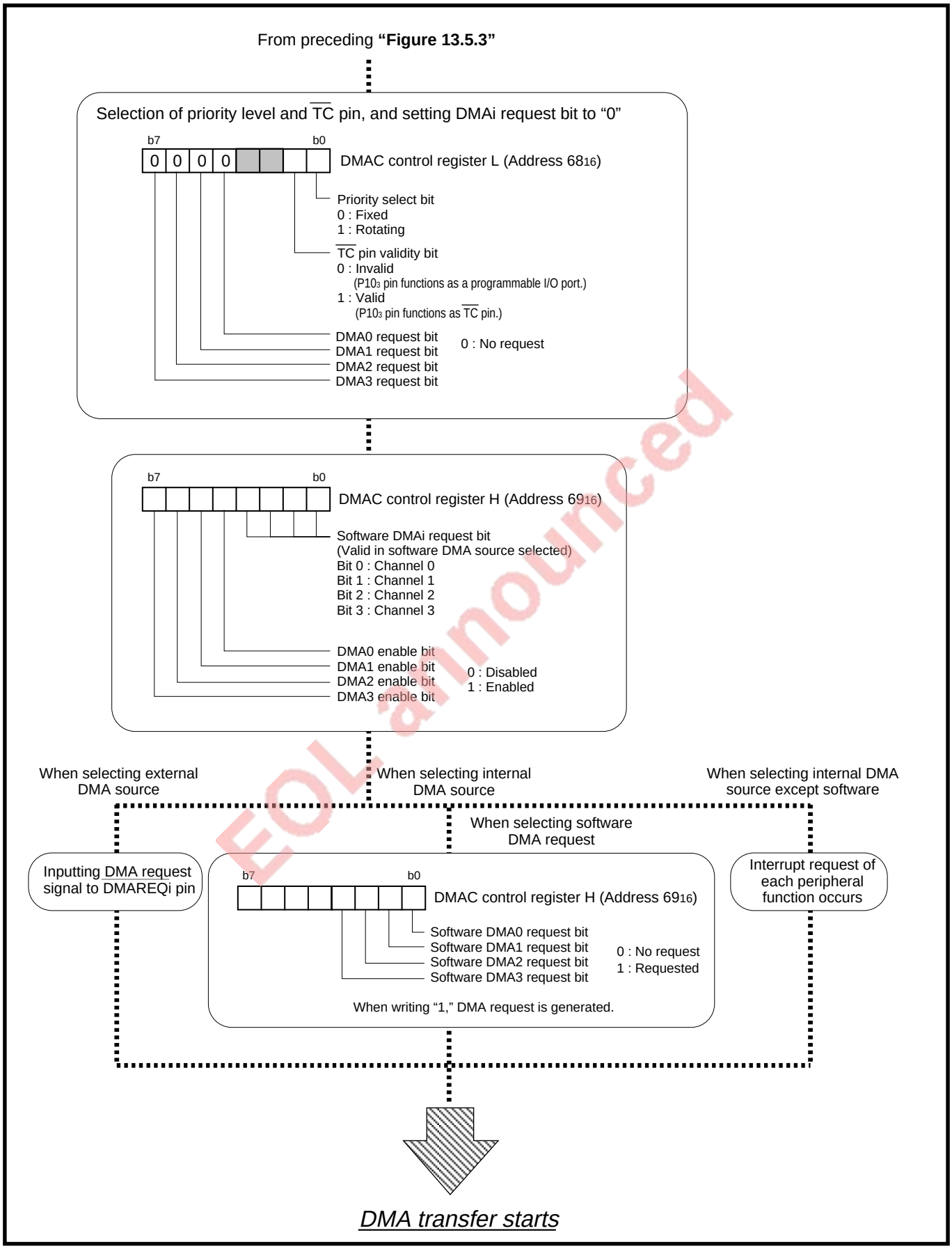
EOL announced

Fig. 13.5.2 Initial setting example for registers relevant to single transfer mode (1)



DOL announced

Fig. 13.5.3 Initial setting example for registers relevant to single transfer mode (2)



EOL announced

Fig. 13.5.4 Initial setting example for registers relevant to single transfer mode (3)

13.5.2 Operation in single transfer mode

Figure 13.5.5 shows the operation flowchart of the single transfer mode, and Figure 13.5.6 shows a timing diagram of the single transfer mode (burst transfer mode).
For the cycle-steal transfer mode, refer to the following:

- All transfers except for the last 1-unit transfer: Figure 13.8.12
- Last 1-unit transfer: Figure 13.8.14

Also, refer to section **"13.2.1 Bus access control circuit"** for the bus request sampling during transfer.

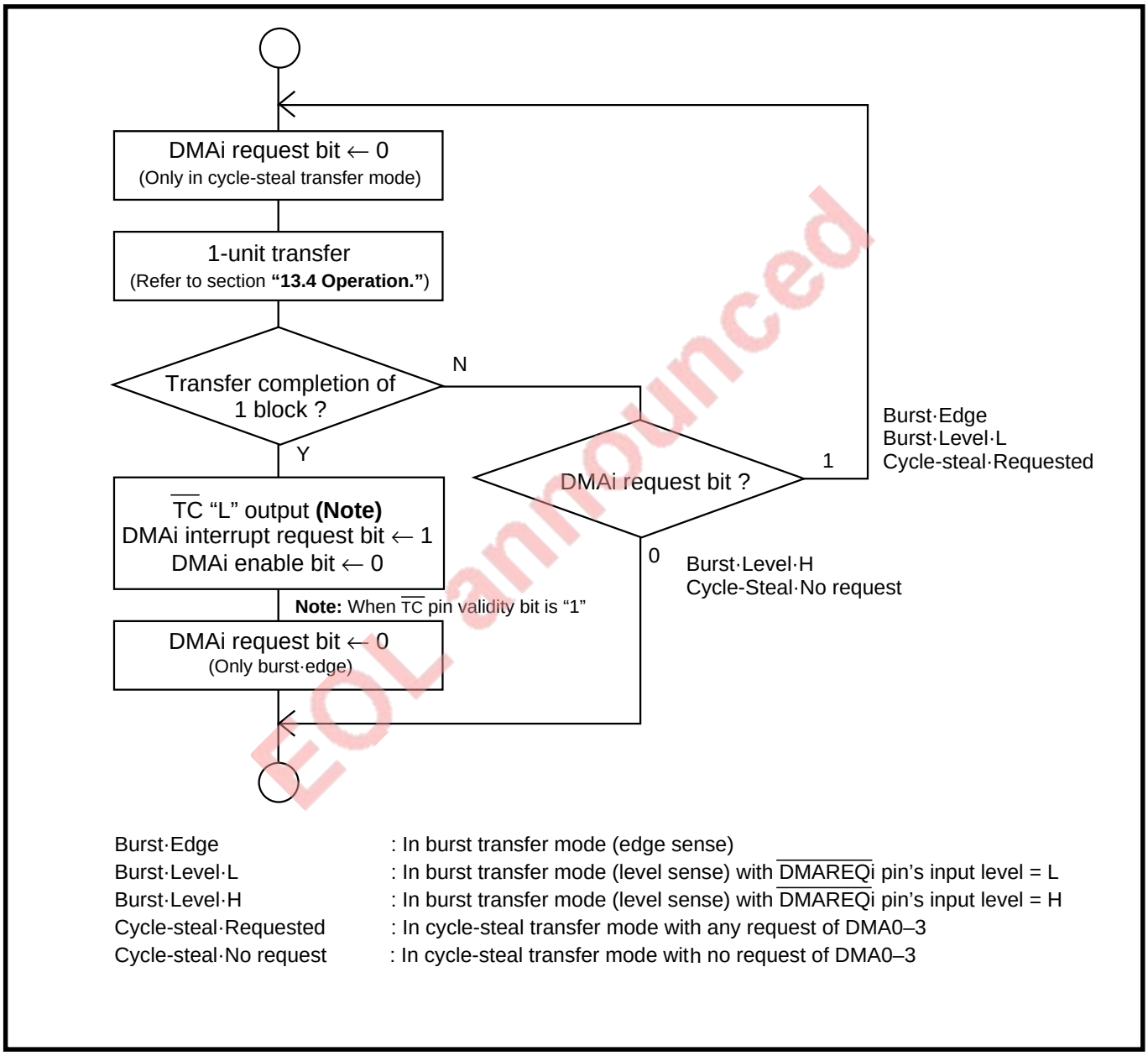
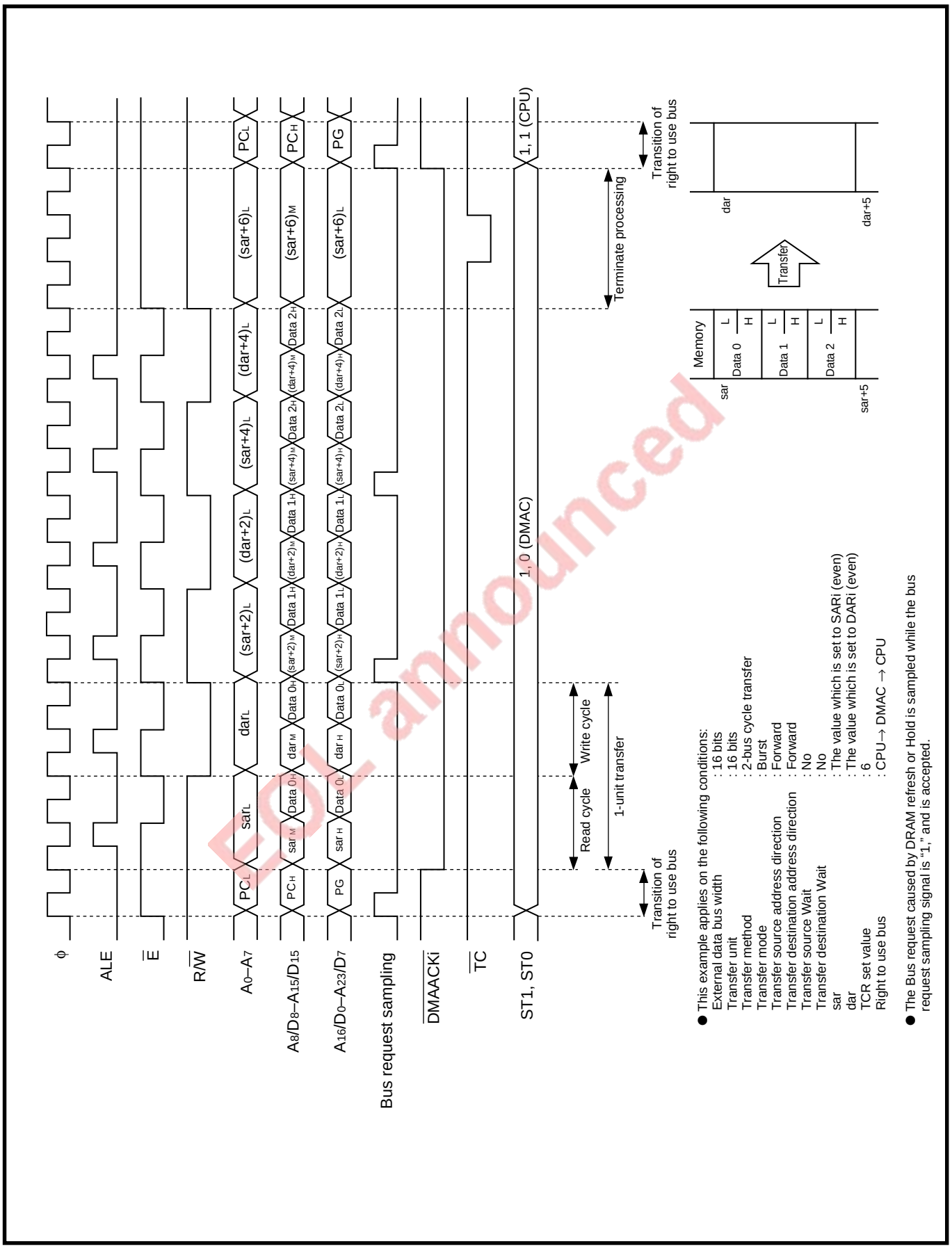


Fig. 13.5.5 Operation flowchart of single transfer mode



EOL announced

Fig. 13.5.6 Timing diagram of single transfer mode (burst transfer mode)

13.6 Repeat transfer mode

This mode is used to transfer one block of data repeatedly. Table 13.6.1 lists the specifications of the repeat transfer mode, and Figure 13.6.1 shows the register structures of SARi, DARi, and TCRi in this mode.

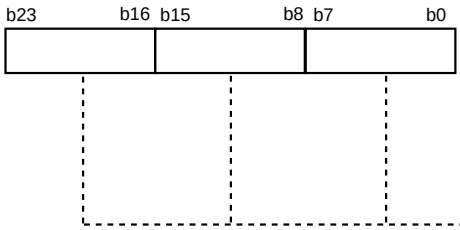
Table 13.6.1 Specifications of repeat transfer mode

Item	Performance specifications
Transfer parameter memory	Not required
Condition of normal termination	— (No normal termination)
Conditions of forced termination	● Falling edge of the $\overline{TC}$ pin's input from "H" to "L" (when the $\overline{TC}$ pin validity bit = "1") ● Write "0" to the DMAi enable bit
Interrupt request generation timing	No request is generated.
Functions of registers	SARi latch: Indicates the transfer start address of data block at the transfer source. SARi: Indicates the address of the next transfer source. DARi latch: Indicates the transfer start address of data block at the transfer destination. DARi: Indicates the address of the next transfer destination. TCRi latch: Indicates the number of transfer bytes. TCRi: Indicates the number of remaining bytes being transferred.

$\overline{TC}$ pin validity bit: Bit 1 at address 68_{16}

EOL announced

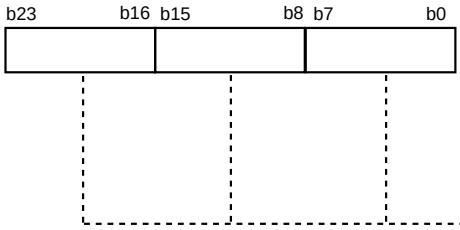
13.6 Repeat transfer mode



Source address register 0 (Addresses $1FC2_{16}$ to $1FC0_{16}$) (SAR0)
Source address register 1 (Addresses $1FD2_{16}$ to $1FD0_{16}$) (SAR1)
Source address register 2 (Addresses $1FE2_{16}$ to $1FE0_{16}$) (SAR2)
Source address register 3 (Addresses $1FF2_{16}$ to $1FF0_{16}$) (SAR3)

Bit	Functions	At reset	RW
23 to 0	[Write] Set the transfer start address of the source. These bits can be set to "000000_{16}" to "$FFFFFF_{16}$." [Read] The read value indicates the source address of data which is next transferred.	Undefined	RW

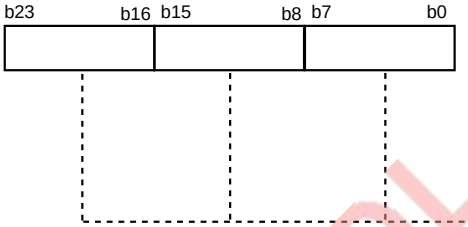
Note: When writing to this register, write to all 24 bits.



Destination address register 0 (Addresses $1FC6_{16}$ to $1FC4_{16}$) (DAR0)
Destination address register 1 (Addresses $1FD6_{16}$ to $1FD4_{16}$) (DAR1)
Destination address register 2 (Addresses $1FE6_{16}$ to $1FE4_{16}$) (DAR2)
Destination address register 3 (Addresses $1FF6_{16}$ to $1FF4_{16}$) (DAR3)

Bit	Functions	At reset	RW
23 to 0	[Write] Set the transfer start address of the destination. These bits can be set to "000000_{16}" to "$FFFFFF_{16}$." [Read] The read value indicates the destination address of data which is next transferred.	Undefined	RW

Note: When writing to this register, write to all 24 bits.



Transfer counter register 0 (Addresses $1FCA_{16}$ to $1FC8_{16}$) (TCR0)
Transfer counter register 1 (Addresses $1FDA_{16}$ to $1FD8_{16}$) (TCR1)
Transfer counter register 2 (Addresses $1FEA_{16}$ to $1FE8_{16}$) (TCR2)
Transfer counter register 3 (Addresses $1FFA_{16}$ to $1FF8_{16}$) (TCR3)

Bit	Functions	At reset	RW
23 to 0	[Write] Set the byte number of transfer data. These bits can be set to "000001_{16}" to "$FFFFFF_{16}$." [Read] The read value indicates the remaining byte number of the block which is being transferred.	Undefined	RW

Note: When writing to this register, write to all 24 bits.
Do not write "000000_{16}" to this register.

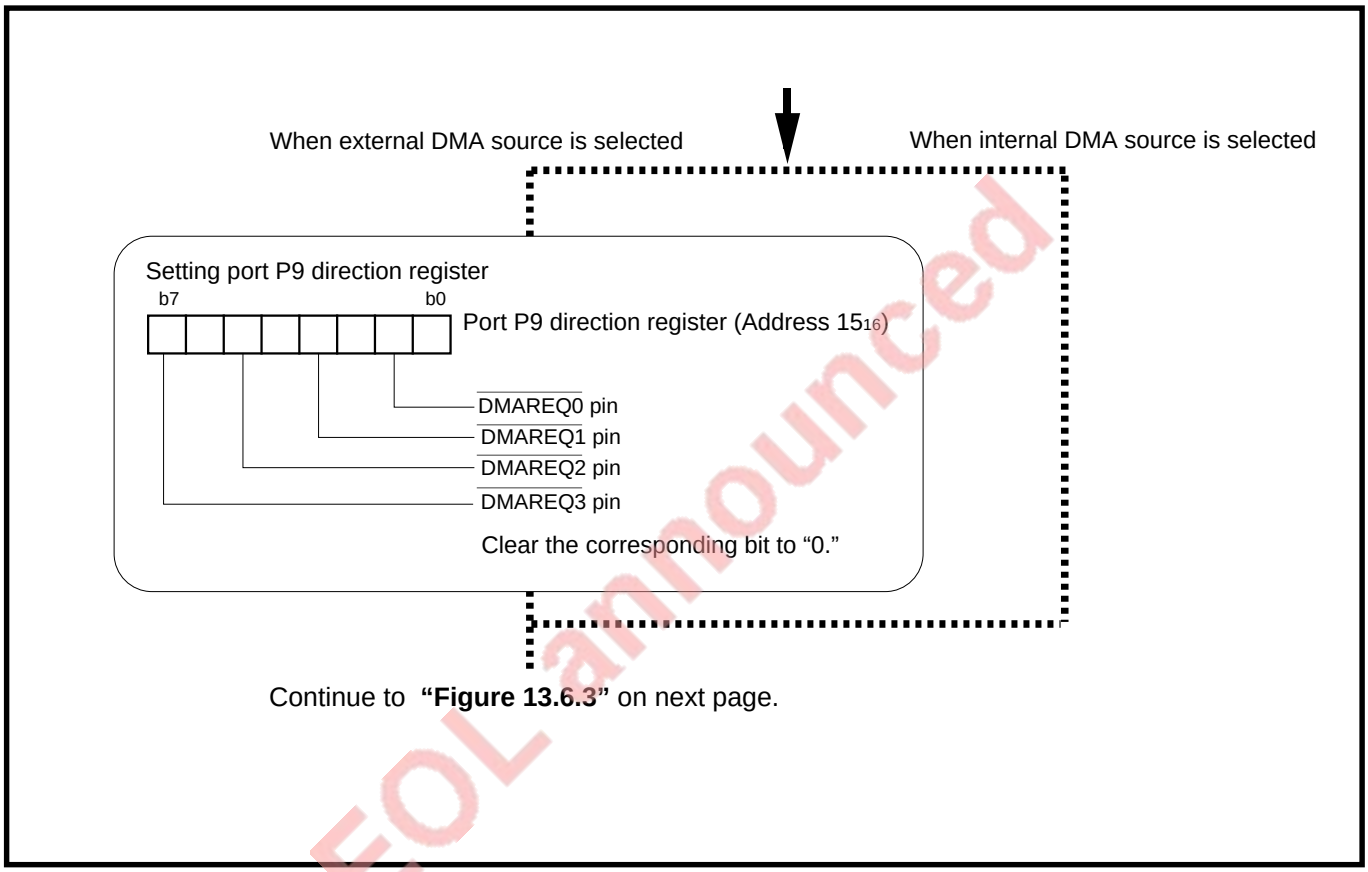
EOL announced

Fig. 13.6.1 Register structures of SARi, DARi, and TCRi in repeat transfer mode

13.6.1 Setting of repeat transfer mode

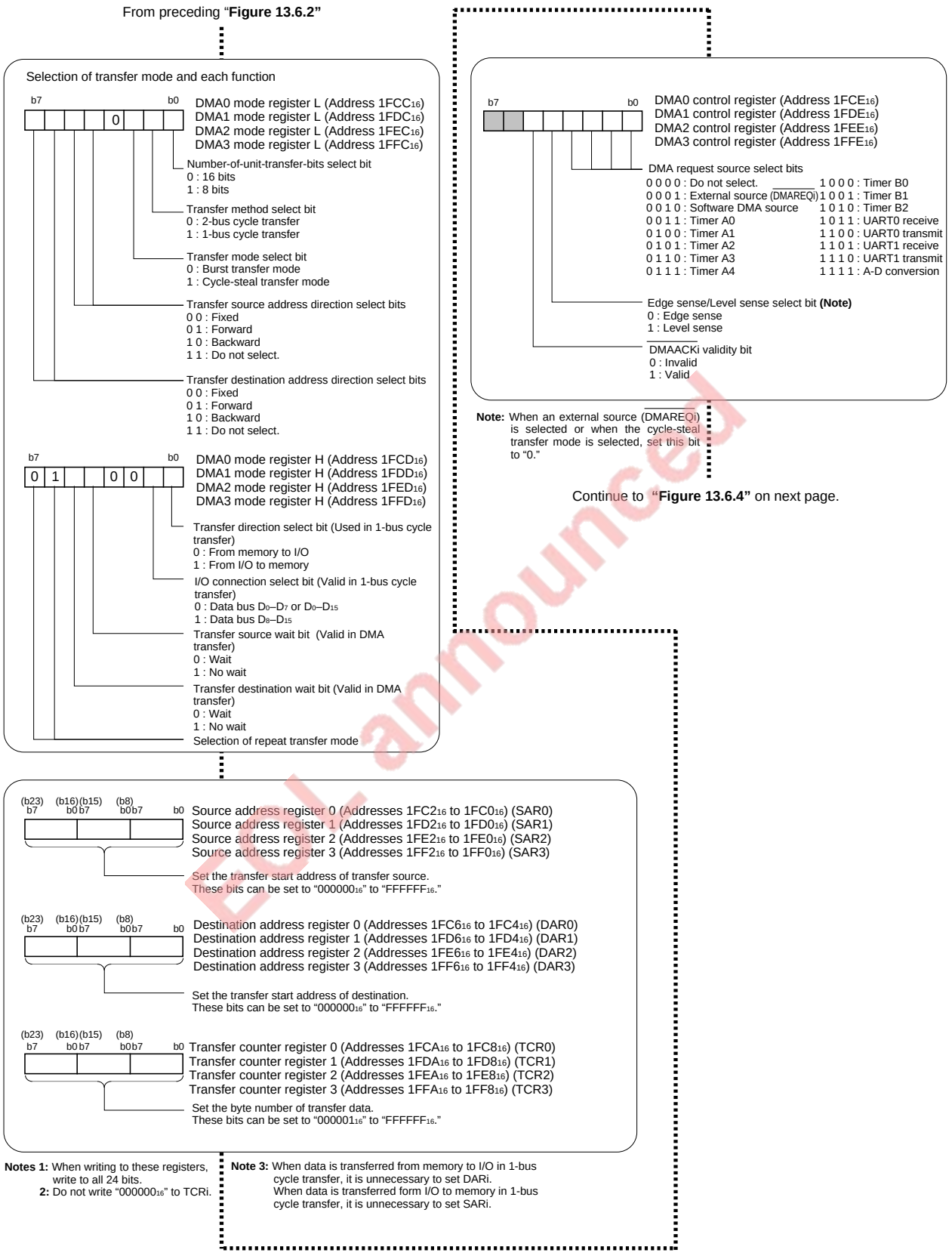
Figures 13.6.2 through 13.6.4 show an initial setting example for registers relevant to the repeat transfer mode. In addition, when timer A, timer B, UART, or the A-D converter is selected as a DMA request source, the setting for the peripheral is required. For details of the setting, refer to the chapter of each peripheral function.

In this mode, only the forced termination can terminate the DMA transfer. (Refer to section "**13.3.5 (2) Forced termination."** Therefore, in the burst transfer mode (edge sense selected), be sure to validate the $\overline{TC}$ pin.



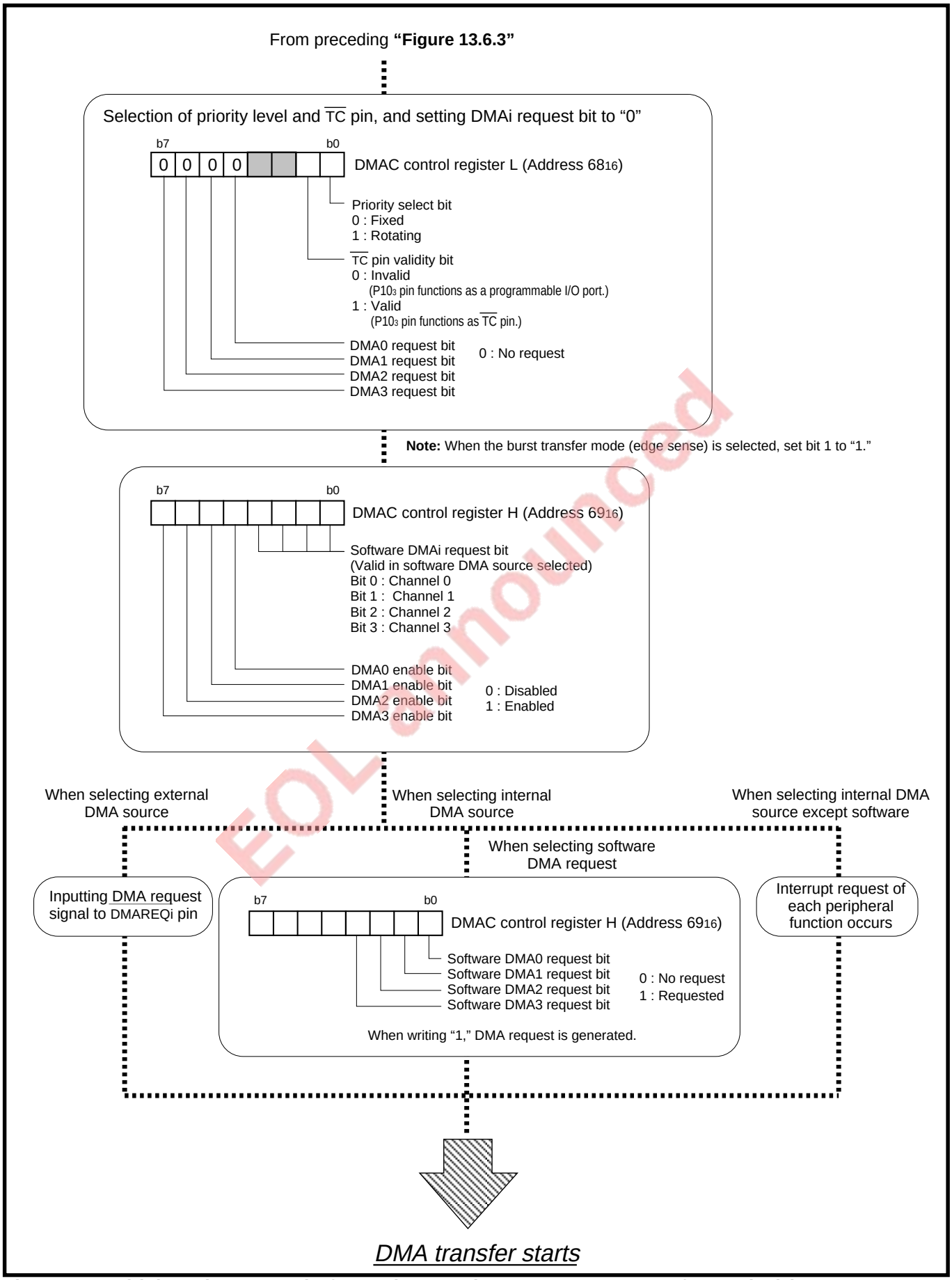
EOL announced

Fig. 13.6.2 Initial setting example for registers relevant to repeat transfer mode (1)



Not announced

Fig. 13.6.3 Initial setting example for registers relevant to repeat transfer mode (2)



EOL announced

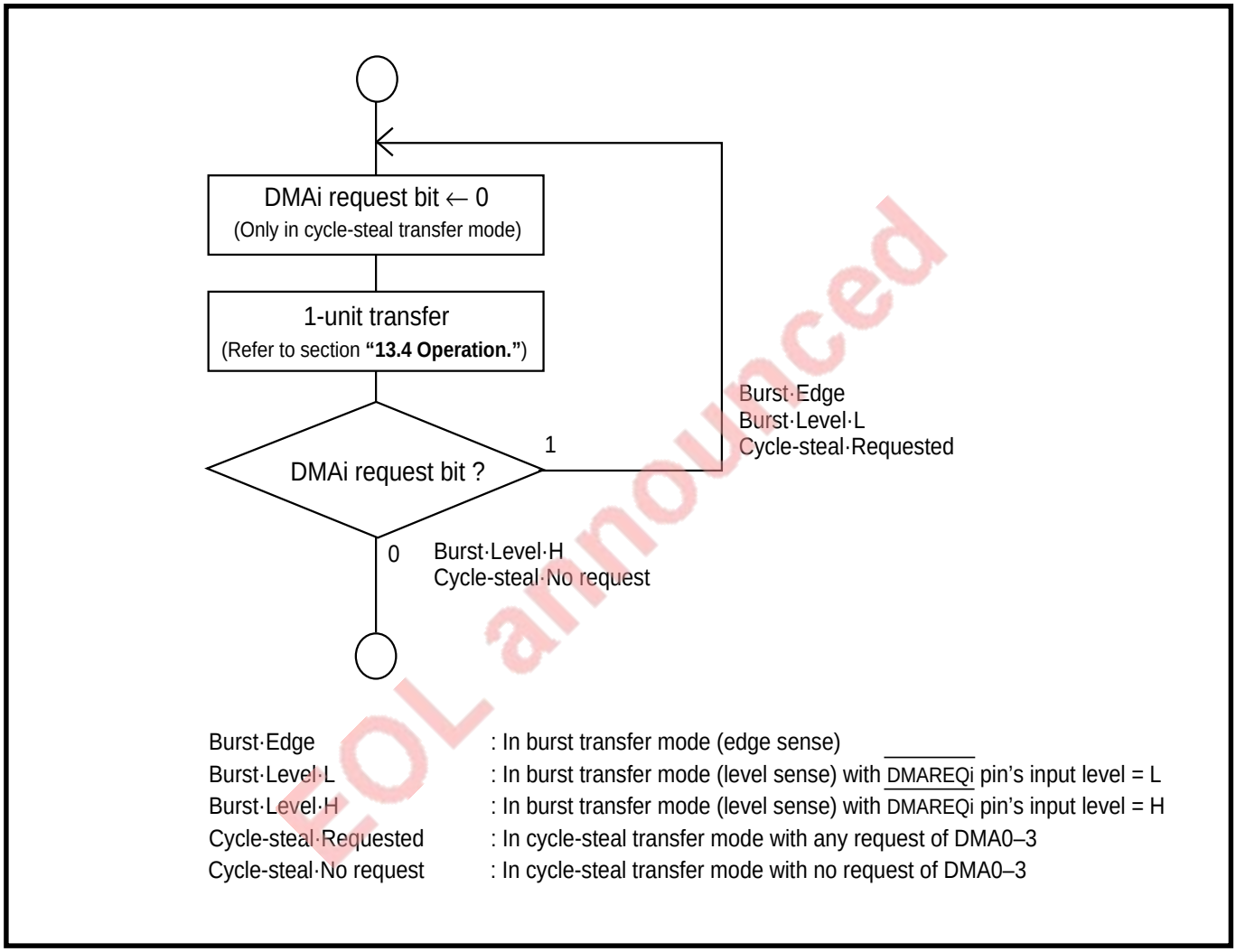
Fig. 13.6.4 Initial setting example for registers relevant to repeat transfer mode (3)

13.6.2 Operation in repeat transfer mode

Figure 13.6.5 shows the operation flowchart of the repeat transfer mode, and Figure 13.6.6 shows a timing diagram of the repeat transfer mode (burst transfer mode).
For the cycle-steal transfer mode, refer to the following:

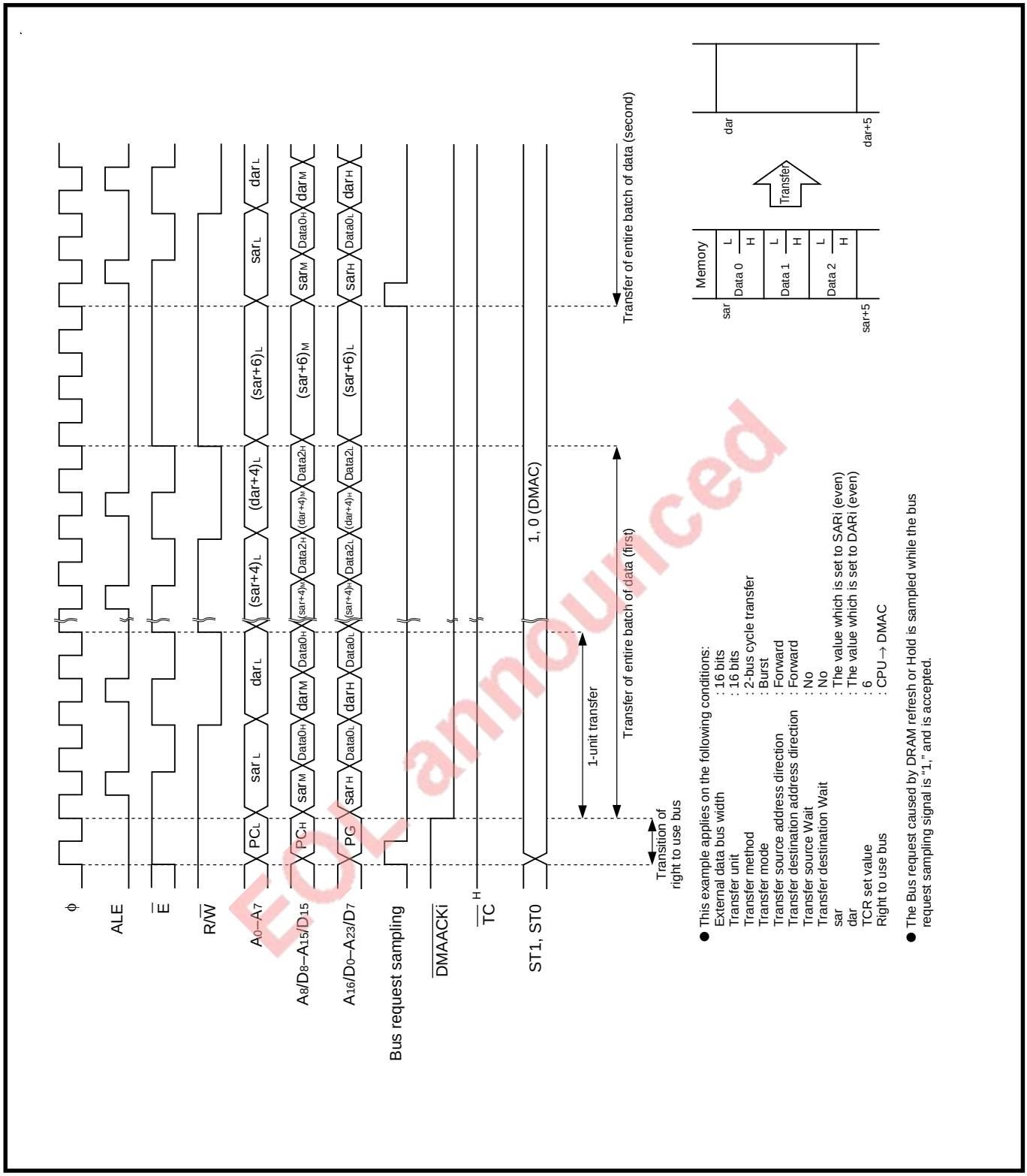
- All transfers except for the last 1-unit transfer: Figure 13.8.12
- Last 1-unit transfer: Figure 13.8.13

Also, refer to section "**13.2.1 Bus access control circuit"** for the bus request sampling during transfer.



EOL announced

Fig. 13.6.5 Operation flowchart of repeat transfer mode



EOL announced

Fig. 13.6.6 Timing diagram of repeat transfer mode (burst transfer mode)

13.7 Array chain transfer mode

This mode is used to transfer several blocks of data. According to the information of each block stored in memory area **(Note)**, several blocks of data are transferred. All of the transfer parameters must be located in series. Table 13.7.1 lists the specifications of the array chain transfer mode, and Figure 13.7.1 shows the register structures of SARi, DARi, and TCRi in this mode.

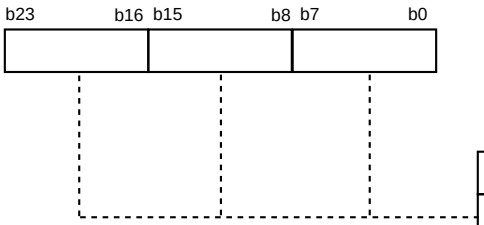
Note: Each of the following information is called "transfer parameter": transfer start addresses of transfer source and destination, and transfer data's byte number.

Table 13.7.1 Specifications of array chain transfer mode

Item	Performance specifications
Transfer parameter memory	Required. ● In 2-bus cycle transfer: 12 bytes per one block (transfer source's transfer start address, transfer destination's transfer start address, transfer data's byte number) ● In 1-bus cycle transfer: 8 bytes per one block (from memory to I/O: transfer source's transfer start address, transfer data's byte number) (from I/O to memory: transfer destination's transfer start address, transfer data's byte number)
Condition of normal termination	TCRi latch = 0 and TCRi = 0
Conditions of forced termination	● Falling edge of the $\overline{TC}$ pin's input from "H" to "L" ($\overline{TC}$ pin validity bit = "1") ● Write "0" to the DMAi enable bit
Interrupt request generation timing	At normal termination
Functions of registers	SARi latch: Indicates the transfer parameter memory's start address of the next block. SARi: Indicates the address of the next transfer source. DARi latch: Not used. DARi: Indicates the address of the next transfer destination. TCRi latch: Indicates the number of remaining transfer blocks. TCRi: Indicates the number of remaining transfer bytes.

$\overline{TC}$ pin validity bit: Bit 1 at address 68_{16}

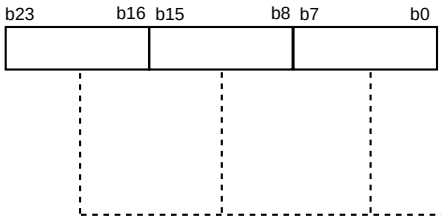
EOL announced



Source address register 0 (Addresses $1FC2_{16}$ to $1FC0_{16}$) (SAR0)
Source address register 1 (Addresses $1FD2_{16}$ to $1FD0_{16}$) (SAR1)
Source address register 2 (Addresses $1FE2_{16}$ to $1FE0_{16}$) (SAR2)
Source address register 3 (Addresses $1FF2_{16}$ to $1FF0_{16}$) (SAR3)

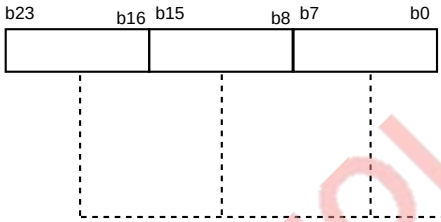
Bit	Functions	At reset	RW
23 to 0	[Write] Set the start address of transfer parameter memory. These bits can be set to "000000_{16}" to "$FFFFFF_{16}$." [Read] • After a value is written to this register and until transfer starts, the read value indicates the written value (the start address of the transfer parameter memory). • After transfer starts, the read value indicates the source address of data which is next transferred.	Undefined	RW

Note: When writing to this register, write to all 24 bits.



Destination address register 0 (Addresses $1FC6_{16}$ to $1FC4_{16}$) (DAR0)
Destination address register 1 (Addresses $1FD6_{16}$ to $1FD4_{16}$) (DAR1)
Destination address register 2 (Addresses $1FE6_{16}$ to $1FE4_{16}$) (DAR2)
Destination address register 3 (Addresses $1FF6_{16}$ to $1FF4_{16}$) (DAR3)

Bit	Functions	At reset	RW
23 to 0	Need not to be set. [Read] After transfer starts, the read value indicates the destination address of data which is next transferred.	Undefined	RW



Transfer counter register 0 (Addresses $1FCA_{16}$ to $1FC8_{16}$) (TCR0)
Transfer counter register 1 (Addresses $1FDA_{16}$ to $1FD8_{16}$) (TCR1)
Transfer counter register 2 (Addresses $1FEA_{16}$ to $1FE8_{16}$) (TCR2)
Transfer counter register 3 (Addresses $1FFA_{16}$ to $1FF8_{16}$) (TCR3)

Bit	Functions	At reset	RW
23 to 0	[Write] Set the number of transfer blocks. These bits can be set to "000001_{16}" to "$FFFFFF_{16}$." [Read] • After a value is written to this register and until transfer starts, the read value indicates the written value (the transfer block number) . • After transfer starts, the read value indicates the remaining byte number of the block which is being transferred.	Undefined	RW

Note: When writing to this register, write to all 24 bits.
Do not write "000000_{16}" to this register.

EOL announced

Fig. 13.7.1 Register structures of SARi, DARi, and TCRi in array chain transfer mode

13.7 Array chain transfer mode

13.7.1 Transfer parameter memory in array chain transfer mode

The transfer parameters required for each transfer method are described below. These parameters must be located in series starting at an even addresses.
Figure 13.7.2 shows a transfer parameter memory map in the array chain transfer mode.

(1) In 2-bus cycle transfer

All of the following transfer parameters are required for each block of data; that is, a transfer parameter memory consumes 12 bytes for each block.

- Transfer source's transfer start address (24 bits) + Dummy data (8 bits)
- Transfer destination's transfer start address (24 bits) + Dummy data (8 bits)
- Transfer data's byte number (24 bits) + Dummy data (8 bits)

(2) In 1-bus cycle transfer from memory to I/O

All of the following transfer parameters are required for each block of data; that is, a transfer parameter memory consumes 8 bytes for each block.

- Transfer source's transfer start address (24 bits) + Dummy data (8 bits)
- Transfer data's byte number (24 bits) + Dummy data (8 bits)

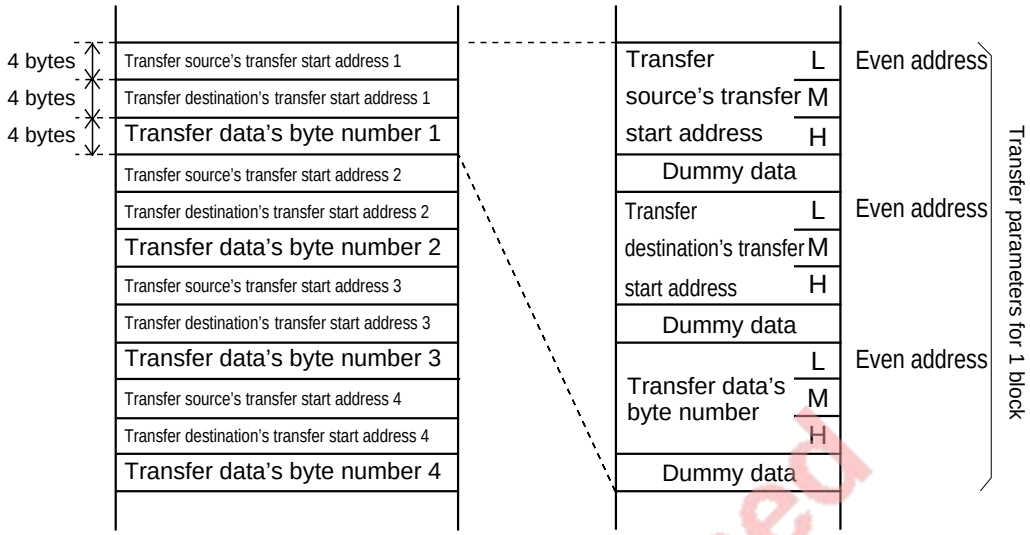
(3) In 1-bus cycle transfer from I/O to memory

All of the following transfer parameters are required for each block of data; that is, a transfer parameter memory consumes 8 bytes for each block.

- Transfer destination's transfer start address (24 bits) + Dummy data (8 bits)
- Transfer data's byte number (24 bits) + Dummy data (8 bits)

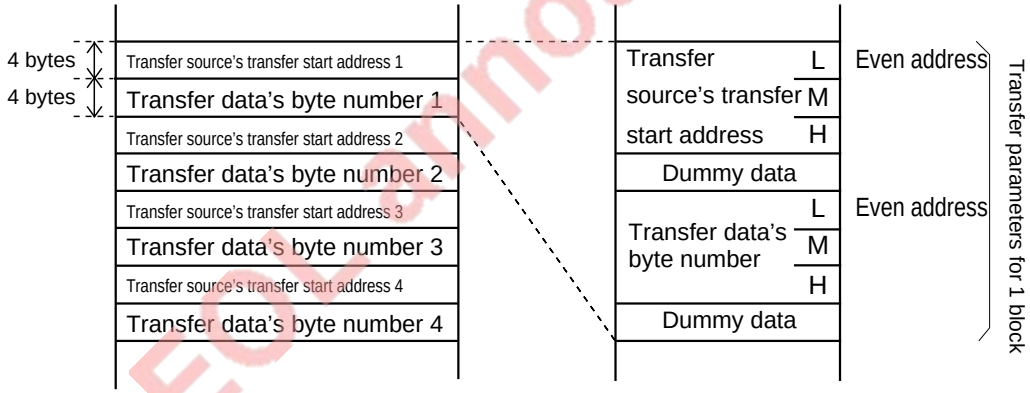
EOL announced

(1) 2-bus cycle transfer



✻ The above applies when 4-block transfer is performed.

(2) 1-bus cycle transfer



✻ The above applies on the following conditions:
- •When data is transferred from memory to I/O
  (When transferring from I/O to memory, replace all the above mentioned "Transfer source's transfer start address" with "Transfer destination's transfer start address.")
- •4-block transfer

EOL announced

Fig. 13.7.2 Transfer parameter memory map in array chain transfer mode

13.7.2 Setting of array chain transfer mode

Figures 13.7.3 through 13.7.5 show an initial setting example for registers relevant to the array chain transfer mode.

In addition, when timer A, timer B, UART, or the A-D converter is selected as a DMA request source, the setting for the peripheral is required. For details of the setting, refer to the chapter of each peripheral function.

When a DMAi interrupt is used, the setting for enabling the interrupt is also required. For details, refer to **"CHAPTER 7. INTERRUPTS."**

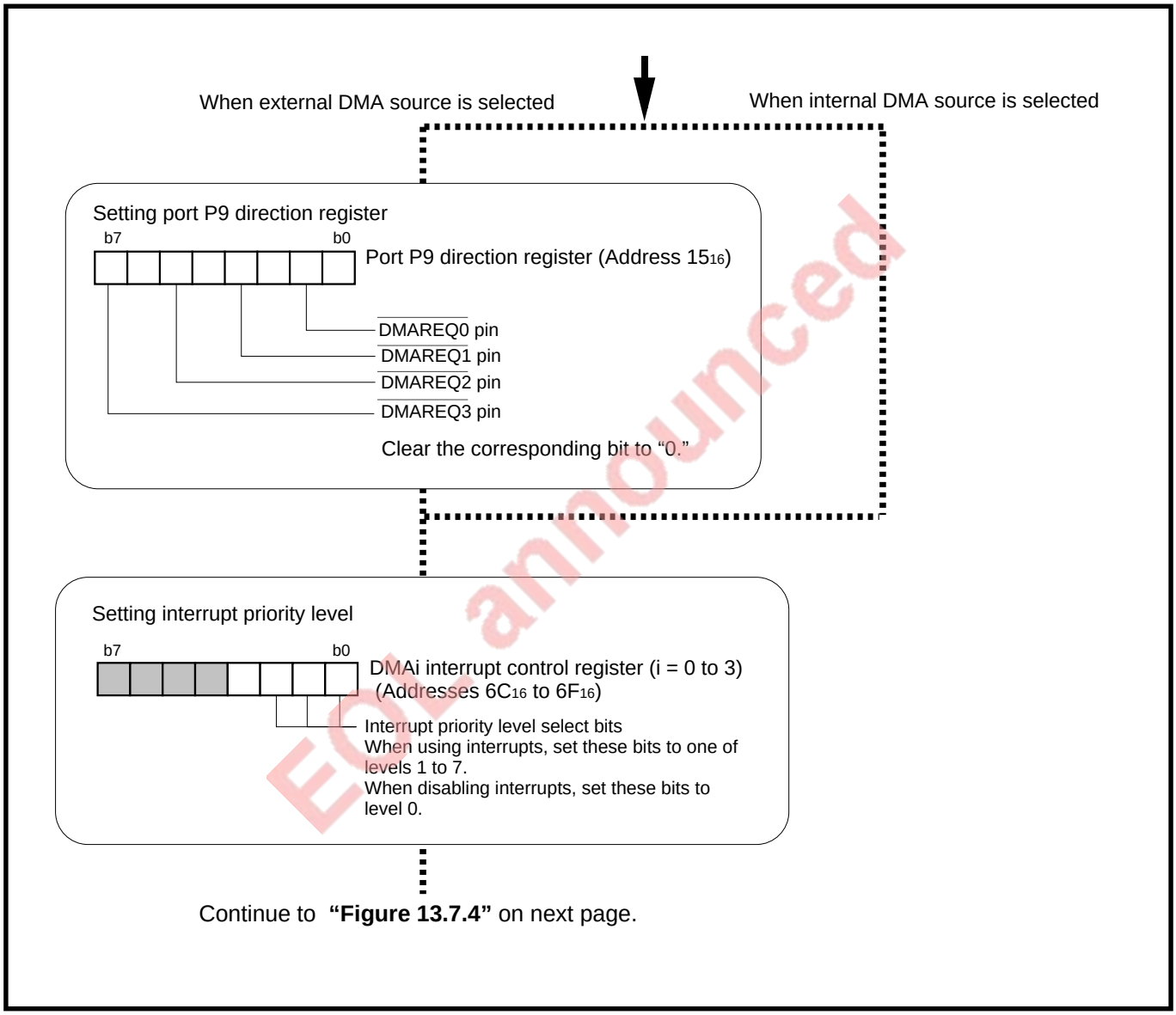
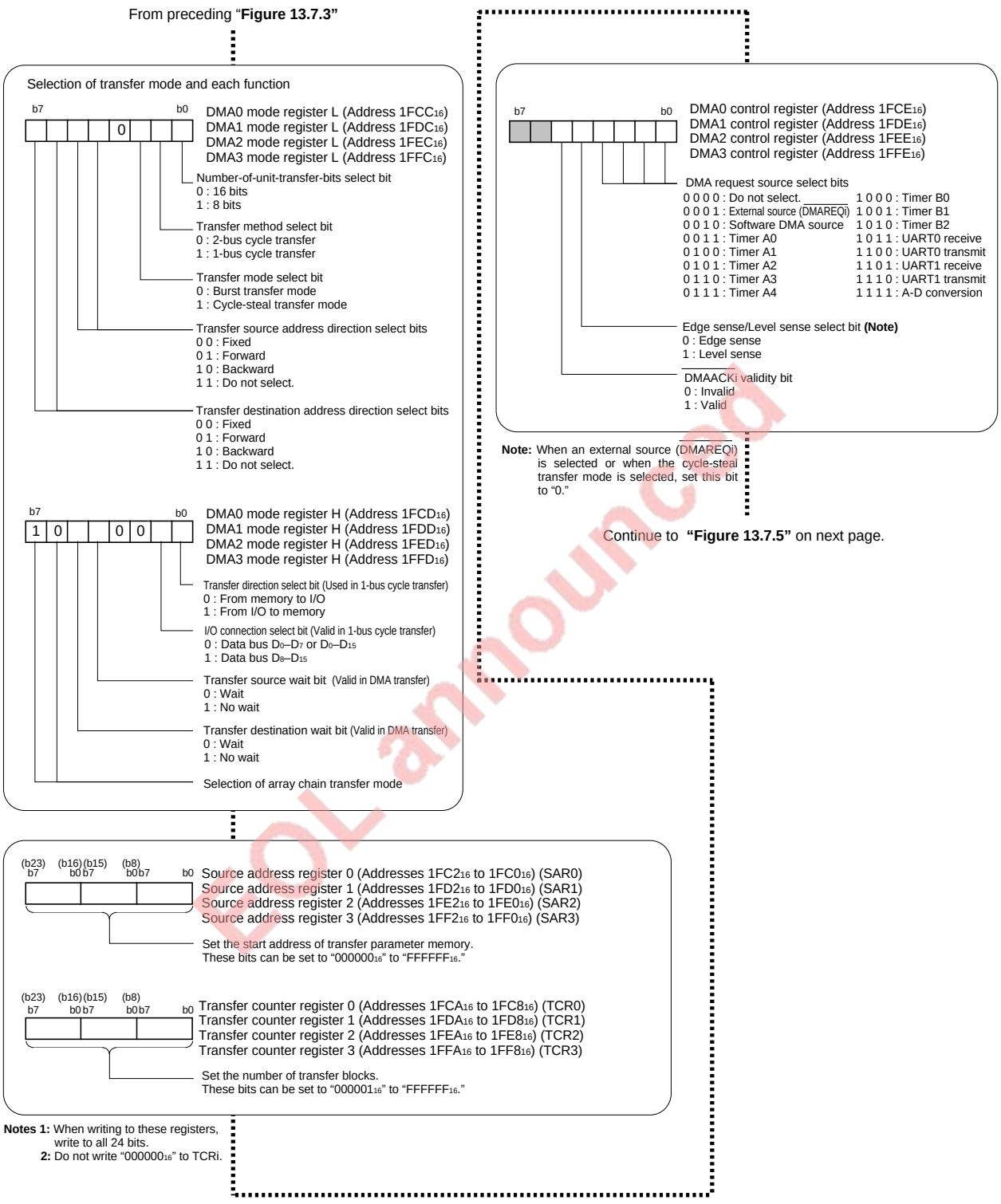


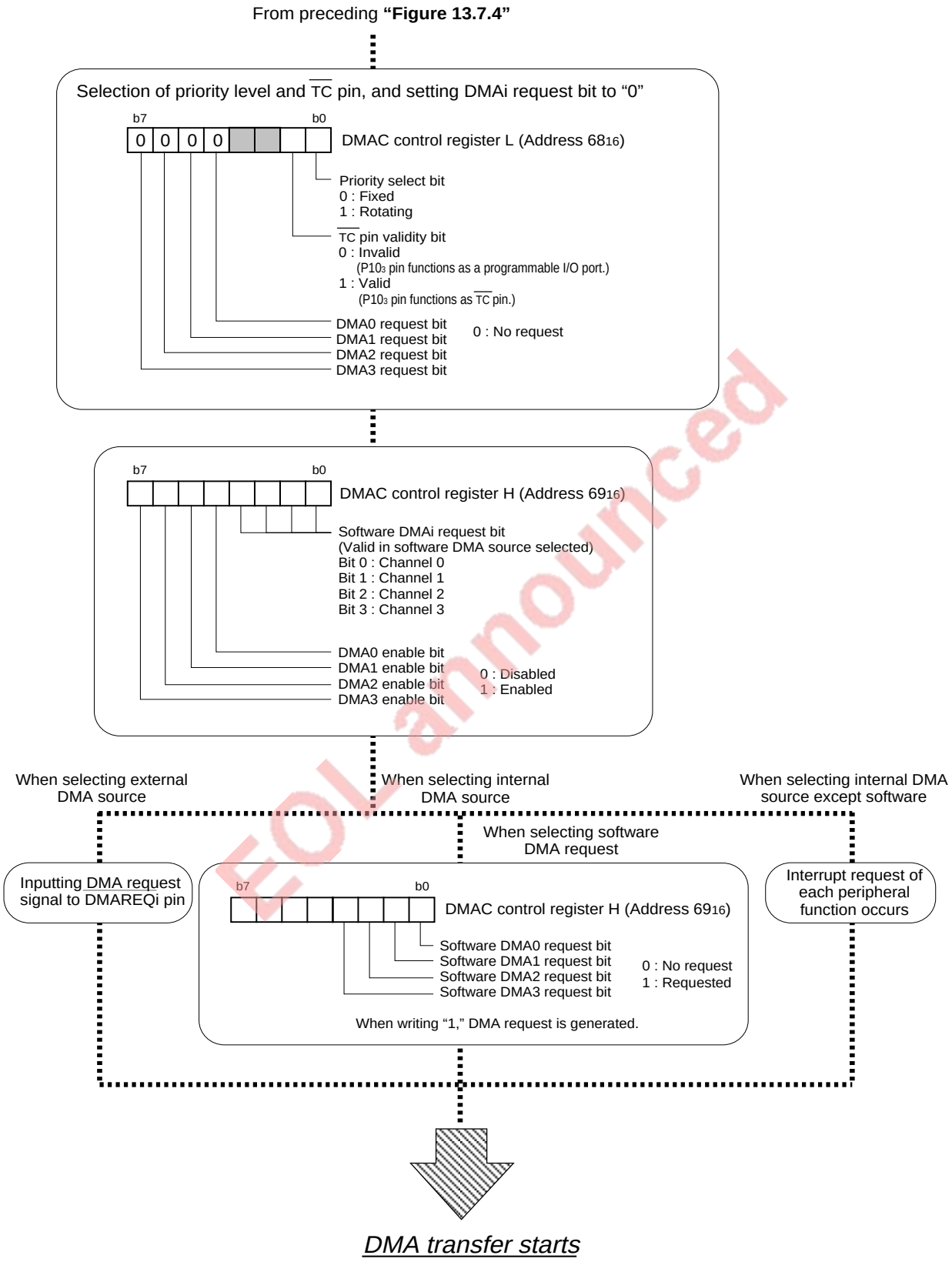
Fig. 13.7.3 Initial setting example for registers relevant to array chain transfer mode (1)

EOL announced



EOL announced

Fig. 13.7.4 Initial setting example for registers relevant to array chain transfer mode (2)



EOL announced

Fig. 13.7.5 Initial setting example for registers relevant to array chain transfer mode (3)

13.7.3 Operation in array chain transfer mode

Figure 13.7.6 shows the operation flowchart of the array chain transfer mode, and Figures 13.7.7 and 13.7.8 show timing diagrams of the array chain transfer mode (burst transfer mode).
For the cycle-steal transfer mode, refer to the following:

- Transfer of transfer parameters in an array state: Figures 13.8.10 and 13.8.11
- All transfers except in an array state and except the last 1-unit transfer of each block: Figure 13.8.12
- Last 1-unit transfer of each block except the last block: Figure 13.8.13
- Last 1-unit transfer of the last block: Figure 13.8.14

The processing performed in the array chain transfer mode consists of an array state and a transfer state.

(1) Array state

In an array state, transfer parameters are read from the transfer parameter memory in a unit of 2 bytes and transferred to registers SARi, DARi, and TCRi and their latches. As shown in Figure 13.7.2, a transfer parameter consists of 4 bytes (24 bits of data + 8 bits of dummy data).
One bus cycle always consumes 3 cycles of ϕ.
During an array state, the $\overline{\text{DMAACKi}}$ pin outputs “H” level.

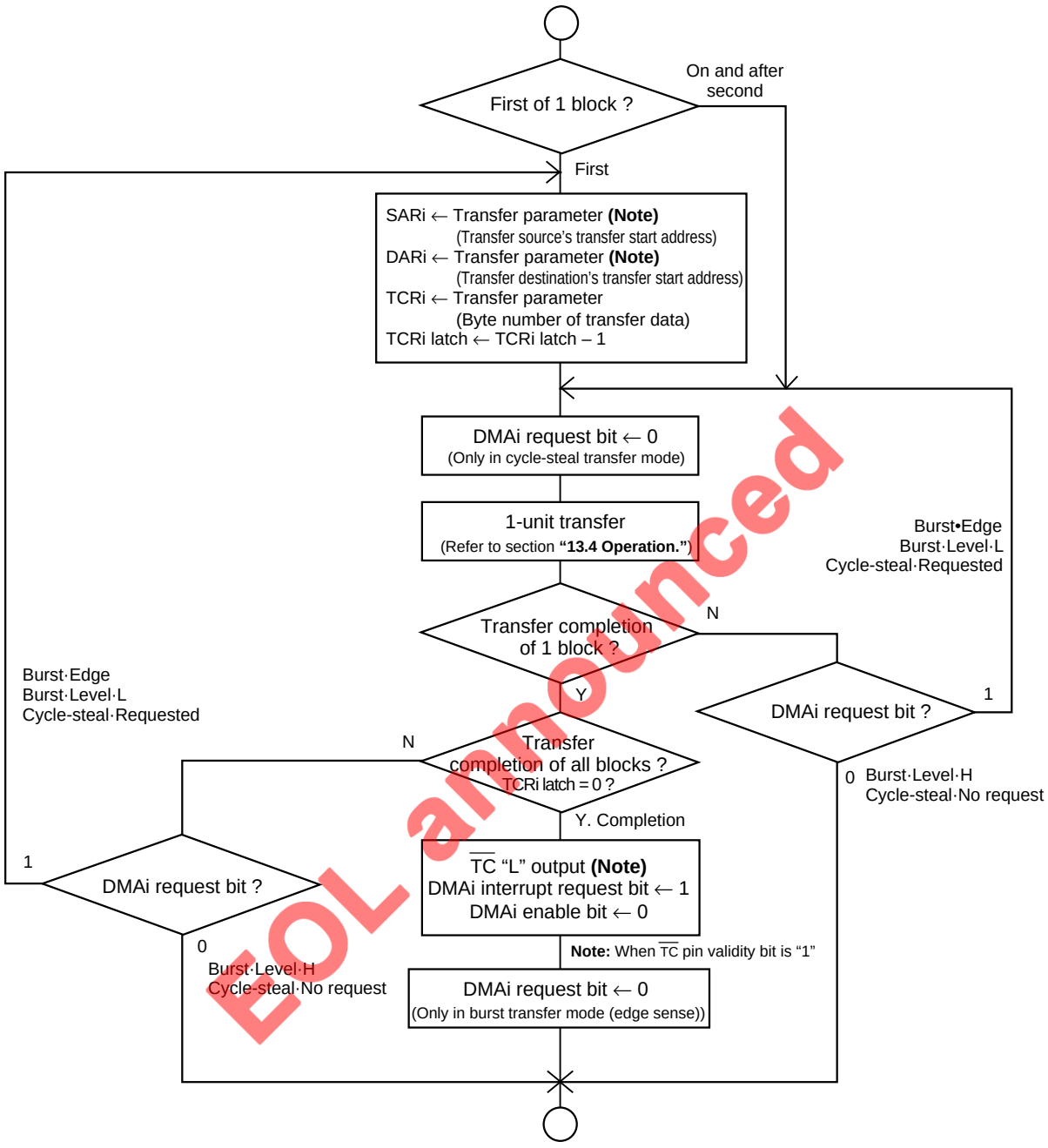
For the bus request sampling in an array state, refer to section **“13.2.1 Bus access control circuit.”**

(2) Transfer state

Data is transferred in a transfer state.

For the bus request sampling in a transfer state, refer to section **“13.2.1 Bus access control circuit.”**

EOL announced

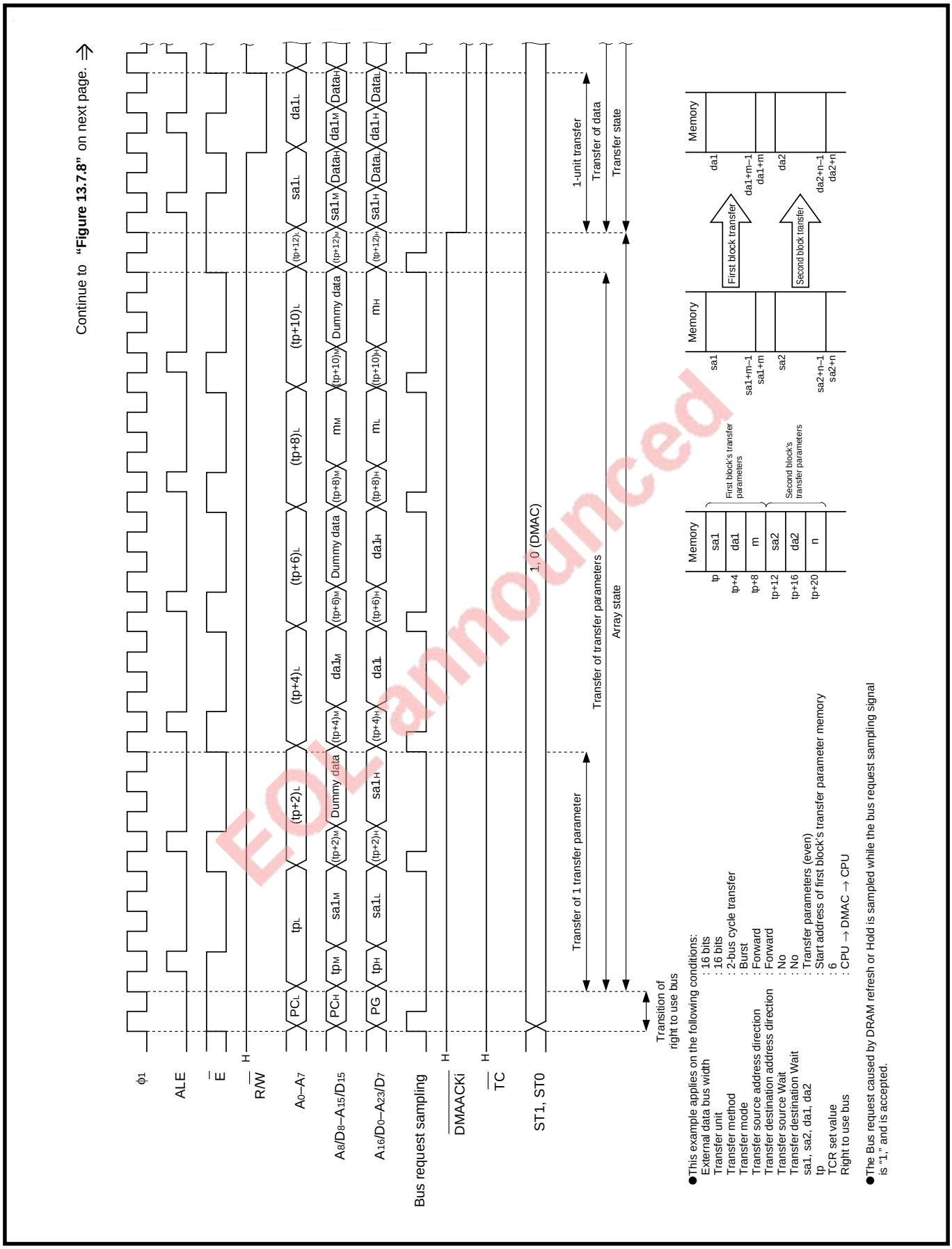


EOL announced

Burst·Edge : In burst transfer mode (edge sense)
Burst·Level·L : In burst transfer mode (level sense) with DMAREQi pin = L
Burst·Level·H : In burst transfer mode (level sense) with DMAREQi pin = H
Cycle-steal·Requested : In cycle-steal transfer mode with any request of DMA0–3
Cycle-steal·No request : In cycle-steal transfer mode with no request of DMA0–3
SARi latch indicates the start address of the transfer parameter memory of the next block.
TCRi latch indicates the number of remaining transfer blocks.

Note: The above figure applies when 2-bus cycle transfer is performed.
When data is transferred from memory to I/O in 1-bus cycle transfer, there is no "DARi ← Transfer parameter."
When data is transferred from I/O to memory in 1-bus cycle transfer, there is no "SARi ← Transfer parameter."

Fig. 13.7.6 Operation flowchart of array chain transfer mode



EOL announced

Fig. 13.7.7 Timing diagram of array chain transfer mode (burst transfer mode) (1)

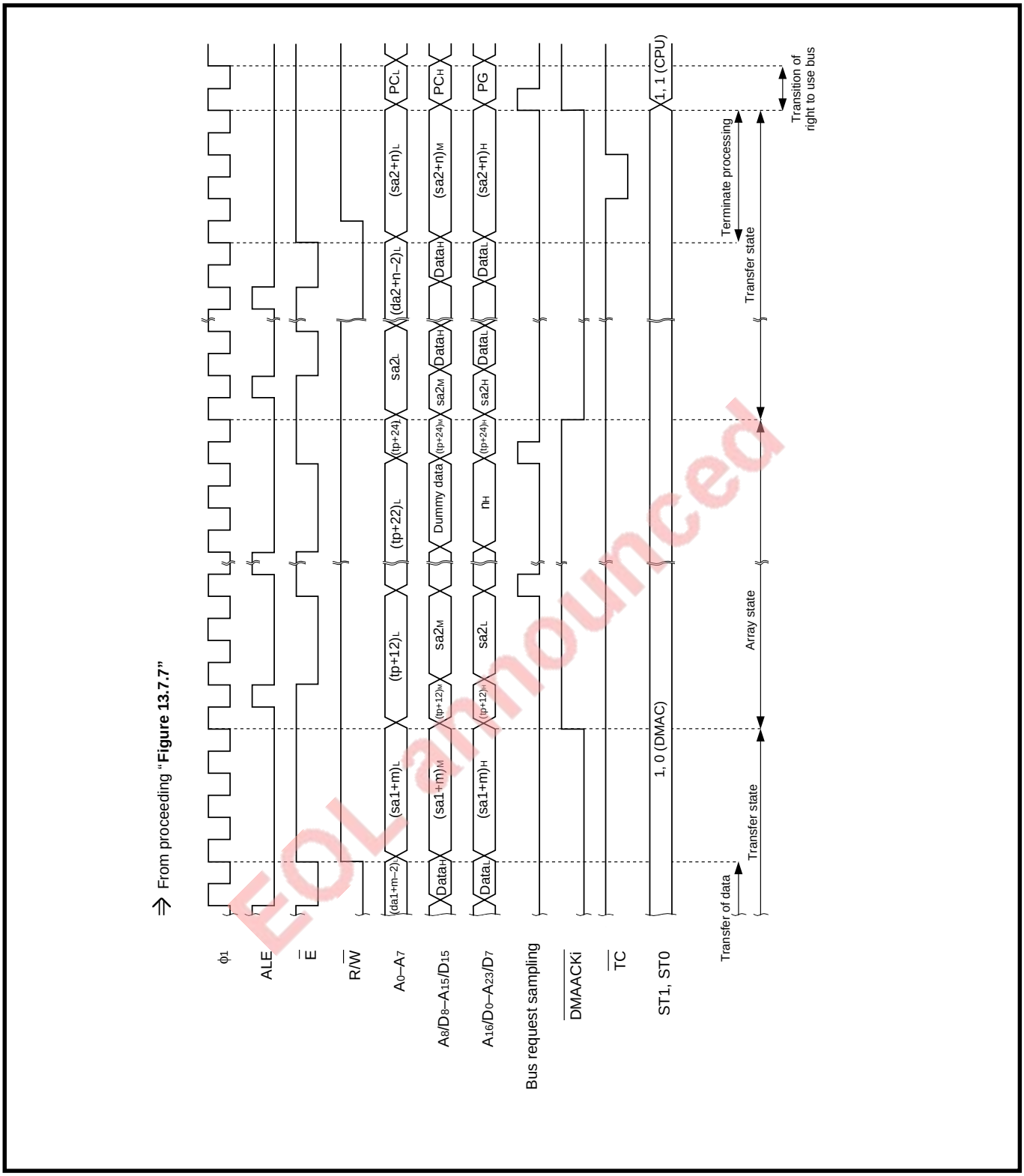


Fig. 13.7.8 Timing diagram of array chain transfer mode (burst transfer mode) (2)

EOL announced

[Precautions for array chain transfer mode]

If the following two conditions are satisfied when the transfer unit is 16 bits and the address direction of transfer source or destination is fixed, the array chain transfer mode can be used:

- The external data bus width = 16 bits or the internal memory is used.
- The transfer start address on the address-direction-fixed side is an even address.

EOL announced

13.8 Link array chain transfer mode

This mode is used to transfer several blocks of data. According to the information of each block stored in memory area **(Note)**, several blocks of data are transferred. Transfer parameters can be located in separate memory locations, in a unit of one block's parameters. Table 13.8.1 lists the specifications of the link array chain transfer mode, and Figure 13.8.1 shows the register structures of SARi, DARi, and TCRi in this mode.

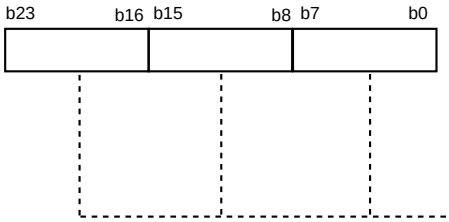
Note: Each of the following information is called "transfer parameter": transfer start addresses of transfer source and destination, and transfer data's byte number.

Table 13.8.1 Specifications of link array chain transfer mode

Item	Performance specifications
Transfer parameter memory	Required. ● In 2-bus cycle transfer: 16 bytes per one block (transfer source's transfer start address, transfer destination's transfer start address, transfer data's byte number, next transfer parameter memory's start address) ● In 1-bus cycle transfer: 12 bytes per one block (from memory to I/O: transfer source's transfer start address, transfer data's byte number, next transfer parameter memory's start address) (from I/O to memory: transfer destination's transfer start address, transfer data's byte number, next transfer parameter memory's start address)
Condition of normal termination	SARi latch = 0 and TCRi = 0
Conditions of forced termination	●Falling edge of $\overline{TC}$ pin's input from "H" to "L" (when the $\overline{TC}$ pin validity bit = "1") ●Write "0" to the DMAi enable bit
Interrupt request generation timing	At normal termination
Functions of registers	SARi latch: Indicates the transfer parameter memory's start address of the next block. SARi: Indicates the address of the next transfer source. DARi latch: Not used. DARi: Indicates the address of the next transfer destination. TCRi latch: Not used. TCRi: Indicates the number of remaining bytes being transferred.

$\overline{TC}$ pin validity bit: Bit 1 at address 68_{16}

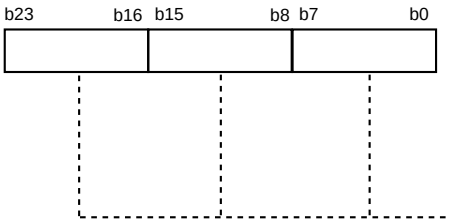
EOL announced



Source address register 0 (Addresses $1FC2_{16}$ to $1FC0_{16}$) (SAR0)
Source address register 1 (Addresses $1FD2_{16}$ to $1FD0_{16}$) (SAR1)
Source address register 2 (Addresses $1FE2_{16}$ to $1FE0_{16}$) (SAR2)
Source address register 3 (Addresses $1FF2_{16}$ to $1FF0_{16}$) (SAR3)

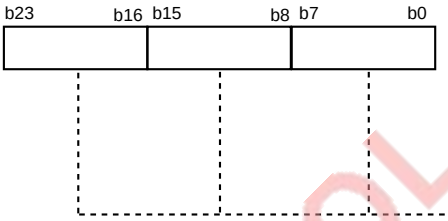
Bit	Functions	At reset	RW
23 to 0	[Write] Set the start address of transfer parameter memory of block which is first transferred. These bits can be set to "000000_{16}" to "$FFFFFF_{16}$." [Read] • After a value is written to this register and until transfer starts, the read value indicates the written value (the start address of the transfer parameter memory of block which is first transferred). • After transfer starts, the read value indicates the source address of data which is next transferred.	Undefined	RW

Note: When writing to this register, write to all 24 bits.



Destination address register 0 (Addresses $1FC6_{16}$ to $1FC4_{16}$) (DAR0)
Destination address register 1 (Addresses $1FD6_{16}$ to $1FD4_{16}$) (DAR1)
Destination address register 2 (Addresses $1FE6_{16}$ to $1FE4_{16}$) (DAR2)
Destination address register 3 (Addresses $1FF6_{16}$ to $1FF4_{16}$) (DAR3)

Bit	Functions	At reset	RW
23 to 0	Need not to be set. [Read] After transfer starts, the read value indicates the destination address of data which is next transferred.	Undefined	RW



Transfer counter register 0 (Addresses $1FCA_{16}$ to $1FC8_{16}$) (TCR0)
Transfer counter register 1 (Addresses $1FDA_{16}$ to $1FD8_{16}$) (TCR1)
Transfer counter register 2 (Addresses $1FEA_{16}$ to $1FE8_{16}$) (TCR2)
Transfer counter register 3 (Addresses $1FFA_{16}$ to $1FF8_{16}$) (TCR3)

Bit	Functions	At reset	RW
23 to 0	[Write] Set the dummy data. These bits can be set to "000001_{16}" to "$FFFFFF_{16}$." [Read] • After a value is written to this register and until transfer starts, the read value indicates the written value (dummy data). • After transfer starts, the read value indicates the remaining byte number of the block which is being transferred.	Undefined	RW

Note: When writing to this register, write to all 24 bits.
Do not write "000000_{16}" to this register.

EOL announced

Fig. 13.8.1 Register structures of SARi, DARi, and TCRi in link array chain transfer mode

13.8 Link array chain transfer mode

13.8.1 Transfer parameter memory in link array chain transfer mode

The transfer parameters required for each transfer method are described below. These parameters can be located in separate memory locations, in a unit of one block's parameters. However, these parameters must be located starting at an even address.

Figure 13.8.2 shows a transfer parameter memory map in the link array chain transfer mode.

(1) In 2-bus cycle transfer

All of the following transfer parameters are required for each block of data; that is, a transfer parameter memory consumes 16 bytes for each block.

- Transfer source's transfer start address (24 bits) + Dummy data (8 bits)
- Transfer destination's transfer start address (24 bits) + Dummy data (8 bits)
- Transfer data's byte number (24 bits) + Dummy data (8 bits)
- Start address of next transfer parameter memory (24 bits) **(Note)** + Dummy data (8 bits)

(2) In 1-bus cycle transfer from memory to I/O

All of the following transfer parameters are required for each block of data; that is, a transfer parameter memory consumes 12 bytes for each block.

- Transfer source's transfer start address (24 bits) + Dummy data (8 bits)
- Transfer data's byte number (24 bits) + Dummy data (8 bits)
- Start address of next transfer parameter memory (24 bits) **(Note)** + Dummy data (8 bits)

(3) In 1-bus cycle transfer from I/O to memory

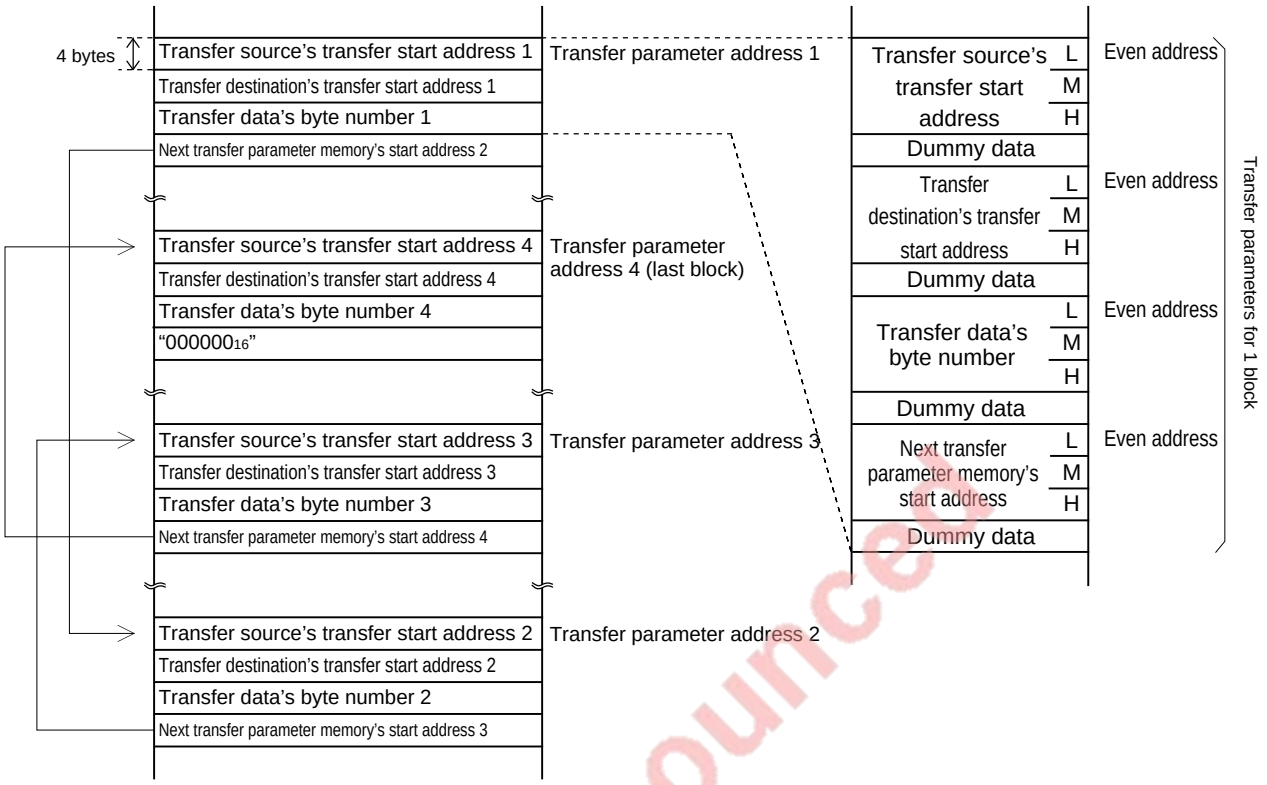
All of the following transfer parameters are required for each block of data; that is, a transfer parameter memory consumes 12 bytes for each block.

- Transfer destination's transfer start address (24 bits) + Dummy data (8 bits)
- Transfer data's byte number (24 bits) + Dummy data (8 bits)
- Start address of next transfer parameter memory (24 bits) **(Note)** + Dummy data (8 bits)

Note: For the last block of data, write "000000_{16}" as the start address of the next transfer parameter memory.

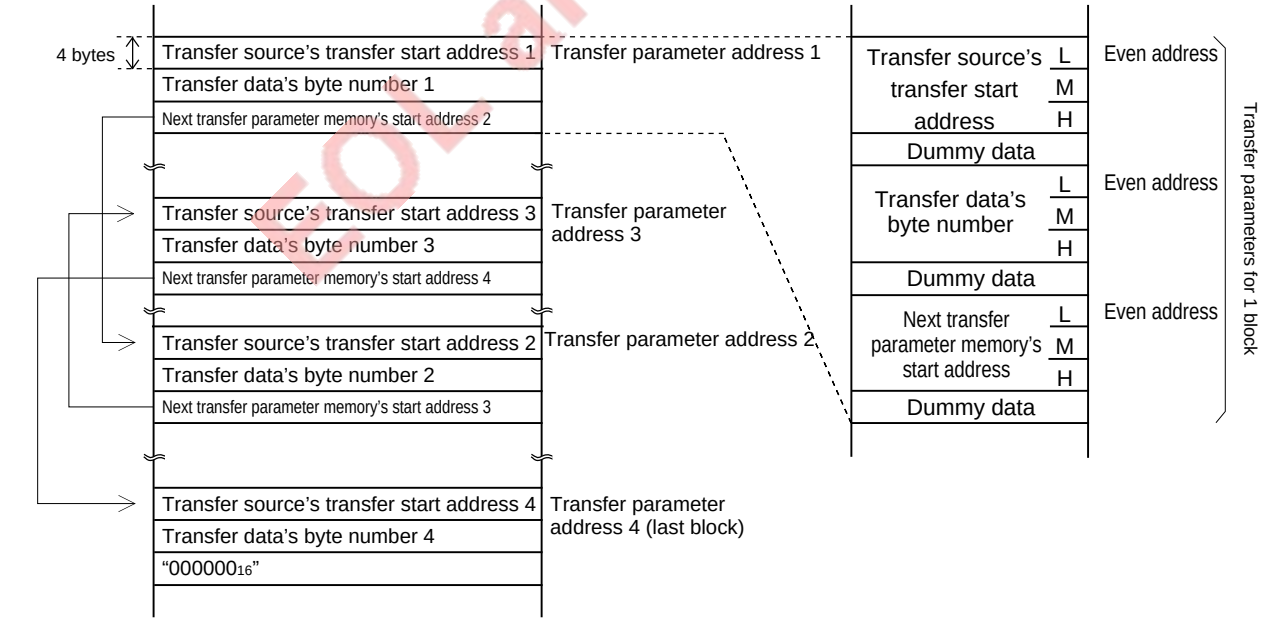
EOL announced

(1) 2-bus cycle transfer



✻ The above figure applies when 4-block transfer is performed.

(2) 1-bus cycle transfer



✻ The above applies on the following conditions:
- •When data is transferred from memory to I/O
  (When transferring from I/O to memory, replace all the above mentioned "Transfer source's transfer start address" with "Transfer destination's transfer start address."
- •4-block transfer

Fig. 13.8.2 Transfer parameter memory map in link array chain transfer mode

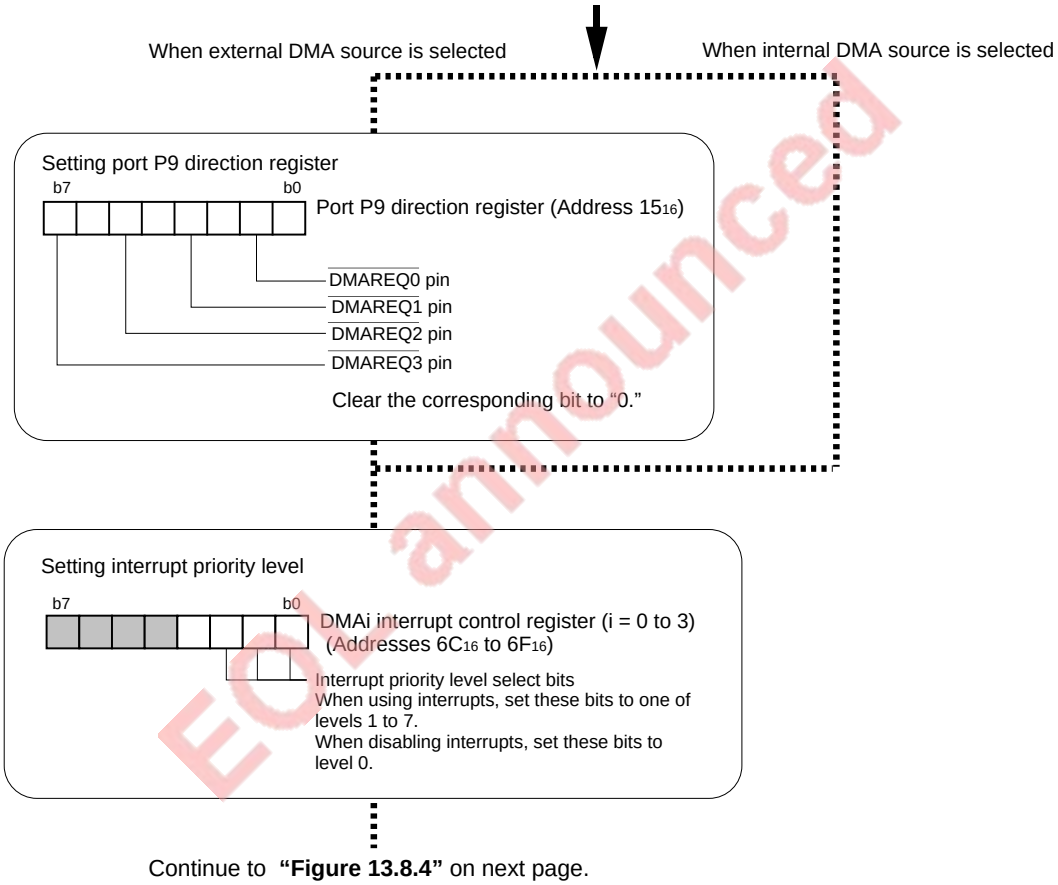
For announcement

13.8.2 Setting of link array chain transfer mode

Figures 13.8.3 through 13.8.5 show an initial setting example for registers relevant to the link array chain transfer mode.

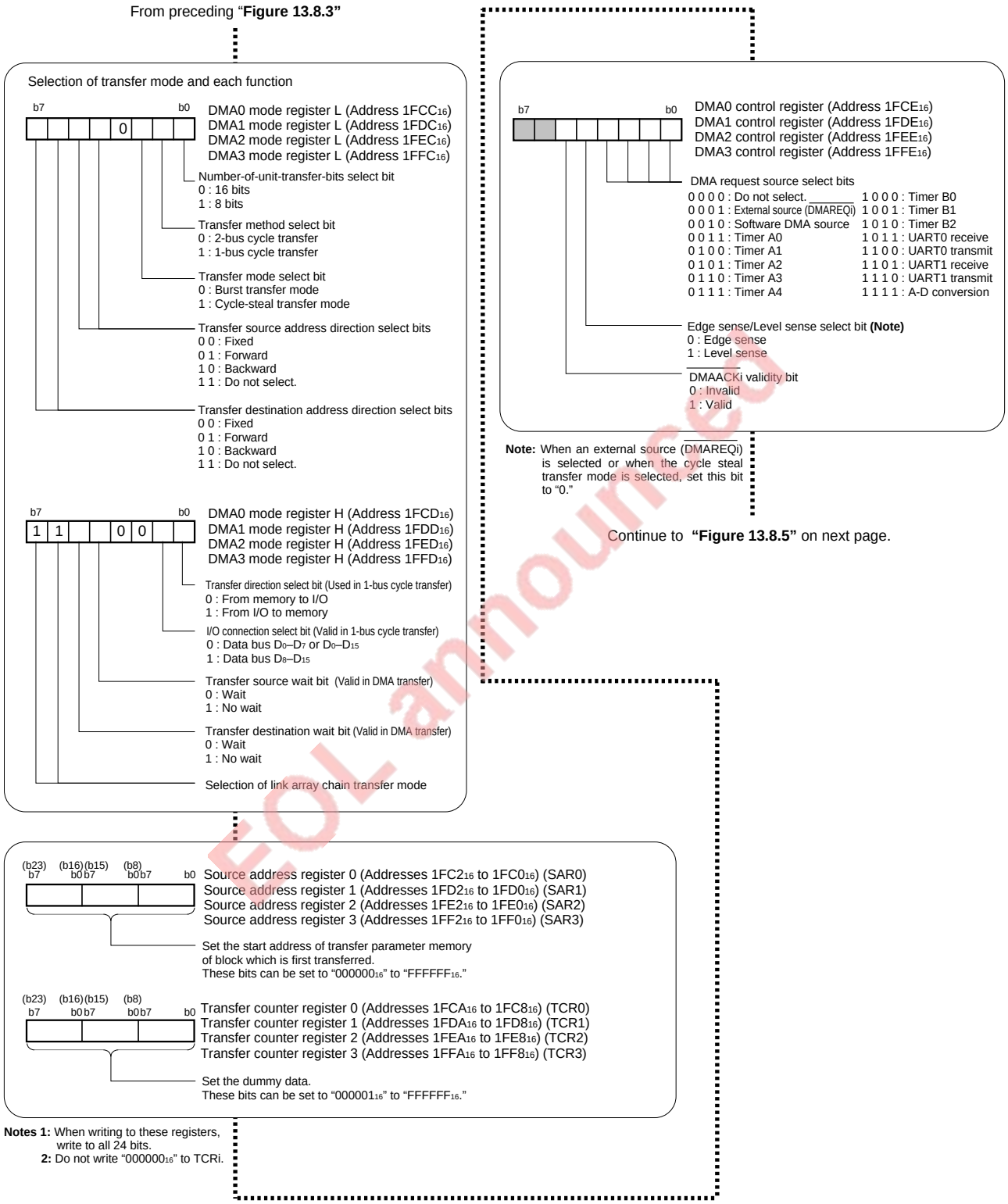
In addition, when timer A, timer B, UART, or the A-D converter is selected as a DMA request source, the setting for the peripheral is required. For details of the setting, refer to the chapter of each peripheral function.

When a DMAi interrupt is used, the setting for enabling the interrupt is also required. For details, refer to **"CHAPTER 7. INTERRUPTS."**



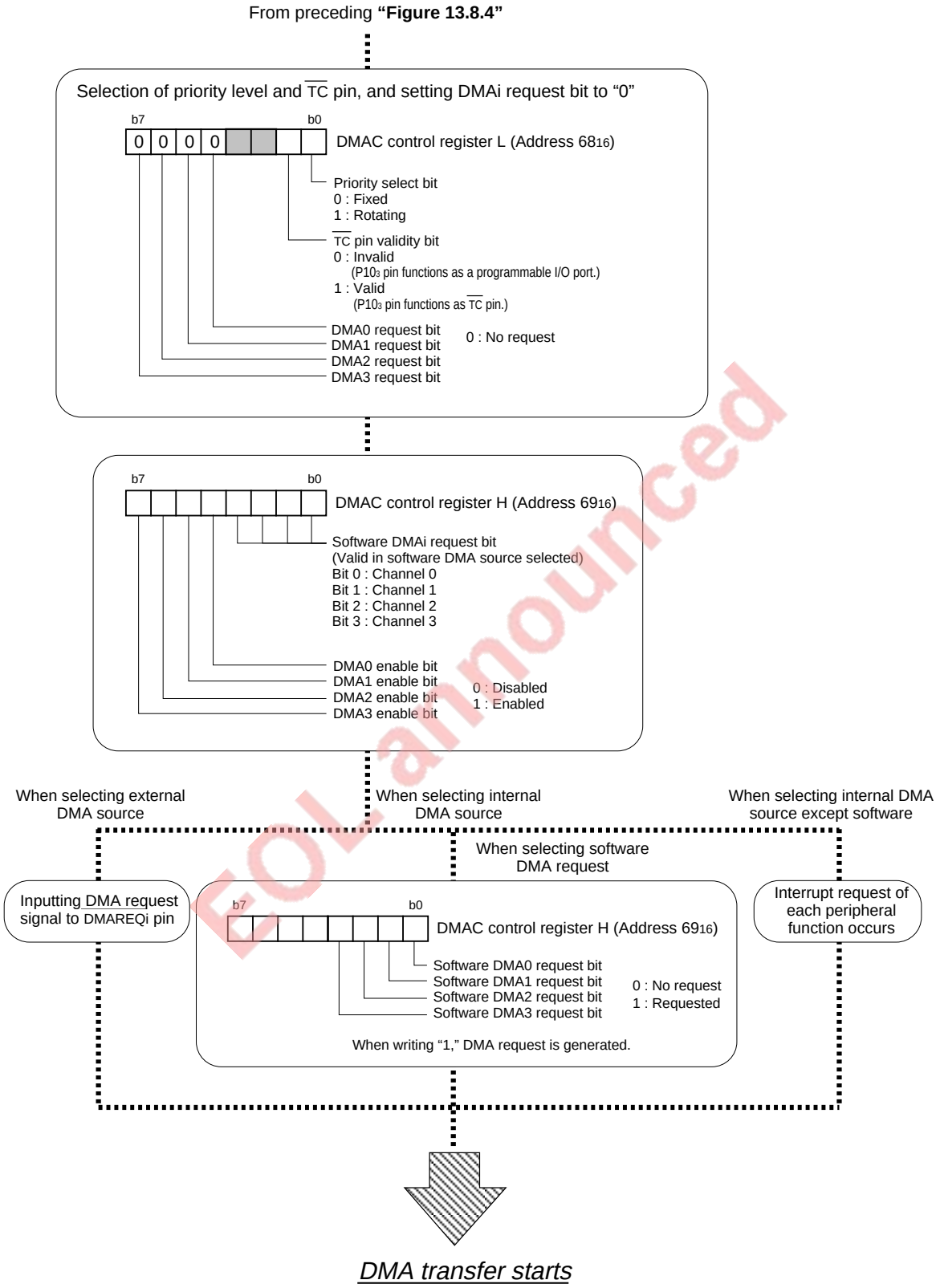
EOL announced

Fig. 13.8.3 Initial setting example for registers relevant to link array chain transfer mode (1)



EOL announced

Fig. 13.8.4 Initial setting example for registers relevant to link array chain transfer mode (2)



EOL announced

Fig. 13.8.5 Initial setting example for registers relevant to link array chain transfer mode (3)

13.8.3 Operation in link array chain transfer mode

Figure 13.8.6 shows the operation flowchart of the link array chain transfer mode, and Figures 13.8.7 and 13.8.8 show timing diagrams of the link array chain transfer mode (burst transfer mode). In addition, Figure 13.8.9 shows the conditions necessary for timings shown in Figures 13.8.7, 13.8.8, and 13.8.10 through 13.8.14.
For the cycle-steal transfer mode, refer to the following:

- Transfer of transfer parameters in an array state: Figures 13.8.10 and 13.8.11
- All transfers except for that in an array state and except for the last 1-unit transfer of each block: Figure 13.8.12
- Last 1-unit transfer of each block except for the last block: Figure 13.8.13
- Last 1-unit transfer of the last block: Figure 13.8.14

The processing performed in the link array chain transfer mode consists of an array state and a transfer state.

(1) Array state

In an array state, transfer parameters are read from the transfer parameter memory in a unit of 2 bytes and transferred to registers SARi, DARi, and TCRi and their latches. As shown in Figure 13.8.2, a transfer parameter consists of 4 bytes (24 bits of data + 8 bits of dummy data).
One bus cycle always consumes 3 cycles of ϕ.
During an array state, the $\overline{\text{DMAACKi}}$ pin outputs “H” level.

For the bus request sampling in an array state, refer to section **“13.2.1 Bus access control circuit.”**

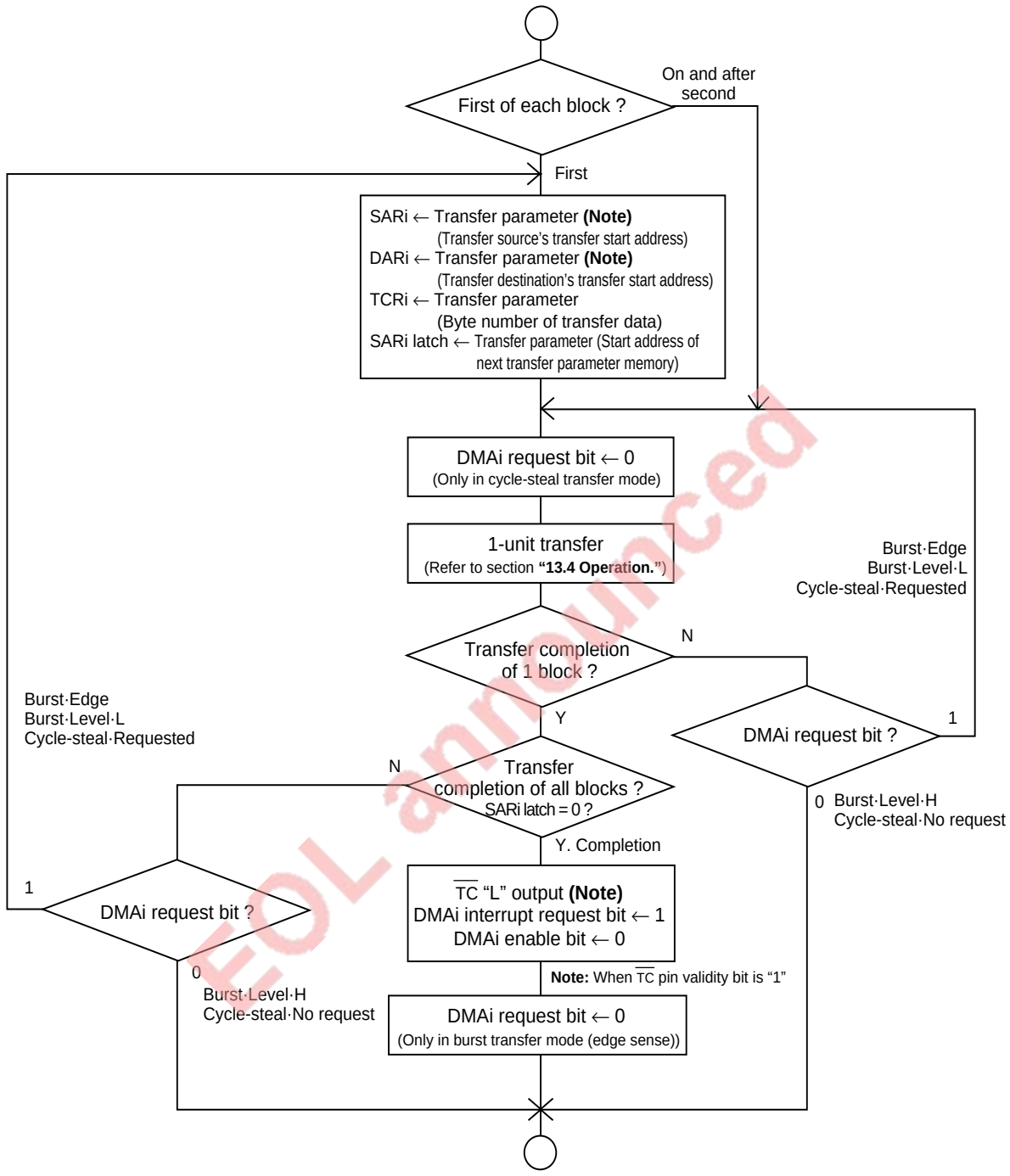
(2) Transfer state

Data is transferred in a transfer state.

For the bus request sampling in a transfer state, refer to section **“13.2.1 Bus access control circuit.”**

EOL announced

13.8 Link array chain transfer mode

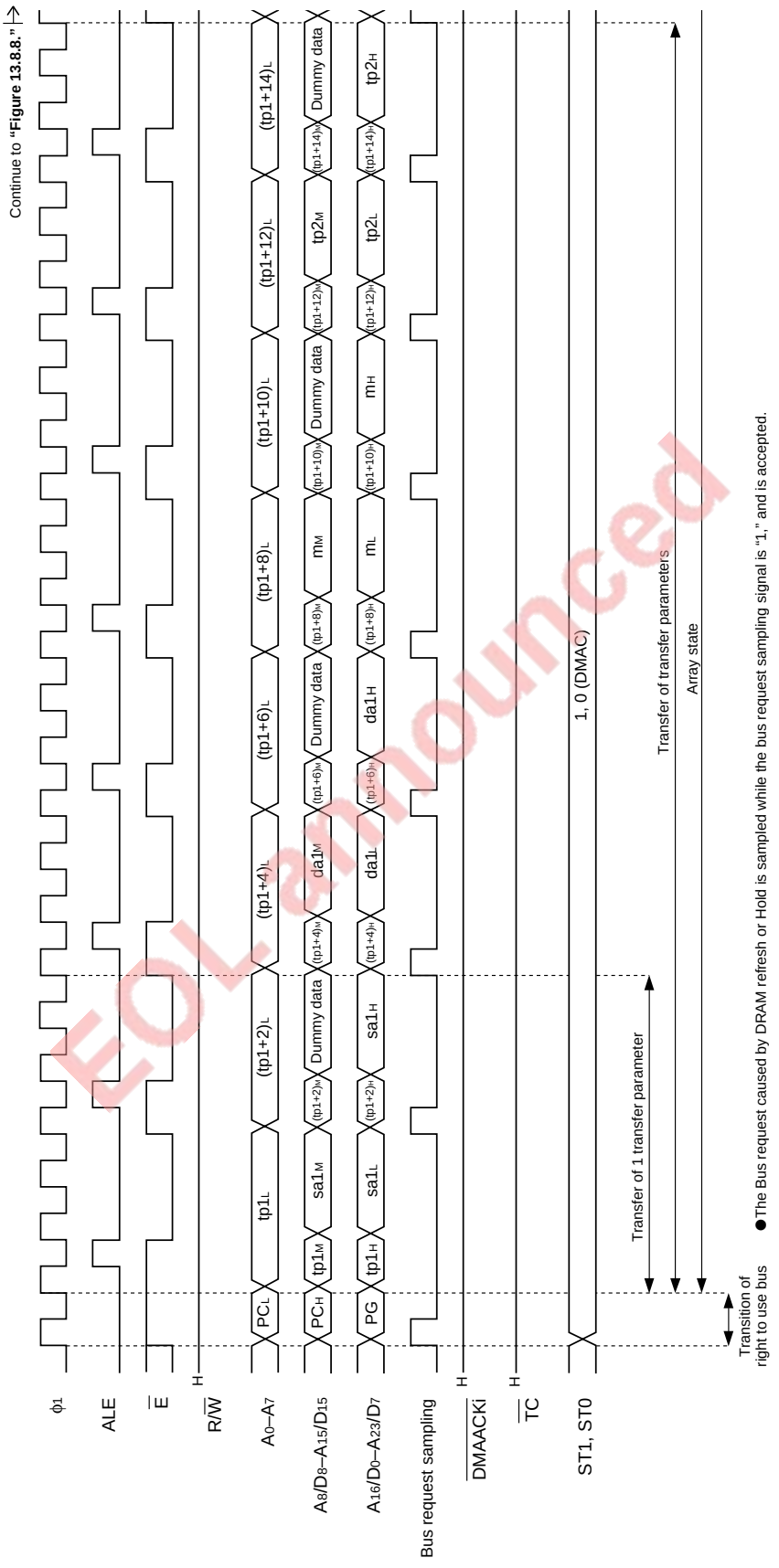


EOL announced

Burst·Edge : In burst transfer mode (edge sense)
Burst·Level·L : In burst transfer mode (level sense) with $\overline{\text{DMAREQi}}$ pin's input level = L
Burst·Level·H : In burst transfer mode (level sense) with $\overline{\text{DMAREQi}}$ pin's input level = H
Cycle-steal·Requested : In cycle-steal transfer mode with any request of DMA0–3
Cycle-steal·No request : In cycle-steal transfer mode with no request of DMA0–3
SARi latch indicates the start address of the transfer parameter memory of the next block.

Note: The above figure applies when 2-bus cycle transfer is performed.
When data is transferred from memory to I/O in 1-bus cycle transfer, there is no "DARi← Transfer parameter."
When data is transferred from I/O to memory in 1-bus cycle transfer, there is no "SARi← Transfer parameter."

Fig. 13.8.6 Operation flowchart of link array chain transfer mode



EOL announced

Fig. 13.8.7 Timing diagram of link array chain transfer mode (burst transfer mode) (1)

13.8 Link array chain transfer mode

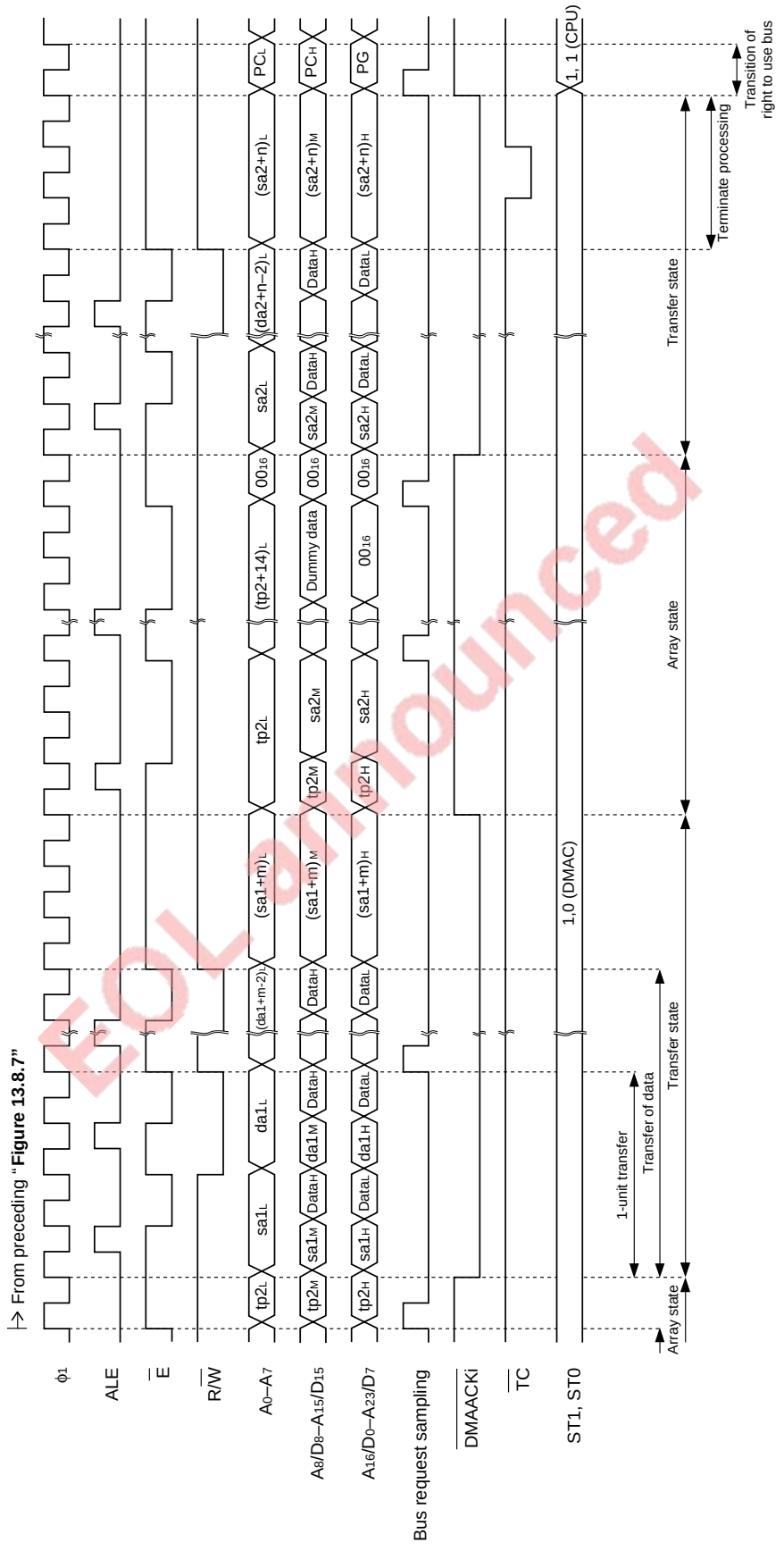


Fig. 13.8.8 Timing diagram of link array chain transfer mode (burst transfer mode) (2)

Figure 13.8.9 shows the conditions necessary for timings shown in Figures 13.8.7, 13.8.8, and 13.8.10 through 13.8.14.

External data bus width : 16 bits
Transfer unit : 16 bits
Transfer method : 2-bus cycle transfer
Transfer mode : Burst (**"Figure 13.8.7"** and **"Figure 13.8.8"**)
: Cycle-steal (**"Figure 13.8.10"** through **"Figure 13.8.14"**)
Transfer source address direction : Forward
Transfer destination address direction : Forward
Transfer source Wait : No
Transfer destination Wait : No
sa1, sa2, da1, da2 : Transfer parameter (even)
tp1 : Start address of first block's transfer parameter memory
Transfer block's number : 2
Right to use bus : CPU → DMAC → CPU

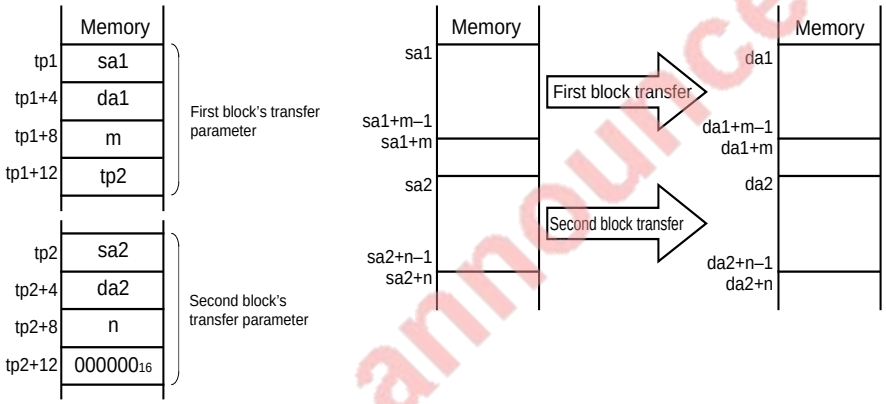
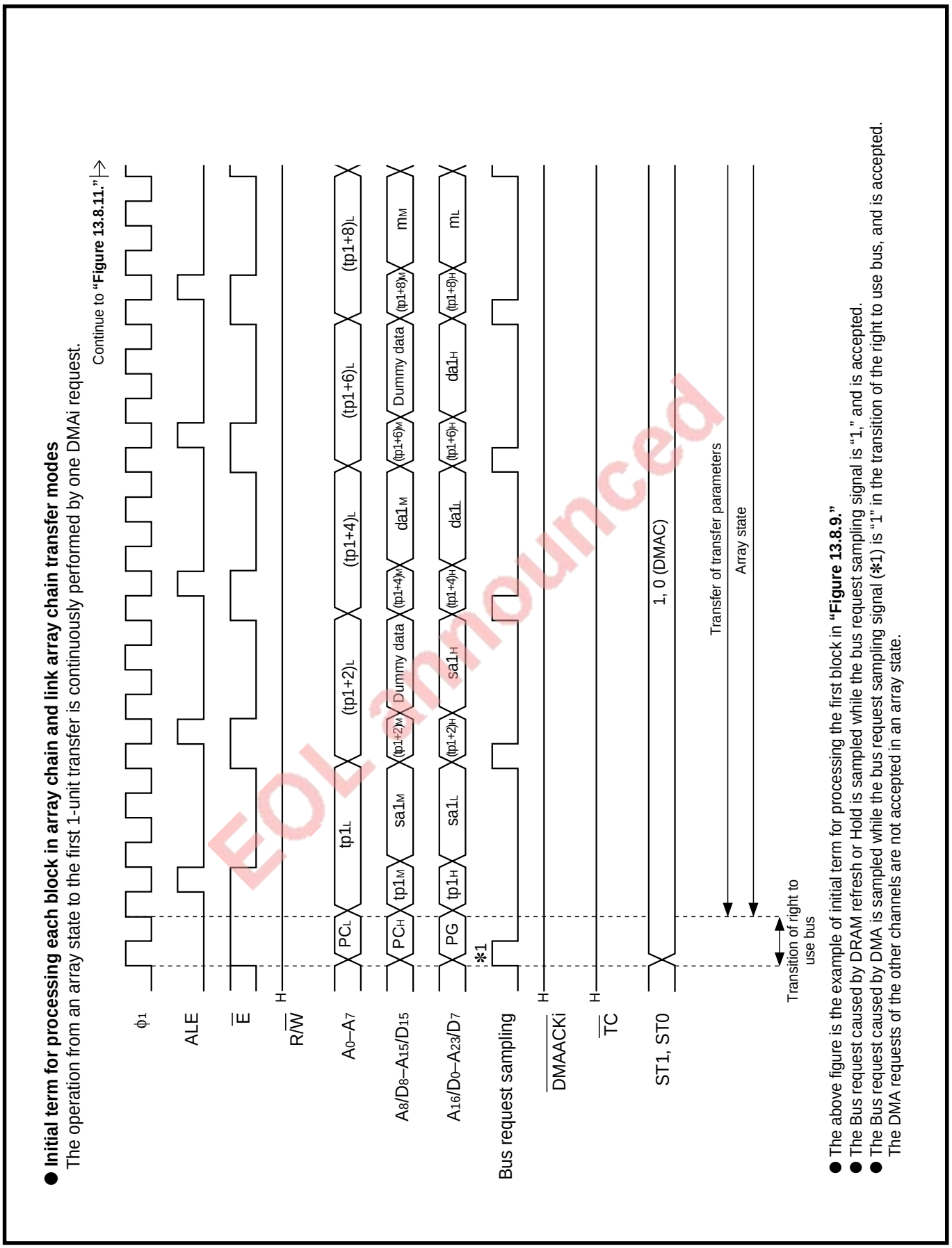


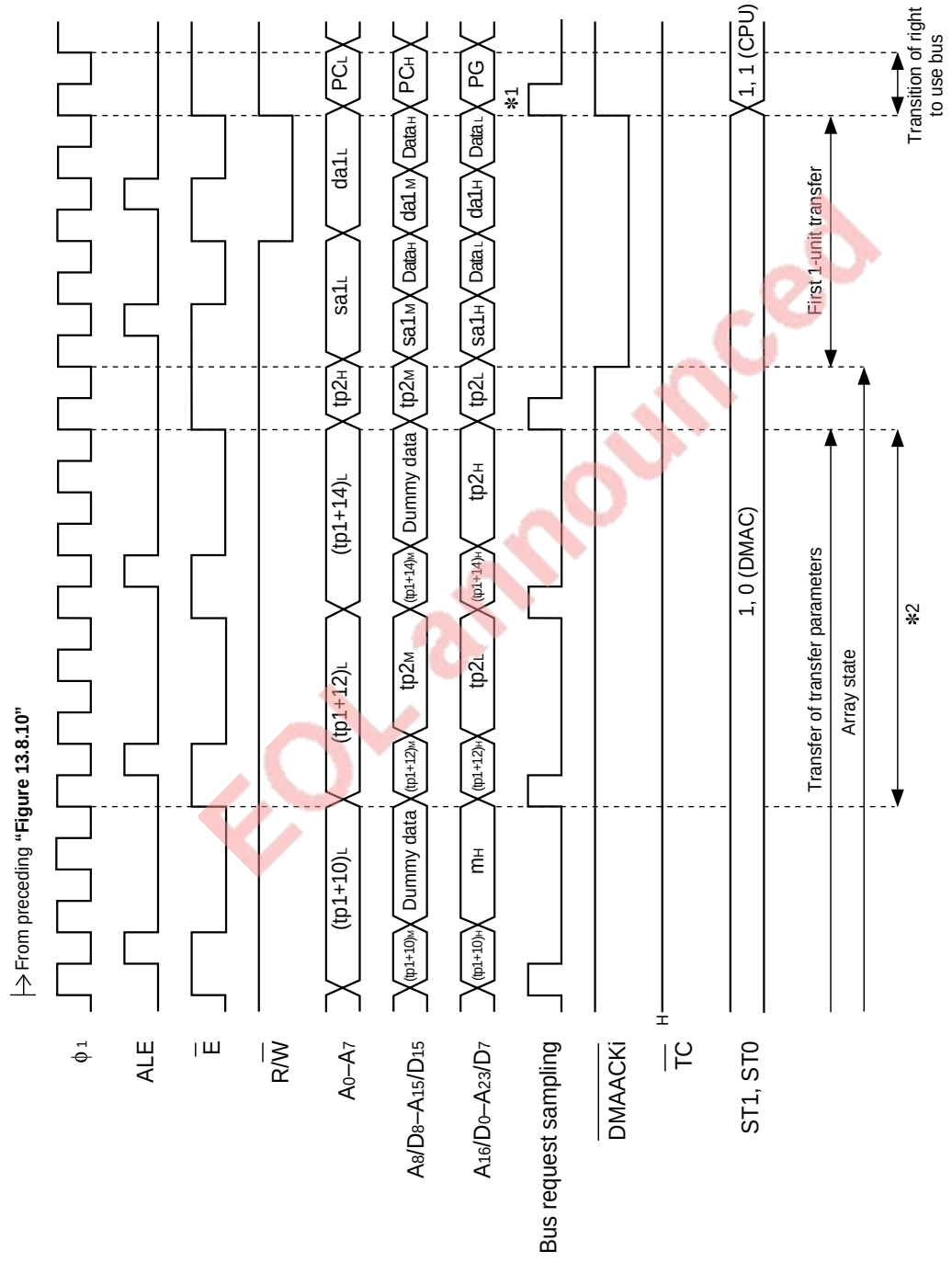
Fig. 13.8.9 Conditions necessary for timings shown in Figures 13.8.7, 13.8.8, and 13.8.10 through 13.8.14

EOL announced



EOL announced

Fig. 13.8.10 Timing diagram of cycle-steal transfer mode (1)



- The above figure is the example of initial term for processing the first block in **"Figure 13.8.9."** When the array chain transfer mode is selected, there is not the term of ✻2.
- The Bus request caused by DRAM refresh or Hold is sampled while the bus request sampling signal is "1," and is accepted.
- The Bus request caused by DMA is sampled while the bus request sampling signal (✻1) is "1" in transition of the right to use bus, and is accepted. The DMA requests of the other channels are not accepted in an array state.

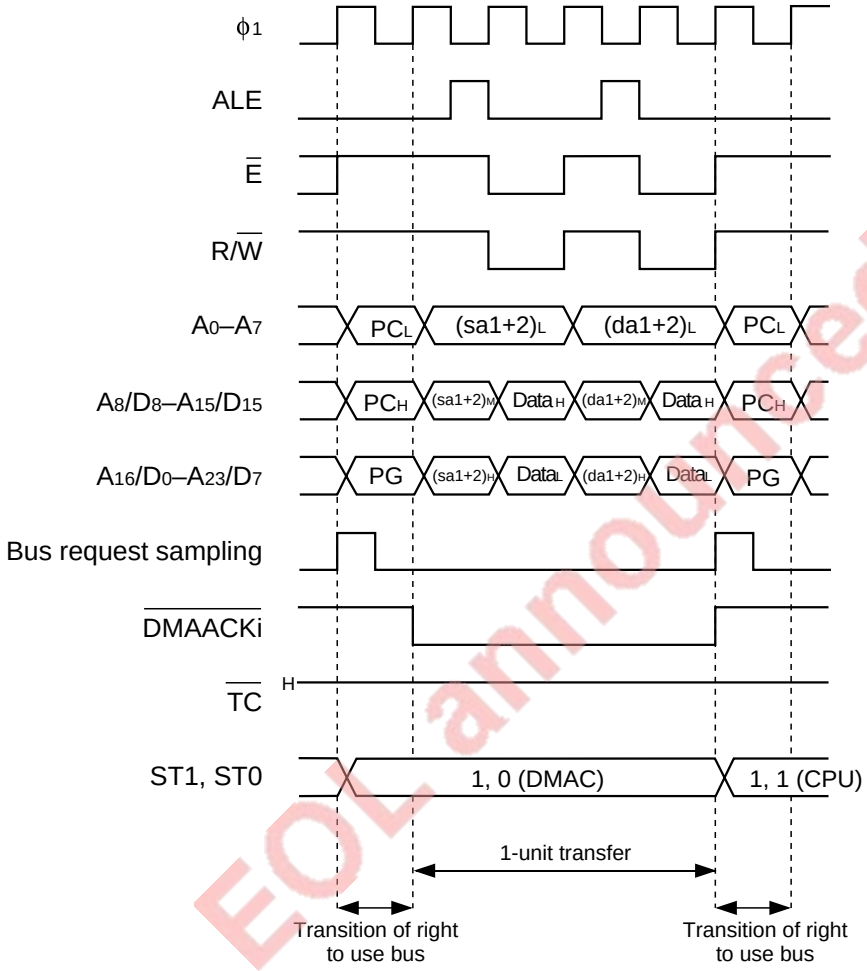
EOL announced

Fig. 13.8.11 Timing diagram of cycle-steal transfer mode (2)

● **1-unit transfer**

1-unit transfer is performed with a DMAi request on the following conditions:

- Single transfer mode (except for the last 1-unit transfer)
- Repeat transfer mode (except for the last 1-unit transfer of block)
- Array chain transfer mode (except for the first and last 1-unit transfers of each block)
- Link array chain transfer mode (except for the first and last 1-unit transfers of each block)



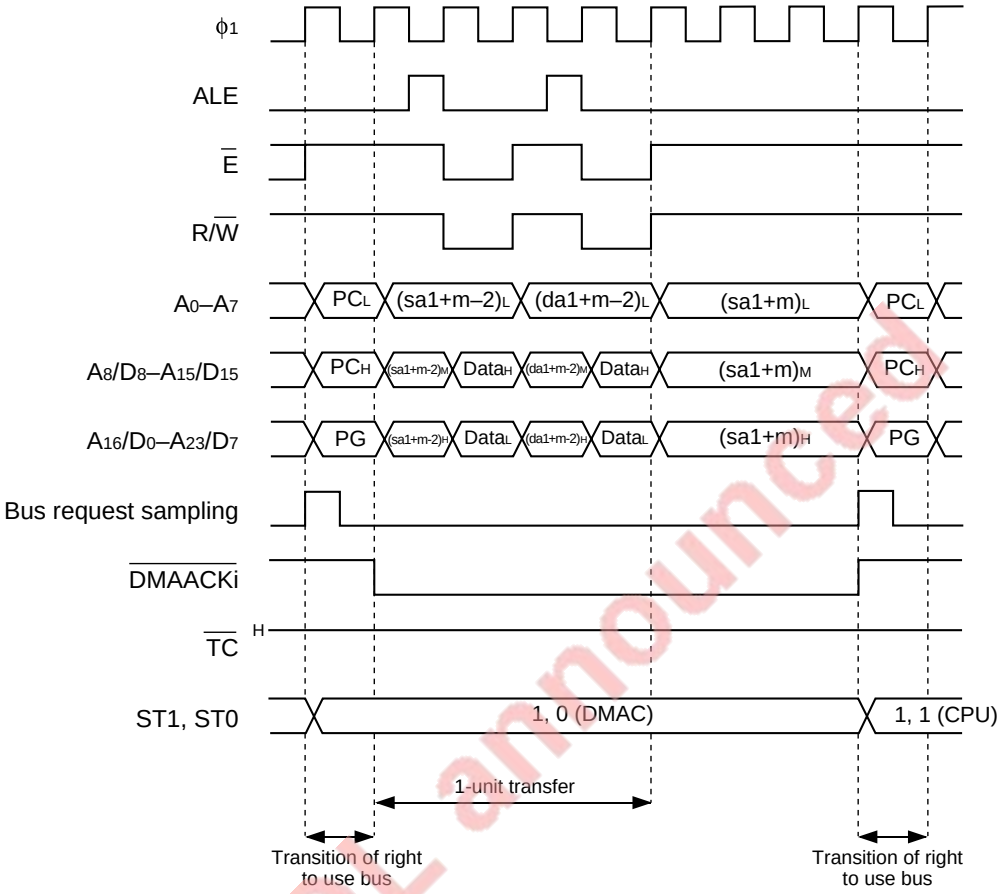
EOL announced

● The above figure is the example of the second 1-unit transfer for processing the first block in **"Figure 13.8.9."**

● The Bus request caused by DRAM refresh, Hold, or DMA is sampled while the Bus request sampling signal is "H," and is accepted.

Fig. 13.8.12 Timing diagram of cycle-steal transfer mode (3)

● **Last transfer of each block**
At the last term (except for the last block) for processing of each block in the repeat, array chain, and link array chain transfer modes, 1-unit transfer is performed with one DMAi request, and the right to use bus is relinquished after 3 cycles of ϕ.

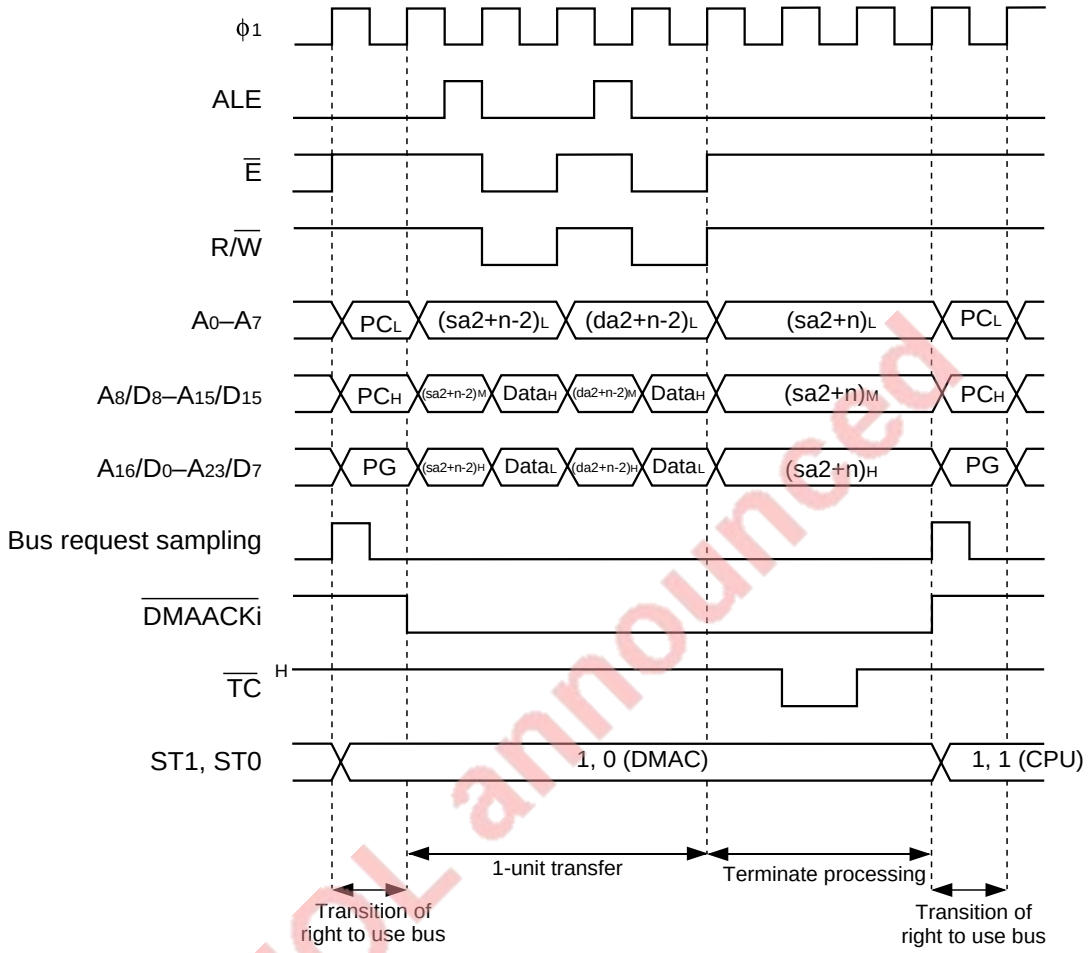


● The above figure is the example of the last term for processing the first block in **"Figure 13.8.9."**
● The Bus request caused by DRAM refresh, Hold, or DMA is sampled while the bus request sampling signal is "H," and is accepted.

EOL announced

Fig. 13.8.13 Timing diagram of cycle-steal transfer mode (4)

● **Last transfer of last block**
At the last term for processing the last block in the single, array chain, and link array chain transfer modes, 1-unit transfer and terminate processing are subsequently performed with one DMAi request.



EOL announced

● The above figure is the example of the last term for processing the second block in **"Figure 13.8.9."**
● The Bus request caused by DRAM refresh, Hold, or DMA is sampled while the bus request sampling signal is "H," and is accepted.

Fig. 13.8.14 Timing diagram of cycle-steal transfer mode (5)

[Precautions for link array chain transfer mode]

If the following two conditions are satisfied when the transfer unit is 16 bits and the address direction of transfer source or destination is fixed, the link array chain transfer mode can be used:

- The external data bus width = 16 bits or the internal memory is used.
- The transfer start address on the address-direction-fixed side is an even address.

EOL announced

13.9 DMA transfer time

Calculation of time from the CPU's relinquishing the right to use bus until its regaining the right under the following conditions is described with reference to cycles of ϕ:

- A DMAi request is generated while the CPU holds the right to use bus.
- The above right is returned to the CPU after completion of DMA transfer for one DMA request.

For the time per 1-unit transfer, refer to section **"13.4.1 (2) Bus operation in 2-bus cycle transfer"** and section **"13.4.2 (2) Bus operation in 1-bus cycle transfer."**
Also, for the time from DMA request generation until the start of the DMA transfer, refer to section **"13.3.4 Processing from DMA request until DMA transfer execution":** and for that from issuing instructions for forced termination until returning the right to use bus to the CPU, refer to section **"13.3.5 (2) Forced termination."**

13.9.1 Cycle-steal transfer mode

(1) 1-unit transfer

In the following cases, 1-unit transfer is performed at one DMAi transfer. (Refer to **"Figure 13.8.12."**)

- Single transfer mode: except for the last 1-unit transfer
- Repeat transfer mode: except for the last 1-unit transfer of a block
- Array chain transfer mode: except for the first and last 1-unit transfers of each block
- Link array chain transfer mode: except for the first and last 1-unit transfers of each block

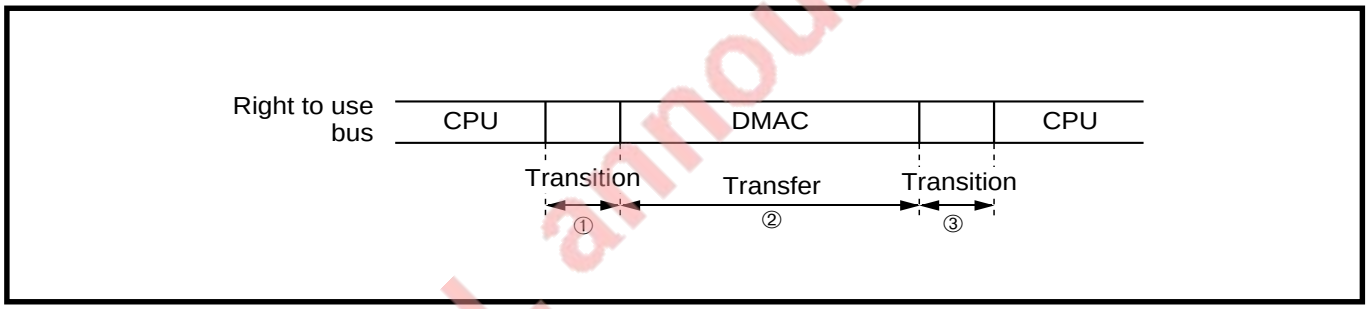


Fig. 13.9.1 1-unit transfer

① Transition of the right to use bus from CPU to DMAC: 1 cycle
② DMA transfer per 1-transfer unit:
- In 2-bus cycle transfer···Read cycle + Write cycle
  (Add a value which satisfies the read/write conditions. Refer to **"Table 13.4.1."**)
- In 1-bus cycle transfer···Refer to **"Table 13.4.5."**

③ Transition of the right to use bus from DMAC to CPU: 1 cycle

[Example]
2-bus cycle transfer, transfer unit =16 bits, external data bus width = 16 bits, and under the following conditions:

- Transfer source: address direction = forward, start address of data = even, with Wait
- Transfer destination: address direction = backward, start address of data = even, without Wait

① + ② + ③ = 1 + (3 + 4) + 1 = 9 cycles

EOL announced

(2) Last transfer of each block

In the following cases, 1-unit transfer and the processing for 3 cycles are performed sequentially. (Refer to **"Figures 13.8.13 and 13.8.14."**)

- Single transfer mode: the last 1-unit transfer
- Repeat transfer mode: the last 1-unit transfer of a block
- Array chain transfer mode: the last 1-unit transfer of each block (including the last block)
- Link array chain transfer mode: the last 1-unit transfer of each block (including the last block)

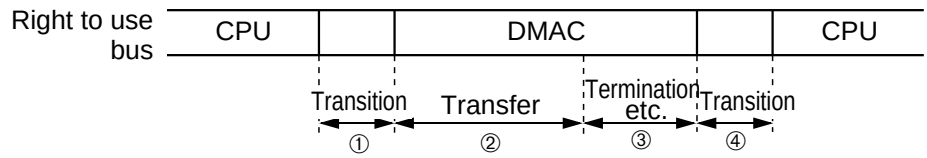


Fig. 13.9.2 Last transfer of each block

① Transition of the right to use bus from CPU to DMAC: 1 cycle
② DMA transfer per 1-unit transfer:
- In 2-bus cycle transfer···Read cycle + Write cycle
  (Add a value which satisfies the read/write conditions. Refer to **"Table 13.4.1."**)
- In 1-bus cycle transfer···Refer to **"Table 13.4.5."**

③ Terminate processing or the last processing of each block: 3 cycles
④ Transition of the right to use bus from DMAC to CPU: 1 cycle

[Example]
2-bus cycle transfer, transfer unit =16 bits, external data bus width = 16 bits, and under the following conditions:

- Transfer source: address direction = forward, start address of data = even, with Wait
- Transfer destination: address direction = backward, start address of data = even, without Wait

① + ② + ③ + ④ = 1 + (3 + 4) + 3 + 1 = 12 cycles

EOL announced

(3) Transfer of array state

In the following cases, the processing in an array state and the first 1-unit transfer are performed sequentially. (Refer to **"Figures 13.8.10 and 13.8.11."**)

- Array chain transfer mode: the first transfer of each block
- Link array chain transfer mode: the first transfer of each block

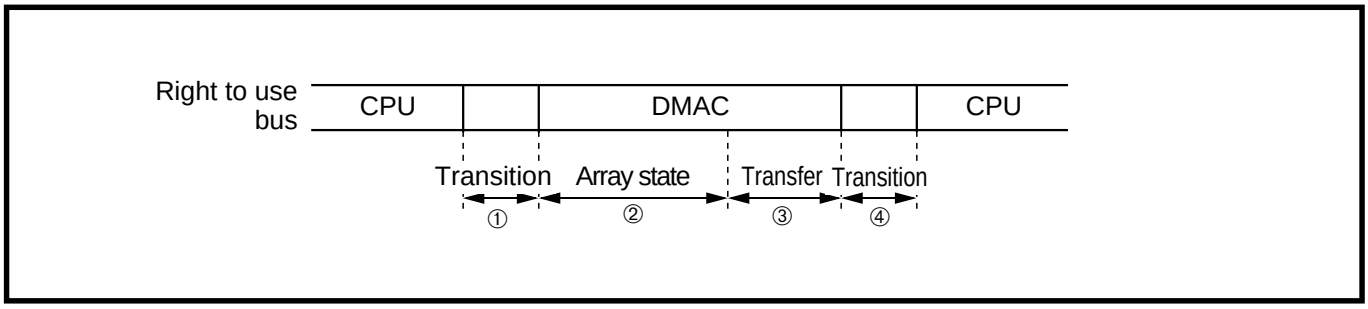


Fig. 13.9.3 Transfer of array state

① Transition of the right to use bus from CPU to DMAC: 1 cycle

② Array state:
The number of transfer parameters × the number of reads of a transfer parameter × the number of bus cycles for a read + 1 cycle (Refer to **"Table 13.9.1."**)

③ DMA transfer per 1-unit transfer:
- In 2-bus cycle transfer···Read cycle + Write cycle
  (Add a value which satisfies the read/write conditions. Refer to **"Table 13.4.1."**)
- In 1-bus cycle transfer···Refer to **"Table 13.4.5."**

④ Transition of the right to use bus from DMAC to CPU: 1 cycle

[Example]
Link array chain transfer mode, external data bus width = 16 bits, 2-bus cycle transfer, transfer unit =16 bits, and under the following conditions:
- Transfer source: address direction = forward, start address of data = even, with Wait
- Transfer destination: address direction = backward, start address of data =odd, with Wait

① + ② + ③ + ④ = 1 + 25 + (3 + 4) + 1 = 34 cycles

Not announced

Table 13.9.1 Time required for processing in array state

Mode	External data bus width	Transfer method	Number of transfer parameters	Number of reads of a transfer parameter	Time required for processing in array state (Unit: ϕ cycle)
Array chain transfer mode	16 bits (Including internal bus)	2-bus cycle transfer	3	2	3 × 2 × 3 + 1 = 19
		1-bus cycle transfer	2	2	2 × 2 × 3 + 1 = 13
	8 bits	2-bus cycle transfer	3	4	3 × 4 × 3 + 1 = 37
		1-bus cycle transfer	2	4	2 × 4 × 3 + 1 = 25
Link array chain transfer mode	16 bits (Including internal bus)	2-bus cycle transfer	4	2	4 × 2 × 3 + 1 = 25
		1-bus cycle transfer	3	2	3 × 2 × 3 + 1 = 19
	8 bits	2-bus cycle transfer	4	4	4 × 4 × 3 + 1 = 49
		1-bus cycle transfer	3	4	3 × 4 × 3 + 1 = 37

13.9.2 Burst transfer mode

(1) Single transfer mode

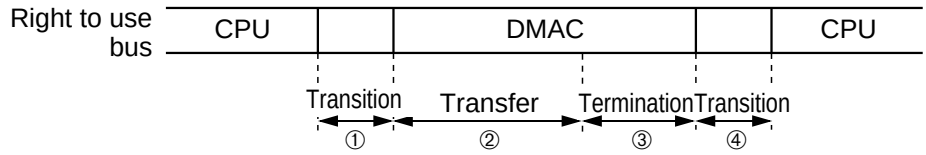


Fig. 13.9.4 Single transfer mode (burst transfer mode selected)

① Transition of the right to use bus from CPU to DMAC: 1 cycle
② DMA transfer per an entire batch of data:
- In 2-bus cycle transfer···(Read cycle + Write cycle※1) × the number of transfers※2

※1: Add a value which satisfies the read/write conditions. Refer to **"Table 13.4.1."**

※2: When the transfer unit is 16 bits, the number of transfers = the number of transfer bytes/2

When the transfer unit is 8 bits, the number of transfers = the number of transfer bytes

- In 1-bus cycle transfer···Refer to **"Table 13.4.5."**

③ Terminate processing: 3 cycles
④ Transition of the right to use bus from DMAC to CPU: 1 cycle

[Example]
External data bus width = 16 bits, 2-bus cycle transfer, transfer unit =16 bits, the number of the transfer bytes = 10 bytes, and under the following conditions:
- Transfer source: address direction = forward, start address of data = even, with Wait
- Transfer destination: address direction = backward, start address of data = even, without Wait

① + ② + ③ + ④ = 1 + 5 (3 + 4) + (3 + 1) = 40 cycles

EOL announced

(2) Repeat transfer mode

In the repeat transfer mode of burst transfer (edge sense), the method of terminating DMA transfer is only the forced termination by the $\overline{TC}$ input. Therefore, the time from the CPU's relinquishing the right to use bus until regaining the right depends on the timing of the $\overline{TC}$ input.

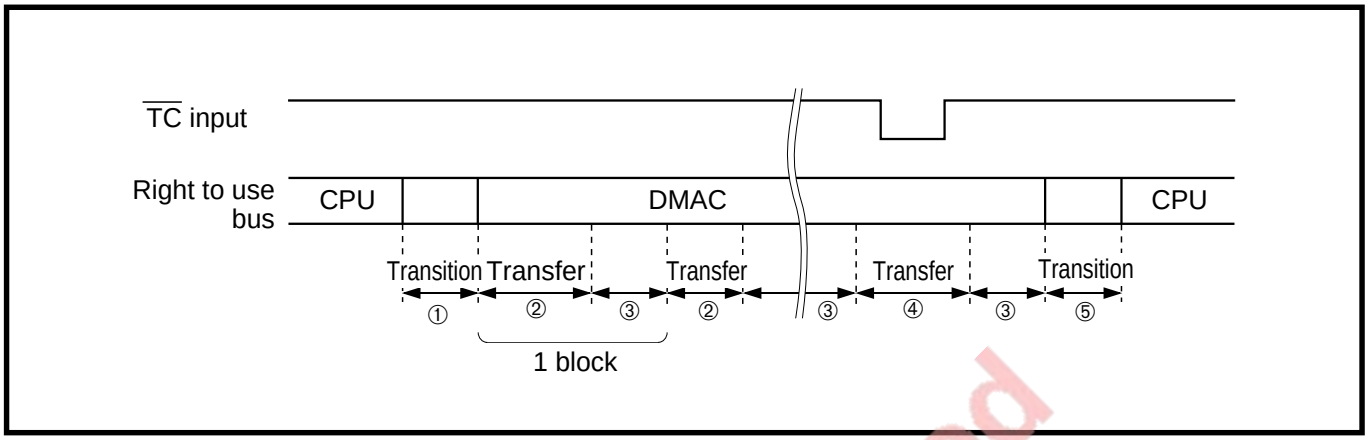


Fig. 13.9.5 Repeat transfer mode (burst transfer mode and edge sense selected)

① Transition of the right to use bus from CPU to DMAC: 1 cycle
② DMA transfer per 1 block:
- In 2-bus cycle transfer···(Read cycle + Write cycle※1) × the number of transfers※2

※1: Add a value which satisfies the read/write conditions. Refer to **"Table 13.4.1."**

※2: When the transfer unit is 16 bits, the number of transfers = the number of transfer bytes/2
When the transfer unit is 8 bits, the number of transfers = the number of transfer bytes

- In 1-bus cycle transfer···Refer to **"Table 13.4.5."**

③ Terminate processing: 3 cycles
④ DMA transfer of the block at the $\overline{TC}$ input: above ②
The number of transfers is assumed to be up to the DMA transfer of 1-unit transfer which was in progress at the $\overline{TC}$ input.
⑤ Transition of the right to use bus from DMAC to CPU: 1 cycle

[Example]
External data bus width = 16 bits, 2-bus cycle transfer, transfer unit =16 bits, the number of the transfer bytes = 10 bytes, and under the following conditions:
- Transfer source: address direction = forward, start address of data = even, with Wait
- Transfer destination: address direction = backward, start address of data = even, without Wait
- $\overline{TC}$ is input when the "m"th byte (m = even) of the "n"th block is in transfer.

① + (n – 1) (② + ③) + ④ + ⑤ = 1 + (n – 1){5(3 + 4) + 3} + m/2 + 1 = 38n + m/2 – 36 cycles

EOL announced

(3) Array chain transfer mode and Link array chain transfer mode

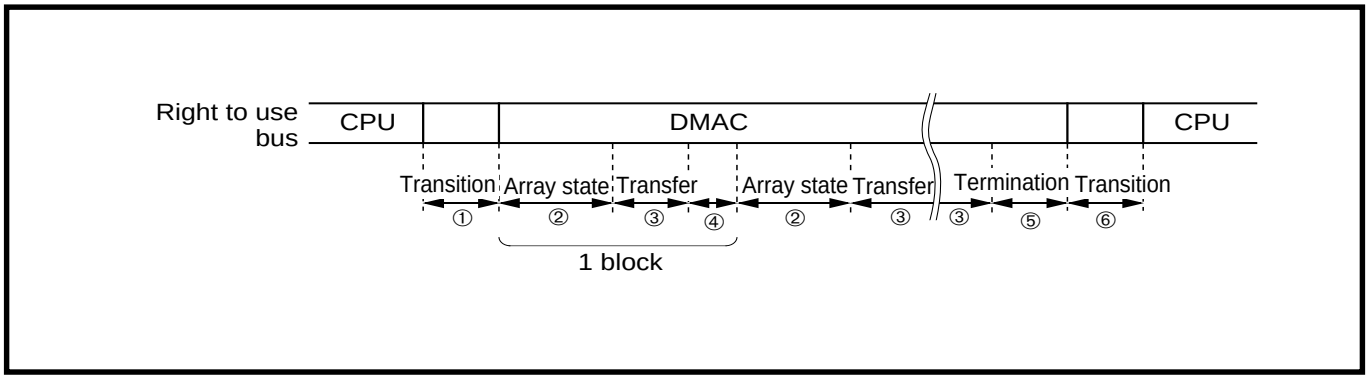


Fig 13.9.6 Array chain transfer mode and Link array chain transfer mode

① Transition of the right to use bus from CPU to DMAC: 1 cycle
② Array state:
The number of transfer parameters × the number of reads of a transfer parameter× the number of bus cycles for a read + 1 cycle (Refer to **"Table 13.9.1."**)
③ DMA transfer per an entire batch of data:
- In 2-bus cycle transfer···(Read cycle + Write cycle※1) × the number of transfers※2

※1: Add a value which satisfies the read/write conditions. Refer to **"Table 13.4.1."**
※2: When the transfer unit is 16 bits, the number of transfers = the number of transfer bytes/2
When the transfer unit is 8 bits, the number of transfers = the number of transfer bytes

- In 1-bus cycle transfer···Refer to **"Table 13.4.5."**

④ Last processing of each block: 3 cycles
⑤ Terminate processing: 3 cycles
⑥ Transition of the right to use bus from DMAC to CPU: 1 cycle

[Example]
Array chain transfer mode, external data bus width = 16 bits, 2-bus cycle transfer, transfer unit =16 bits, the number of transfer blocks = 3, and under the following conditions:
- Transfer source: address direction = forward, without Wait
- Transfer destination: address direction = backward, without Wait
- First block: transfer source's data start address = even, transfer destination's data start address = even, the number of transfer bytes =10 bytes
- Second block: transfer source's data start address = even, transfer destination's data start address =odd, the number of transfer bytes =12 bytes
- Third block: transfer source's data start address =odd, transfer destination's data start address =odd, the number of transfer bytes =14 bytes

① + ② + ③ + ④ + ② + ③ + ④ + ② + ③ + ⑤ + ⑥
= 1 + 19 + 5(2 + 4) + 3 + 19 + 6(2 + 3) + 3 + 19 + 7(4 + 3) + 3 + 1 = 177 cycles

EOL announced

MEMORANDUM

EOL announced

CHAPTER 14
DRAM CONTROLLER

14.1 Overview
14.2 Block description
14.3 Setting for DRAMC
14.4 DRAMC operation
14.5 Precautions for DRAMC

EOL announced

14.1 Overview

Table 14.1.1 lists the performance specifications of DRAM controller (hereafter called DRAMC).

Table 14.1.1 Performance specifications of DRAMC

Item	Performance specifications
DRAM area	0 to 15 Mbytes; programmable in a unit of 1 Mbyte
Refreshing method	$\overline{CAS}$ before $\overline{RAS}$; dispersive refreshing
Refresh timer	8 bits
Multiplexed address pins	10

14.2 Block description

Figure 14.2.1 shows the block diagram of DRAMC. Registers relevant to DRAMC are described below.

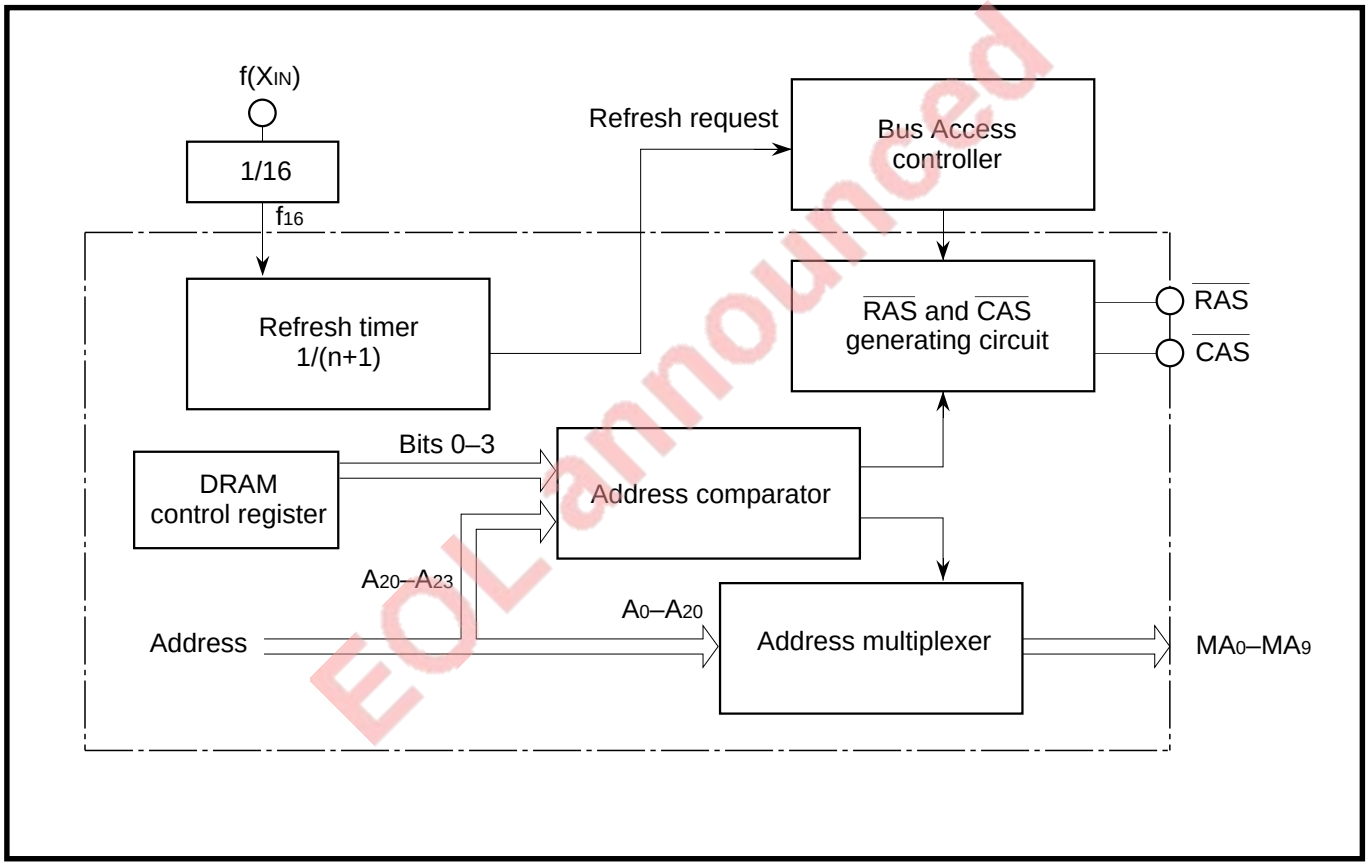


Fig. 14.2.1 Block diagram of DRAMC

14.2.1 DRAM control register

Figure 14.2.2 shows the structure of the DRAM control register.



DRAM control register (Address 64_{16})

Bit	Bit name	Functions	At reset	RW
0	DRAM area select bits	b3 b2 b1 b0 0 0 0 0 : No DRAM area 0 0 0 1 : $F00000_{16}$–$FFFFFF_{16}$ (1 Mbyte) 0 0 1 0 : $E00000_{16}$–$FFFFFF_{16}$ (2 Mbytes) 0 0 1 1 : $D00000_{16}$–$FFFFFF_{16}$ (3 Mbytes) 0 1 0 0 : $C00000_{16}$–$FFFFFF_{16}$ (4 Mbytes) 0 1 0 1 : $B00000_{16}$–$FFFFFF_{16}$ (5 Mbytes) 0 1 1 0 : $A00000_{16}$–$FFFFFF_{16}$ (6 Mbytes) 0 1 1 1 : 900000_{16}–$FFFFFF_{16}$ (7 Mbytes) 1 0 0 0 : 800000_{16}–$FFFFFF_{16}$ (8 Mbytes) 1 0 0 1 : 700000_{16}–$FFFFFF_{16}$ (9 Mbytes) 1 0 1 0 : 600000_{16}–$FFFFFF_{16}$ (10 Mbytes) 1 0 1 1 : 500000_{16}–$FFFFFF_{16}$ (11 Mbytes) 1 1 0 0 : 400000_{16}–$FFFFFF_{16}$ (12 Mbytes) 1 1 0 1 : 300000_{16}–$FFFFFF_{16}$ (13 Mbytes) 1 1 1 0 : 200000_{16}–$FFFFFF_{16}$ (14 Mbytes) 1 1 1 1 : 100000_{16}–$FFFFFF_{16}$ (15 Mbytes)	0	RW
1			0	RW
2			0	RW
3			0	RW
6 to 4	Nothing is assigned. The value is “0” at reading.		0	–
7	DRAM validity bit **(Note)**	0 : Invalid ($P10_4$–$P10_7$ pins function as programmable input ports. A_0–A_7 pins function as address output pins. Refresh timer stops counting.) 1 : Valid ($P10_4$–$P10_7$ pins function as $\overline{CAS}$, $\overline{RAS}$, MA_8, and MA_9. A_0–A_7 function as MA_0–MA_7. Refresh timer starts counting.)	0	RW

Note: Set the refresh timer (address 66_{16}) before setting this bit to “1.”

EOL announced

Fig. 14.2.2 Structure of DRAM control register

14.2 Block description

(1) DRAM area select bits (bits 0 to 3)

These 4 bits specify a DRAM area of 15 Mbytes maximum in a unit of 1 Mbyte. Figure 14.2.3 shows setting examples of DRAM areas.

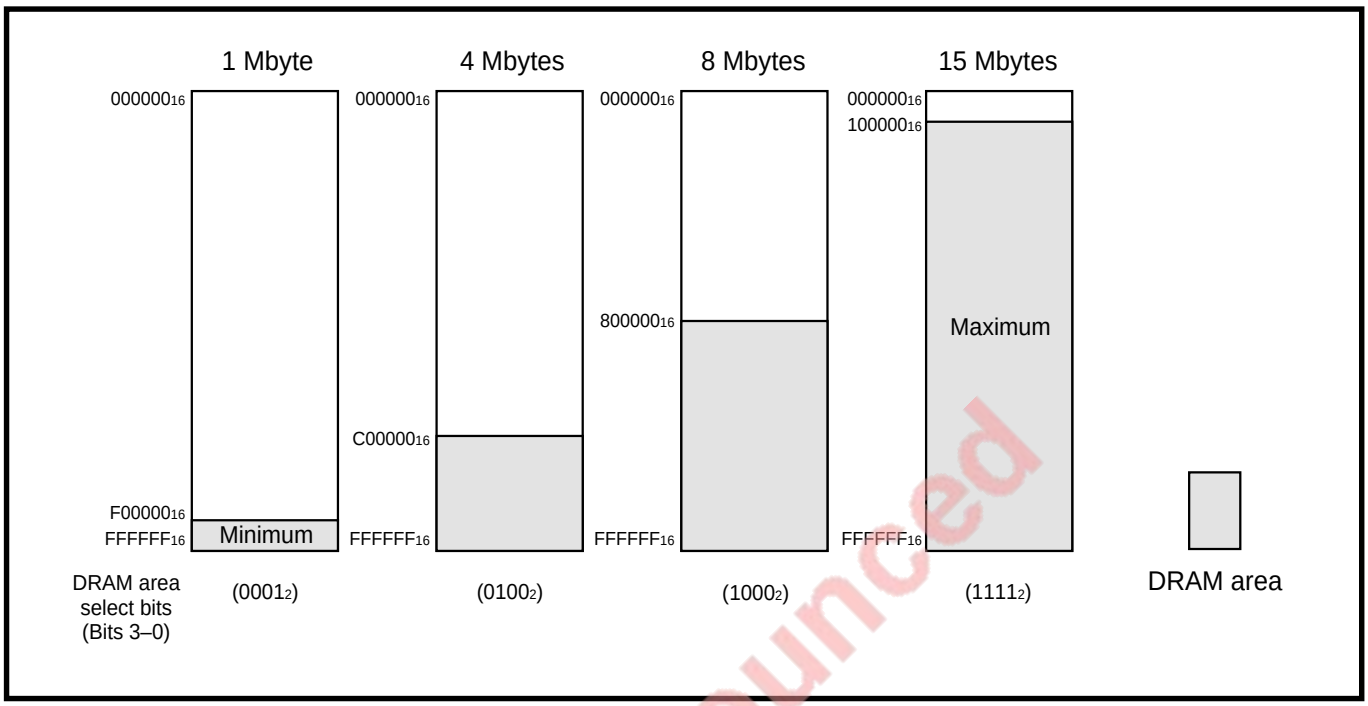


Fig. 14.2.3 Setting examples of DRAM areas

(2) DRAM validity bit (bit 7)

When this bit is set to "1," pin functions for DRAM control become valid, and the refresh timer starts counting. Table 14.2.1 lists the pin functions for DRAM control.

Table 14.2.1 Pin functions for DRAM control

DRAM validity bit	Operation	Pins				
		A_0/MA_0 –A_7/MA_7	$P10_6/MA_8$, $P10_7/MA_9$	$P10_4/\overline{CAS}$, $P10_5/\overline{RAS}$	A_{16}/D_0–A_{23}/D_7, A_8/D_8–A_{15}/D_{15}, R/$\overline{W}$, $\overline{E}$, $\overline{BLE}$, $\overline{BHE}$	ST0, ST1
1	Accessing DRAM area	MA_0–MA_7	MA_8, MA_9	$\overline{CAS}$, $\overline{RAS}$	A_{16}/D_0–A_{23}/D_7, A_8/D_8–A_{15}/D_{15}, R/$\overline{W}$, $\overline{E}$, $\overline{BLE}$, $\overline{BHE}$	ST0, ST1
	DRAM refresh	A_0–A_7	MA_8, MA_9	$\overline{CAS}$, $\overline{RAS}$	A_{16}/D_0–A_{23}/D_7, A_8/D_8–A_{15}/D_{15} R/$\overline{W}$, $\overline{E}$, $\overline{BLE}$, $\overline{BHE}$	ST0, ST1 ([0,0] is output.)
	Other than the above	A_0–A_7	MA_8, MA_9	$\overline{CAS}$, $\overline{RAS}$	A_{16}/D_0–A_{23}/D_7, A_8/D_8–A_{15}/D_{15}, R/$\overline{W}$, $\overline{E}$, $\overline{BLE}$, $\overline{BHE}$	ST0, ST1
0	—	A_0–A_7	$P10_6$, $P10_7$	$P10_4$,$P10_5$	A_{16}/D_0–A_{23}/D_7, A_8/D_8–A_{15}/D_{15} R/$\overline{W}$, $\overline{E}$, $\overline{BLE}$, $\overline{BHE}$	ST0, ST1

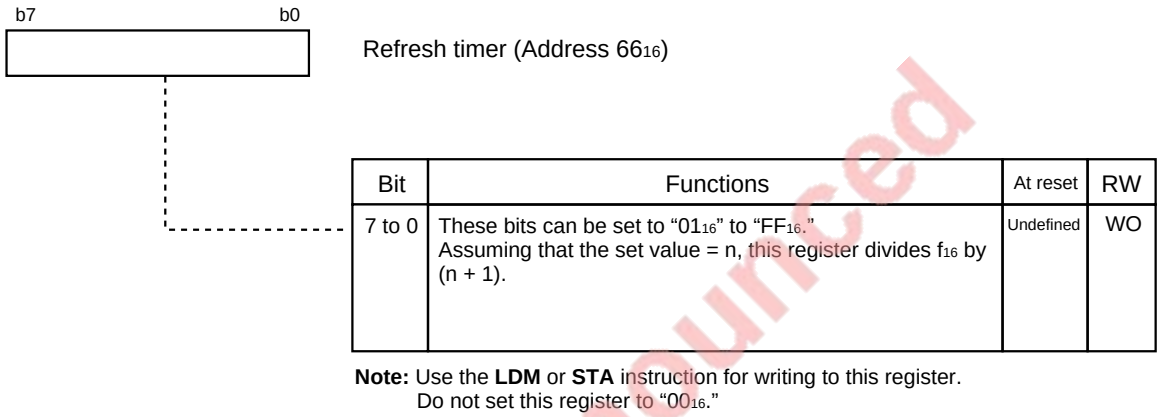
EOL announced

14.2.2 Refresh timer

The refresh timer is an 8-bit timer with a reload register and is used to generate refresh requests for DRAM data. Assuming that the set value of the refresh timer = "n," the refresh timer counts f_{16} (n + 1) times. Figure 14.2.4 shows the structure of the refresh timer, and the following formula gives the value to be written to the refresh timer.

$$n = \{m\ [\mu s] \times \frac{f(X_{IN})}{16}\} - 1$$

n: a set value of the refresh timer ($n = 01_{16}$–FF_{16})
m: a refresh interval
Examples of "m": an average of 15.625 μs for 512 refresh cycles at 8-ms intervals
an average of 125 μs for 512 refresh cycles at 64-ms intervals



Refresh timer (Address 66_{16})

Bit	Functions	At reset	RW
7 to 0	These bits can be set to "01_{16}" to "FF_{16}." Assuming that the set value = n, this register divides f_{16} by (n + 1).	Undefined	WO

Note: Use the **LDM** or **STA** instruction for writing to this register. Do not set this register to "00_{16}."

Fig. 14.2.4 Structure of refresh timer

EOL announced

14.2 Block description

14.2.3 Address comparator

The address comparator examines whether the address to be accessed is within the DRAM area. When this address is within DRAM area, control signals are sent to the $\overline{RAS}$ and $\overline{CAS}$ generating circuit and the address multiplexer.

14.2.4 $\overline{RAS}$ and $\overline{CAS}$ generating circuit

The $\overline{RAS}$ signal (a timing signal to latch a row address) and the $\overline{CAS}$ signal (a timing signal to latch a column address) are generated by a control signal from the address comparator.

14.2.5 Address multiplexer

Address data is time-shared by the control signal from the address comparator and is output to the MA_0–MA_9 pins. The time-sharing method depends on the external bus width. Table 14.2.2 lists the time-sharing method for the address at DRAM access. When the 8-bit external bus width is selected, A_0–A_{19} are time-shared and are output; when the 16-bit external bus width is selected, A_1–A_{20} are time-shared and are output.

Table 14.2.2 Time-sharing method for address at DRAM access

Pin name			A_0/MA_0	A_1/MA_1	A_2/MA_2	A_3/MA_3	A_4/MA_4	A_5/MA_5	A_6/MA_6	A_7/MA_7	$P10_6/MA_8$	$P10_7/MA_9$
Output signal	8-bit external bus width	Row address	A_0	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_{16}	A_{18}
		Column address	A_8	A_9	A_{10}	A_{11}	A_{12}	A_{13}	A_{14}	A_{15}	A_{17}	A_{19}
	16-bit external bus width	Row address	A_{16}	A_1	A_2	A_3	A_4	A_5	A_6	A_7	A_{18}	A_{20}
		Column address	A_8	A_9	A_{10}	A_{11}	A_{12}	A_{13}	A_{14}	A_{15}	A_{17}	A_{19}

EOL announced

14.3 Setting for DRAMC

Figure 14.3.1 shows an initial setting example for registers relevant to DRAMC.

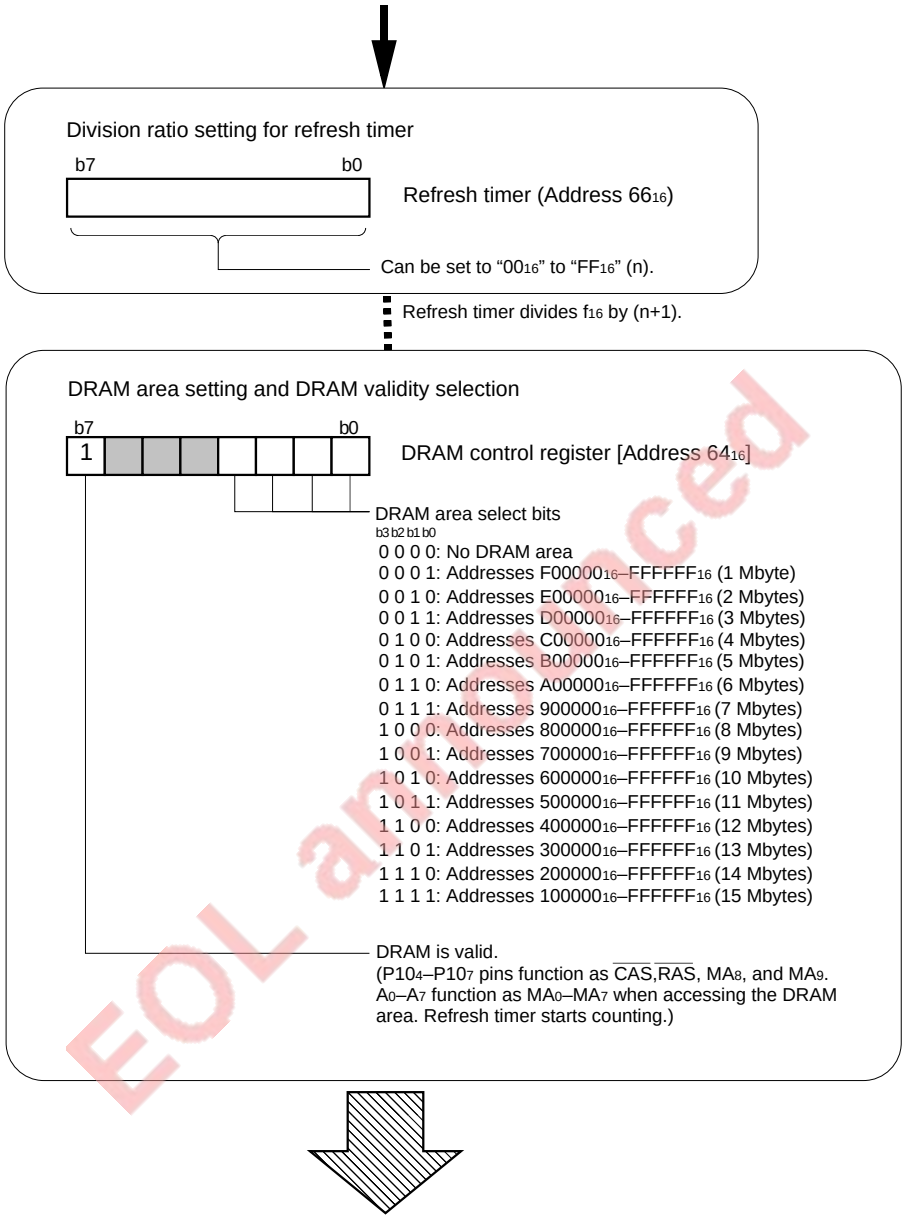


Fig. 14.3.1 Initial setting example for registers relevant to DRAMC

EOL announced

14.4 DRAMC operation

14.4.1 Waveform example of DRAM control signals

Figure 14.4.1 shows a waveform example of the DRAM control signals. When DRAM is accessed, the bus cycle is always with "wait" (the low-level width of $\overline{E}$ is equivalent to 2 cycles of ϕ). It is not affected by the wait bit, the wait bit of the transfer source, and the wait bit of the transfer destination.

(1) Read Cycle

In the read cycle, the $\overline{CAS}$ signal falls with a delay of 0.5 cycle of ϕ after the $\overline{RAS}$ signal has changed from "H" to "L." The address bus signal changes from "row address" to "column address" within a period from a fall of $\overline{RAS}$ until a fall of $\overline{CAS}$.

Pins A_{16}/D_0–A_{23}/D_7 and A_8/D_8–A_{15}/D_{15} output addresses and input data in the same way as in reading external devices other than DRAM.

(2) Write Cycle

In the write cycle, the $\overline{CAS}$ signal falls with a delay of 1 cycle of ϕ after the $\overline{RAS}$ signal has changed from "H" to "L." The address bus signal changes "row address" to "column address" within a period from a fall of $\overline{RAS}$ until a fall of $\overline{CAS}$.

Pins A_{16}/D_0–A_{23}/D_7 and A_8/D_8–A_{15}/D_{15} output addresses and data in the same way as in writing external devices other than DRAM.

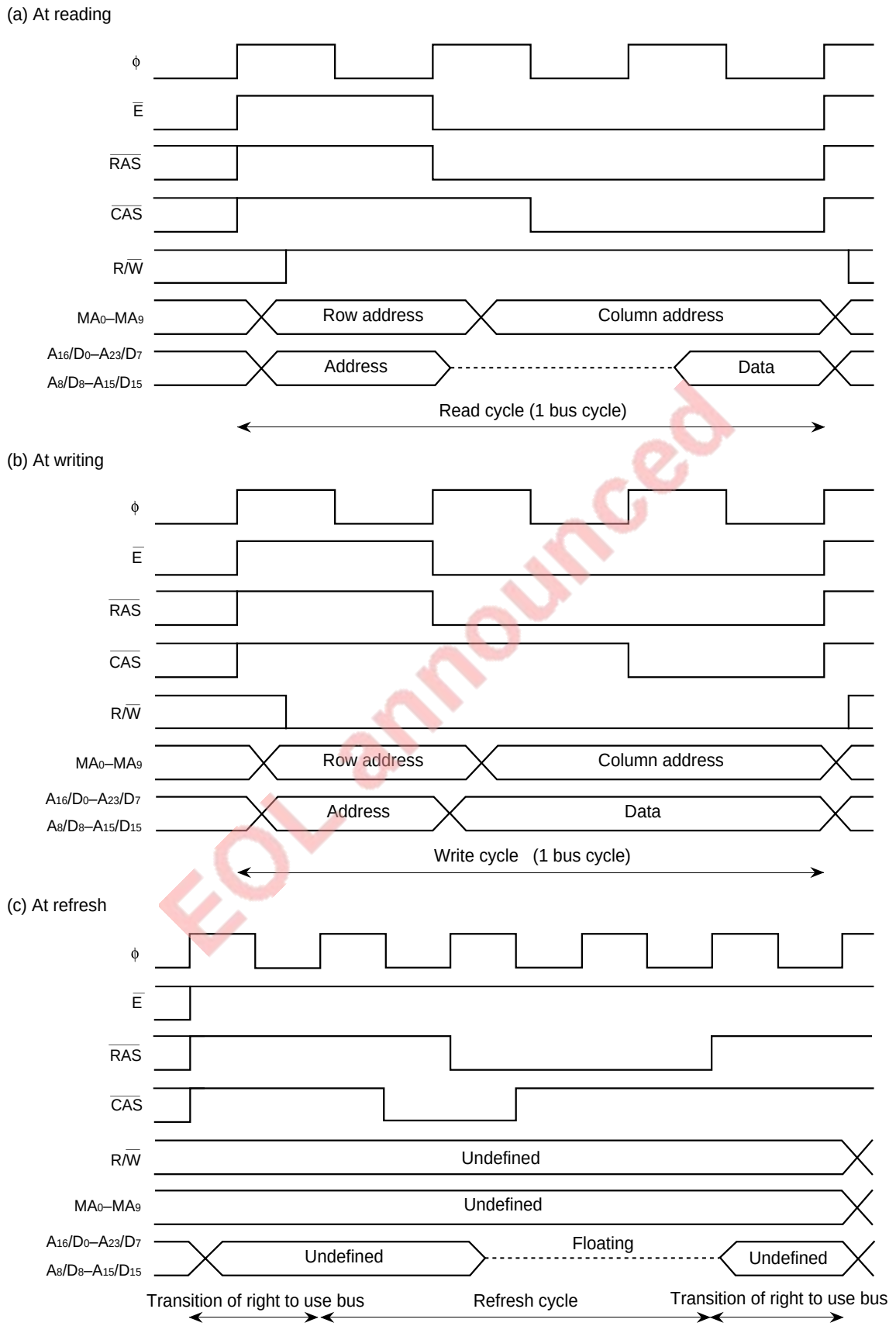
(3) Refresh Cycle

In the refresh cycle, the $\overline{RAS}$ signal falls with a delay of 0.5 cycle of ϕ after the $\overline{CAS}$ signal has changed from "H" to "L."

R/$\overline{W}$ is undefined.

One refresh request requires 5 cycle of ϕ, including the time for passing the right to use buses.

EOL announced



EOL announced

Fig. 14.4.1 Waveform example of DRAM control signals

14.4.2 Refresh request

① When the DRAM validity bit is set to "1," the refresh timer starts counting down. The count source is f_{16}.

② When the contents of the refresh timer reach "00_{16}," a refresh request occurs. The refresh timer reloads the contents of address 66_{16} and continues counting.

③ Refresh requests are sampled as bus requests (DRAMC) by using the bus access controller.
As soon as a refresh request is acknowledged by sampling, the following ④ is performed because DRAM refresh has the highest priority in using the bus.
However, when the CPU or DMAC uses the bus, no bus request is sampled until the CPU or DMAC releases the bus.
Therefore, in a period from when a refresh request occurs until DRAM refresh is performed, the delay listed in Table 14.4.1 occurs depending on the refresh request generating timing.
Figures 14.4.2 and 14.4.3 show refresh delay time examples when CPU is operating and during DMA transfer. For a bus request, refer to **"13.2.1 Bus access control circuit."**

④ When the refresh request is accepted, the right to use the bus is passed to DRAM refresh (1 cycle of ϕ). Both of the output levels of ST1 and ST0 are "L." (The bus status is indicated as [0, 0].)

⑤ The $\overline{RAS}$ and the $\overline{CAS}$ signals are output and the DRAM data is refreshed (refresh cycle: 3 cycles of ϕ).

⑥ The right to use the bus is passed to the CPU, DRAM or Hold (1 cycle of ϕ).
The outputs of ST1 and ST0 change.

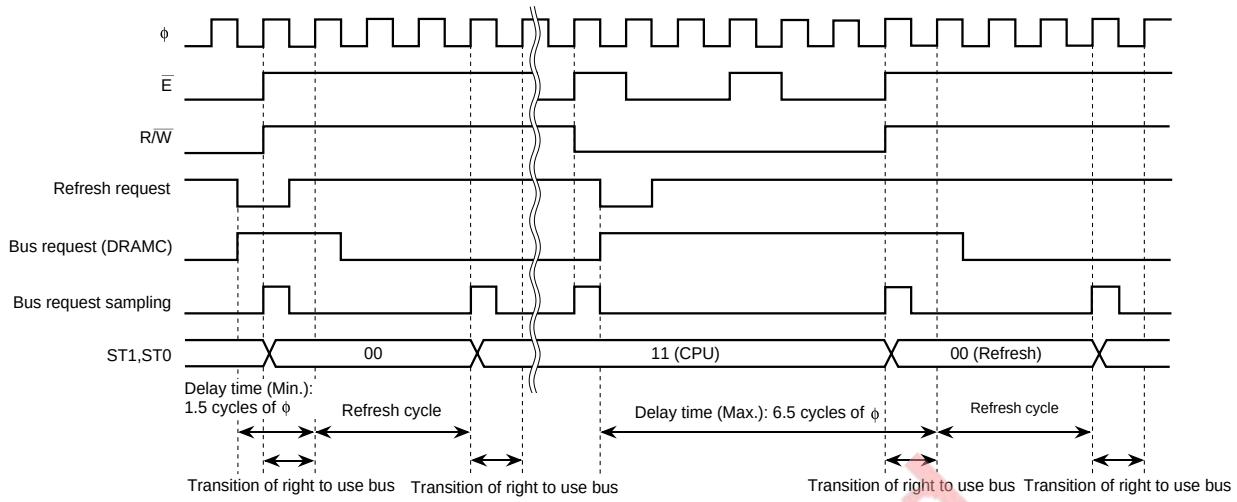
Note: In Stop or Wait mode, DRAM refresh is not performed because no refresh request occurs.

Table 14.4.1 Delay time from when refresh request occurs until DRAM refresh is performed

Source of using bus		Delay time (unit: ϕ cycle)		
		Minimum	Maximum (no Wait)	Maximum (with Wait)
CPU		1.5	4.5	6.5
DMAC	Transfer (a unit of 1 transfer)	1.5	8.5	12.5
	Transfer (a unit of 1 transfer) + Complete cycle		11.5	15.5
	Array state		6.5	6.5
Hold		1.5	1.5	1.5

Note: The above is applied when Ready is not used. The delay time includes the time for passing the right to use buses to DRAM refresh (1 cycle).

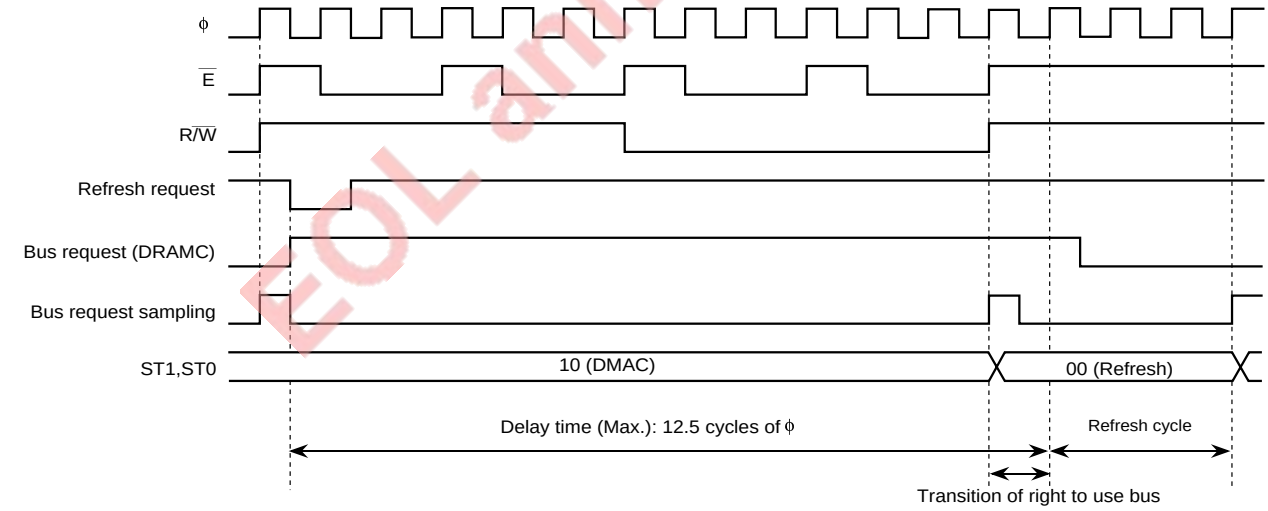
EOL announced



The following are internal signals:
•Refresh request
•Bus request (DRAMC)
•Bus request sampling

Refresh request becomes “0” at an underflow of the refresh timer.

Fig. 14.4.2 Refresh delay time example when CPU is operating



The following are internal signals:
•Refresh request
•Bus request (DRAMC)
•Bus request sampling

Refresh request becomes “0” at an underflow of the refresh timer.

Fig. 14.4.3 Refresh delay time example during DMA transfer

EOL announced

14.5 Precautions for DRAMC

1. Set the refresh timer (address 66_{16}) to any of 01_{16}–FF_{16}.

2. When a DRAM refresh request occurs during Hold state, a refresh cycle is activated regardless of the bus state. It is because a bus request is always sampled during Hold state. Therefore, in order to use the DRAMC together with the Hold function, an external circuit which is controlled depending on the states of ST0 and ST1 is required.

3. DRAM refresh is not performed in Stop or Wait mode.

EOL announced

CHAPTER 15
WATCHDOG TIMER

15.1 Block description
15.2 Operation description
15.3 Precautions for Watchdog timer

EOL announced

Watchdog functions as follows:

- Detects a program runaway.
- Measures a certain time from when oscillation starts owing to terminating Stop mode. (Refer to section **"5.3 Stop mode."**)

15.1 Block description

Figure 15.1.1 shows the block diagram of Watchdog timer.

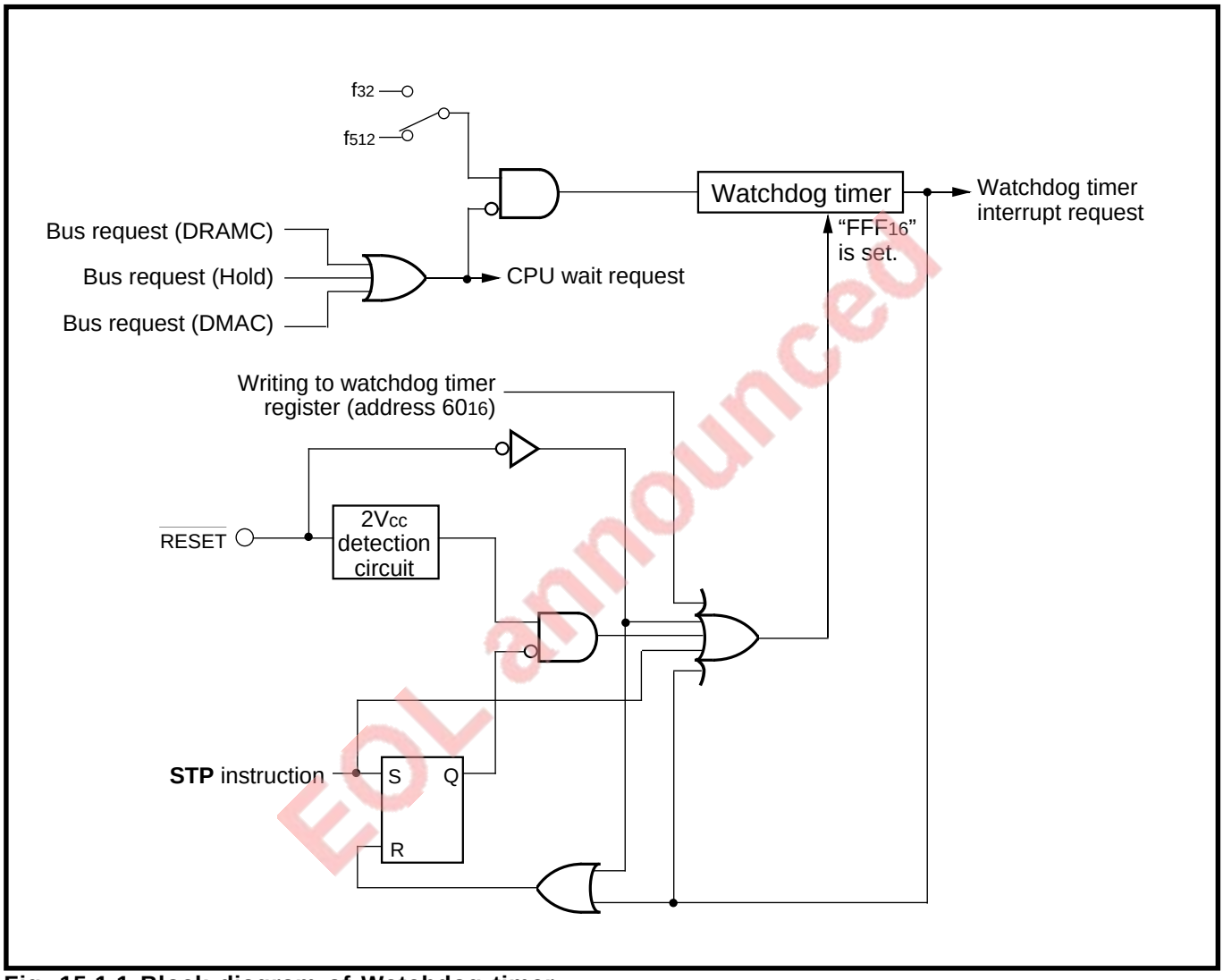


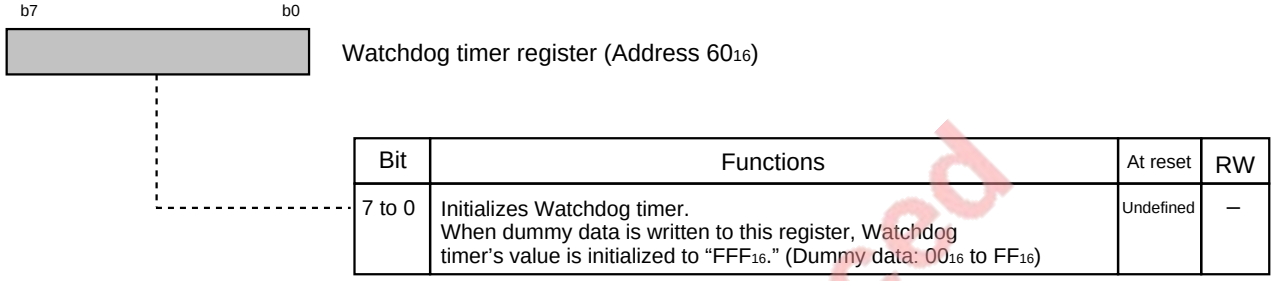
Fig. 15.1.1 Block diagram of Watchdog timer

EOL announced

15.1.1 Watchdog timer

Watchdog timer is a 12-bit counter where the count source which is selected with the watchdog timer frequency select bit (bit 0 at address 61_{16}) is counted down. A value "FFF_{16}" is automatically set in Watchdog timer in the cases listed below. An arbitrary value cannot be set to Watchdog timer.

- When dummy data is written to the watchdog timer register (Refer to **"Figure 15.1.2."**)
- When the most significant bit of Watchdog timer becomes "0"
- When the **STP** instruction is executed (Refer to section **"5.3 Stop mode."**)
- At reset

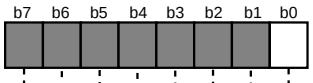


Bit	Functions	At reset	RW
7 to 0	Initializes Watchdog timer. When dummy data is written to this register, Watchdog timer's value is initialized to "FFF_{16}." (Dummy data: 00_{16} to FF_{16})	Undefined	–

Fig. 15.1.2 Structure of watchdog timer register

15.1.2 Watchdog timer frequency select register

This is used to select a Watchdog timer's count source. Figure 15.1.3 shows the structure of the watchdog timer frequency select register.



Watchdog timer frequency select register (Address 61_{16})

Bit	Bit name	Functions	At reset	RW
0	Watchdog timer frequency select bit	0 : f_{512} 1 : f_{32}	0	RW
7 to 1	Nothing is assigned.		Undefined	–

Fig. 15.1.3 Structure of watchdog timer frequency select register

EOL announced

15.2 Operation description

15.2.1 Basic operation

① Watchdog timer starts counting down from "FFF_{16}."
② When the Watchdog timer's most significant bit becomes "0" (counted 2048 times), a watchdog timer interrupt request occurs. (Refer to **"Table 15.2.1."**)
③ When the interrupt request occurs at above ②, a value "FFF_{16}" is set to Watchdog timer.

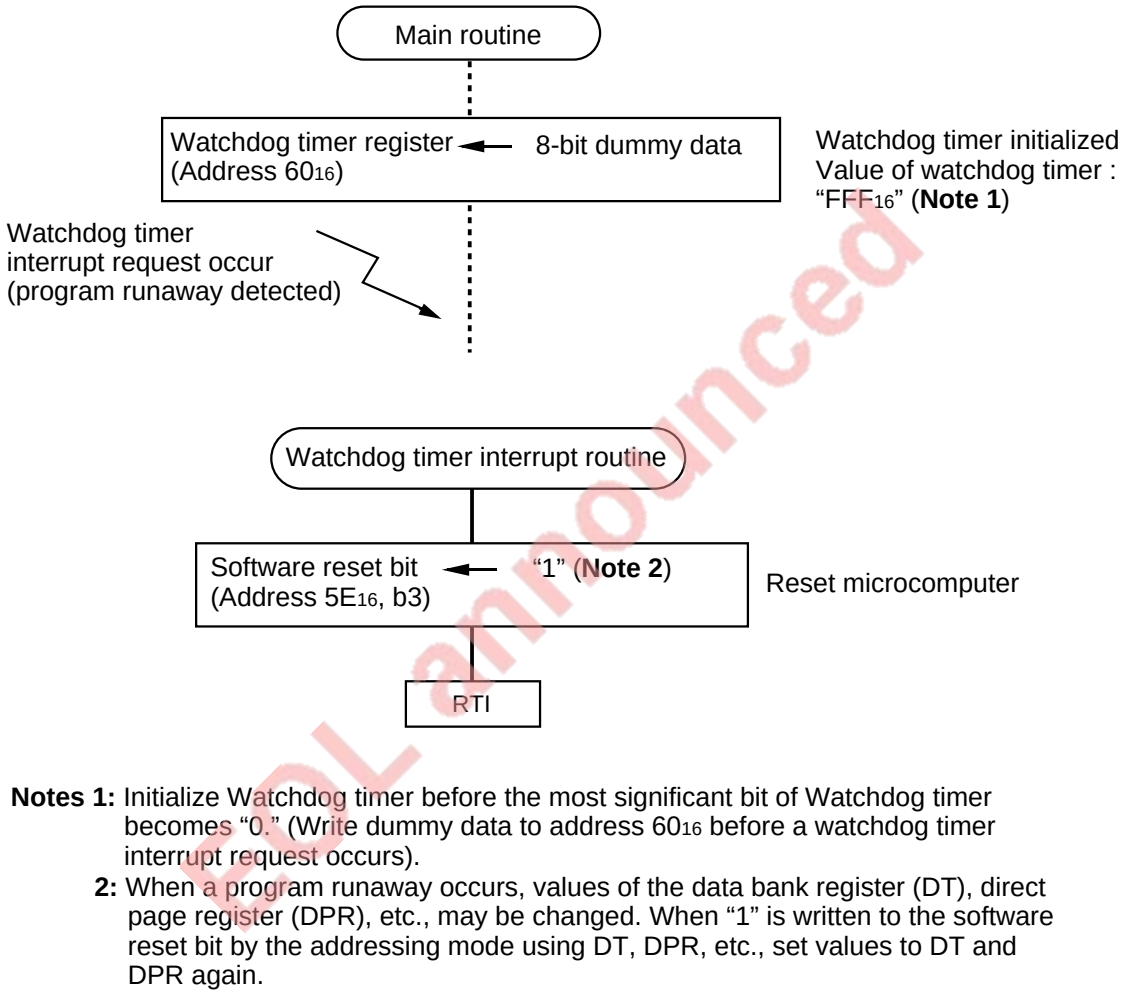
The watchdog timer interrupt is a non-maskable interrupt. When the watchdog timer interrupt request is accepted, the processor interrupt priority level (IPL) is set to "111_2."

Table 15.2.1 Occurrence interval of watchdog timer interrupt request

Watchdog timer frequency select bit	$f(X_{IN})$ = 25 MHz	
	Count source	Occurrence interval
0	f_{512}	41.94 ms
1	f_{32}	2.62 ms

EOL announced

Write dummy data to the watchdog timer register (address 60_{16}) before the most significant bit of Watchdog timer becomes "0." When Watchdog timer is used to detect a program runaway, a watchdog timer interrupt request occurs if writing to address 60_{16} is not performed owing to a program runaway and the most significant bit of Watchdog timer becomes "0." This means that a program runaway has occurred.
In order to reset the microcomputer when a program runaway is detected, write "1" to the software reset bit (bit 3 at address $5E_{16}$) in the watchdog timer interrupt routine.



EOL announced

Notes 1: Initialize Watchdog timer before the most significant bit of Watchdog timer becomes "0." (Write dummy data to address 60_{16} before a watchdog timer interrupt request occurs).
2: When a program runaway occurs, values of the data bank register (DT), direct page register (DPR), etc., may be changed. When "1" is written to the software reset bit by the addressing mode using DT, DPR, etc., set values to DT and DPR again.

Fig. 15.2.1 Example of program runaway detection by Watchdog timer

15.2 Operation description

15.2.2 Stop period

Watchdog timer stops operation in the following period:

① Hold state (Refer to section **"3.4 Hold function."**)
② During DMAC operation (Refer to **"CHAPTER 13. DMA CONTROLLER."**)
③ During DRAM refresh (Refer to **"CHAPTER 14. DRAM CONTROLLER."**)
④ Stop mode

When states ① to ③ are terminated, Watchdog timer restarts counting from the state before it stops operation. For Watchdog timer's operation when state ④ is terminated, refer to section **"15.2.3 Operation in Stop mode."**

15.2.3 Operation in Stop mode

In Stop mode, Watchdog timer stops operation. Immediately after Stop mode is terminated, Watchdog timer operates as follows. (Refer to section **"5.3 Stop mode."**)

(1) When Stop mode is terminated by hardware reset

Supply of ϕ and ϕ_{CPU} starts immediately after Stop mode is terminated, and the microcomputer performs "operation after reset." (Refer to **"CHAPTER 4. RESET."**) The watchdog timer frequency select bit becomes "0," and Watchdog timer starts counting of f_{512} from "FFF_{16}."

(2) When Stop mode is terminated by interrupt request occurrence

Immediately after Stop mode is terminated, Watchdog timer starts counting of f_{32} from "FFF_{16}" regardless of the contents of watchdog timer frequency select bit (bit 0 at address 61_{16}). Supply of ϕ and ϕ_{CPU} starts when Watchdog timer's most significant bit becomes "0." (At this time, a watchdog timer interrupt request does not occur.)

When supply of ϕ_{CPU} starts, the microcomputer executes the routine of the interrupt which is used to terminate Stop mode. Watchdog timer restarts counting of the count source (f_{32} or f_{512}), which was counted immediately before executing the **STP** instruction, from "FFF_{16}."

EOL announced

15.3 Precautions for Watchdog timer

1. When dummy data is written to address 60_{16} with the 16-bit data length, writing to address 61_{16} is simultaneously performed. Accordingly, when the user does not want to change a value of the watchdog timer frequency select bit (bit 0 at address 61_{16}), write the previous value to the bit simultaneously with writing to address 60_{16}.

2. When the **STP** instruction is executed, Watchdog timer stops. (Refer to section **"5.3 Stop mode."**)

3. Watchdog timer stops during DRAM refresh, hold state, and DMAC operation. (For Watchdog timer's structure, refer to **"Figure 15.1.1."**) Accordingly, when a bus request is changed in the period which is shorter than 1 cycle of the count source **(Note)**, Watchdog timer's count may gain. (Refer to **"Figure 15.3.1."**)

Note: f_{32} or f_{512}, which is selected by the watchdog timer frequency select bit

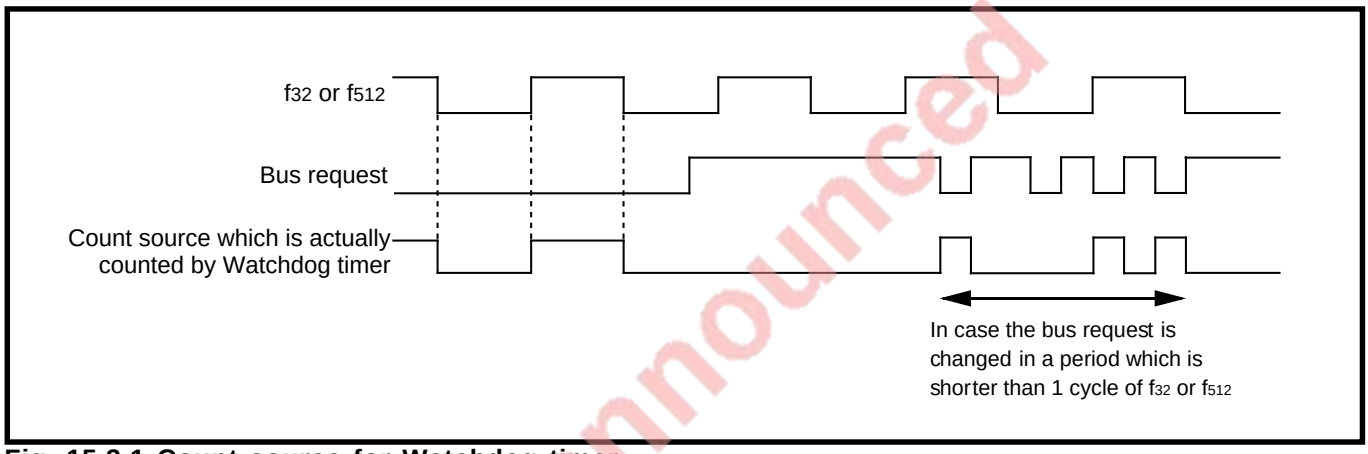


Fig. 15.3.1 Count source for Watchdog timer

EOL announced

MEMORANDUM

EOL announced

CHAPTER 16
APPLICATION

16.1 Memory connection
16.2 Examples of using DMA controller
16.3 Comparison of sample program execution rate

EOL announced

APPLICATION

16.1 Memory connection

This chapter describes application. Application shown here is just examples. The user shall modify them according to the actual application and test them.

16.1 Memory connection

This section shows examples for memory and I/O connection. Refer to **"CHAPTER 3. CONNECTION WITH EXTERNAL DEVICES"** for details about the functions and operations of used pins when connecting a memory or I/O. Refer to section **"Appendix 11. Electrical characteristics"** for timing requirements of the microcomputer.

16.1.1 Memory connection model

For the M37721, the level of the external data bus width select signal makes it possible to select the memory connection model from the four models listed in Table 16.1.1.

(1) Minimum model

This is a connection model of which external data bus width is 8 bits and access space is expanded up to 64 Kbytes. It is unnecessary to connect the address latch externally, so this model gives priority to cost and is most suitable when connecting the memory of which data bus width is 8 bits.

(2) Medium model A

This is a connection model of which external data bus width is 8 bits and access space is expanded up to 16 Mbytes. In this model, the high-order 8 bits of the external address bus (A_{16} to A_{23}) are multiplexed with the external data bus. Therefore, an n-bit ($n \leq 8$) address latch is required for latching n bits of the address in A_{16} to A_{23}.

(3) Medium model B

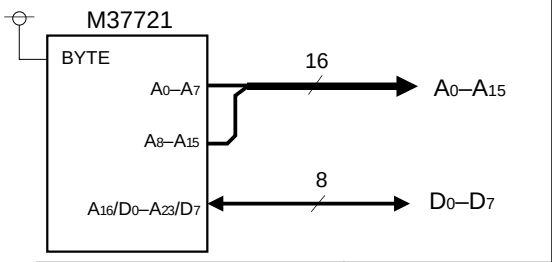
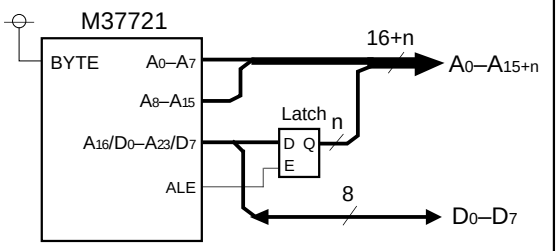
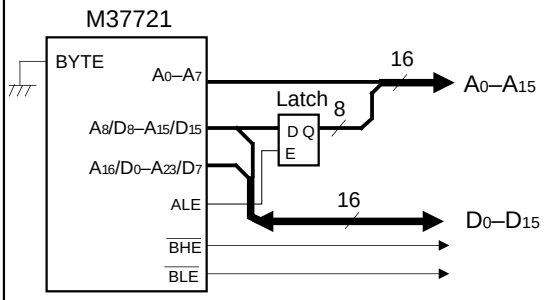
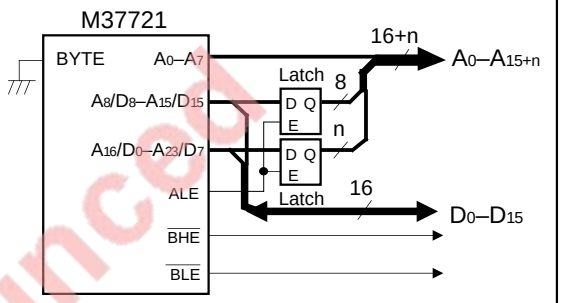
This is a connection model of which external data bus width is 16 bits and access space is expanded up to 64 Kbytes. This model gives priority to rate performance. In this model, the middle-order 8 bits of the external address bus (A_8 to A_{15}) are multiplexed with the external data bus. Therefore, an 8-bit address latch is required for latching A_8 to A_{15}.

(4) Maximum model

This is a connection model of which external data bus width is 16 bits and access space is expanded up to 16 Mbytes. In this model, the high- and middle-order 16 bits of the external address bus (A_8 to A_{23}) are multiplexed with the external data bus. Therefore, an 8-bit address latch for latching A_8 to A_{15} and an n-bit ($n \leq 8$) address latch for latching n bits of A_{16} to A_{23} are required.

EOL announced

Table 16.1.1 Memory connection model

Access space / External data bus width	Maximum 64 Kbytes	Maximum 16 Mbytes
8-bit width; BYTE = "H"	 Memory connection model: Minimum model	 Memory connection model: Medium model A
16-bit width; BYTE = "L"	 Memory connection model: Medium model B	 Memory connection model: Maximum model

Notes 1: Refer to **"CHAPTER 3. CONNECTION WITH EXTERNAL DEVICES"** for details about the functions and operations of used pins when connecting a memory. Refer to section **"Appendix 11. Electrical characteristics"** for timing requirements.

2: Because the address bus can be expanded up to 24 bits when connecting a memory, strengthen the M37721's Vss and Vcc lines on the system. (Refer to section **"Appendix 8. Countermeasure against noise."**)

EOL announced

APPLICATION

16.1 Memory connection

16.1.2 How to calculate timing

Timings at which data is read or written when connecting a memory and precautions when connecting a memory are described below.

For timing requirements of the memory and detailed account except limits described below, also refer to the memory's Data book etc. When using bus buffers, various logical circuits, etc., be sure to consider the propagation delay time etc.

(1) Timing for reading data

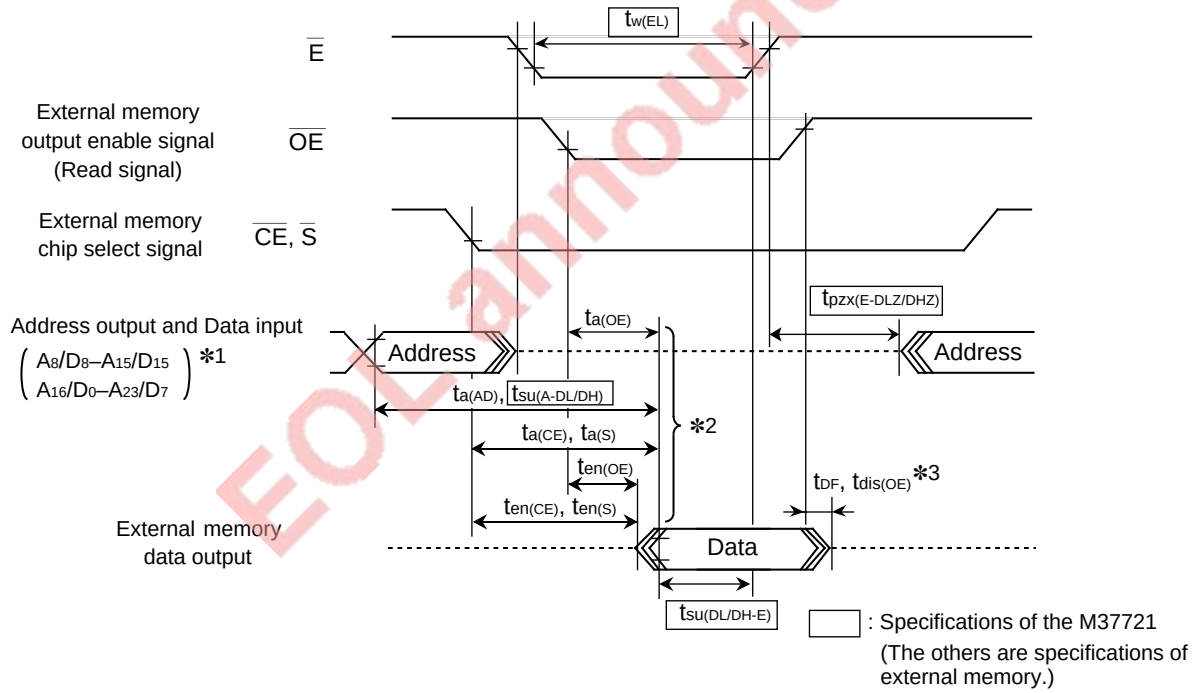
When reading data, the external data bus is placed in a floating state, and data is read from the external memory. This floating state is maintained after the falling edge of the $\overline{E}$ signal until an interval of $t_{pzx(E-DLZ/DHZ)}$ has passed after the rising edge of the $\overline{E}$ signal. Satisfy $t_{su(DL/DH-E)}$ when inputting data read from the external memory.

The following are described below:

- Timing for reading data from the flash memory, SRAM, and DRAM
- Calculation formulas for the external memory's access time, which are for $t_{su(DL/DH-E)}$ to be satisfied

The memory output enable signal ($\overline{OE}$) is assumed to be generated from the $\overline{E}$ signal.

● **Timing for reading data from flash memory and SRAM**



EOL announced

*1: This applies when the external data bus has a width of 16 bits (BYTE = "L").

*2: If data is output from the external memory before the falling edge of $\overline{E}$, there is a possibility that the tail of address collides with the head of data. → Refer to section **"(3) Precautions on memory connection."**

*3: If one of the external memory's specifications is greater than $t_{pzx(E-DLZ/DHZ)}$, there is a possibility that the tail of data collides with the head of address. → Refer to section **"(3) Precautions on memory connection."**

Note: $t_{su(A-DL/DH)}$: $t_{su(A-DL)}$ or $t_{su(A-DH)}$
$t_{pzx(E-DLZ/DHZ)}$: $t_{pzx(E-DLZ)}$ or $t_{pzx(E-DHZ)}$
$t_{su(DL/DH-E)}$: $t_{su(DL-E)}$ or $t_{su(DH-E)}$

Fig. 16.1.1 Timing for reading data from flash memory and SRAM

Address access time : $t_{a(AD)} \leq t_{su(A\text{-}DL/DH)}$ – address latch delay time*1
$\overline{OE}$ access time : $t_{a(OE)} \leq t_{w(EL)} - t_{su(DL/DH\text{-}E)}$
Chip select access time : $t_{a(S)} \leq t_{su(A\text{-}DL/DH)}$ – (address decode time*2 + address latch delay time*1)

Address latch delay time*1 : Delay time required when latching address (Unnecessary in minimum model)
Address decode time*2 : Time required for validating chip select signal after decoding address

Table 16.1.2 lists the calculation formulas and values for each parameter in Figure 16.1.1. Figure 16.1.2 shows the relationship between $t_{su(A\text{-}DL/DH)}$ and $f(X_{IN})$.

Table 16.1.2 Calculation formulas and Values for each parameter in Figure 16.1.1 (unit : ns)

	Calculation formulas and Values	
	No Wait	Wait
$t_{w(EL)}$	$\frac{2 \times 10^9}{f(X_{IN})} - 25$	$\frac{4 \times 10^9}{f(X_{IN})} - 25$
$t_{su(A\text{-}DL)}$ $t_{su(A\text{-}DH)}$	$\frac{3 \times 10^9}{f(X_{IN})} - 70$	$\frac{5 \times 10^9}{f(X_{IN})} - 70$
$t_{pzx(E\text{-}DLZ)}$ $t_{pzx(E\text{-}DHZ)}$	$\frac{1 \times 10^9}{f(X_{IN})} - 20$	
$t_{su(DL\text{-}E)}$ $t_{su(DH\text{-}E)}$	30	

EOL announced

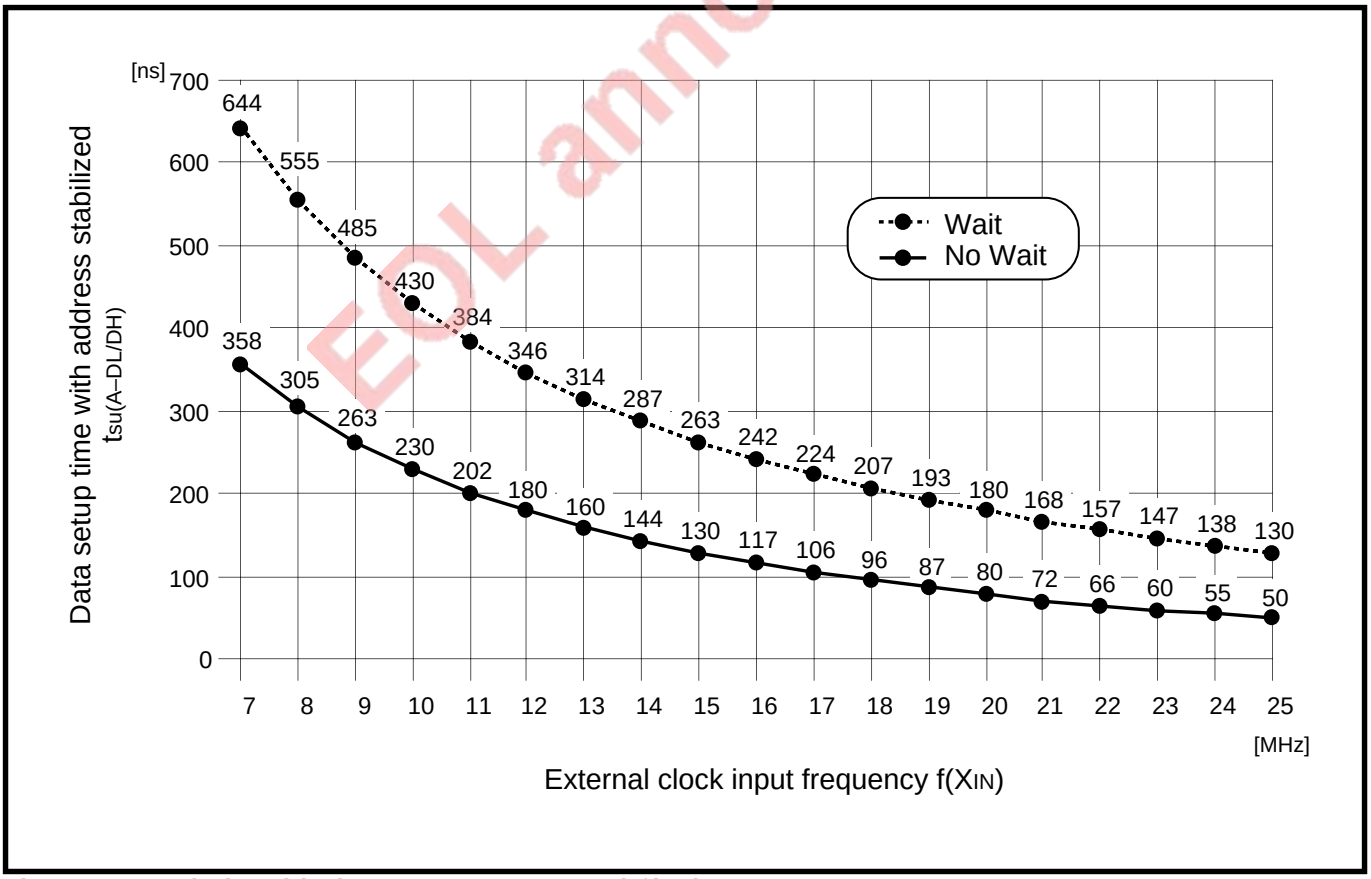


Fig. 16.1.2 Relationship between $t_{su(A\text{-}DL/DH)}$ and $f(X_{IN})$

APPLICATION

16.1 Memory connection

● **Timing for reading data from DRAM**

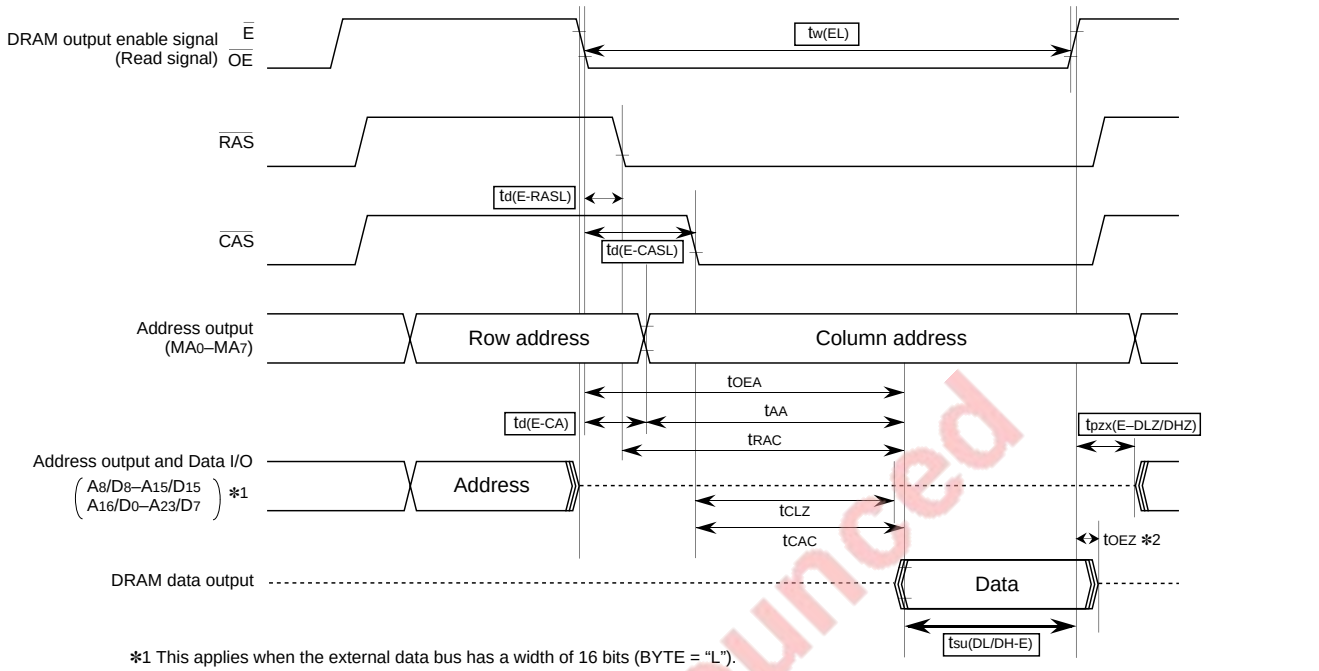


Fig. 16.1.3 Timing for reading data from DRAM

EOL announced

$\overline{CAS}$ access time : $t_{CAC} \leq t_{w(EL)} - t_{d(E\text{-}CASL)} - t_{su(DL/DH\text{-}E)}$
$\overline{RAS}$ access time : $t_{RAC} \leq t_{w(EL)} - t_{d(E\text{-}RASL)} - t_{su(DL/DH\text{-}E)}$
Column address access time : $t_{AA} \leq t_{w(EL)} - t_{d(E\text{-}CA)} - t_{su(DL/DH\text{-}E)}$
$\overline{OE}$ access time : $t_{OEA} \leq t_{w(EL)} - t_{su(DL/DH\text{-}E)}$

Table 16.1.3 lists the calculation formula and value for each parameter in Figure 16.1.3. Figure 16.1.4 shows the relationship between t_{CAC}, t_{RAC}, t_{AA} and $f(X_{IN})$.

Table 16.1.3 Calculation formula and Value for each parameter in Figure 16.1.3 (unit : ns)

	Calculation formula and Value
$t_{w(EL)}$	$\frac{4 \times 10^9}{f(X_{IN})} - 25$
$t_{d(E-RASL)}$	30
$t_{d(E-CASL)}$	$\frac{1 \times 10^9}{f(X_{IN})} + 37.5$
$t_{d(E-CA)}$	$\frac{1 \times 10^9}{f(X_{IN})} + 25$
$t_{pzx(E-DLZ)}$ $t_{pzx(E-DHZ)}$	$\frac{1 \times 10^9}{f(X_{IN})} - 20$
$t_{su(DL-E)}$ $t_{su(DH-E)}$	30

Note: When accessing DRAM, Wait is always inserted regardless of the contents of the Wait bit, source's Wait bit, and destination's Wait bit.

EOL announced

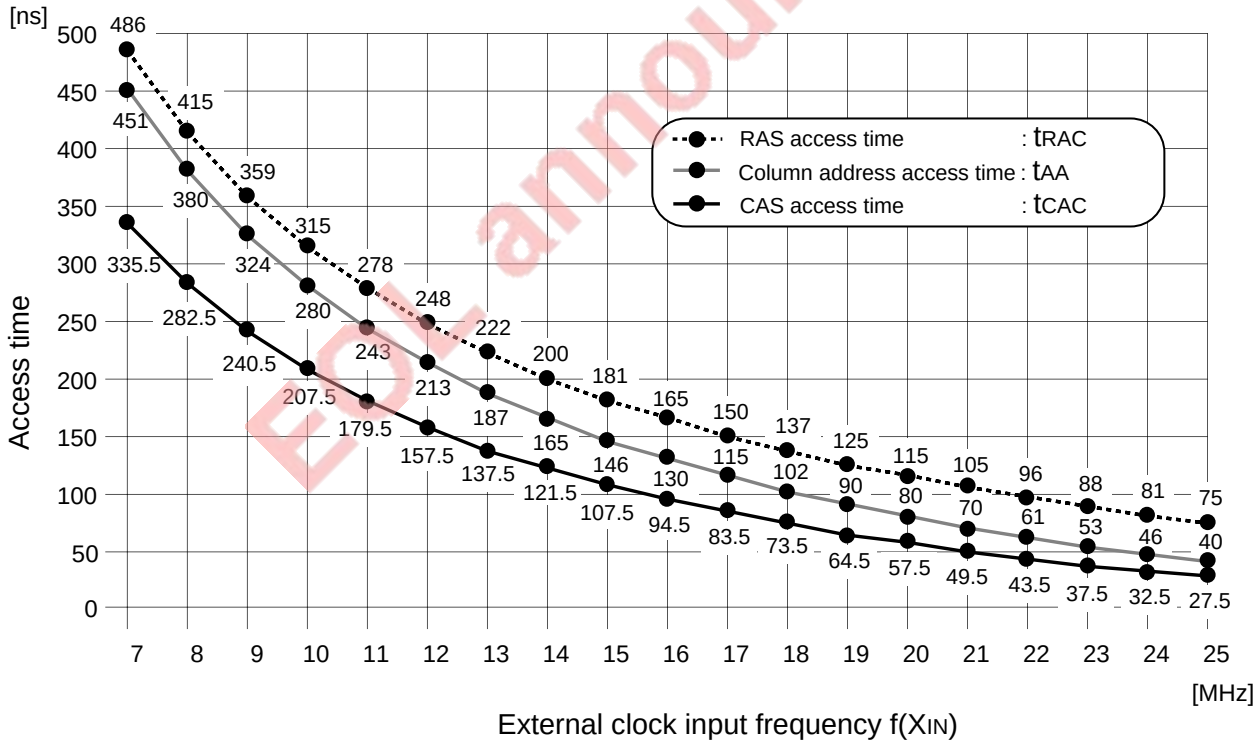


Fig. 16.1.4 Relationship between t_{CAC}, t_{RAC}, t_{AA} and $f(X_{IN})$

16.1 Memory connection

(2) Timing for writing data

When writing data, the output data is stabilized when an interval of $t_{d(E-DLQ/DHQ)}$ has passed after the falling edge of the $\overline{E}$ signal. This data is continuously output until when an interval of $t_{h(E-DLQ/DHQ)}$ has passed after the rising edge of the $\overline{E}$ signal.

Data to be written to an external memory must satisfy the data set up time ($t_{su(D)}$) (for DRAM, the data hold time (t_{DH})) of the external memory.

The following are described below:

•Timing for writing data to flash memory, SRAM, and DRAM

•Calculation formulas which are for $t_{su(D)}$ and t_{DH} to be satisfied

● **Timing for writing data to flash memory and SRAM**

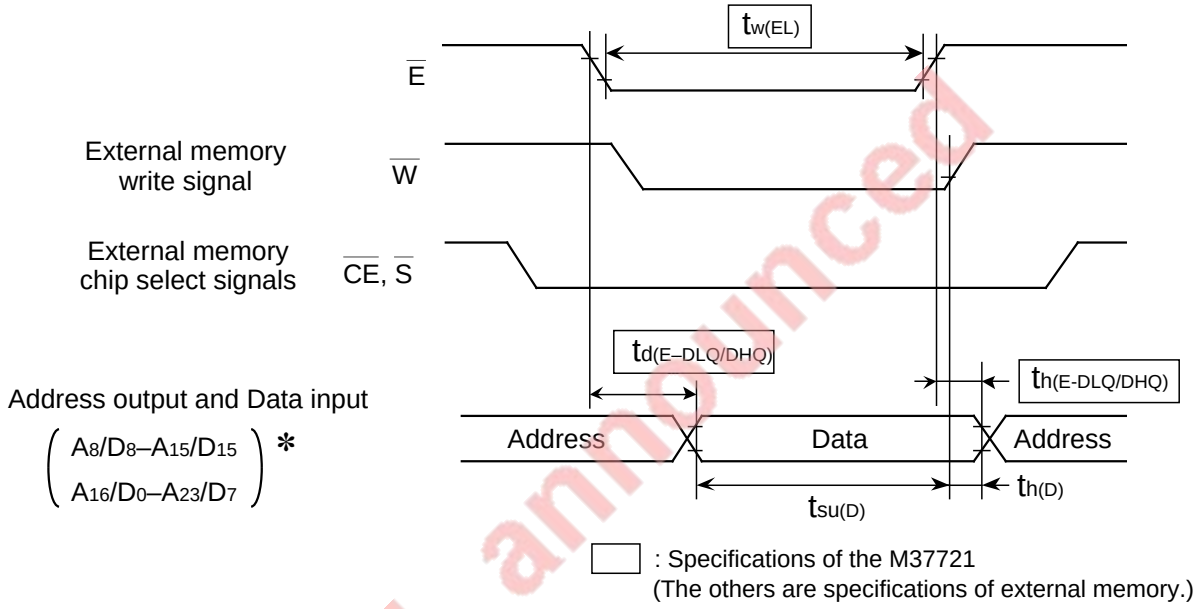


Fig. 16.1.5 Timing for writing data to flash memory and SRAM

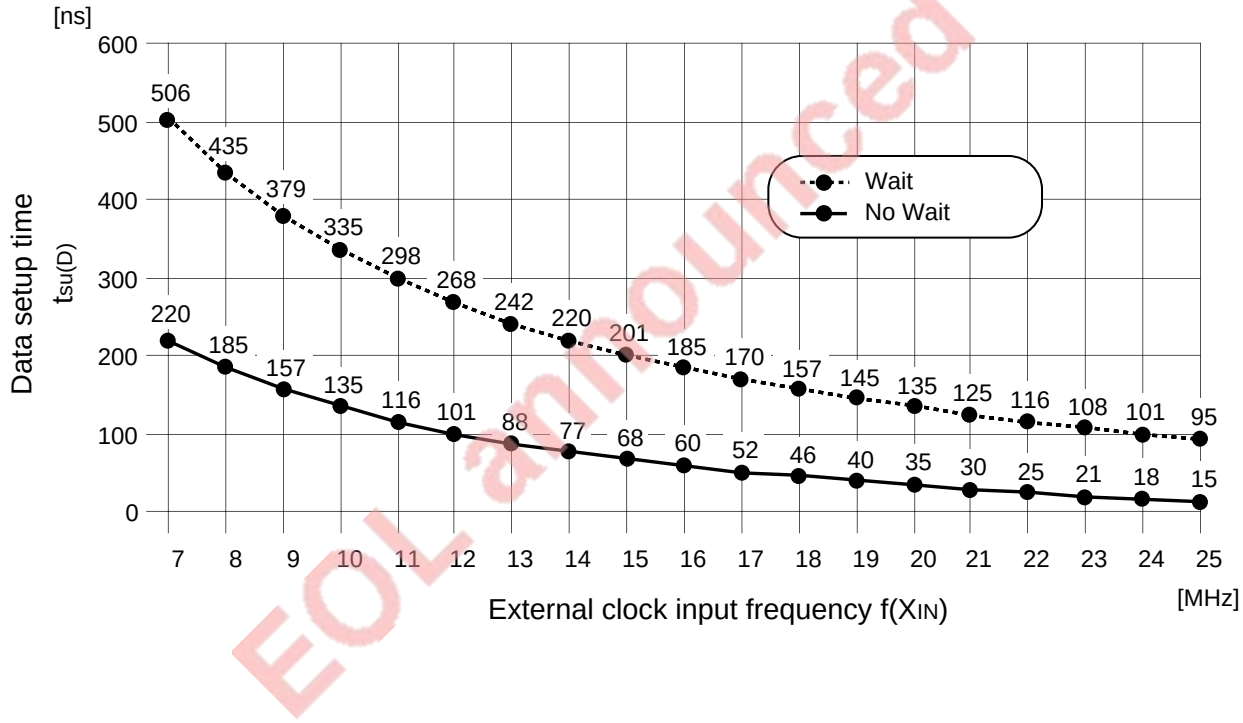
Data setup time : $t_{su(D)} \leq t_{w(EL)} - t_{d(E-DLQ/DHQ)}$

Table 16.1.4 lists the calculation formulas and values for each parameter in Figure 16.1.5, Figure 16.1.6 shows the relationship between $t_{su(D)}$ and $f(X_{IN})$.

EOL announced

Table 16.1.4 Calculation formulas and Values for each parameter in Figure 16.1.5 (unit : ns)

	Calculation formulas and Values	
	No Wait	Wait
$t_{w(EL)}$	$\frac{2 \times 10^9}{f(X_{IN})} - 25$	$\frac{4 \times 10^9}{f(X_{IN})} - 25$
$t_{d(E-DLQ)}$ $t_{d(E-DHQ)}$	35	
$t_{h(E-DLQ)}$ $t_{h(E-DHQ)}$	$\frac{1 \times 10^9}{f(X_{IN})} - 22$	



EOL announced

Fig. 16.1.6 Relationship between $t_{su(D)}$ and $f(X_{IN})$

APPLICATION

16.1 Memory connection

● **Timing for writing data to DRAM**

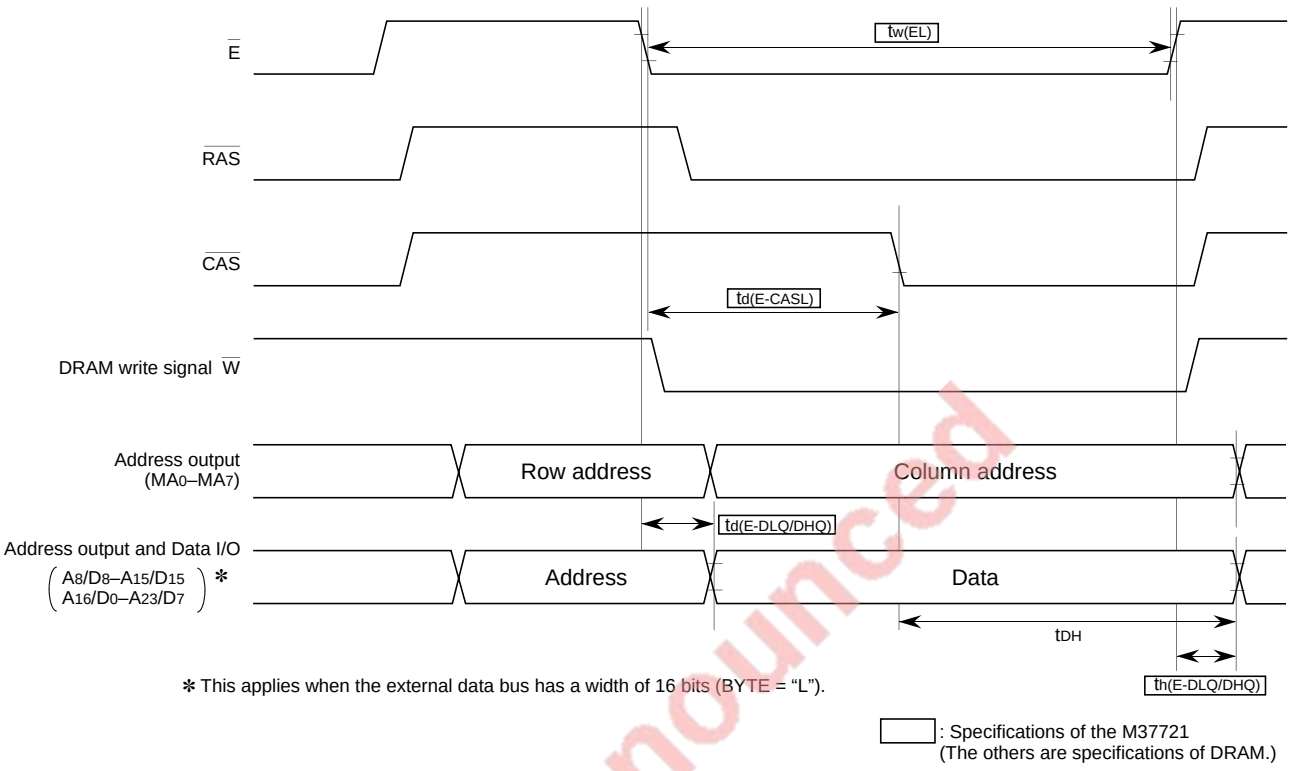


Fig. 16.1.7 Timing for writing data to DRAM

Data hold time : $t_{DH} \leq t_{w(EL)} - t_{d(E\text{-}CASL)} + t_{h(E\text{-}DLQ/DHQ)}$

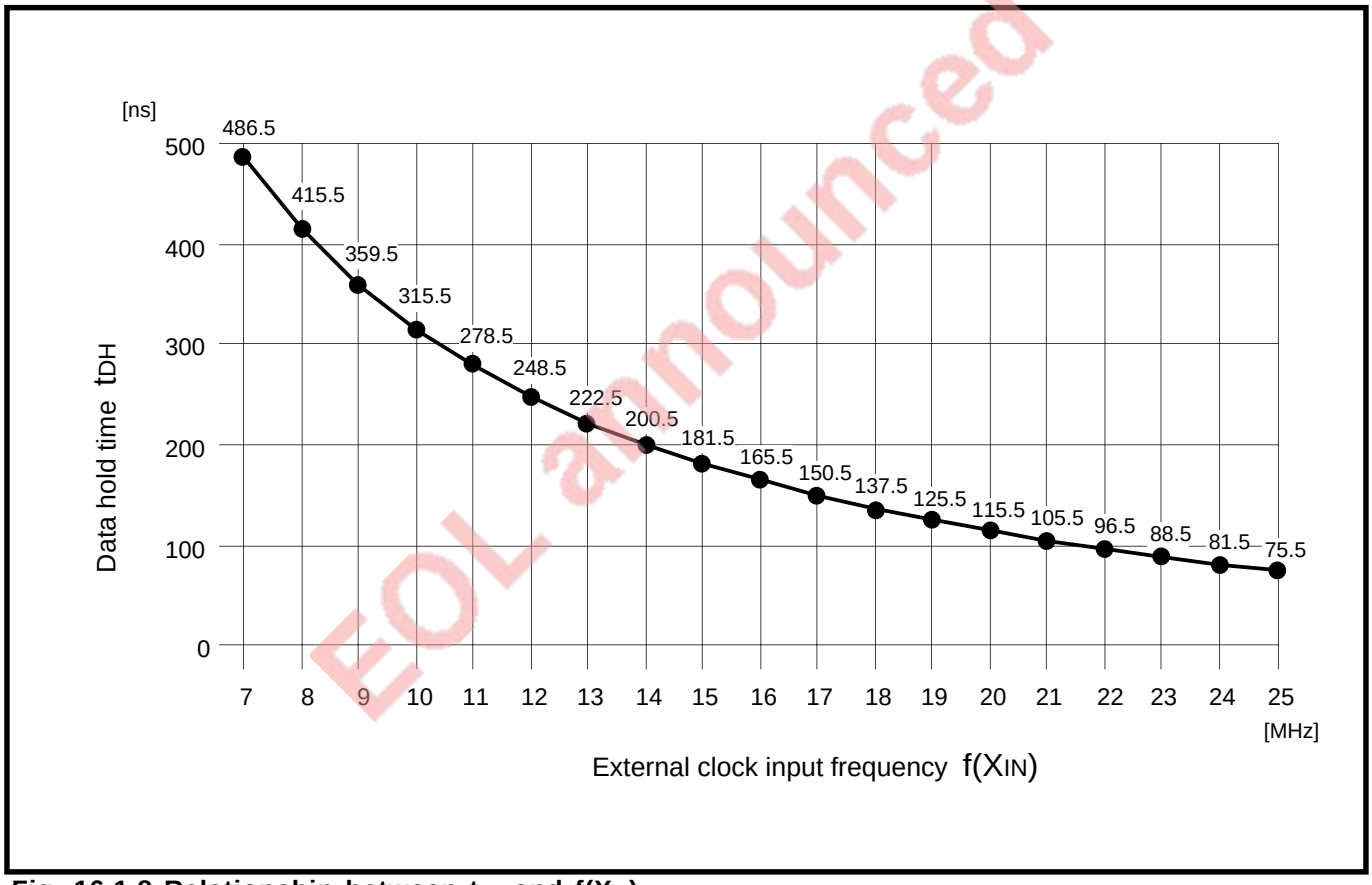
Table 16.1.5 lists the calculation formula and value for each parameter in Figure 16.1.7. Figure 16.1.8 shows the relationship between t_{DH} and $f(X_{IN})$.

EOL announced

Table 16.1.5 Calculation formula and value for each parameter in Figure 16.1.7 (unit : ns)

	Calculation formula and Value
$t_{w(EL)}$	$\frac{4 \times 10^9}{f(X_{IN})} - 25$
$t_{d(E-CASL)}$	80 to 115
$t_{d(E-DLQ)}$ $t_{d(E-DHQ)}$	35
$t_{h(E-DLQ)}$ $t_{h(E-DHQ)}$	$\frac{1 \times 10^9}{f(X_{IN})} - 22$

Note: When accessing DRAM, Wait is always inserted regardless of the contents of the Wait bit, source's Wait bit, and destination's Wait bit.



EOL announced

Fig. 16.1.8 Relationship between t_{DH} and $f(X_{IN})$

16.1 Memory connection

(3) Precautions on memory connection

As described in ① to ③ below, if specifications of the external memory do not match those of the M37721, some considerations must be incorporated into circuit design:

① When using an external memory that requires a long access time
② When data is output from an external memory before falling edge of the $\overline{E}$ signal
③ When using an external memory that outputs data for more than $t_{pzx(E\text{-}DLZ/DHZ)}$ after rising edge of the $\overline{E}$ signal

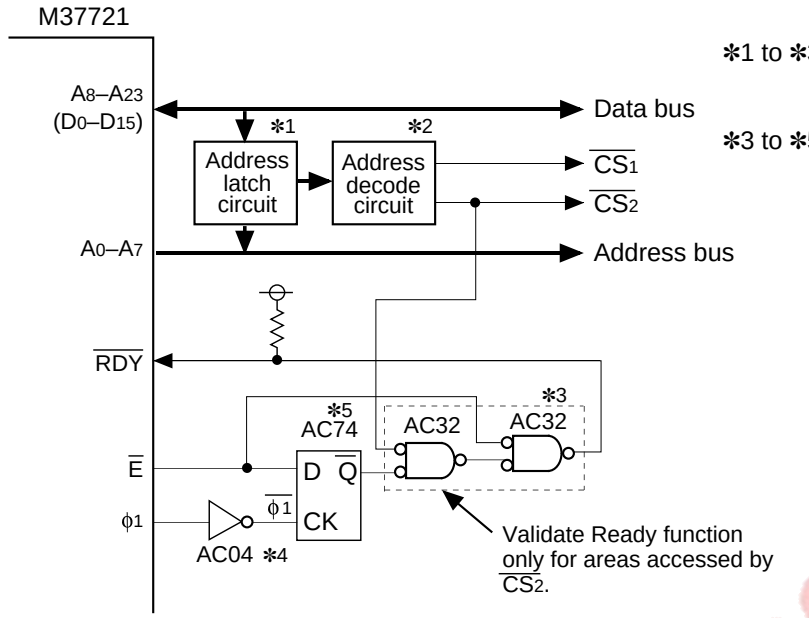
① **When using external memory that requires long access time**

If the M37721's $t_{su(DL/DH\text{-}E)}$ cannot be satisfied because the external memory requires a long access time, try to carry out the following:

- Lower f(X_{IN}).
- Select "software Wait is inserted." (Refer to section **"3.2 Software Wait."**)
- Use Ready function. (Refer to section **"3.3 Ready function."**)

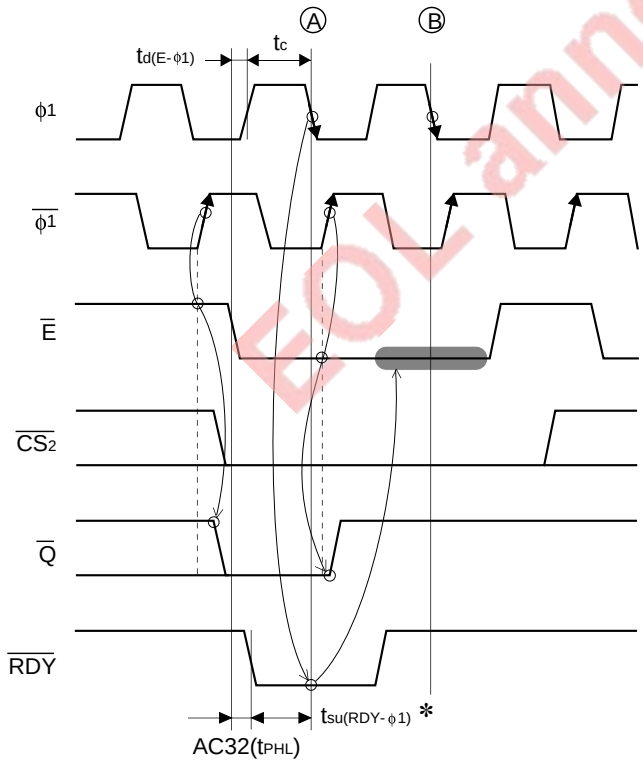
Figure 16.1.9 shows an example of using Ready function (no software Wait). Figure 16.1.10 shows an example of using Ready function (software Wait).
Ready function is valid for the internal areas, so that the circuits in Figures 16.1.9 and 16.1.10 use the chip select signal ($\overline{CS_2}$) to specify areas where Ready function is valid. In these cases, the $\overline{CS_2}$ signal is externally generated.

EOL announced



∗1 to ∗3 : Make sure that the sum of propagation delay time is within 15 ns.
∗3 to ∗5 : Make sure that the sum of propagation delay time is within 72 ns.

Circuit conditions : f(XIN) ≤ 15.7 MHz, no software Wait



Ready request is accepted at Ⓐ.
Termination request for Ready state is accepted at Ⓑ.

Ⓐ,Ⓑ : Judgement timing of RDY pin's input level

: E (“L” level) stopped by Ready function

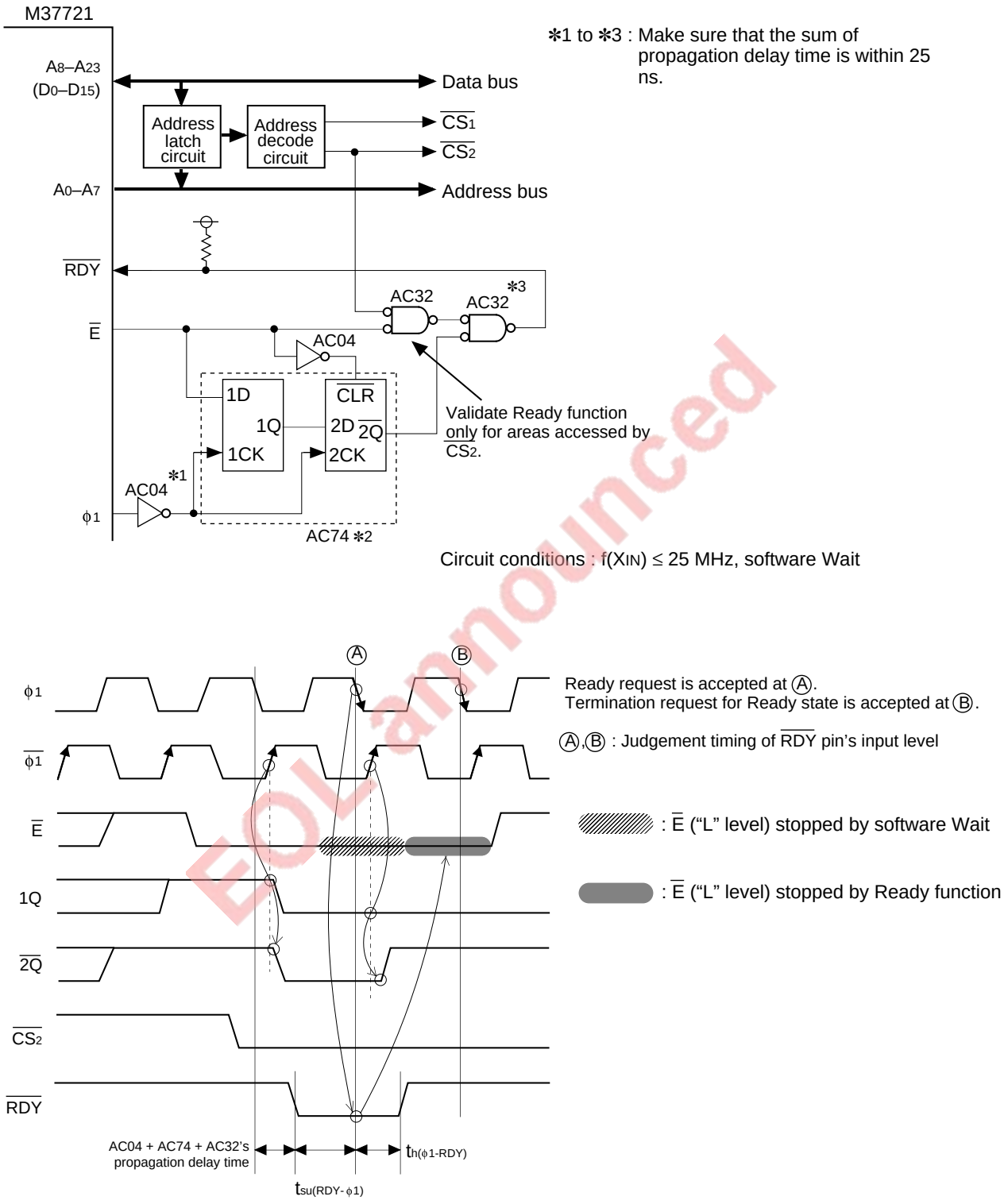
∗ The condition satisfy $t_{su(RDY-\phi1)} \geq 55$ ns is $t_c \geq 63.5$ ns. (This applies when AC32's propagation delay time is within 8.5 ns.) Accordingly, when f(XIN) ≤ 15.7 MHz, this circuit example satisfies $t_{su(RDY-\phi1)} \geq 55$ ns.

EOL announced

Fig. 16.1.9 Example of using Ready function (no software Wait)

APPLICATION

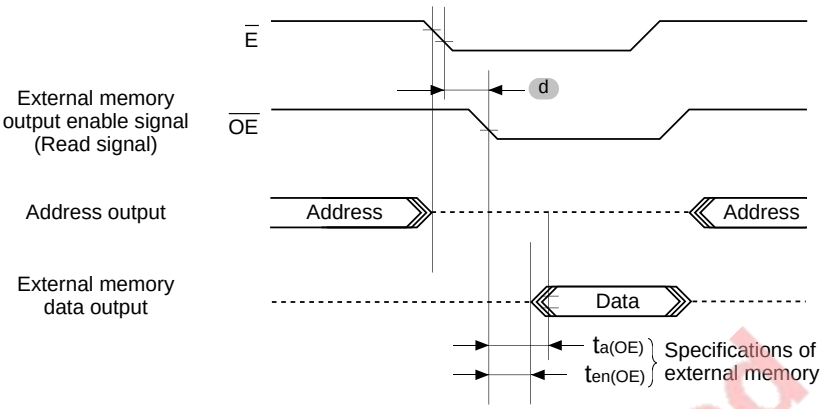
16.1 Memory connection



EOL announced

Fig. 16.1.10 Example of using Ready function (software Wait)

② **When data is output from external memory before falling edge of $\overline{E}$ signal**
Because the external memory outputs data before the falling edge of the $\overline{E}$ signal, there is a possibility that the tail of address collides with the head of data. In such a case, generate the external memory read signal ($\overline{OE}$) by using $\overline{E}$. (Refer to **"Figure 16.1.11."**)



Note: Make sure that d ≥ 0 is satisfied when generating the external memory read signal ($\overline{OE}$).

Fig. 16.1.11 Example of making data output timing delayed

③ **When using external memory that outputs data for more than $t_{pzx(E-DLZ/DHZ)}$ after rising edge of $\overline{E}$ signal**
Because the external memory outputs data for more than $t_{pzx(E-DLZ/DHZ)}$ after the rising edge of the $\overline{E}$ signal, there is a possibility that the tail of data collides with the head of address. In such a case, try to carry out the following:

- Cut the tail of data output from the memory by using, for example, a bus buffer.
- Use the Mitsubishi's memory chips that can be connected without a bus buffer.

Figures 16.1.12 to 16.1.15 show examples of using bus buffers and the timing charts. Table 16.1.6 lists the Mitsubishi's memory chips that can be connected without a bus buffer. When using one of these memory chips, timing parameters t_{DF} and $t_{dis(OE)}$ listed below are guaranteed. Accordingly, no bus buffer is necessary for the system where the external memory's read signal ($\overline{OE}$) goes high within $t_{pzx(E-DLZ/DHZ)}$-t_{DF} (or $t_{dis(OE)}$) [ns] after the rising edge of the $\overline{E}$ signal.

Table 16.1.6 Mitsubishi's memory chips that can be connected without bus buffers

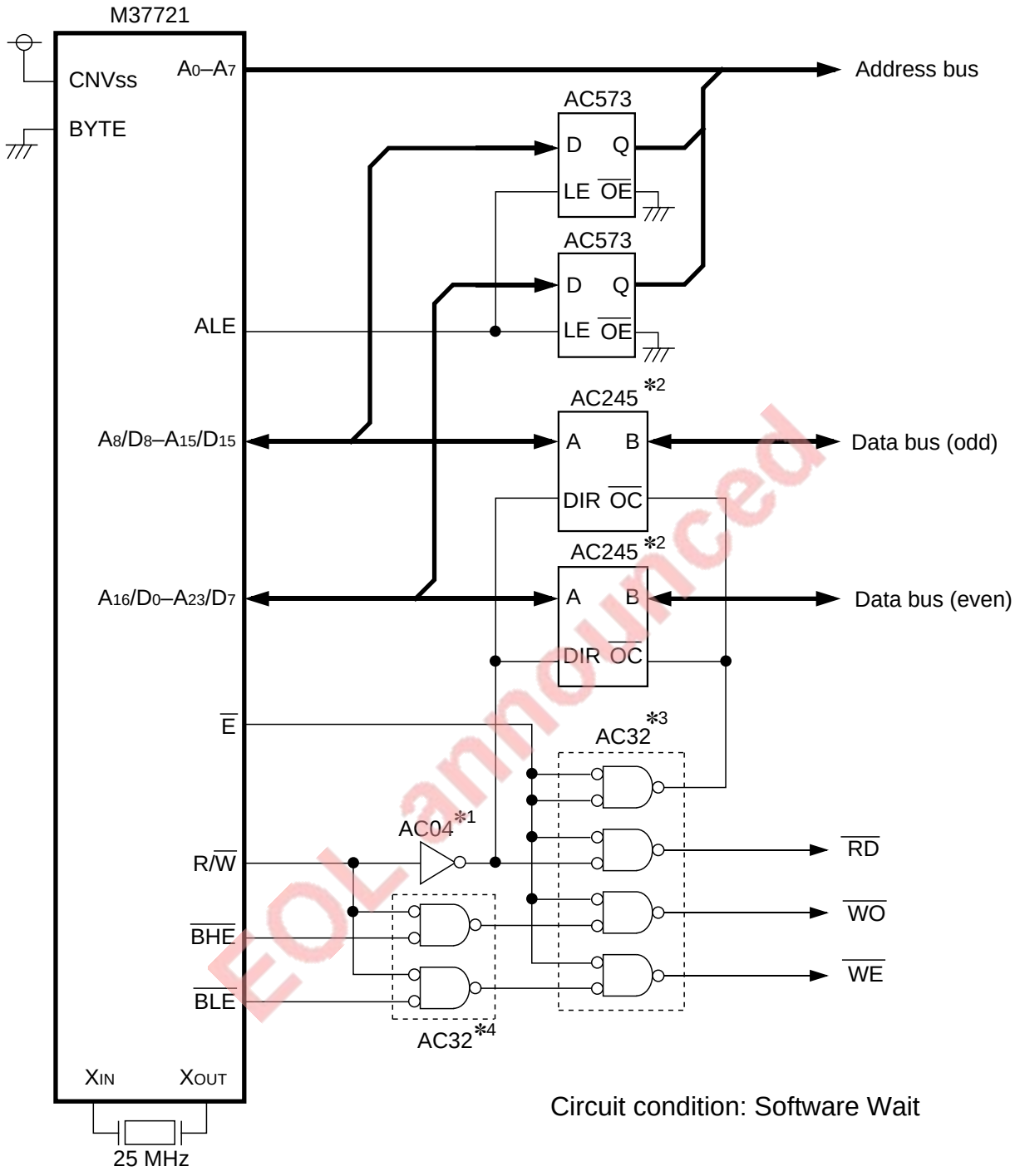
Memory	Type	$t_{DF}/t_{dis(OE)}$ (Maximum)
Flash memory	M5M28F101AP, FP, J, VP, RV-85, -10 M5M28F102AFP, J, VP-85, -10	15 ns (Guaranteed as kit.)
SRAM	M5M5256DP, FP, KP, VP, RV-45LL, -45XL, -55LL, -55XL, -70LL, -70XL	**(Note)**
	M5M5278DP, J-12	6 ns
	M5M5278DP, FP, J-15, -15L	7 ns
	M5M5278DP, FP, J-20, -20L	8 ns

Note: t_{DF} or $t_{dis(OE)}$ listed above is guaranteed when these memory chips are connected with the M37721. When the user wants specifications of these memory chips, add a comment "$t_{DF}/t_{dis(OE)}$ = 15 ns, microcomputer and kit."

EOL announced

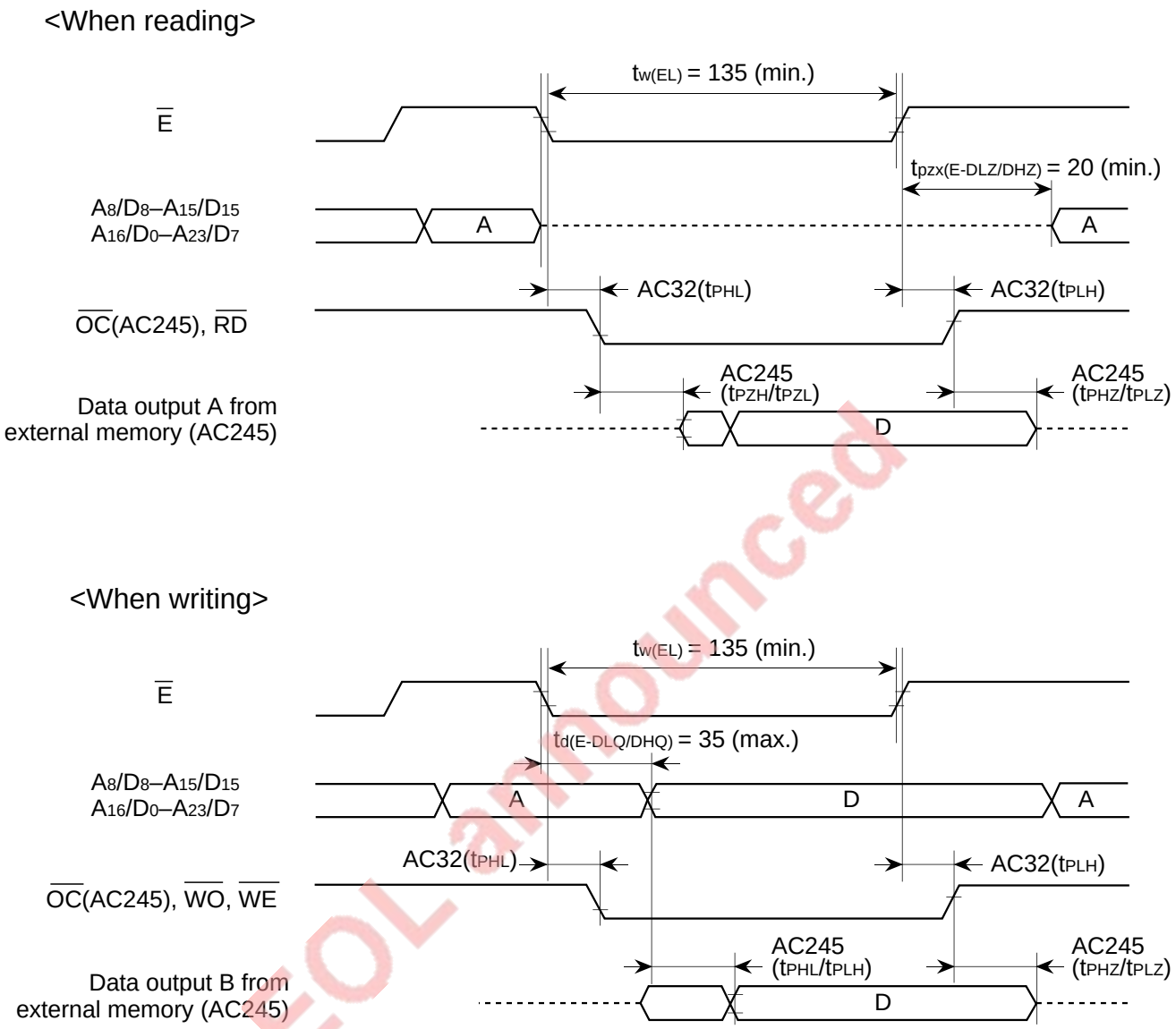
APPLICATION

16.1 Memory connection



*1: Make sure that the propagation delay time is within 20 ns.
*2, *3: Make sure that the sum of output disable time in *2 and propagation delay time in *3 is within 20 ns.
*4: Make sure that the propagation delay time is within 15 ns.

Fig. 16.1.12 Example of using bus buffers (1)

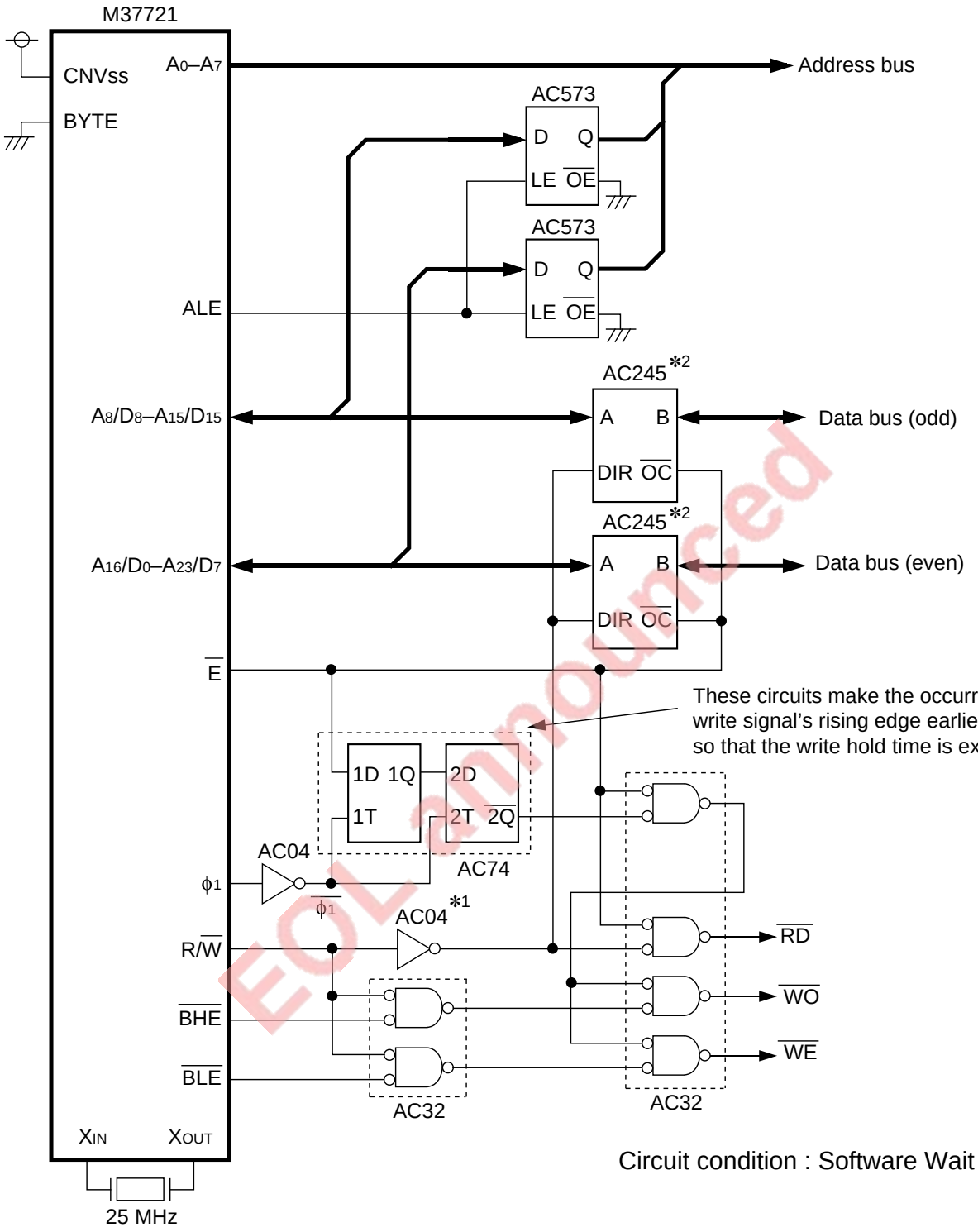


EOL announced

Fig. 16.1.13 Timing chart for circuit example using bus buffers (1)

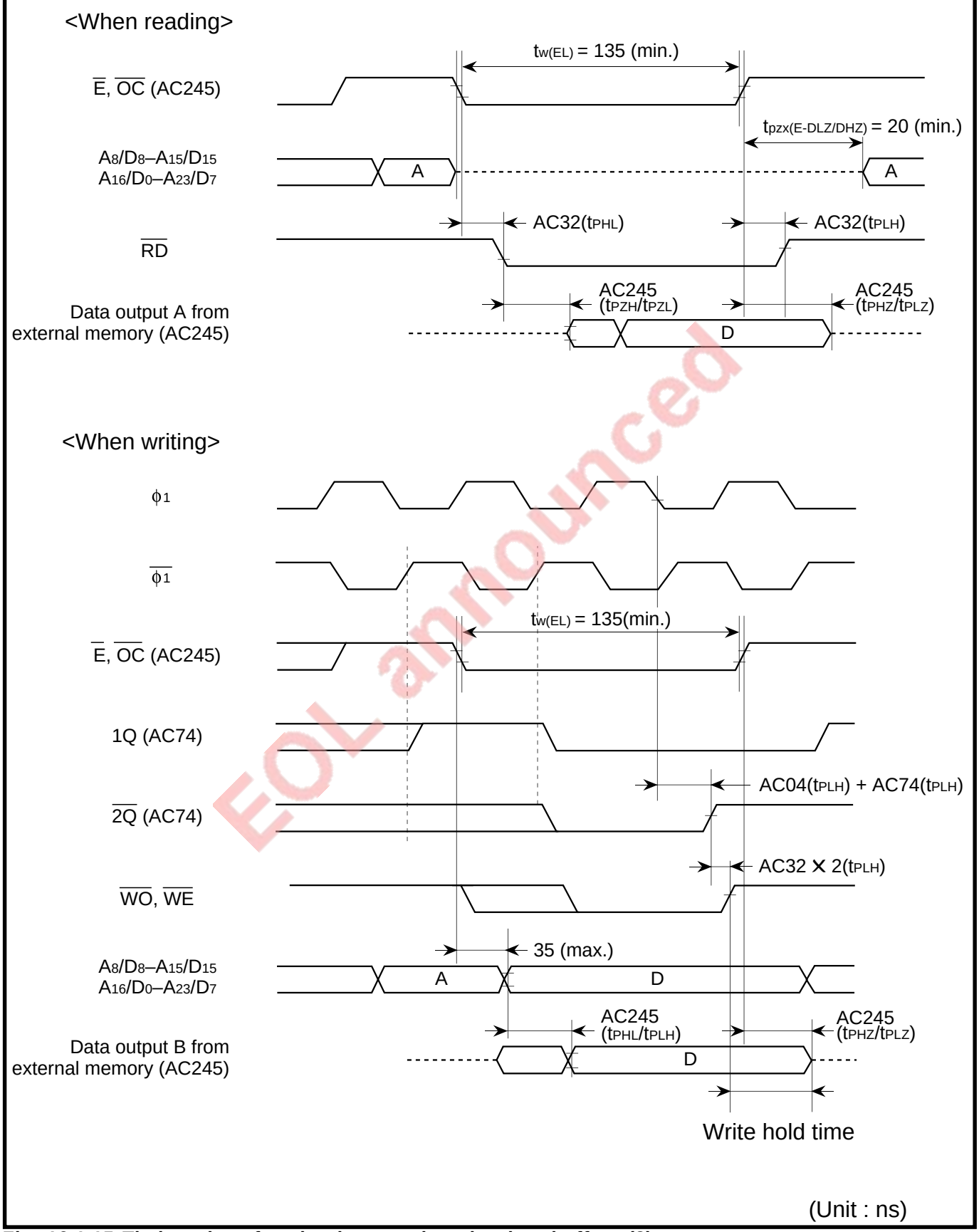
APPLICATION

16.1 Memory connection



*1: Make sure that the propagation delay time is within 20 ns.
*2: Make sure that the output disable time is within 20 ns.

Fig. 16.1.14 Example for using bus buffers (2) (connecting with memory requiring long data hold time for writing)



EOL announced

Fig. 16.1.15 Timing chart for circuit example using bus buffers (2)

16.1 Memory connection

16.1.3 Example of memory connection

Examples of the flash memory, SRAM, and DRAM connection and the timing charts are described as follows.

(1) Example of flash memory connection (minimum model)

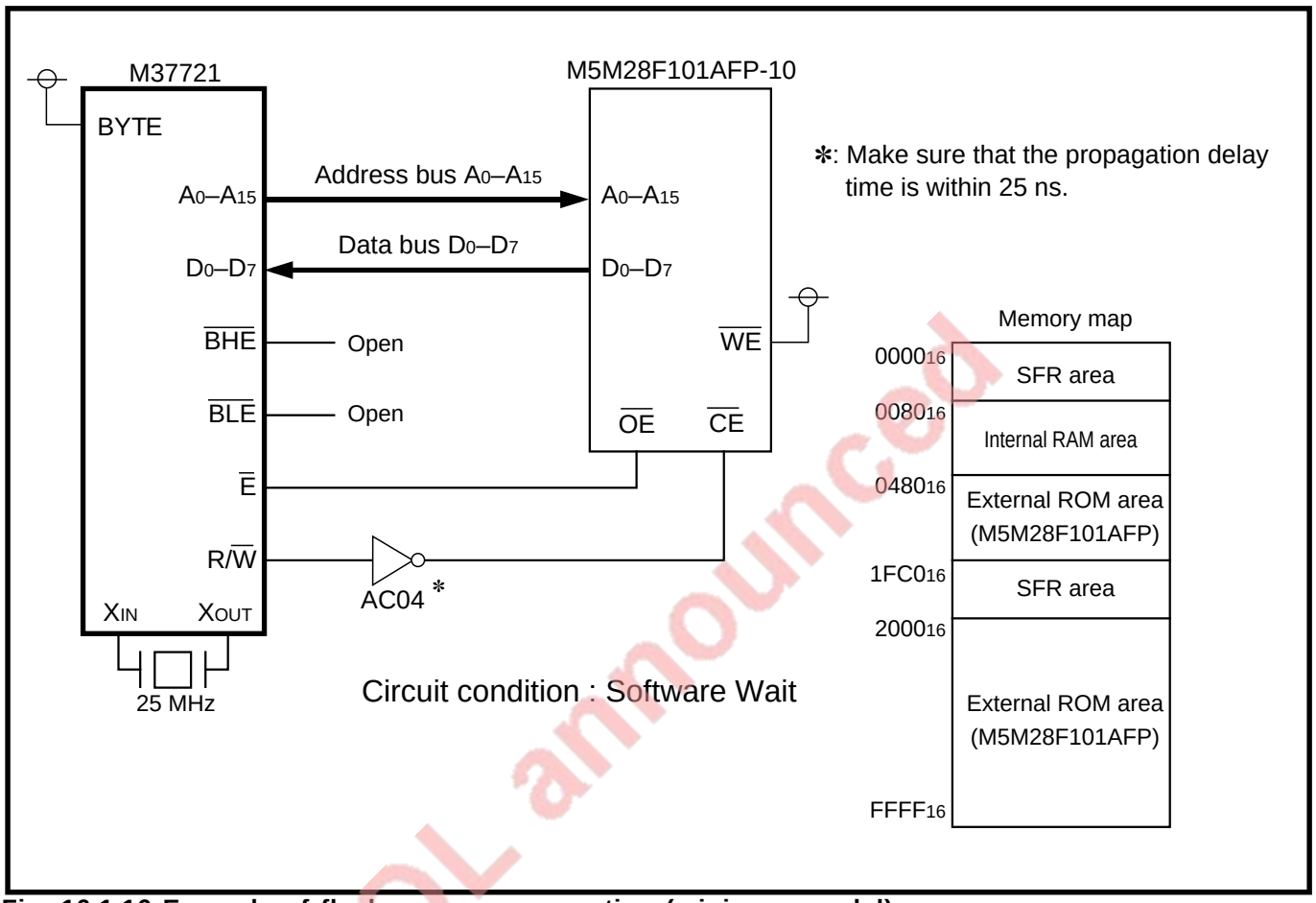


Fig. 16.1.16 Example of flash memory connection (minimum model)

EOL announced

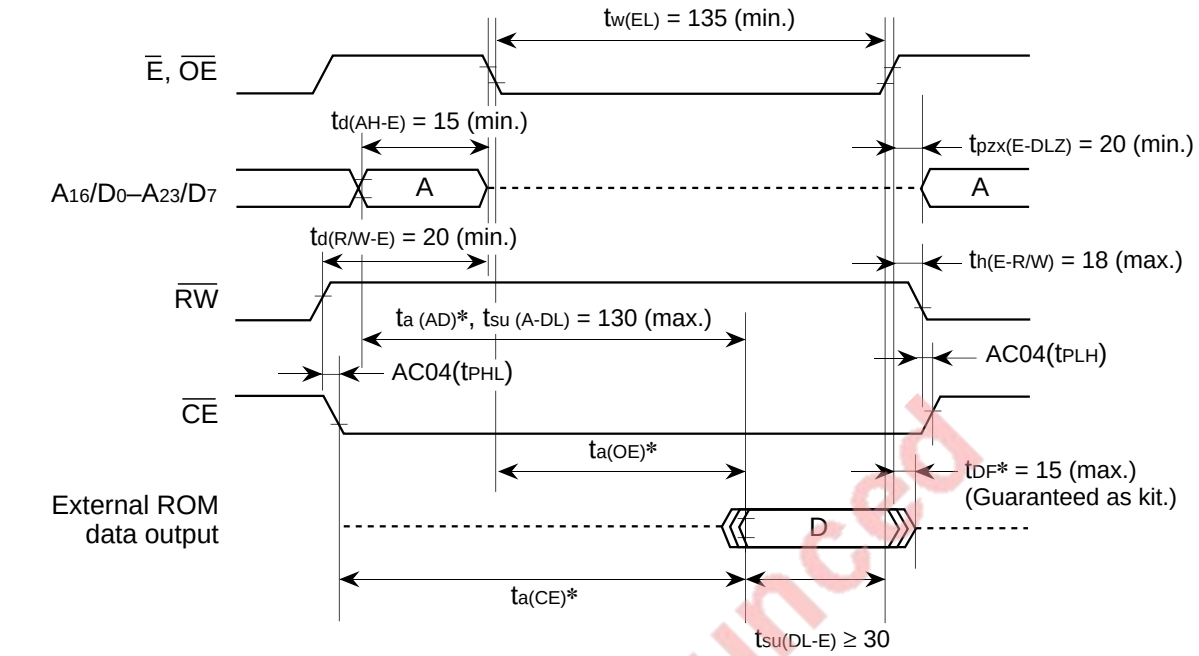


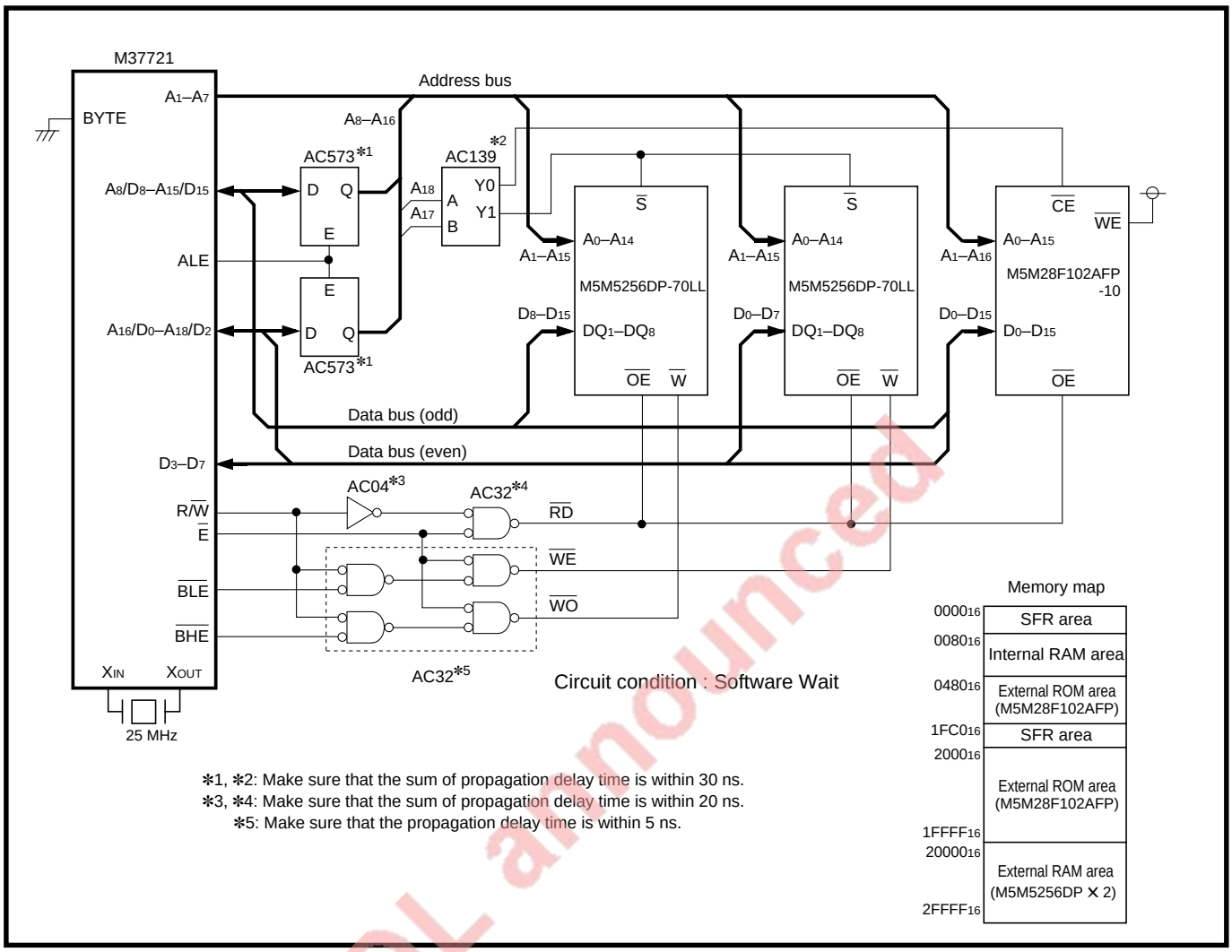
Fig. 16.1.17 Timing chart for flash memory connection example (minimum model)

EOL announced

APPLICATION

16.1 Memory connection

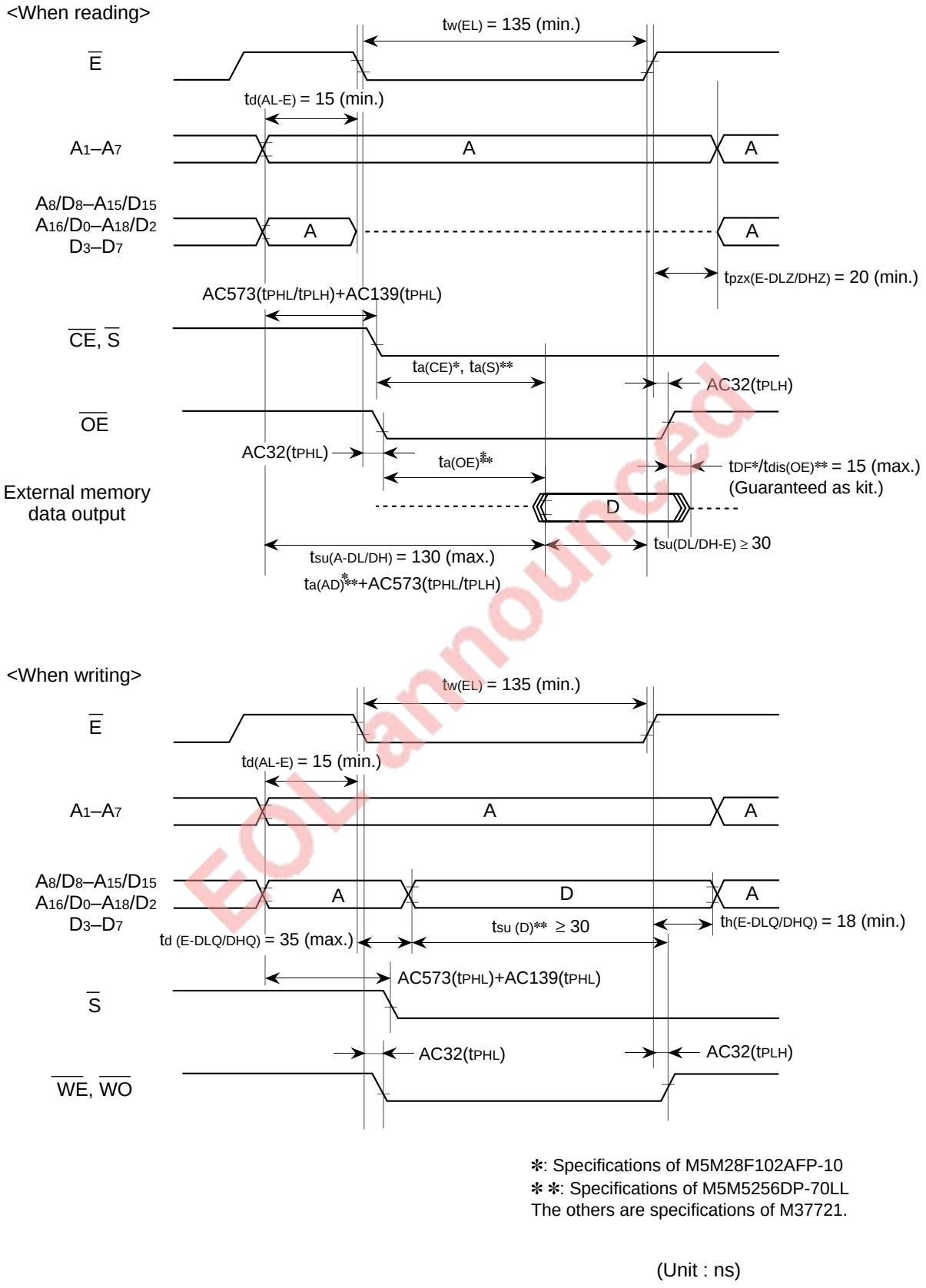
(2) Example of flash memory and SRAM connection (maximum model)



*1, *2: Make sure that the sum of propagation delay time is within 30 ns.
*3, *4: Make sure that the sum of propagation delay time is within 20 ns.
*5: Make sure that the propagation delay time is within 5 ns.

Fig. 16.1.18 Example of flash memory and SRAM connection (maximum model)

EOL announced



EOL announced

Fig. 16.1.19 Timing chart for example of flash memory and SRAM connection (maximum model)

APPLICATION

16.1 Memory connection

(3) Example of DRAM connection (external bus width = 8 bits) ①

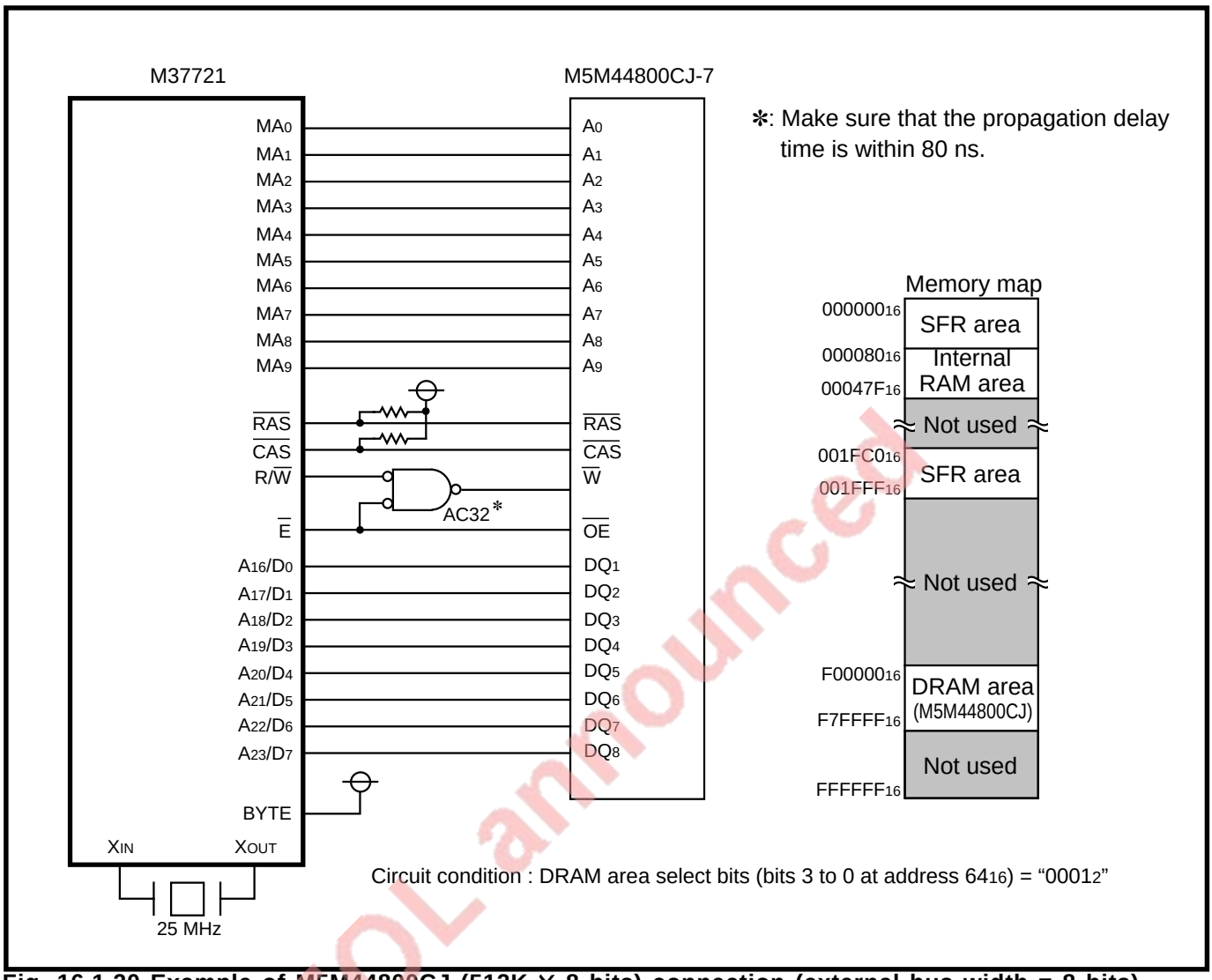
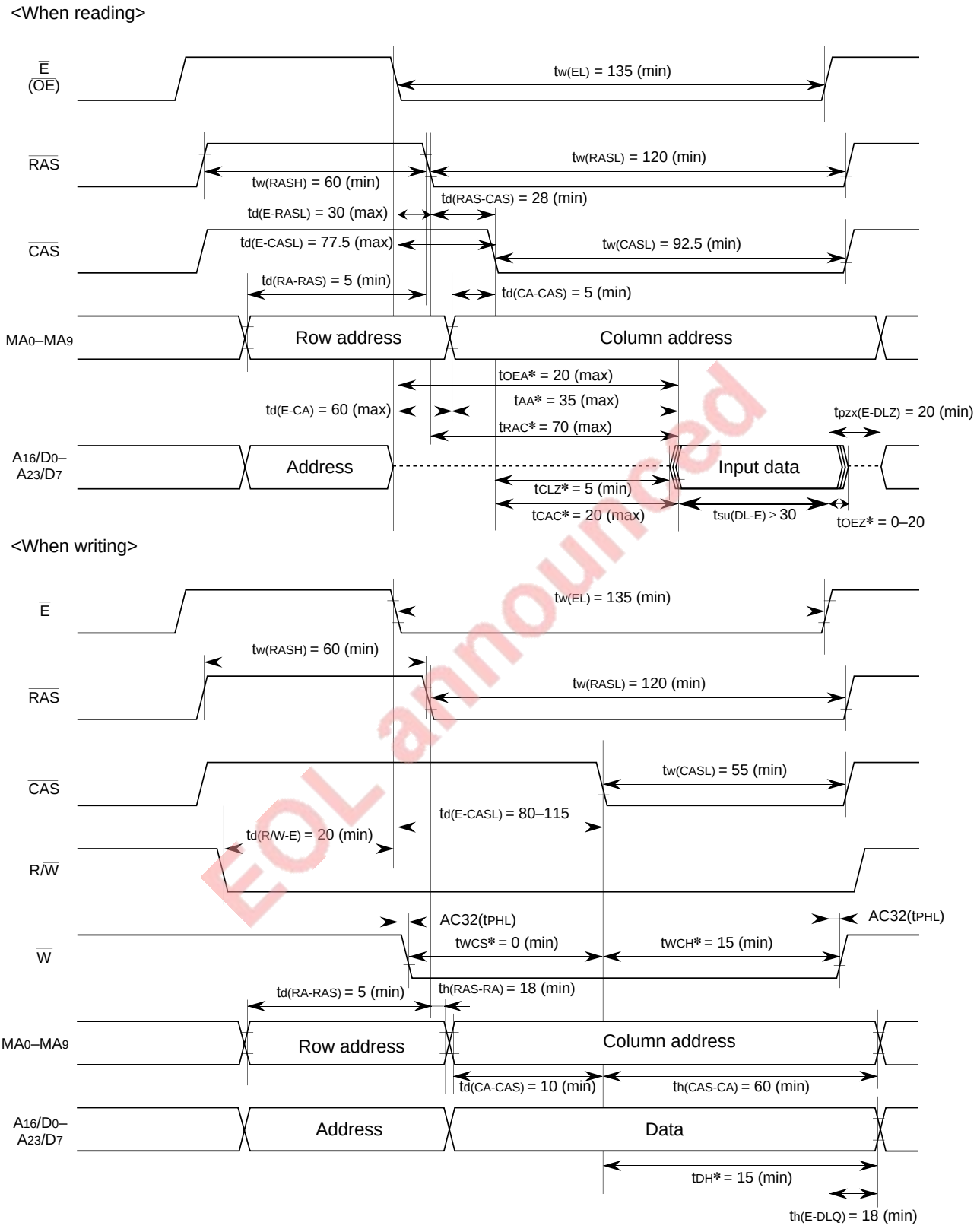


Fig. 16.1.20 Example of M5M44800CJ (512K × 8 bits) connection (external bus width = 8 bits)

EOL announced



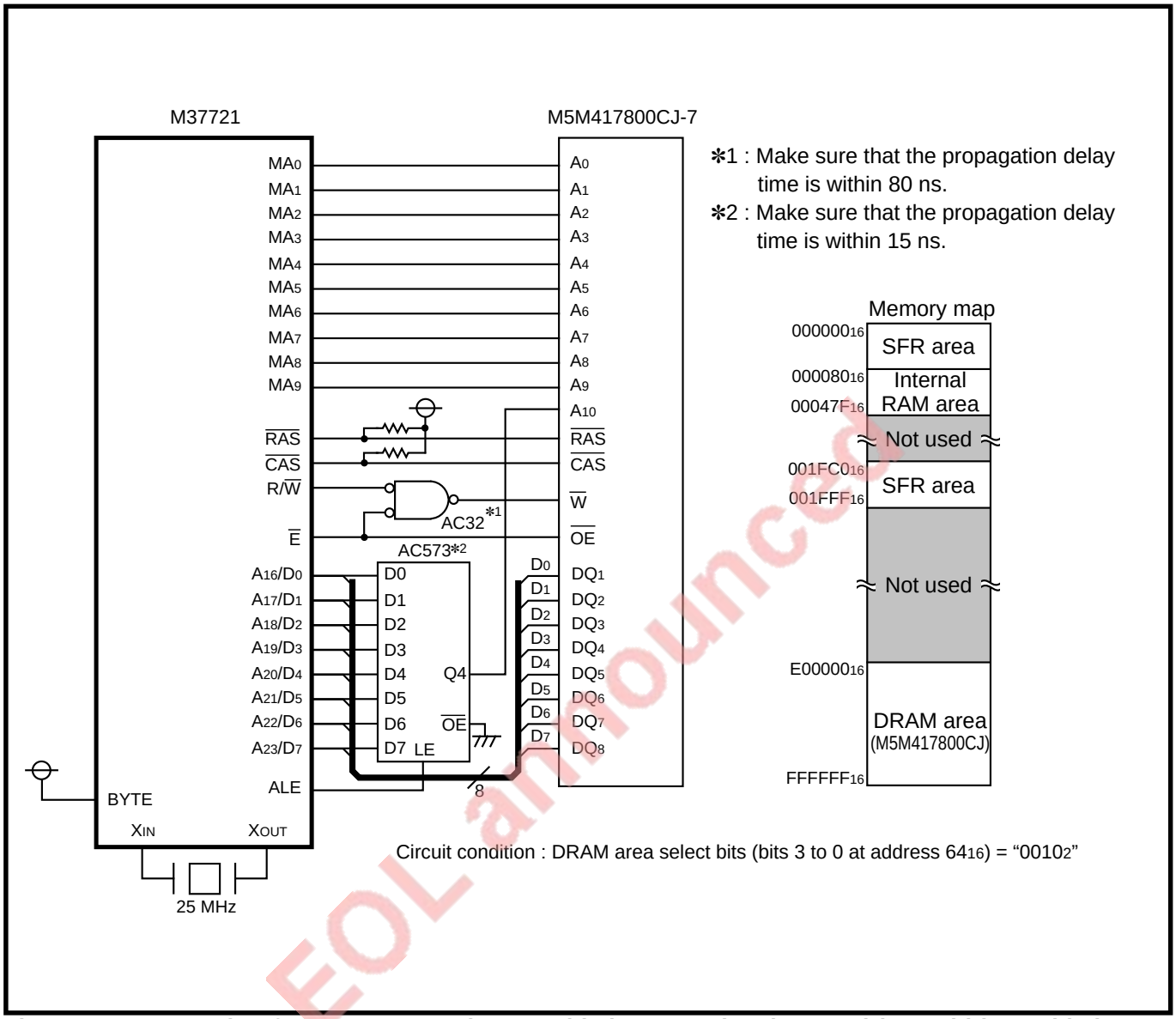
EOL announced

Fig. 16.1.21 Timing chart for example of M5M44800CJ (512K × 8 bits) connection (external bus width = 8 bits)

APPLICATION

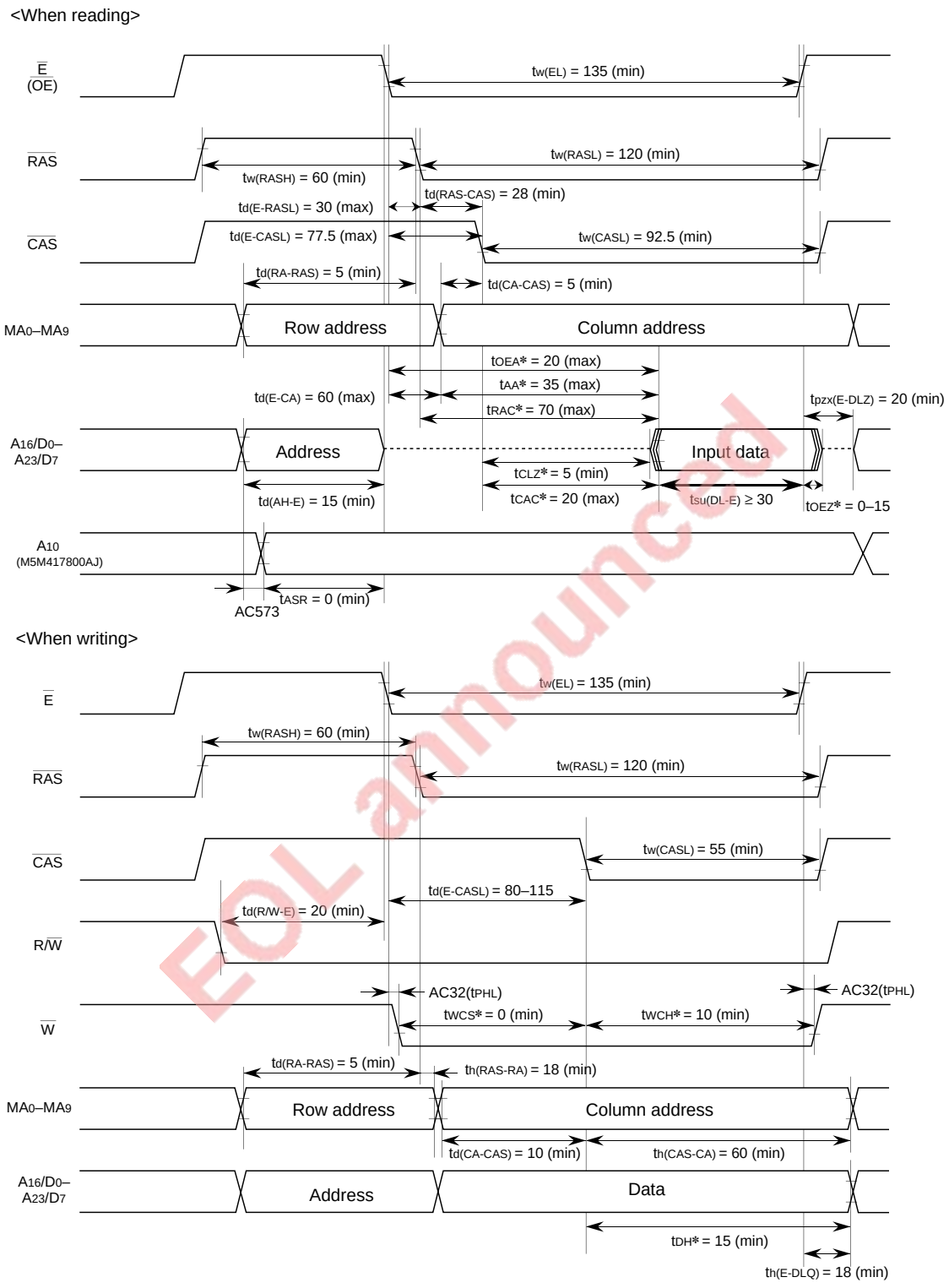
16.1 Memory connection

(4) Example of DRAM connection (external bus width = 8 bits) ②



EOL announced

Fig. 16.1.22 Example of M5M417800CJ (2M × 8 bits) connection (external bus width = 8 bits)



EOL announced

✻ : Specifications of M5M417800CJ-7
The others are specifications of M37721.

(Unit : ns)

Fig. 16.1.23 Timing chart for example of M5M417800CJ (2M × 8 bits) connection (external bus width = 8 bits)

APPLICATION

16.1 Memory connection

(5) Example of DRAM connection (external bus width = 8 bits) ③

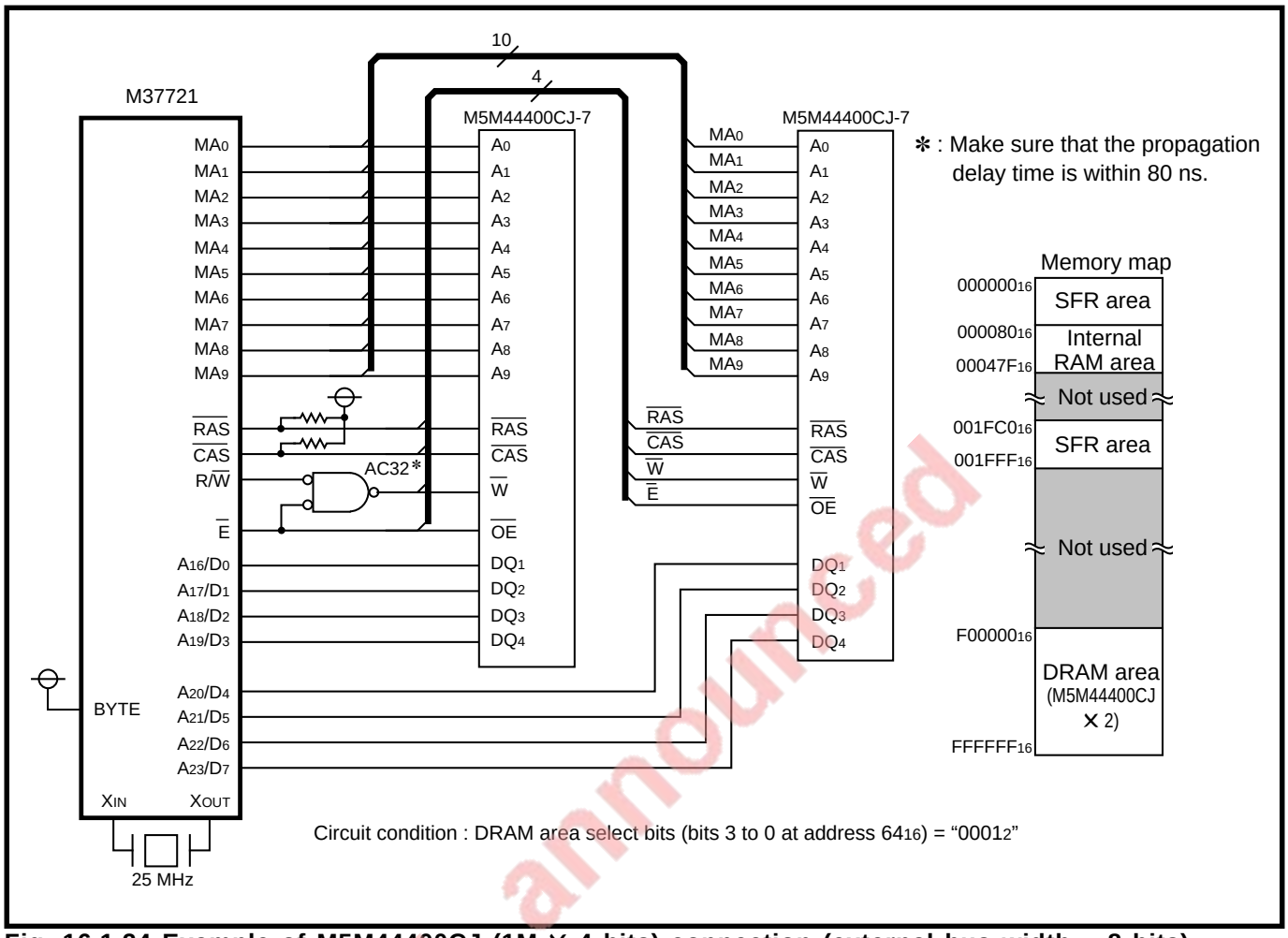
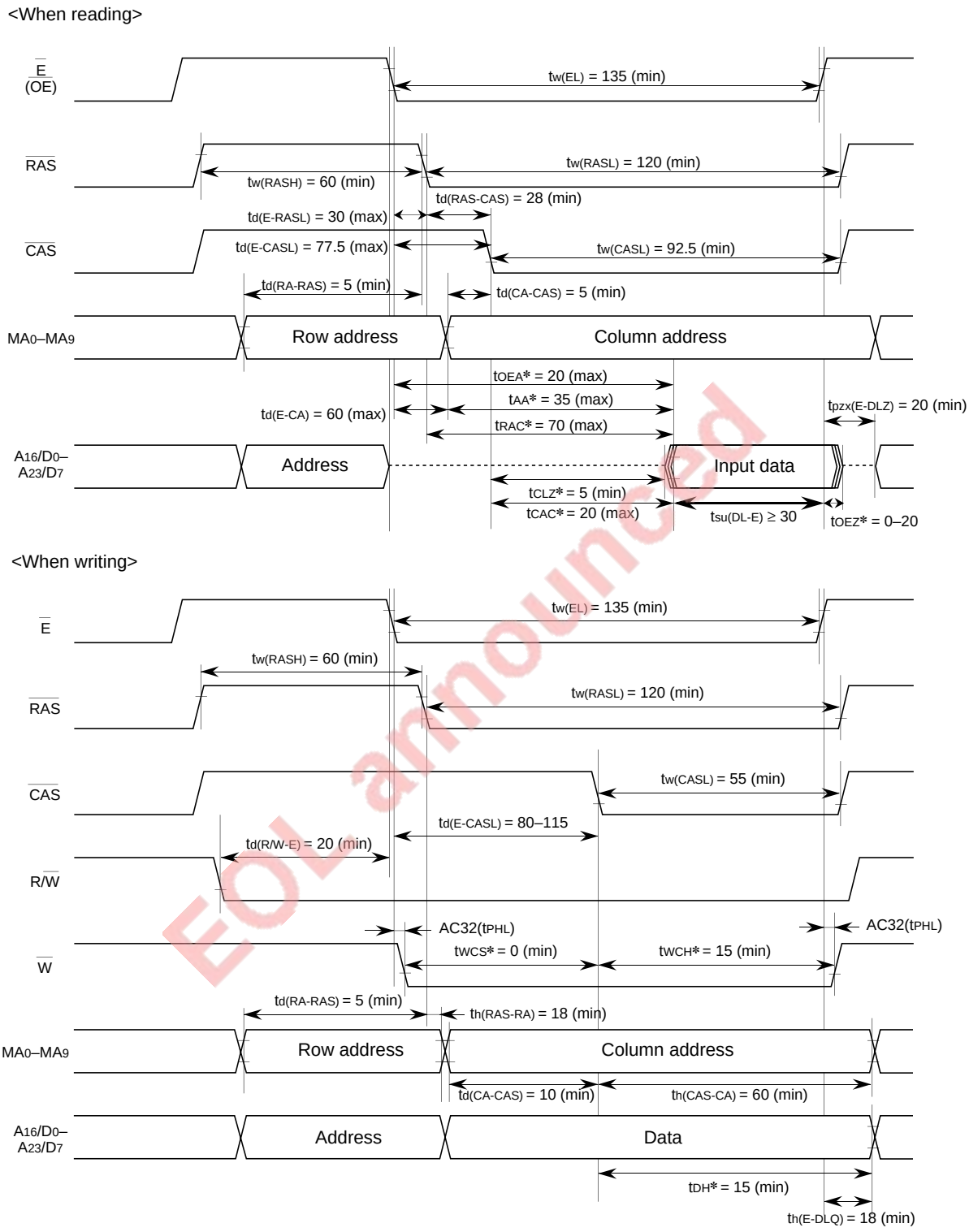


Fig. 16.1.24 Example of M5M44400CJ (1M × 4 bits) connection (external bus width = 8 bits)

EOL announced



✻ : Specifications of M5M44400CJ-7
The others are specifications of M37721.

(Unit : ns)

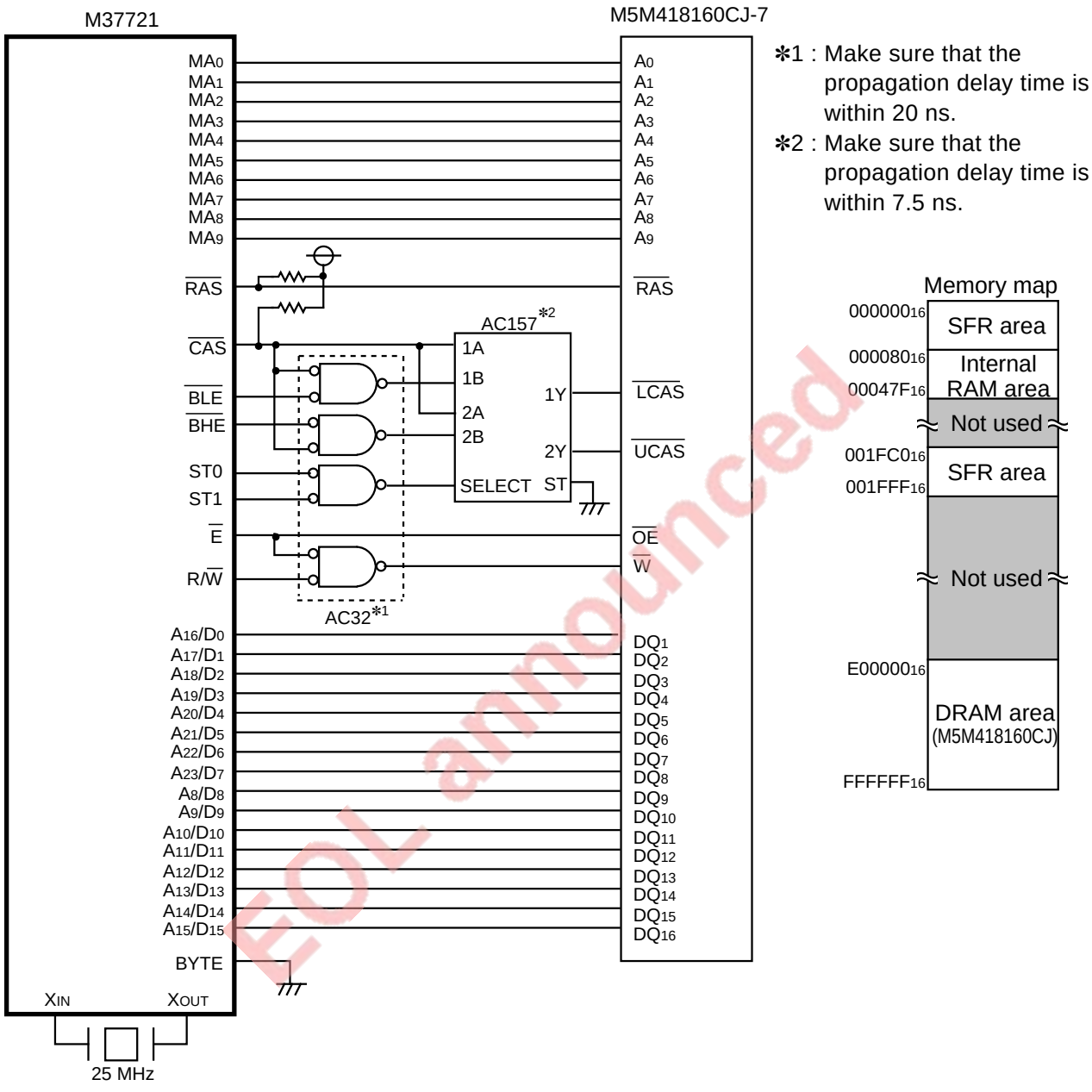
EOL announced

Fig. 16.1.25 Timing chart for example of M5M44400CJ (1M × 4 bits) connection (external bus width = 8 bits)

APPLICATION

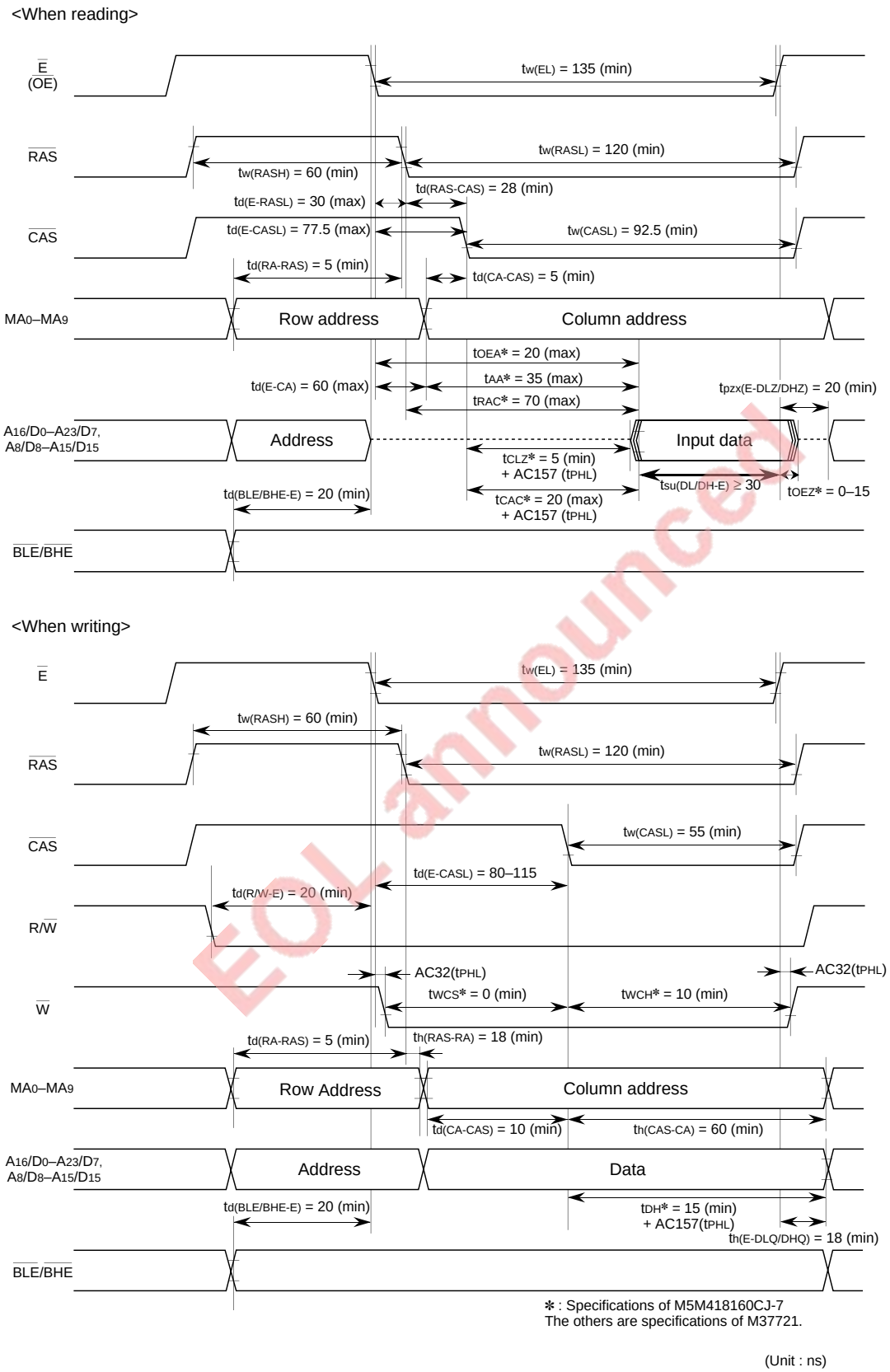
16.1 Memory connection

(6) Example of DRAM connection (external bus width = 16 bits) ①



EOL announced

Fig. 16.1.26 Example of M5M418160CJ (1M × 16 bits) connection (external bus width = 16 bits)

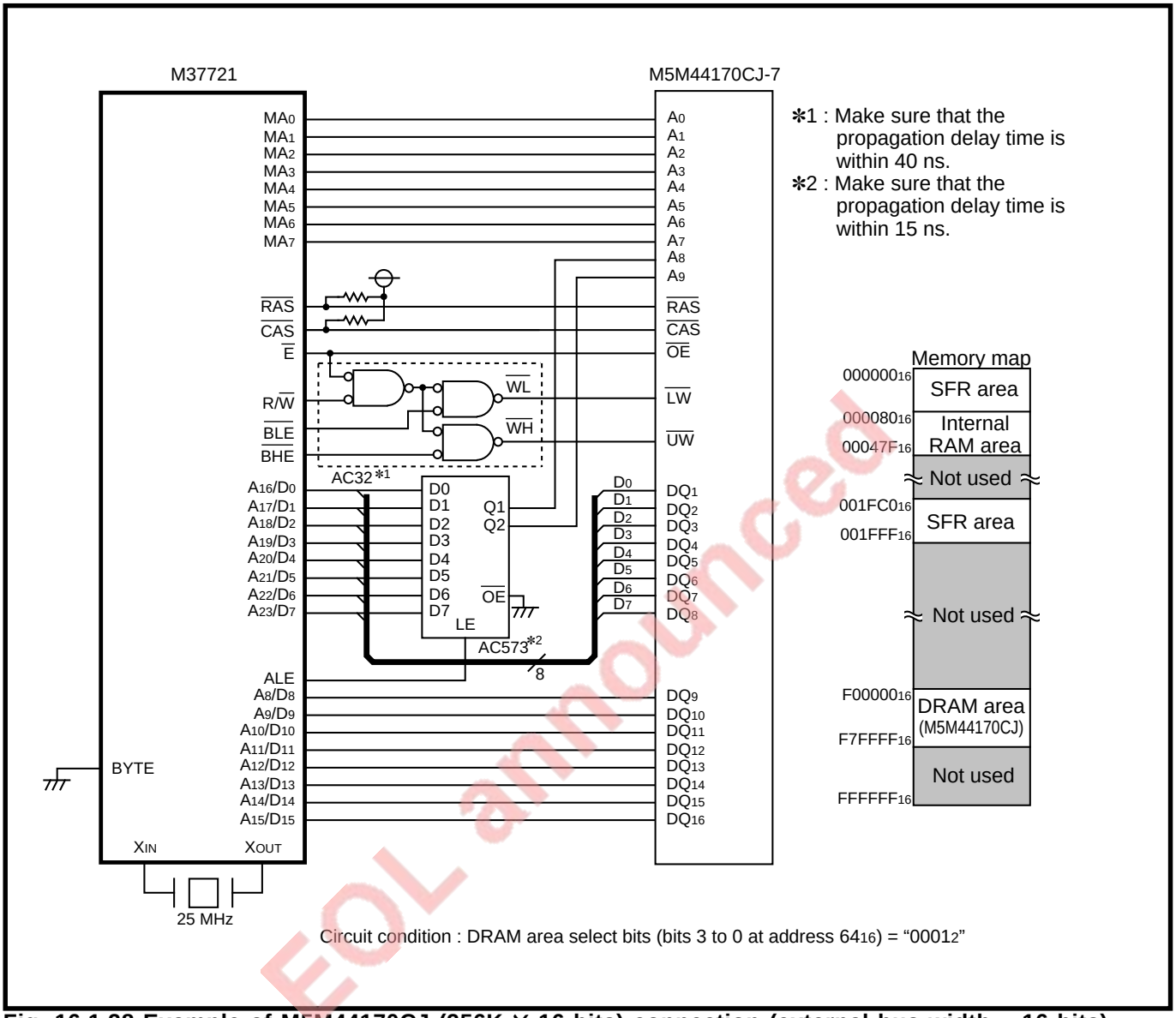


EOL announced

Fig. 16.1.27 Timing chart for example of M5M418160CJ (1M × 16 bits) connection (external bus width = 16 bits)

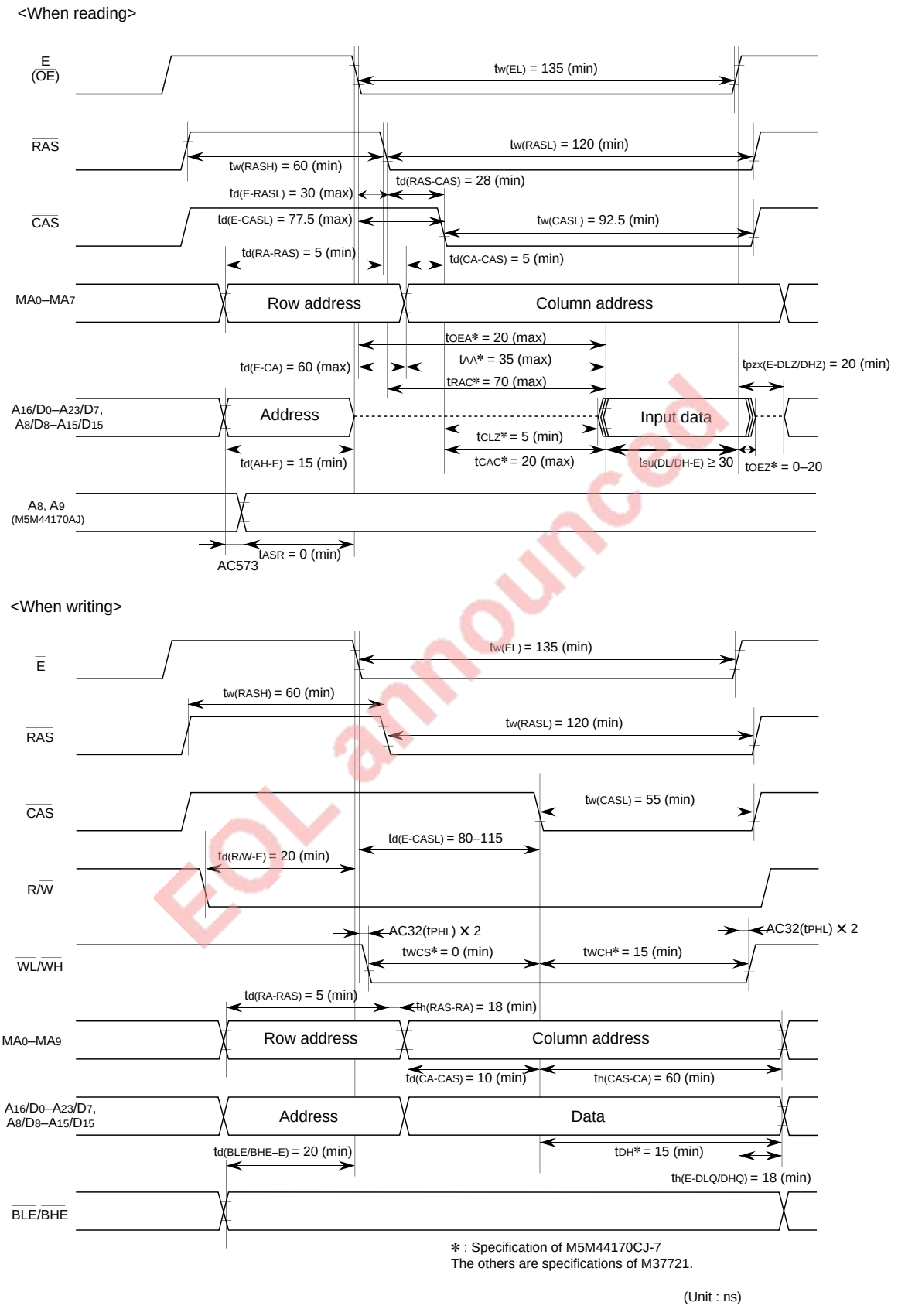
16.1 Memory connection

(7) Example of DRAM connection (external bus width = 16 bits) ②



EOL announced

Fig. 16.1.28 Example of M5M44170CJ (256K × 16 bits) connection (external bus width = 16 bits)



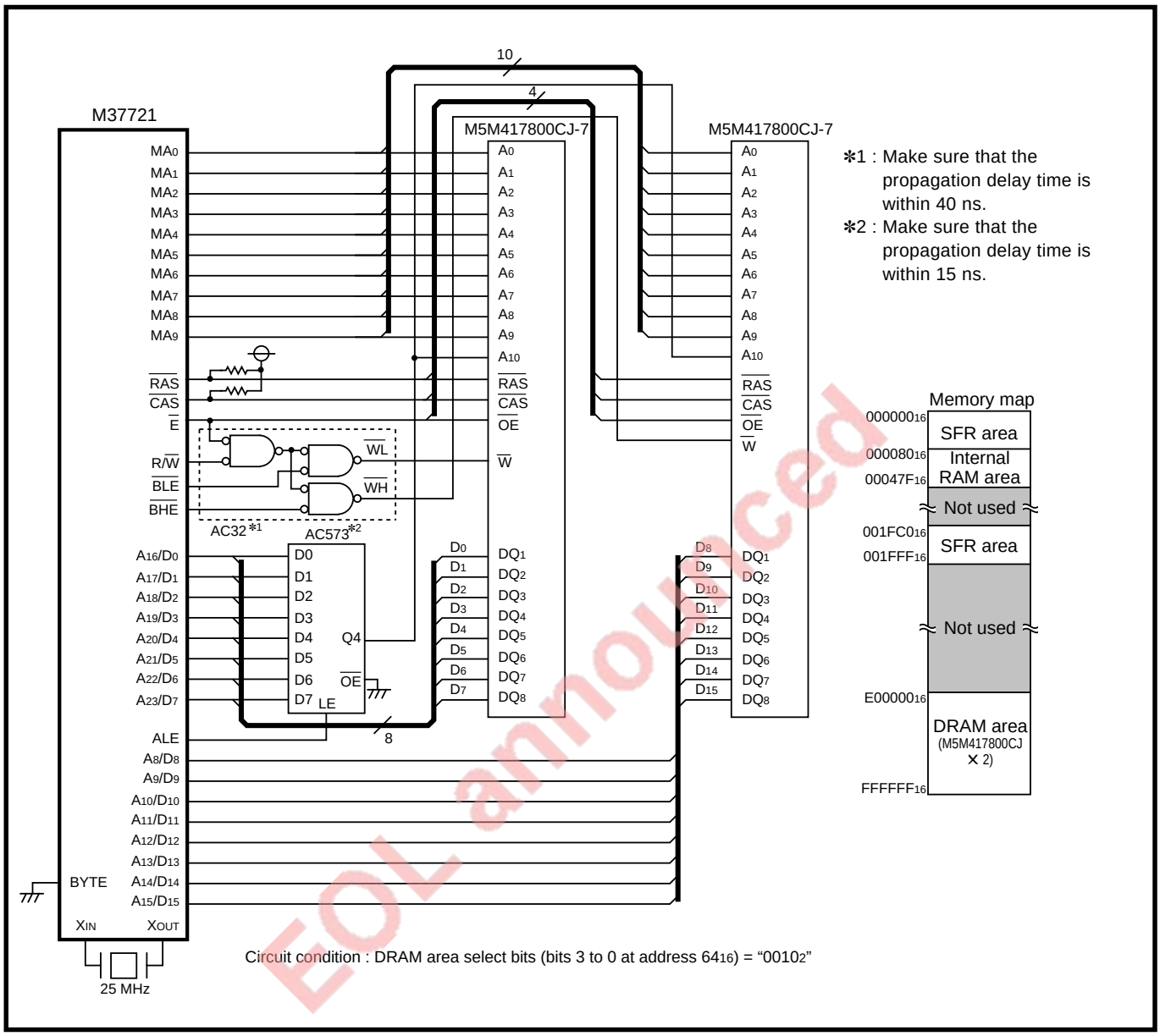
EOL announced

Fig. 16.1.29 Timing chart for example of M5M44170CJ (256K × 16 bits) connection (external bus width = 16 bits)

APPLICATION

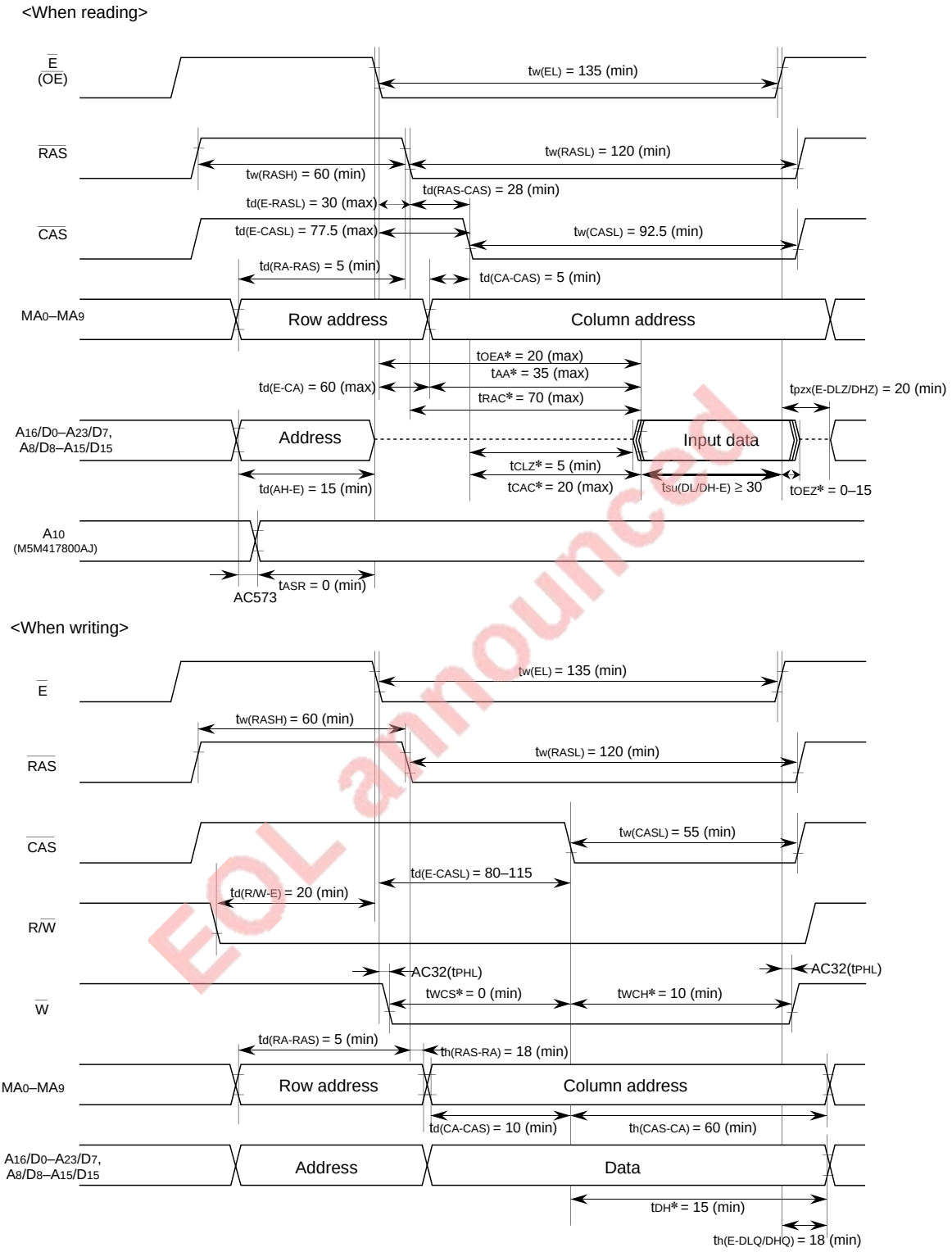
16.1 Memory connection

(8) Example of DRAM connection (external bus width = 16 bits) ③



EOL announced

Fig. 16.1.30 Example of M5M417800CJ (2M × 8 bits) connection (external bus width = 16 bits)



EOL announced

✻ : Specifications of M5M417800CJ-7
The others are specifications of M37721.

(Unit : ns)

Fig. 16.1.31 Timing chart for example of M5M417800CJ (2M × 8 bits) connection (external bus width = 16 bits)

APPLICATION

16.1 Memory connection

(9) Example of DRAM connection (external bus width = 16 bits) ④

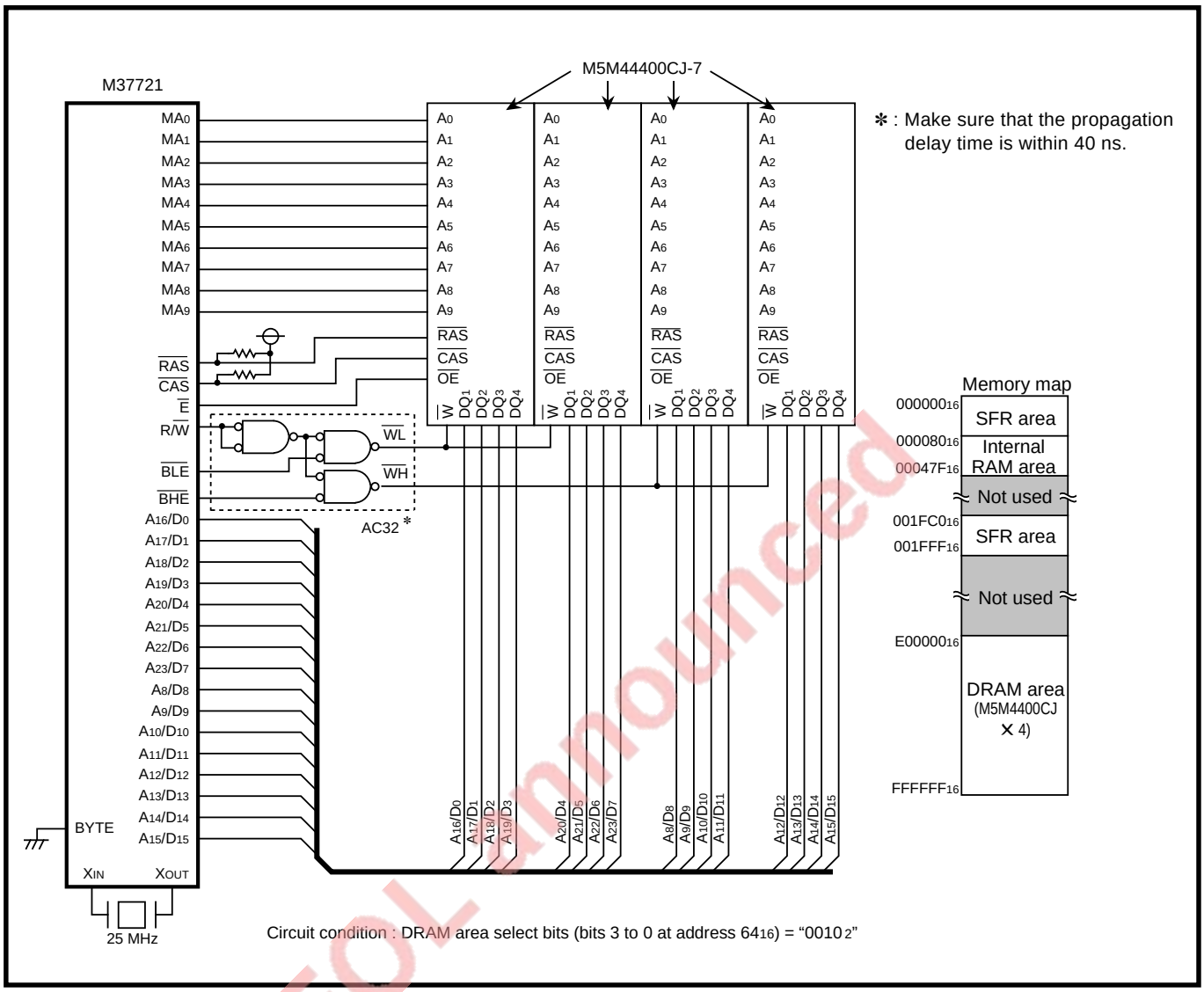
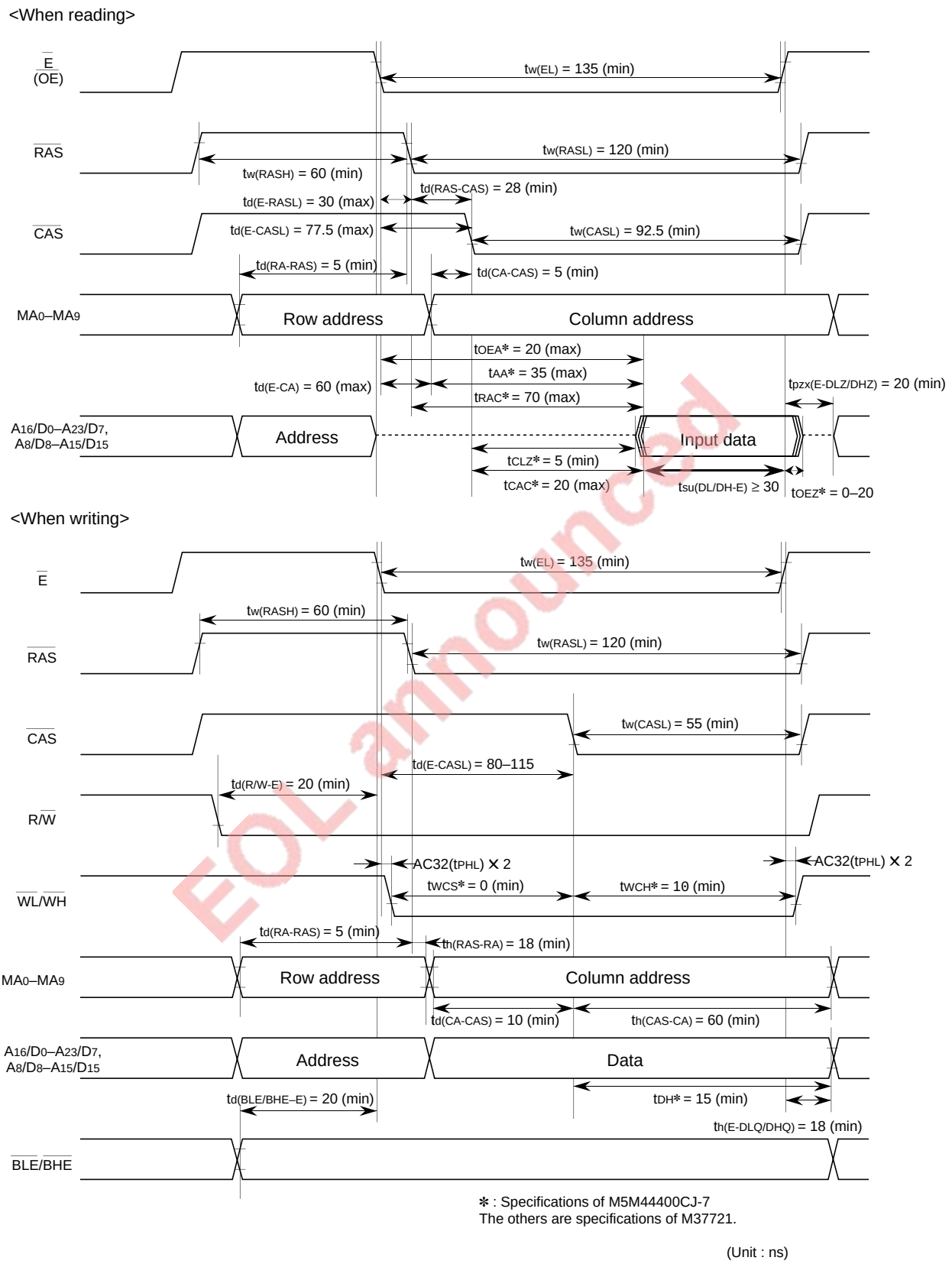


Fig. 16.1.32 Example of M5M44400CJ (1M × 4 bits) connection (external bus width = 16 bits)

EOL announced



✻ : Specifications of M5M44400CJ-7
The others are specifications of M37721.

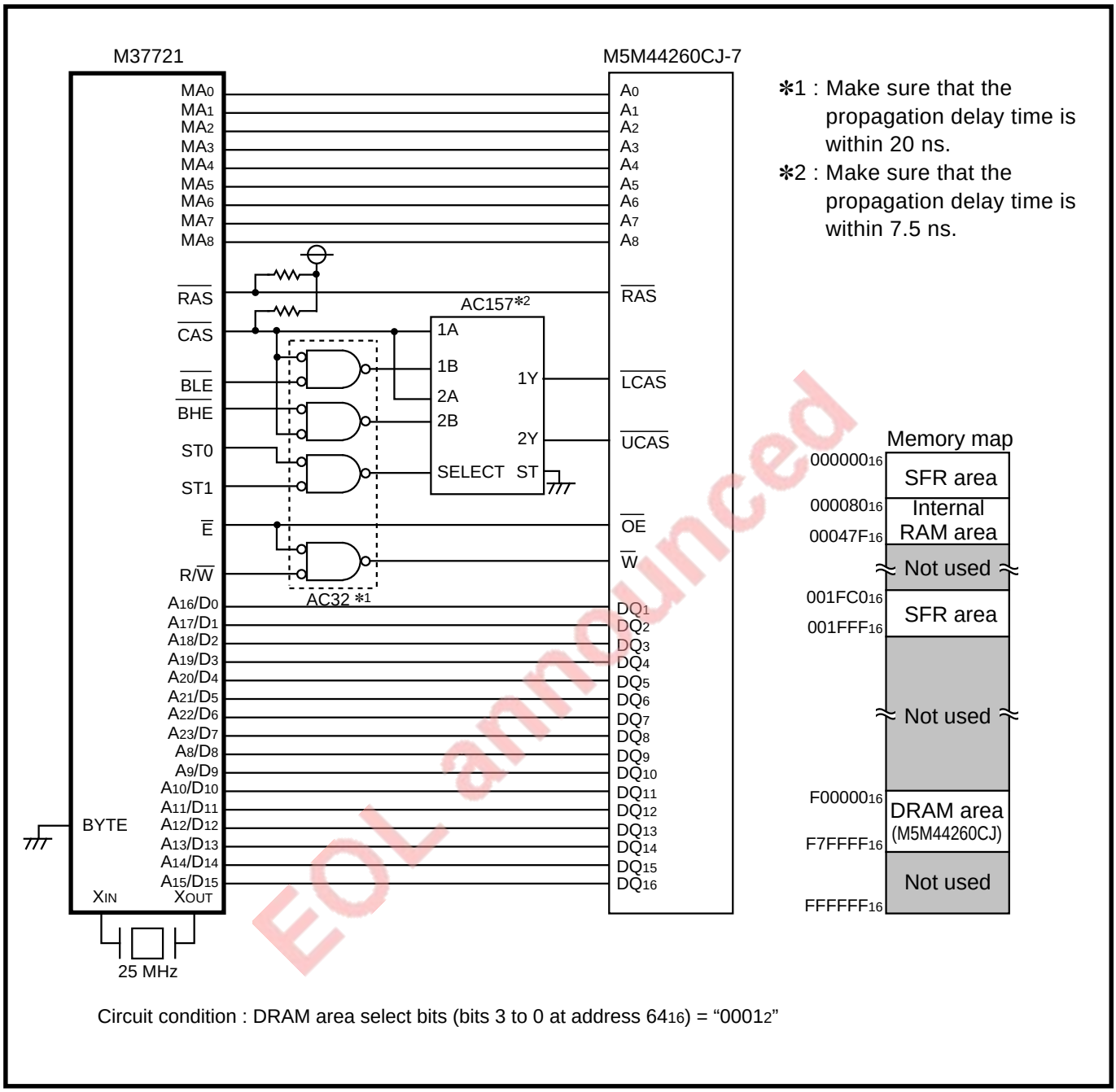
(Unit : ns)

Fig. 16.1.33 Timing chart for example of M5M44400CJ (1M × 4 bits) connection (external bus width = 16 bits)

APPLICATION

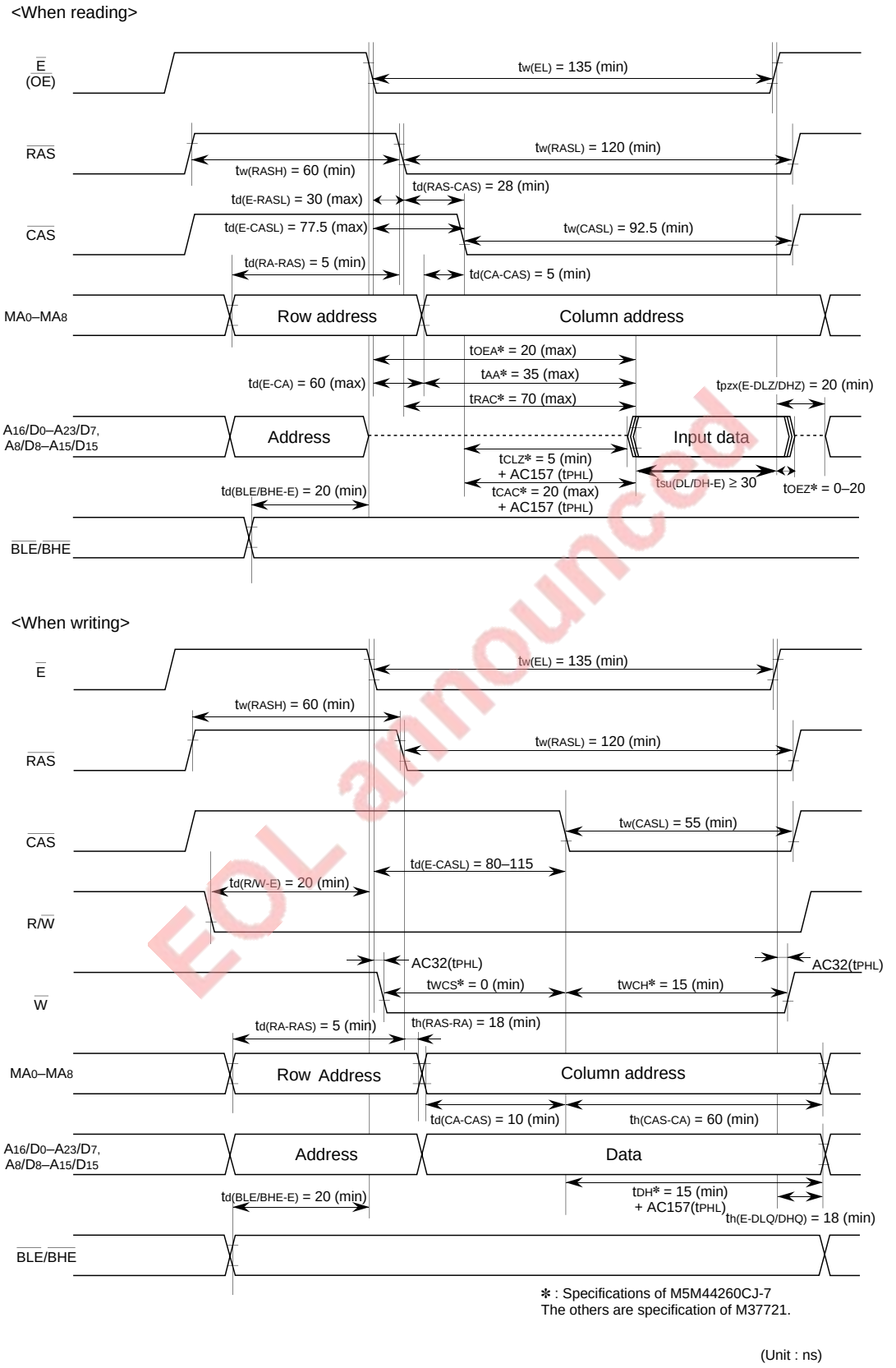
16.1 Memory connection

(10) Example of DRAM connection (external bus width = 16 bits) ⑤



EOL announced

Fig. 16.1.34 Example of M5M44260CJ (256K × 16 bits) connection (external bus width = 16 bits)



EOL announced

Fig. 16.1.35 Timing chart for example of M5M44260CJ (256K × 16 bits) connection (external bus width = 16 bits)

16.1 Memory connection

16.1.4 Example of I/O expansion

(1) Example of port expansion circuit using M66010FP

Figure 16.1.36 shows an example of a port expansion circuit using the M66010FP. Make sure that the frequency of Serial I/O transfer clock must be 1.923 MHz or less.
About Serial I/O control in this expansion example is described below.
In this example, 8-bit data transmission/reception is performed 3 times by using UART0, so that 24-bit port expansion is realized. Setting of UART0 is described below:

- Clock synchronous serial I/O mode: Transmission/Reception enable state
- Internal clock is selected. Transfer clock frequency is 1.66 MHz.
- LSB first

The control procedure is described below:

① Output “L” level from port $P4_5$. (Expanded I/O ports of the M66010FP enter a floating state by this signal.)
② Output “H” level from port $P4_5$.
③ Output “L” level from port $P4_4$.
④ Transmit/Receive 24-bit data by using UART0.
⑤ Output “H” level from port $P4_4$.

Figure 16.1.37 shows the serial transfer timing between the M37721 and the M66010FP.

EOL announced

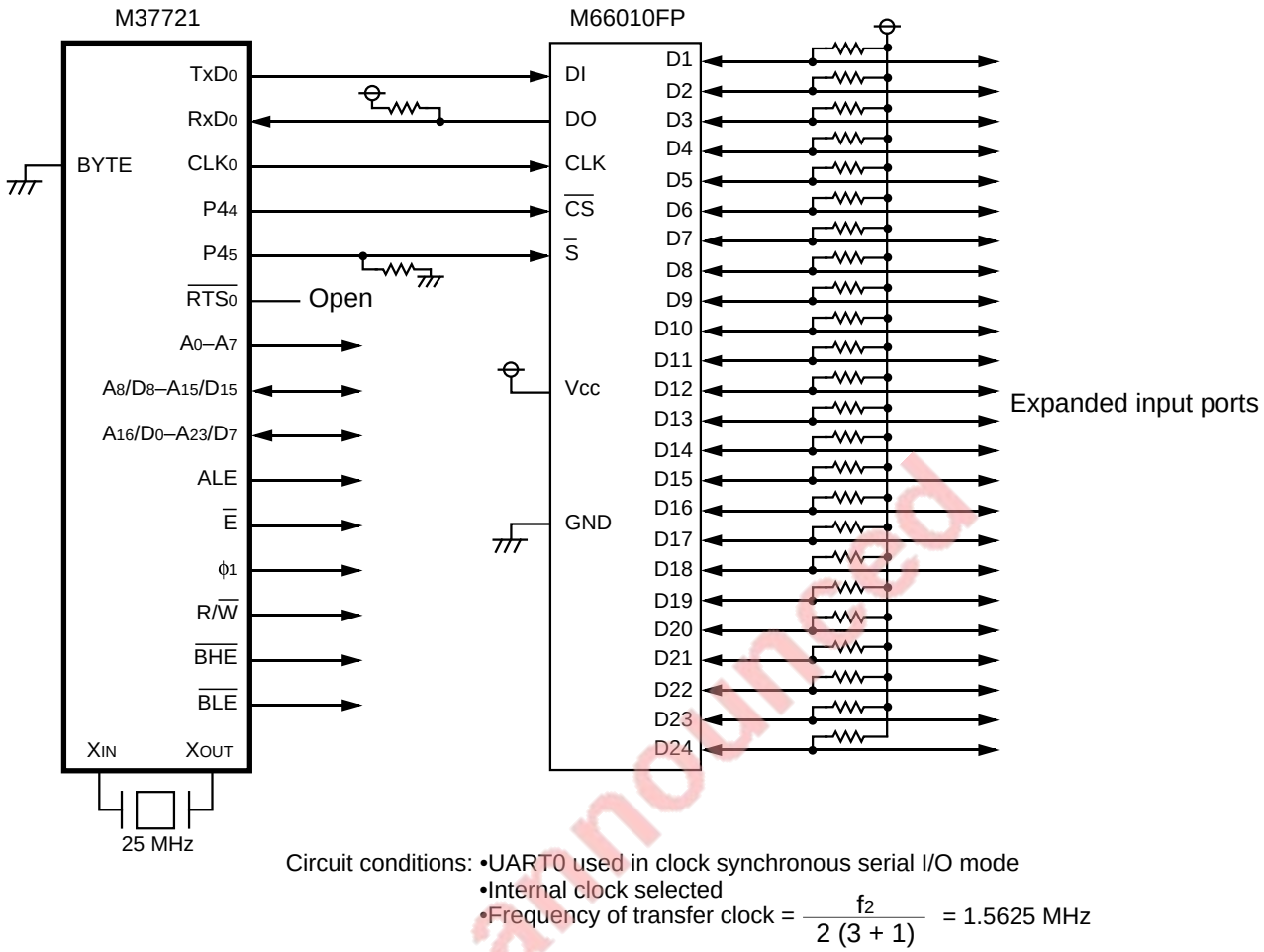
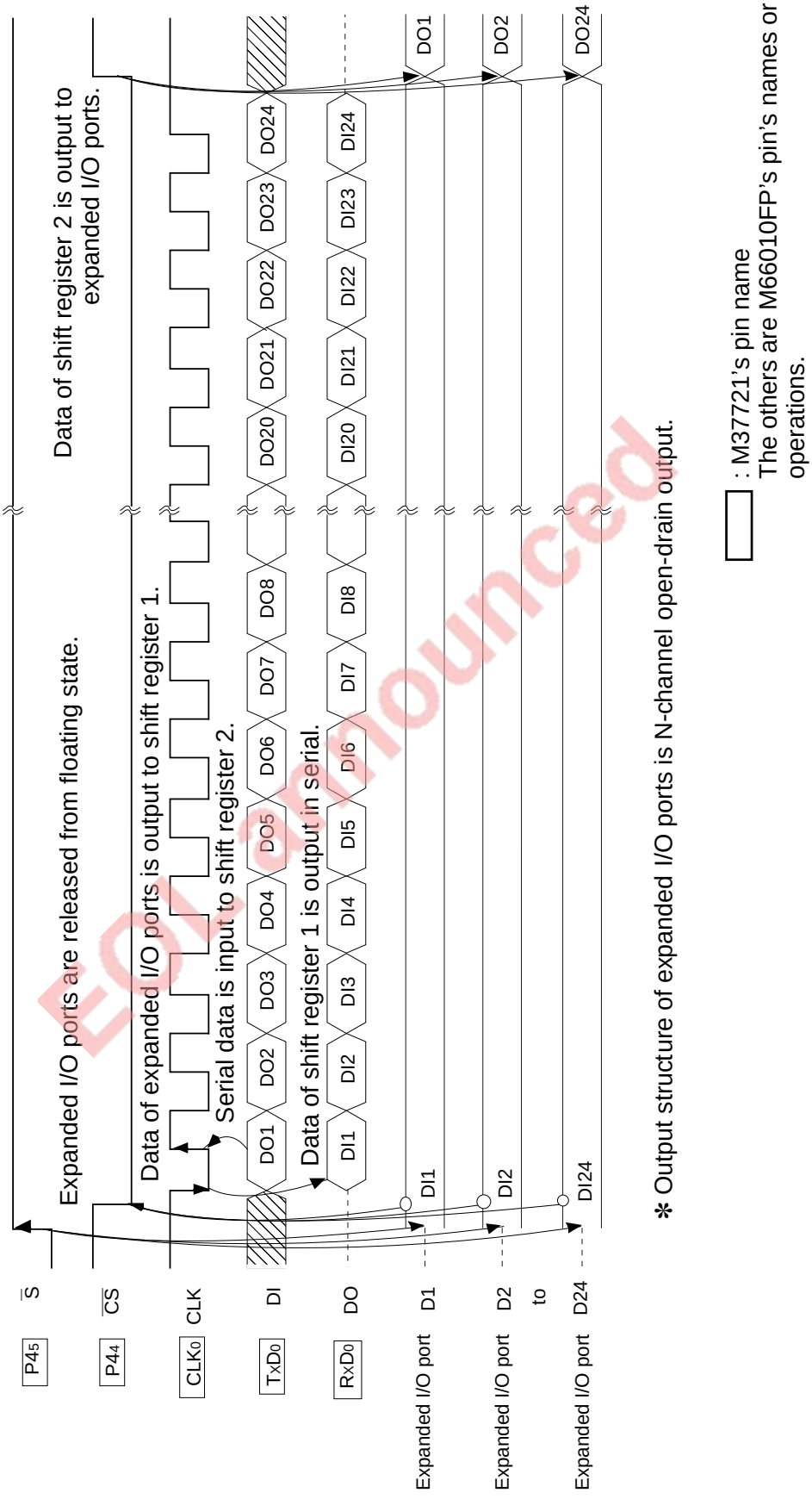


Fig. 16.1.36 Example of port expansion circuit using M66010FP

EOL announced

APPLICATION

16.1 Memory connection



EOL announced

Fig. 16.1.37 Serial transfer timing between M37721 and M66010FP

16.2 Examples of using DMA controller

16.2.1 Example of Centronics interface configuration

The following is an example of Centronics interface configurated by using DMA0, Timers A2 and A3.

(1) Specifications

- Octal latch's contents are transferred to the data buffer (RAM) by using DMA0. The trigger is the $\overline{STB}$ signal. (Refer to **"Figure 16.2.1."**)
- "L" level width of the $\overline{ACK}$ signal is generated by using Timer A2; one-shot pulse mode; the trigger is the rising edge of the $\overline{DMAACK0}$ signal. (Refer to **"Figure 16.2.2."**)
- Timer A3 generates the time from when the $\overline{ACK}$ signal rises until the BUSY signal falls; one-shot pulse mode; the trigger is the rising edge of the $\overline{DMAACK0}$ signal. (Refer to **"Figure 16.2.2."**)
- $P4_3$ is used for BUSY signal generation. When outputting "H" level, the next transfer can wait. In that case, the contents of the preceding transfer are hold in the octal latch.
- When the data buffer is filled (in other words, DMA transfer is completed), a DMA interrupt occurs.

EOL announced

16.2 Examples of using DMA controller

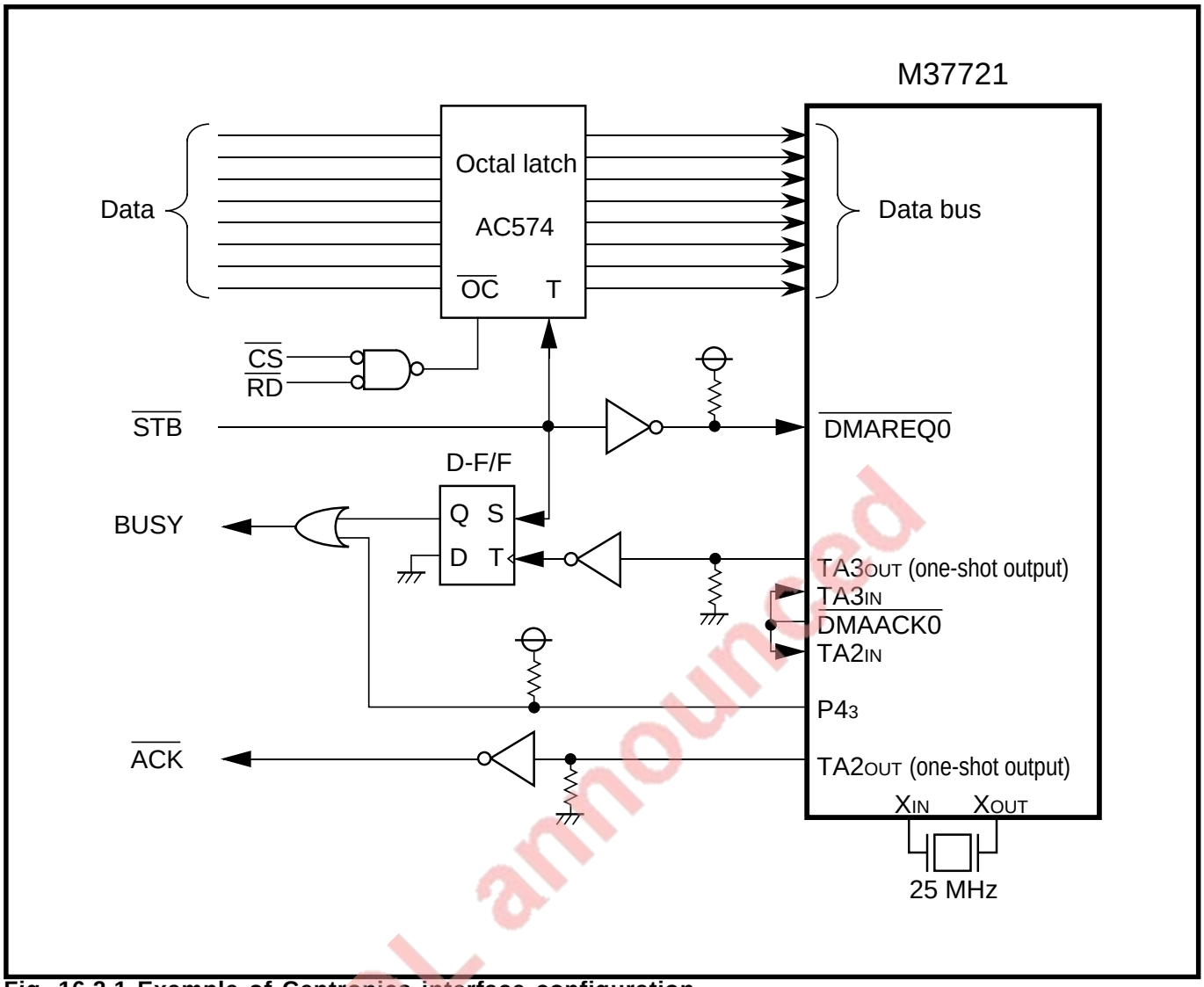


Fig. 16.2.1 Example of Centronics interface configuration

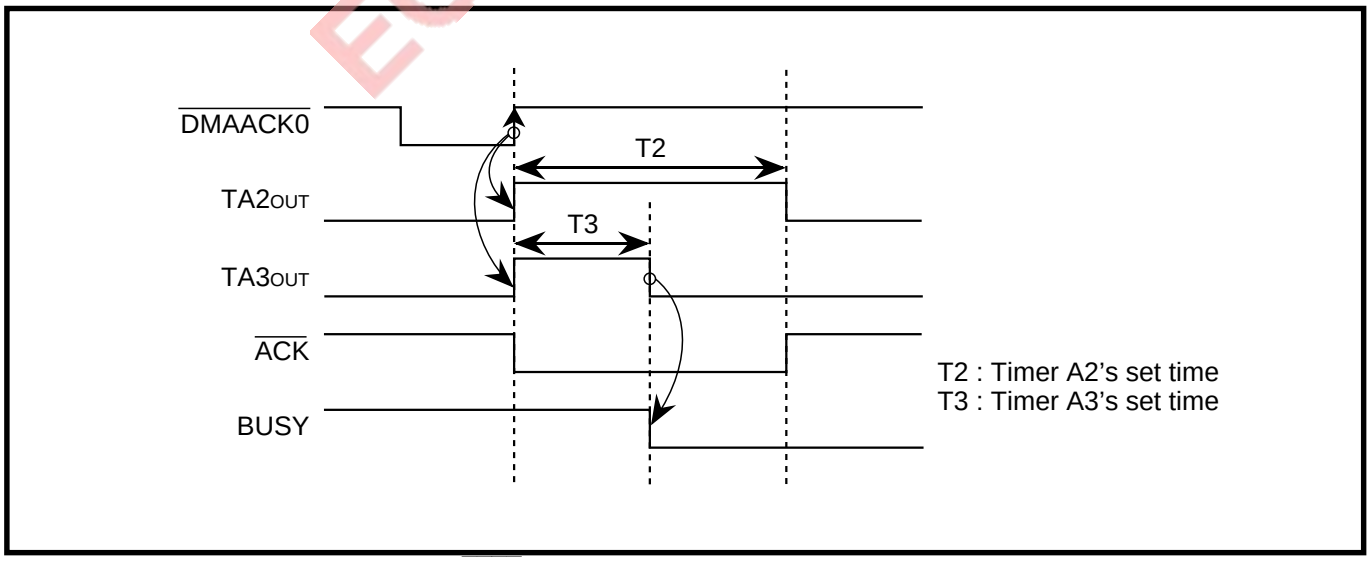


Fig. 16.2.2 Relationship between ACK and BUSY

EOL announced

(2) Initial setting example for relevant register

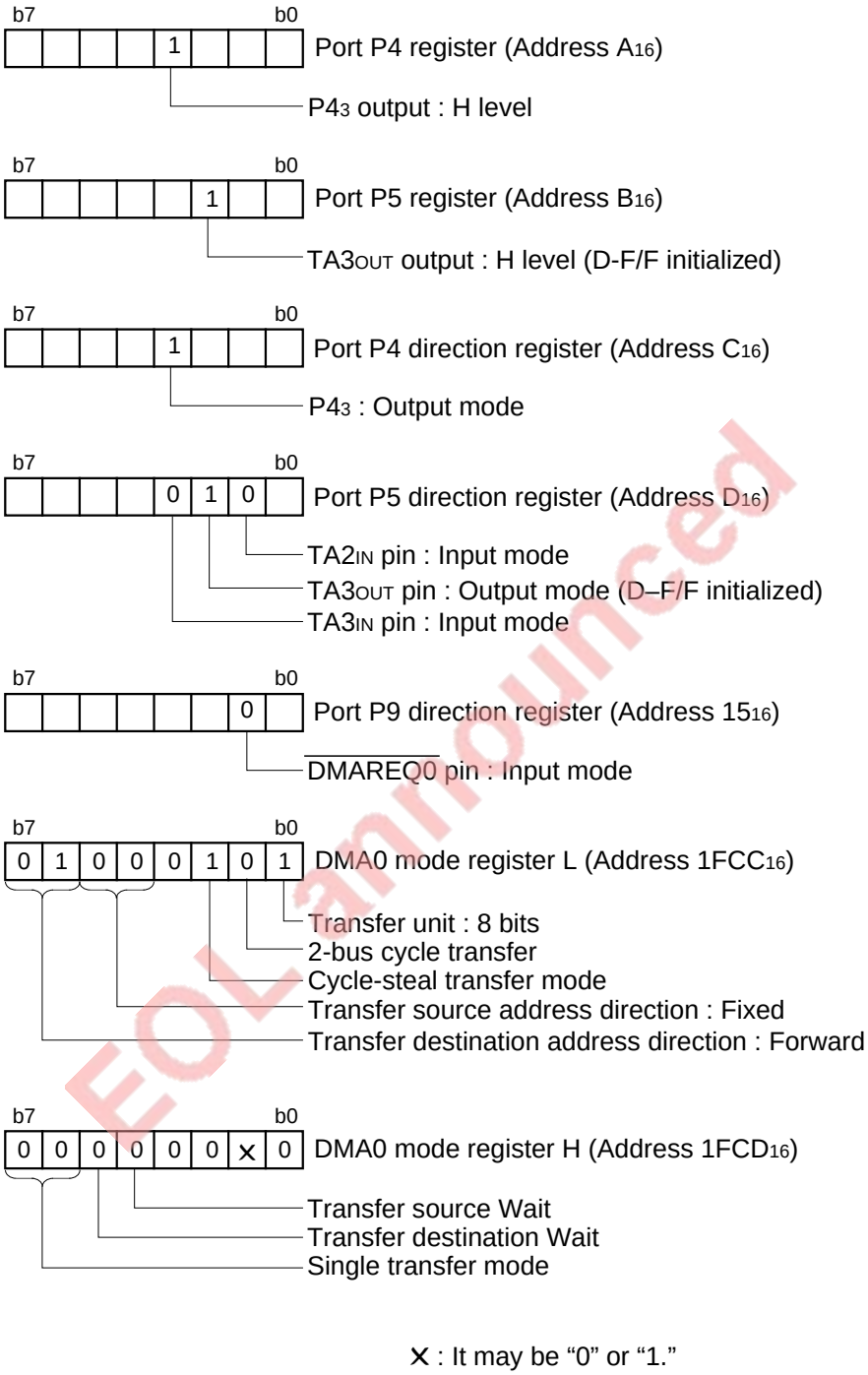
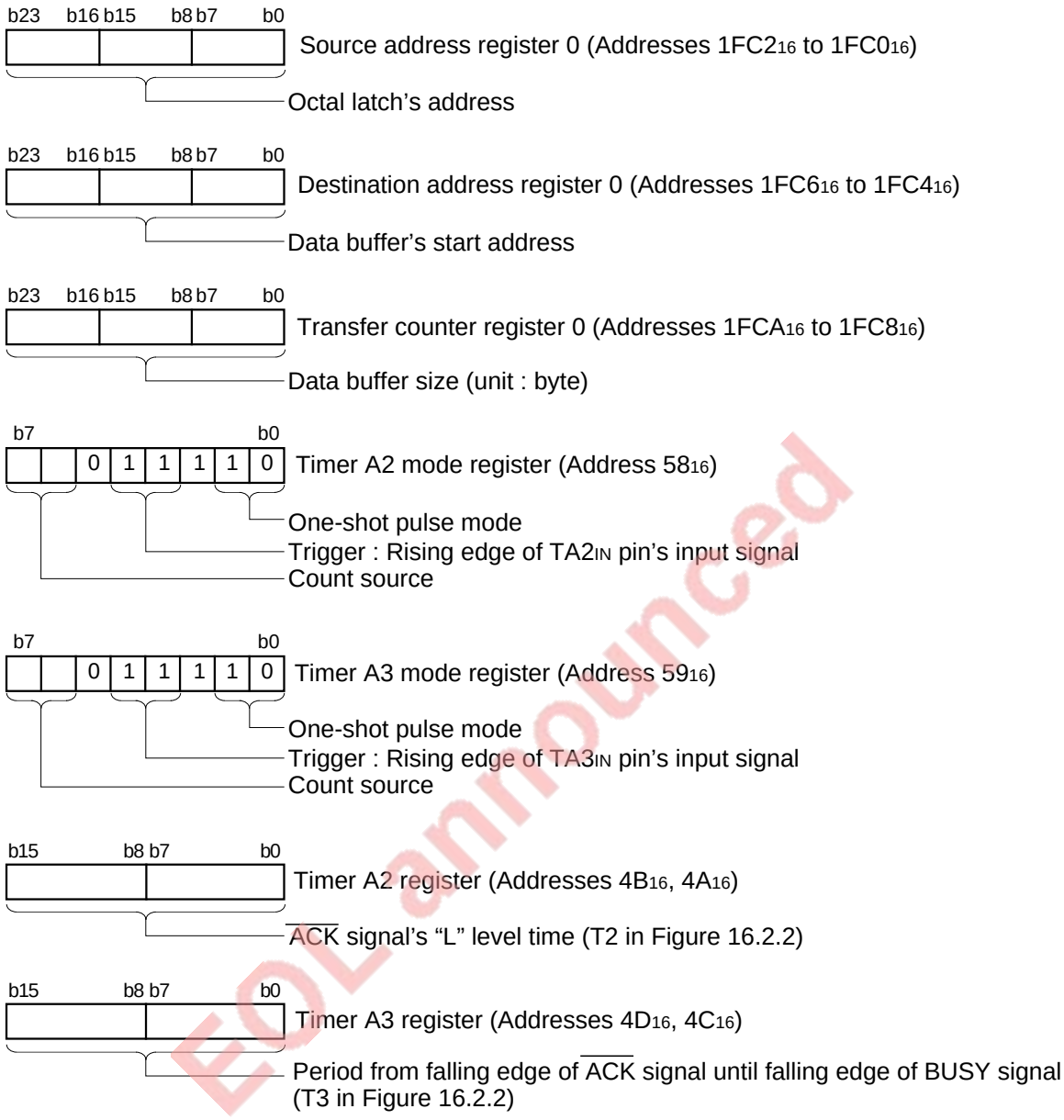


Fig. 16.2.3 Initial setting example for relevant register (1)

EOL announced

16.2 Examples of using DMA controller



EOL announced

Fig. 16.2.4 Initial setting example for relevant register (2)

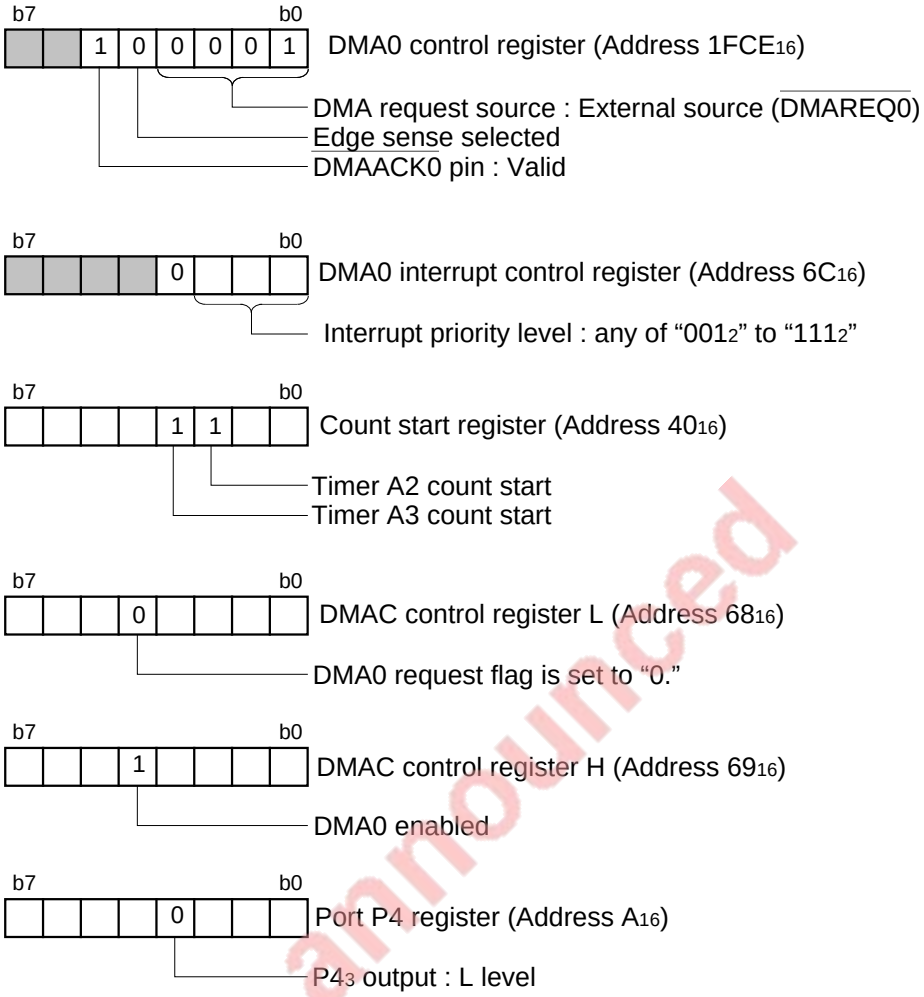


Fig. 16.2.5 Initial setting example for relevant register (3)

EOL announced

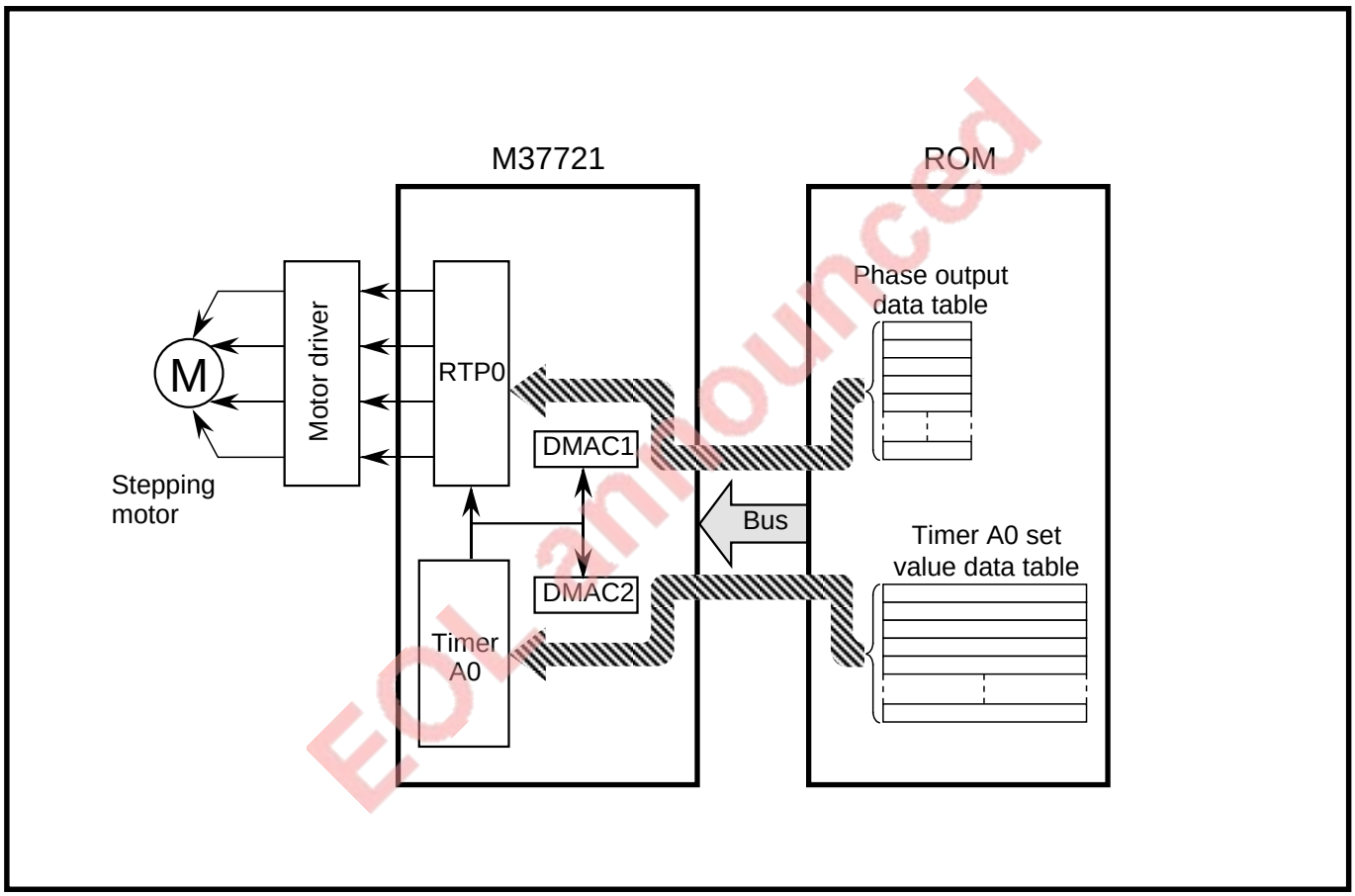
16.2 Examples of using DMA controller

16.2.2 Example of stepping motor control

The following is an example where the slow-up or slow-down control for the stepping motor is performed by using DMA1, DMA2, and RTP0.

(1) Specifications

- DMA1 transfers the stepping motor's phase output data from the phase output data table to the RTP0 pulse output data register. (Refer to **"Figure 16.2.6"** and **"Table 16.2.1."**)
- DMA2 transfers the step time for slow up or slow down from the timer A0 set value data table to the timer A0 register. (Refer to **"Figure 16.2.6."**)
  After slow up or slow down is completed, a DMA2 interrupt occurs.
- Phase output is performed by RTP0; pulse output mode 0 (Refer to **"Figures 16.2.6 and 16.2.7."**)
- After slow up or slow down is completed, the motor operates with the definite rate.



EOL announced

Fig. 16.2.6 Example of stepping motor control

Table 16.2.1 Example of phase output data table

	2-2 phase	1-2 phase
0	0011	0011
1	1001	0001
2	1100	1001
3	0110	1000
4	0011	1100
5	1001	0100
6	1100	0110
7	0110	0010

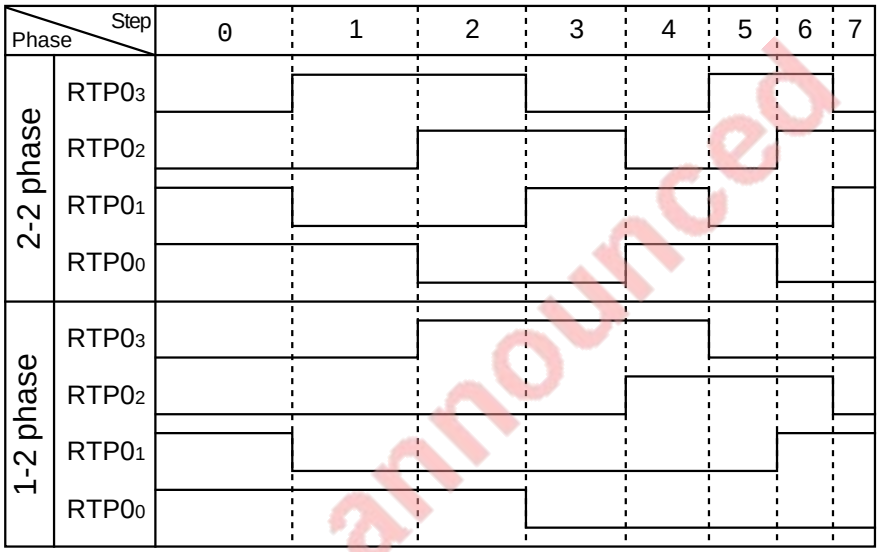
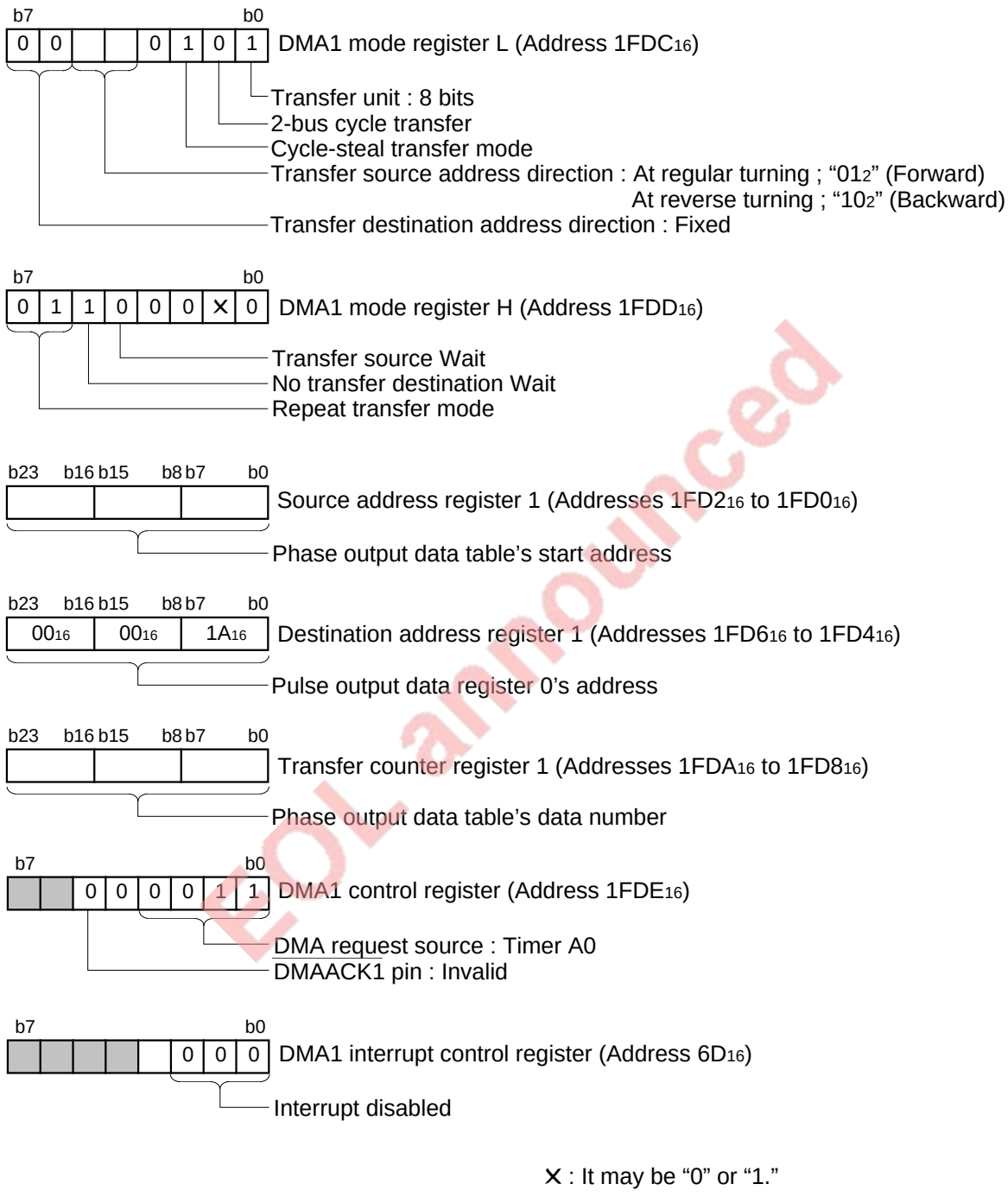


Fig. 16.2.7 Example of phase output

EOL announced

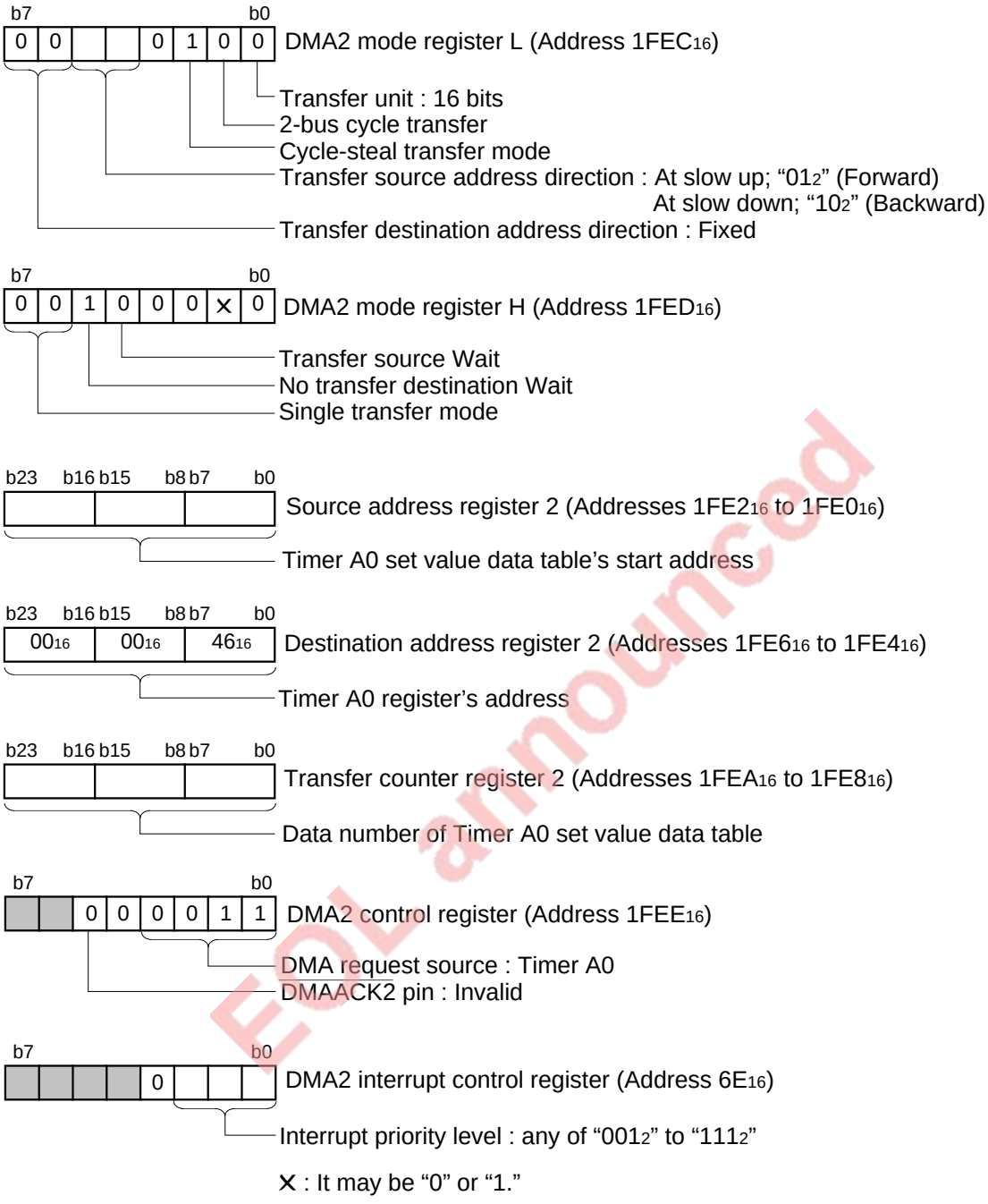
16.2 Examples of using DMA controller

(2) Initial setting example for relevant register



EOL announced

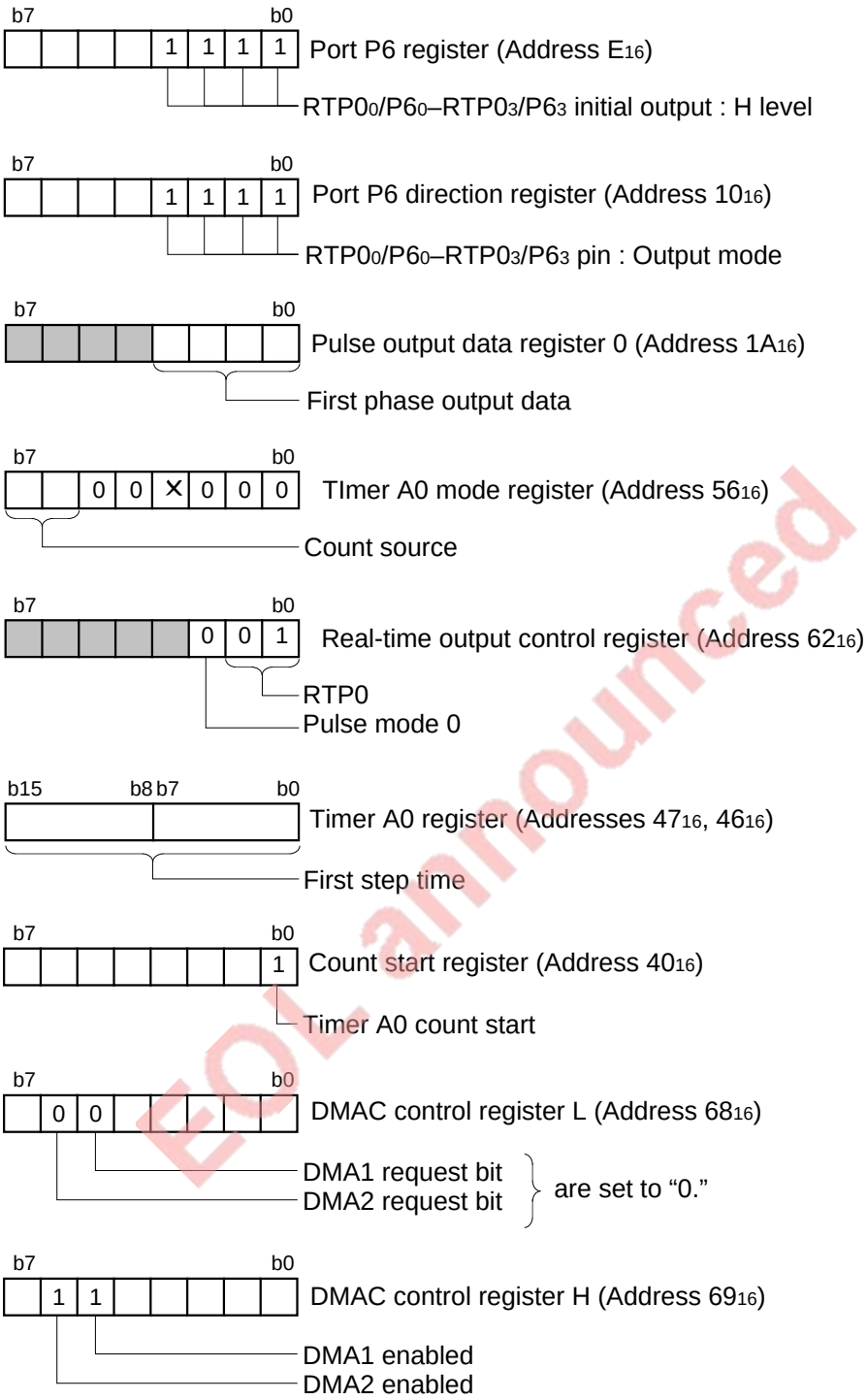
Fig. 16.2.8 Initial setting example for relevant register (1)



EOL announced

Fig. 16.2.9 Initial setting example for relevant register (2)

16.2 Examples of using DMA controller



EOL announced

Fig. 16.2.10 Initial setting example for relevant register (3)

16.2.3 Example of dynamic lighting for LED

The following is an example of dynamic lighting for LED by using DMA3 and Timer B0.

(1) Specifications

- The eight 7-segment LEDs are lighted up; port P6 outputs the segment data; port P7 outputs the digit data. (Refer to **"Figure 16.2.11."**)
- The display data and the segment data are transferred from the data buffer to the port P6 and P7 registers by DMA3.
- Digit switch interval is generated by Timer B0.
- 16 bytes of RAM are used as the data buffer. 1-digit display data consists of 2 bytes; the digit data is placed in the high-order byte; the segment data is placed in the low-order byte. (Refer to **"Table 16.2.2."**)

  When the digit data and segment data are "0," the LED is lighted up (ON): when they are "1," the light goes out (OFF).

  Assuming that the segment pattern is generated by another processing.

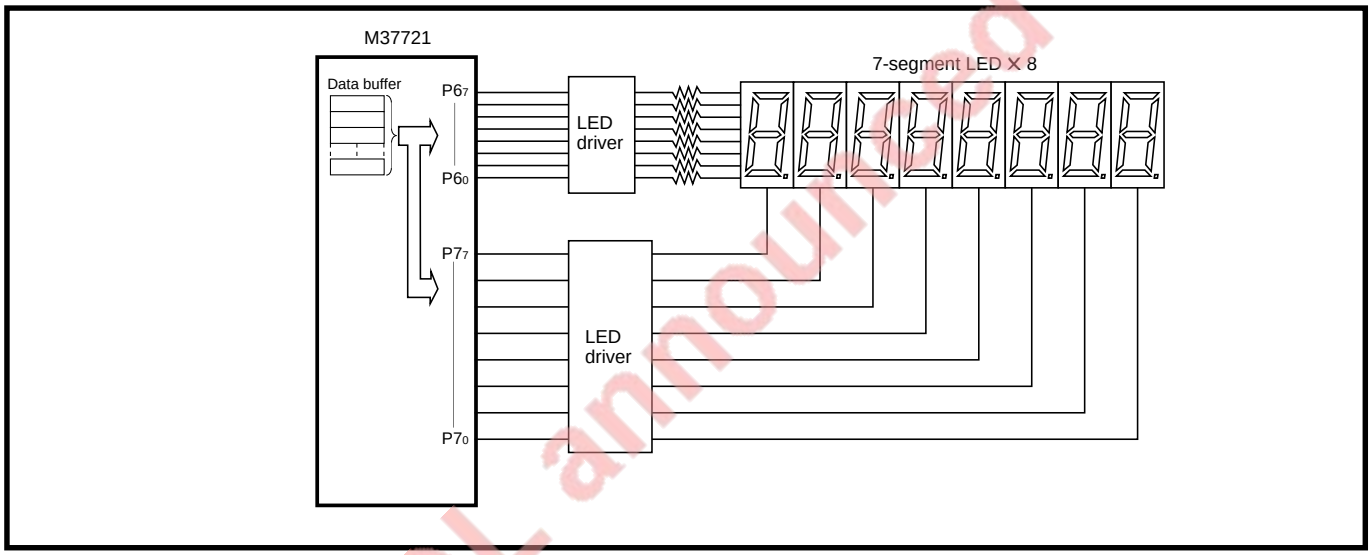


Fig. 16.2.11 Example of dynamic lighting for LED

Table 16.2.2 Data buffer

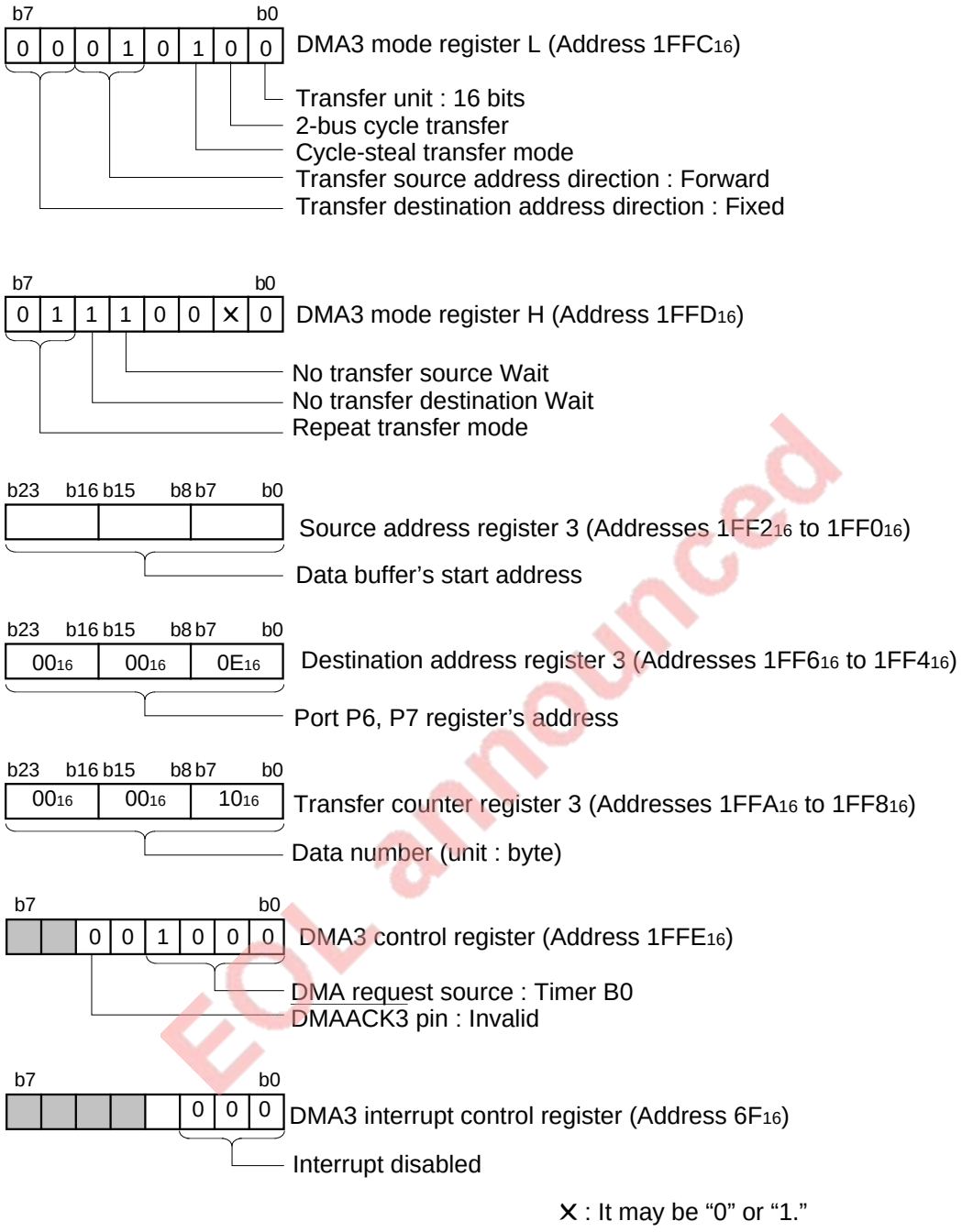
Data buffer	
Digit data	Segment pattern
00000001	Segment pattern of the contents to be displayed in each digit
00000010	
00000100	
00001000	
00010000	
00100000	
01000000	
10000000	

Notes 1: This applies in the following:
- when the digit data is "0," the light goes out.
- when the digit data is "1," the LED is lighted up.

2: Assuming that the segment pattern is generated by another processing.

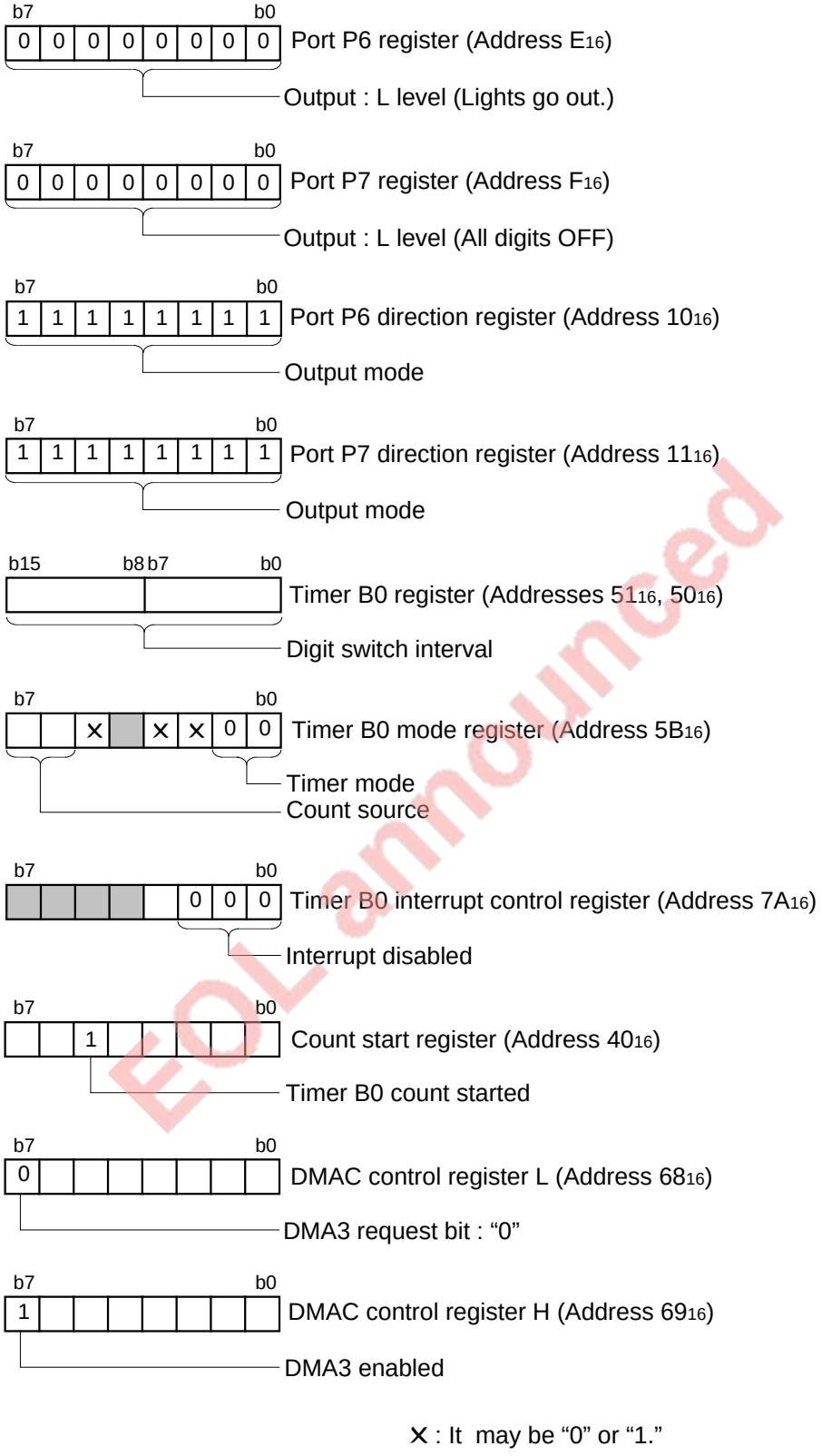
EOL announced

16.2 Examples of using DMA controller



EOL announced

Fig. 16.2.12 Initial setting example for relevant register (1)



EOL announced

Fig. 16.2.13 Initial setting example for relevant register (2)

16.3 Comparison of sample program execution rate

Sample program execution rates are compared in this paragraph.
The execution time ratio depends on the program or the usage conditions.

16.3.1 Differences depending on data bus width and software Wait

Internal areas are always accessed with data bus of which width is 16 bits and no software Wait. In the external areas, the external data bus width and software Wait are selectable. Table 16.3.1 lists the sample program (Refer to **"Figure 16.3.1."**) execution time ratio depending on these selection and usable memory areas.

Table 16.3.1 Sample program execution time ratio (external data bus width and software Wait)

Memory area		External data bus width (unit : bit)	Software Wait	Sample program execution time ratio	
RAM	ROM			Sample A	Sample B
Internal	External	16	None	1.00	1.00
			Inserted	1.17	1.10
		8	None	1.19	1.08
			Inserted	1.67	1.46
External	Internal	16	None	1.00	1.00
			Inserted	1.25	1.17
		8	None	1.19	1.13
			Inserted	1.78	1.65
Calculated value *				0.92	0.90

Calculated value * : The value is calculated from the shortest execution cycle number of each instruction described in **"7700 Family Software Manual."**

EOL announced

Sample A

```
        SEP     M,X
        LDA.B   A,#0
        STA     A,DEST+64
        STA     A,DEST+65
        STA     A,DEST+66
        LDX.B   #63
ITALIC: LDA     A,SOUR,X
        TAY
        AND.B   A,#00000011B
        STA     A,DEST,X
        TYA
        AND.B   A,#00001100B
        ORA     A,DEST+1,X
        STA     A,DEST+1,X
        TYA
        AND.B   A,#00110000B
        ORA     A,DEST+2,X
        STA     A,DEST+2,X
        TYA
        AND.B   A,#11000000B
        ORA     A,DEST+3,X
        STA     A,DEST+3,X
        DEX
        BPL     ITALIC
```

Sample B

```
        SEP     X
        CLM
        .DATA   16
        .INDEX  8
        LDY     #69
LOOP0:  LDX     #69
LOOP1:  ASL     SOUR,X
        SEM
        .DATA   8
        ROL     SOUR+2,X
        ROL     B
        CLM
        .DATA   16
        ROR     A
        DEX
        DEX
        DEX
        BNE     LOOP1
        STA     A,DEST,Y
        SEM
        .DATA   8
        STA     B,DEST+2,Y
        CLM
        .DATA   16
        DEY
        DEY
        DEY
        BNE     LOOP0
```

✻ SOUR, DEST : Work area
(Direct page area : Access this area by using the following modes.)
- •Direct addressing mode
- •Direct Indexed X addressing mode
- •Absolute Indexed Y addressing mode

EOL announced

Fig. 16.3.1 Sample program list

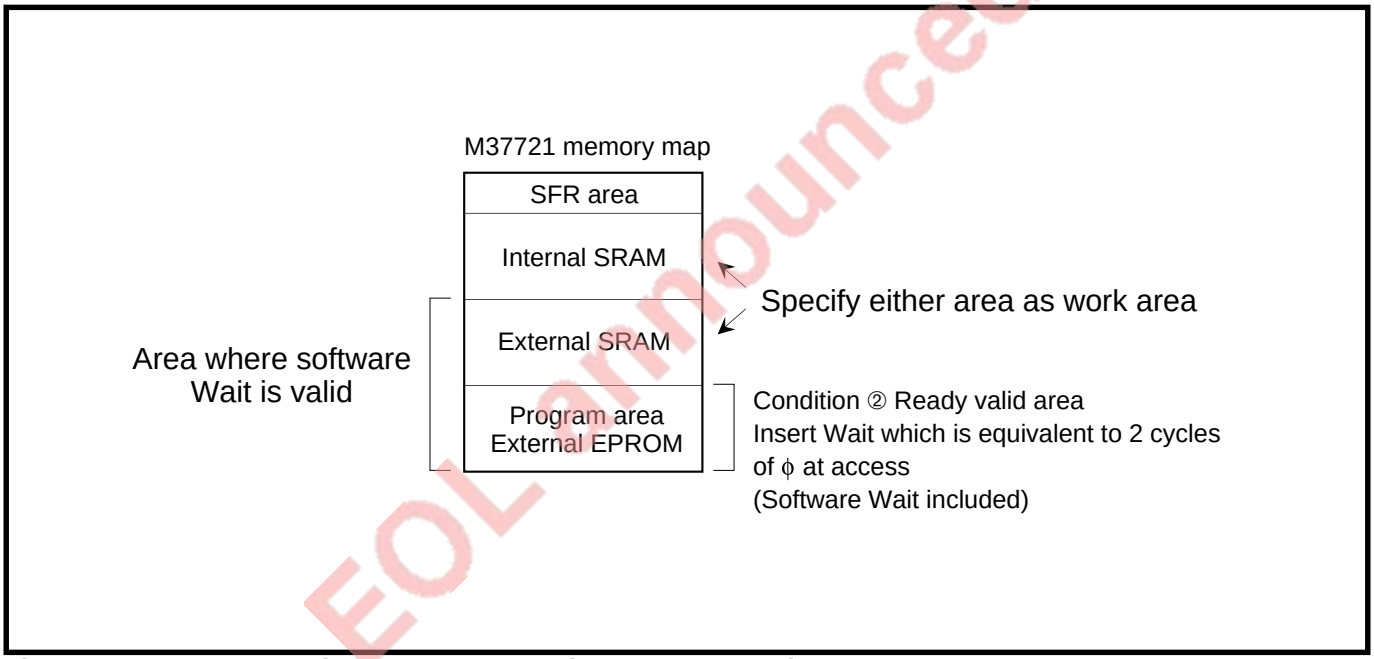
16.3 Comparison of sample program execution rate

16.3.2 Comparison between software Wait ($f(X_{IN})$ = 20 MHz) and software Wait + Ready ($f(X_{IN})$ = 25 MHz)

Figure 16.3.3 shows the execution time ratio when sample programs in Figure 16.3.1 are executed on the two conditions in Table 16.3.2. Figure 16.3.2 shows the memory assignment at execution rate comparison. The execution time ratio depends on the program or the usage conditions.

Table 16.3.2 Comparison conditions

Item	Condition ①	Condition ②
Processor mode	Microprocessor mode	Microprocessor mode
$f(X_{IN})$	20 MHz	25 MHz
External data bus width	16 bits	16 bits
Software Wait	Inserted	Inserted
Ready	Invalid	Valid only for external EPROM area
Program area	External EPROM	External EPROM
Work area	Internal or External SRAM	Internal or External SRAM



EOL announced

Fig. 16.3.2 Memory assignment at execution rate comparison

Figure 16.3.3 shows that there is almost no difference between conditions ① and ② about the execution time. The bus buffers become unnecessary when using the specified memory. (Refer to **"Table 16.1.6."**) Considering this, the case where software Wait is inserted with f(X_{IN}) = 20 MHz (condition ①) is superior in the cost performance.

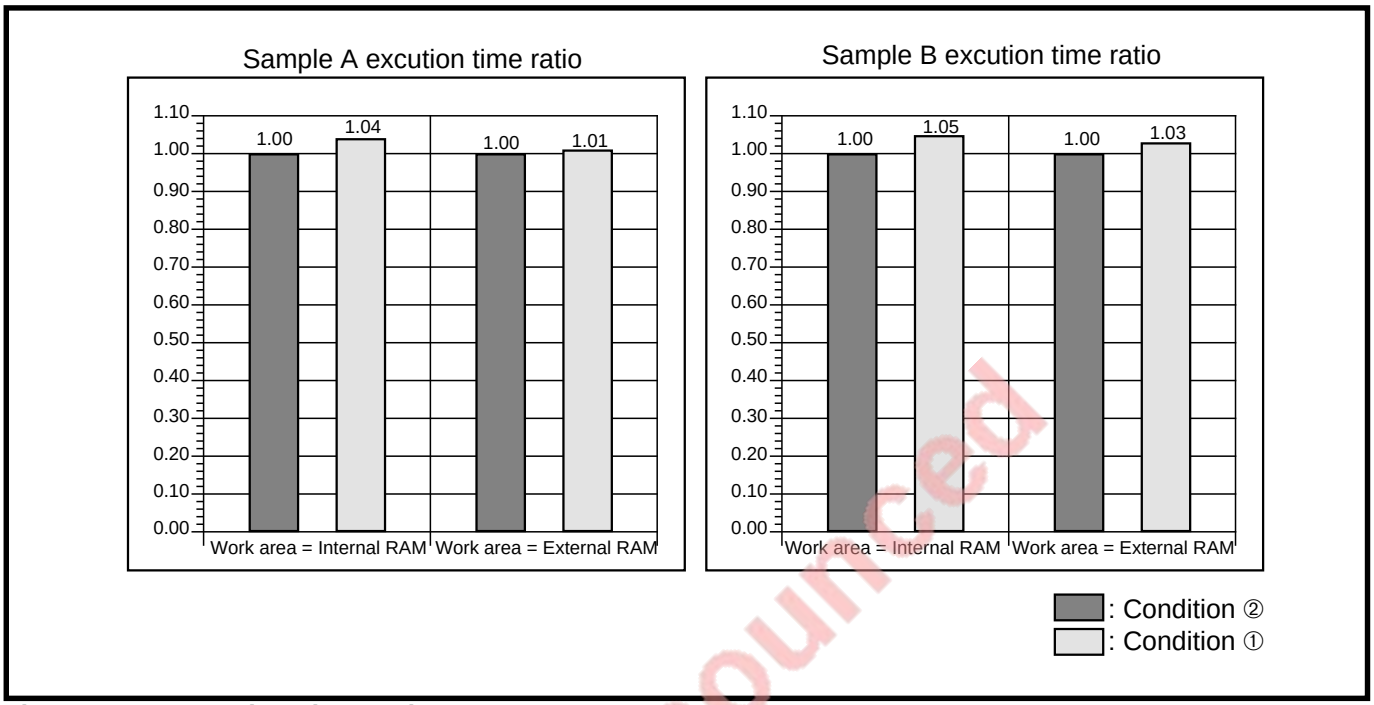


Fig. 16.3.3 Execution time ratio

EOL announced

MEMORANDUM

EOL announced

APPENDIX

Appendix 1. Memory assignment of 7721 Group
Appendix 2. Memory assignment in SFR area
Appendix 3. Control registers
Appendix 4. Package outline
Appendix 5. Examples of handling unused pins
Appendix 6. Machine instructions
Appendix 7. Hexadecimal instruction code table
Appendix 8. Countermeasure against noise
Appendix 9. 7721 Group Q & A
Appendix 10. Differences between 7721 Group and 7720 Group
Appendix 11. Electrical characteristics
Appendix 12. Standard characteristics

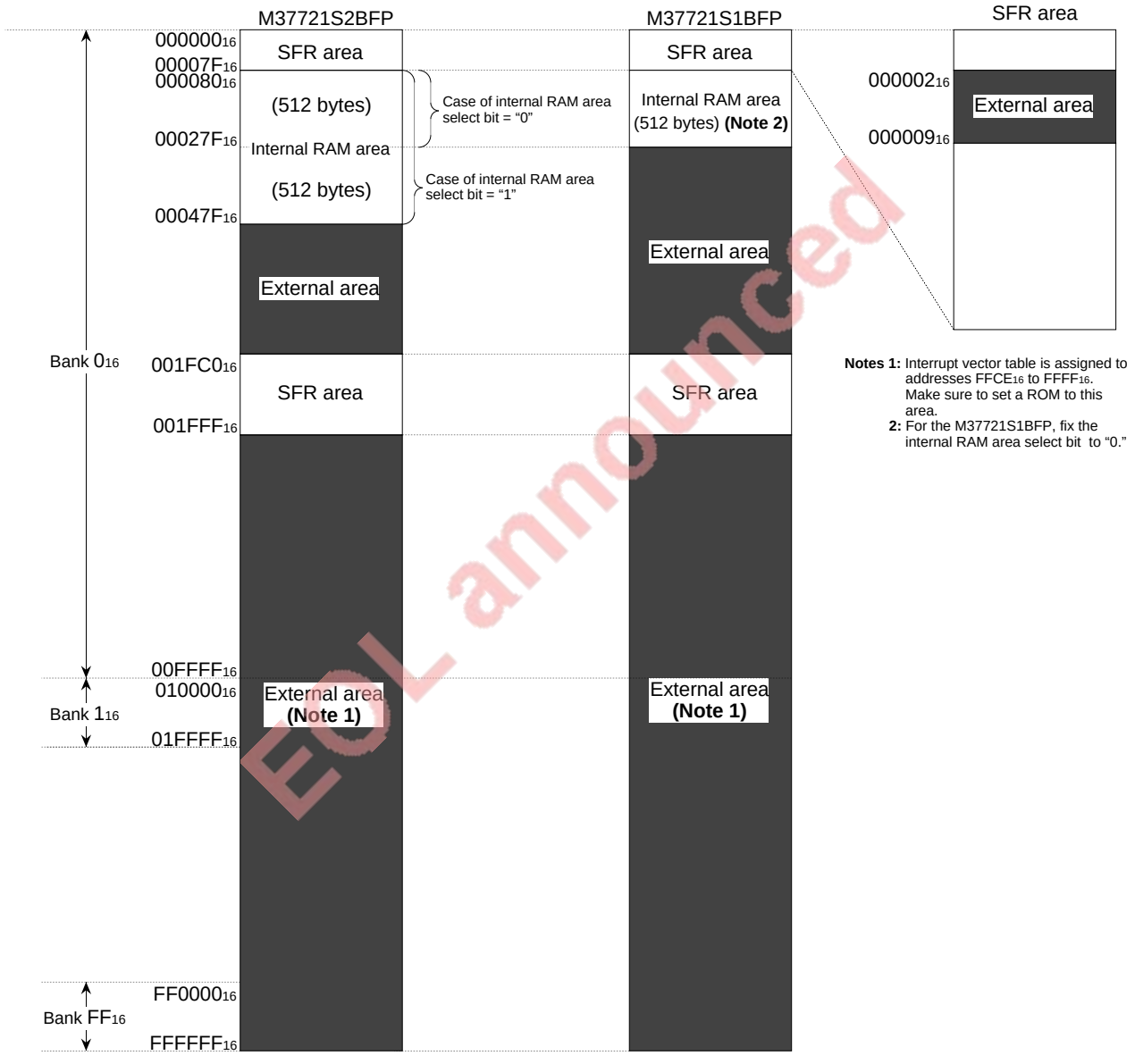
EOL announced

APPENDIX

Appendix 1. Memory assignment of 7721 Group

Appendix 1. Memory assignment of 7721 Group

Microprocessor mode



EOL announced

Fig. 1 Memory assignment (microprocessor mode)

Appendix 2. Memory assignment in SFR area

Access characteristics

RW : It is possible to read the bit state at reading. The written value becomes valid.
RO : It is possible to read the bit state at reading. The written value becomes invalid.
WO : The written value becomes valid. It is impossible to read the bit state.
: Nothing is assigned. It is impossible to read the bit state. The written value becomes invalid.

State immediately after reset

0 : "0" immediately after reset.
1 : "1" immediately after reset.
? : Undefined immediately after reset.

[0] : Always "0" at reading.
[1] : Always "1" at reading.
[?] : Always undefined at reading.
[0] : "0" immediately after reset. Fix this bit to "0."

Address	Register name	Access characteristics b7 b0	State immediately after reset b7 b0
0₁₆			?
1₁₆			?
2₁₆			?
3₁₆			?
4₁₆			?
5₁₆			?
6₁₆			?
7₁₆			?
8₁₆			?
9₁₆			?
A₁₆	Port P4 register	RW	? 0 0 0
B₁₆	Port P5 register	RW	?
C₁₆	Port P4 direction register	RW	0 0 0 0 0 0 0 0
D₁₆	Port P5 direction register	RW	00₁₆
E₁₆	Port P6 register	RW	?
F₁₆	Port P7 register	RW	?
10₁₆	Port P6 direction register	RW	00₁₆
11₁₆	Port P7 direction register	RW	00₁₆
12₁₆	Port P8 register	RW	?
13₁₆	Port P9 register	RW	?
14₁₆	Port P8 direction register	RW	00₁₆
15₁₆	Port P9 direction register	RW	00₁₆
16₁₆	Port P10 register	RW	?
17₁₆			?
18₁₆	Port P10 direction register	RW	00₁₆
19₁₆			?
1A₁₆	Pulse output data register 0	WO	?
1B₁₆			?
1C₁₆	Pulse output data register 1	WO	?
1D₁₆			?
1E₁₆	A-D control register	RW	0 0 0 0 0 ? ? ?
1F₁₆	A-D sweep pin select register	RW	? ? ? ? ? ? 1 1

EOL announced

APPENDIX

Appendix 2. Memory assignment in SFR area

Access characteristics

RW : It is possible to read the bit state at reading. The written value becomes valid.
RO : It is possible to read the bit state at reading. The written value becomes invalid.
WO : The written value becomes valid. It is impossible to read the bit state.
: Nothing is assigned. It is impossible to read the bit state. The written value becomes invalid.

State immediately after a reset

0 : "0" immediately after reset.
1 : "1" immediately after reset.
? : Undefined immediately after reset.

0 : Always "0" at reading.
1 : Always "1" at reading.
? : Always undefined at reading.
0 : "0" immediately after reset. Fix this bit to "0."

Address	Register name	Access characteristics b7 b0	State immediately after reset b7 b0
2016	A-D register 0	RO	?
2116			?
2216	A-D register 1	RO	?
2316			?
2416	A-D register 2	RO	?
2516			?
2616	A-D register 3	RO	?
2716			?
2816	A-D register 4	RO	?
2916			?
2A16	A-D register 5	RO	?
2B16			?
2C16	A-D register 6	RO	?
2D16			?
2E16	A-D register 7	RO	?
2F16			?
3016	UART0 transmit/receive mode register	RW	0016
3116	UART0 baud rate register	WO	?
3216	UART0 transmit buffer register	WO	?
3316		WO	?
3416	UART0 transmit/receive control register 0	RO RW	? ? ? ? 1 0 0 0
3516	UART0 transmit/receive control register 1	RO RW RO RW	0 0 0 0 0 0 1 0
3616	UART0 receive buffer register	RO	?
3716		RO	0 0 0 0 0 0 0 ?
3816	UART1 transmit/receive mode register	RW	0016
3916	UART1 baud rate register	WO	?
3A16	UART1 transmit buffer register	WO	?
3B16		WO	?
3C16	UART1 transmit/receive control register 0	RO RW	? ? ? ? 1 0 0 0
3D16	UART1 transmit/receive control register 1	RO RW RO RW	0 0 0 0 0 0 1 0
3E16	UART1 receive buffer register	RO	?
3F16		RO	0 0 0 0 0 0 0 ?

EOL announced

Access characteristics

RW : It is possible to read the bit state at reading. The written value becomes valid.
RO : It is possible to read the bit state at reading. The written value becomes invalid.
WO : The written value becomes valid. It is impossible to read the bit state.
: Nothing is assigned. It is impossible to read the bit state. The written value becomes invalid.

State immediately after reset

0 : "0" immediately after reset.
1 : "1" immediately after reset.
? : Undefined immediately after reset.

0 : Always "0" at reading.
1 : Always "1" at reading.
? : Always undefined at reading.
0 : "0" immediately after reset. Fix this bit to "0."

Address	Register name	Access characteristics b7 b0	State immediately after reset b7 b0
4016	Count start register	RW	0016
4116			?
4216	One-shot start register	WO	? 0 0 0 0 0
4316			?
4416	Up-down register	WO RW	0 0 0 0 0 0 0 0
4516			?
4616	Timer A0 register	RW	?
4716		RW	?
4816	Timer A1 register	RW	?
4916		RW	?
4A16	Timer A2 register	**(Note 1)**	?
4B16		**(Note 1)**	?
4C16	Timer A3 register	**(Note 1)**	?
4D16		**(Note 1)**	?
4E16	Timer A4 register	**(Note 1)**	?
4F16		**(Note 1)**	?
5016	Timer B0 register	**(Note 2)**	?
5116		**(Note 2)**	?
5216	Timer B1 register	**(Note 2)**	?
5316		**(Note 2)**	?
5416	Timer B2 register	RW	?
5516		RW	?
5616	Timer A0 mode register	RW	0 0 0 0 0 0 0 0
5716	Timer A1 mode register	RW	0 0 0 0 0 0 0 0
5816	Timer A2 mode register	RW	0016
5916	Timer A3 mode register	RW	0016
5A16	Timer A4 mode register	RW	0016
5B16	Timer B0 mode register	RW (Note 3) RW	0 0 ? ? 0 0 0 0
5C16	Timer B1 mode register	RW (Note 3) RW	0 0 ? ? 0 0 0 0
5D16	Timer B2 mode register	RW (Note 3) RW	0 0 ? ? 0 0 0 0
5E16	Processor mode register 0	RW WO RW RW	0 0 0 0 0 0 1 0
5F16	Processor mode register 1	RW	? (Note 4) ?

EOL announced

Notes 1: The access characteristics at addresses 4A16 to 4F16 vary according to Timer A's operating mode. (Refer to **"CHAPTER 8. TIMER A."**)
2: The access characteristics at addresses 5016 to 5316 vary according to Timer B's operating mode. (Refer to **"CHAPTER 9. TIMER B."**)
3: The access characteristics for bit 5 at addresses 5B16 and 5C16 vary according to Timer B's operating mode. Bit 5 at address 5D16 is invalid. (Refer to **"CHAPTER 9. TIMER B."**)
4: Bit 1 at address 5F16 becomes "0" immediately after reset. For the M37721S1BFP, fix this bit to "0."

APPENDIX

Appendix 2. Memory assignment in SFR area

Access characteristics

RW : It is possible to read the bit state at reading. The written value becomes valid.
RO : It is possible to read the bit state at reading. The written value becomes invalid.
WO : The written value becomes valid. It is impossible to read the bit state.
: Nothing is assigned. It is impossible to read the bit state. The written value becomes invalid.

State immediately after reset

0 : "0" immediately after reset.
1 : "1" immediately after reset.
? : Undefined immediately after reset.

0 : Always "0" at reading.
1 : Always "1" at reading.
? : Always undefined at reading.
0 : "0" immediately after reset. Fix this bit to "0."

Address	Register name	Access characteristics b7 b0	State immediately after reset b7 b0
6016	Watchdog timer register	(Note 5)	?(Note 6)
6116	Watchdog timer frequency select register	RW	? 0
6216	Real-time output control register	RW	0 0 0 0 0 0 0 0
6316			?
6416	DRAM control register	RW RW	0 0 0 0 0 0 0 0
6516			?
6616	Refresh timer	WO	?
6716			?
6816	DMAC control register L	(Note 7) RW	0 0 0 0 ? ? 0 0
6916	DMAC control register H	RW WO	0 0 0 0 0 0 0 0
6A16			?
6B16			?
6C16	DMA0 interrupt control register	RW	? 0 0 0 0
6D16	DMA1 interrupt control register	RW	? 0 0 0 0
6E16	DMA2 interrupt control register	RW	? 0 0 0 0
6F16	DMA3 interrupt control register	RW	? 0 0 0 0
7016	A-D conversion interrupt control register	RW	? 0 0 0 0
7116	UART0 transmit interrupt control register	RW	? 0 0 0 0
7216	UART0 receive interrupt control register	RW	? 0 0 0 0
7316	UART1 transmit interrupt control register	RW	? 0 0 0 0
7416	UART1 receive interrupt control register	RW	? 0 0 0 0
7516	Timer A0 interrupt control register	RW	? 0 0 0 0
7616	Timer A1 interrupt control register	RW	? 0 0 0 0
7716	Timer A2 interrupt control register	RW	? 0 0 0 0
7816	Timer A3 interrupt control register	RW	? 0 0 0 0
7916	Timer A4 interrupt control register	RW	? 0 0 0 0
7A16	Timer B0 interrupt control register	RW	? 0 0 0 0
7B16	Timer B1 interrupt control register	RW	? 0 0 0 0
7C16	Timer B2 interrupt control register	RW	? 0 0 0 0
7D16	$\overline{INT_0}$ interrupt control register	RW	? 0 0 0 0 0 0
7E16	$\overline{INT_1}$ interrupt control register	RW	? 0 0 0 0 0 0
7F16	$\overline{INT_2}$ interrupt control register	RW	? 0 0 0 0 0 0

Not announced

Notes 5: By writing dummy data to address 6016, the value "FFF16" is set to the watchdog timer. The dummy data is not retained anywhere.
6: The value "FFF16" is set to the watchdog timer. (Refer to "**CHAPTER 15. WATCHDOG TIMER**.")
7: It is possible to read the bit state at reading. When writing "0" to this bit, this bit becomes "0." But when writing "1" to this bit, this bit does not change.

Access characteristics

RW : It is possible to read the bit state at reading. The written value becomes valid.
RO : It is possible to read the bit state at reading. The written value becomes invalid.
WO : The written value becomes valid. It is impossible to read the bit state.
[ ] : Nothing is assigned. It is impossible to read the bit state. The written value becomes invalid.

State immediately after reset

0 : "0" immediately after reset.
1 : "1" immediately after reset.
? : Undefined immediately after reset.

[0] : Always "0" at reading.
[1] : Always "1" at reading.
[?] : Always undefined at reading.
[0] : "0" immediately after reset. Fix this bit to "0."

Address	Register name	Access characteristics b7 b0	State immediately after reset b7 b0
1FC0₁₆		RW	?
1FC1₁₆	Source address register 0	RW	?
1FC2₁₆		RW	?
1FC3₁₆			?
1FC4₁₆		RW	?
1FC5₁₆	Destination address register 0	RW	?
1FC6₁₆		RW	?
1FC7₁₆			?
1FC8₁₆		RW	?
1FC9₁₆	Transfer counter register 0	RW	?
1FCA₁₆		RW	?
1FCB₁₆			?
1FCC₁₆	DMA0 mode register L	RW	0 0 0 0 [0] 0 0 0
1FCD₁₆	DMA0 mode register H	RW	0 0 0 0 [0] [0] 0 0
1FCE₁₆	DMA0 control register	[ ] [ ] RW	? ? 0 0 0 0 0 0
1FCF₁₆			?
1FD0₁₆		RW	?
1FD1₁₆	Source address register 1	RW	?
1FD2₁₆		RW	?
1FD3₁₆			?
1FD4₁₆		RW	?
1FD5₁₆	Destination address register 1	RW	?
1FD6₁₆		RW	?
1FD7₁₆			?
1FD8₁₆		RW	?
1FD9₁₆	Transfer counter register 1	RW	?
1FDA₁₆		RW	?
1FDB₁₆			?
1FDC₁₆	DMA1 mode register L	RW	0 0 0 0 [0] 0 0 0
1FDD₁₆	DMA1 mode register H	RW	0 0 0 0 [0] [0] 0 0
1FDE₁₆	DMA1 control register	[ ] [ ] RW	? ? 0 0 0 0 0 0
1FDF₁₆			?

EOL announced

Access characteristics

RW : It is possible to read the bit state at reading. The written value becomes valid.
RO : It is possible to read the bit state at reading. The written value becomes invalid.
WO : The written value becomes valid. It is impossible to read the bit state.
: Nothing is assigned. It is impossible to read the bit state. The written value becomes invalid.

State immediately after reset

0 : "0" immediately after reset.
1 : "1" immediately after reset.
? : Undefined immediately after reset.

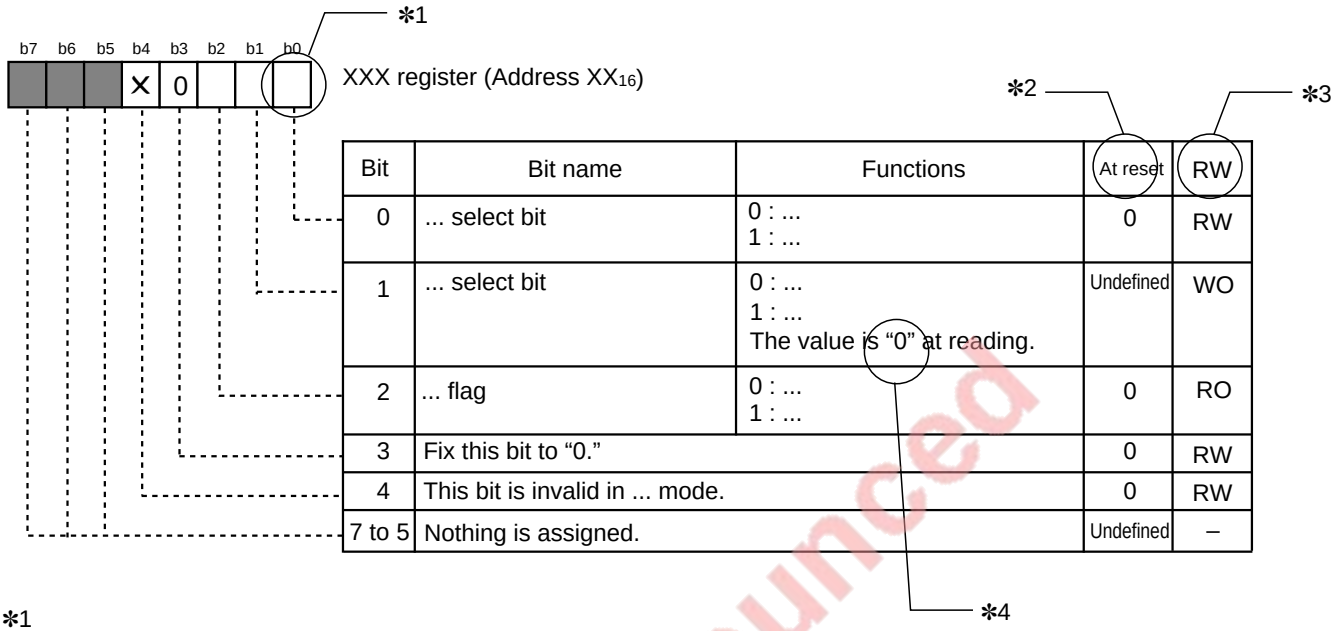
[0] : Always "0" at reading.
[1] : Always "1" at reading.
[?] : Always undefined at reading.
[0] : "0" immediately after reset. Fix this bit to "0."

Address	Register name	Access characteristics b7 b0	State immediately after reset b7 b0
1FE0₁₆		RW	?
1FE1₁₆	Source address register 2	RW	?
1FE2₁₆		RW	?
1FE3₁₆			?
1FE4₁₆		RW	?
1FE5₁₆	Destination address register 2	RW	?
1FE6₁₆		RW	?
1FE7₁₆			?
1FE8₁₆		RW	?
1FE9₁₆	Transfer counter register 2	RW	?
1FEA₁₆		RW	?
1FEB₁₆			?
1FEC₁₆	DMA2 mode register L	RW	0 0 0 0 0 0 0 0
1FED₁₆	DMA2 mode register H	RW	0 0 0 0 0 0 0 0
1FEE₁₆	DMA2 control register	RW	? ? 0 0 0 0 0 0
1FEF₁₆			?
1FF0₁₆		RW	?
1FF1₁₆	Source address register 3	RW	?
1FF2₁₆		RW	?
1FF3₁₆			?
1FF4₁₆		RW	?
1FF5₁₆	Destination address register 3	RW	?
1FF6₁₆		RW	?
1FF7₁₆			?
1FF8₁₆		RW	?
1FF9₁₆	Transfer counter register 3	RW	?
1FFA₁₆		RW	?
1FFB₁₆			?
1FFC₁₆	DMA3 mode register L	RW	0 0 0 0 0 0 0 0
1FFD₁₆	DMA3 mode register H	RW	0 0 0 0 0 0 0 0
1FFE₁₆	DMA3 control register	RW	? ? 0 0 0 0 0 0
1FFF₁₆			?

EOL announced

Appendix 3. Control registers

The control registers allocated in the SFR area are shown on the following pages.
Below is the structure diagram for all registers.



Bit	Bit name	Functions	At reset	RW
0	... select bit	0 : ... 1 : ...	0	RW
1	... select bit	0 : ... 1 : ... The value is “0” at reading.	Undefined	WO
2	... flag	0 : ... 1 : ...	0	RO
3	Fix this bit to “0.”		0	RW
4	This bit is invalid in ... mode.		0	RW
7 to 5	Nothing is assigned.		Undefined	–

✻1

Blank : Set to “0” or “1” according to the usage.
0 : Set to “0” at writing.
1 : Set to “1” at writing.
X : Invalid depending on the mode or state. It may be “0” or “1.”
: Nothing is assigned.

✻2

0 : “0” immediately after reset.
1 : “1” immediately after reset.
Undefined : Undefined immediately after reset.

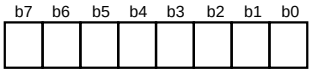
✻3

RW : It is possible to read the bit state at reading. The written value becomes valid.
RO : It is possible to read the bit state at reading. The written value becomes invalid. Accordingly, the written value may be “0” or “1.”
WO : The written value becomes valid. It is impossible to read the bit state. The value is undefined at reading. However, when [“0” is at reading”] is indicated in the “Function” or “Note” column, the bit is always “0” at reading. (See ✻4 above.)
— : It is impossible to read the bit state. The value is undefined at reading. However, when [“0” is at reading”] is indicated in the “Function” or “Note” column, the bit is always “0” at reading. (See ✻4 above.)
The written value becomes invalid. Accordingly, the written value may be “0” or “1.”

FOE announced

Appendix 3. Control registers

Port Pi register

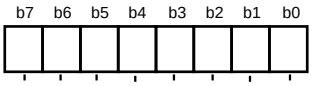


Port Pi register (i = 4 to 10)
(Addresses A_{16}, B_{16}, E_{16}, F_{16}, 12_{16}, 13_{16}, 16_{16})

Bit	Bit name	Functions	At reset	RW
0	Port Pi_0's pin	Data is input from or output to a pin by reading from or writing to the corresponding bit. 0 : "L" level 1 : "H" level	Undefined	RW
1	Port Pi_1's pin		Undefined	RW
2	Port Pi_2's pin		Undefined	RW
3	Port Pi_3's pin		Undefined	RW
4	Port Pi_4's pin		Undefined	RW
5	Port Pi_5's pin		Undefined	RW
6	Port Pi_6's pin		Undefined	RW
7	Port Pi_7's pin		Undefined	RW

Note: For bits 0 to 2 of the port P4 register, nothing is assigned and these bits are fixed to "0" at reading.

Port Pi direction register



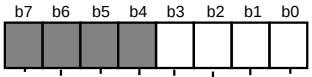
Port Pi direction register (i = 4 to 10)
(Addresses C_{16}, D_{16}, 10_{16}, 11_{16}, 14_{16}, 15_{16}, 18_{16})

Bit	Bit name	Functions	At reset	RW
0	Port Pi_0 direction bit	0 : Input mode (The port functions as an input port) 1 : Output mode (The port functions as an output port)	0	RW
1	Port Pi_1 direction bit		0	RW
2	Port Pi_2 direction bit		0	RW
3	Port Pi_3 direction bit		0	RW
4	Port Pi_4 direction bit		0	RW
5	Port Pi_5 direction bit		0	RW
6	Port Pi_6 direction bit		0	RW
7	Port Pi_7 direction bit		0	RW

Note: For bits 0 to 2 of the port P4 direction register, nothing is assigned and these bits are fixed to "0" at reading.

EOL announced

Pulse output data register 0

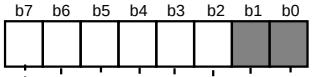


Pulse output data register 0 (Address $1A_{16}$)

Bit	Bit name	Functions	At reset	RW
0	$RTP0_0$ pulse output data bit	0 : "L" level output 1 : "H" level output	Undefined	WO
1	$RTP0_1$ pulse output data bit		Undefined	WO
2	$RTP0_2$ pulse output data bit (Valid in pulse mode 0)		Undefined	WO
3	$RTP0_3$ pulse output data bit (Valid in pulse mode 0)		Undefined	WO
7 to 4	Nothing is assigned.		Undefined	—

Note: Use the **LDM** or **STA** instruction for writing to this register

Pulse output data register 1



Pulse output data register 1 (Address $1C_{16}$)

Bit	Bit name	Functions	At reset	RW
0, 1	Nothing is assigned.		Undefined	—
2	$RTP0_2$ pulse output data bit (Valid in pulse mode 1)	0 : "L" level output 1 : "H" level output	Undefined	WO
3	$RTP0_3$ pulse output data bit (Valid in pulse mode 1)		Undefined	WO
4	$RTP1_0$ pulse output data bit		Undefined	WO
5	$RTP1_1$ pulse output data bit		Undefined	WO
6	$RTP1_2$ pulse output data bit		Undefined	WO
7	$RTP1_3$ pulse output data bit		Undefined	WO

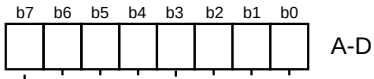
Note: Use the **LDM** or **STA** instruction for writing to this register.

Eol announced

APPENDIX

Appendix 3. Control registers

A-D control register

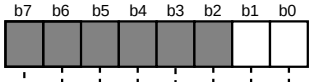


A-D control register (Address $1E_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Analog input select bits (Valid in one-shot and repeat modes) **(Note 1)**	b2 b1 b0 0 0 0 : AN_0 selected 0 0 1 : AN_1 selected 0 1 0 : AN_2 selected 0 1 1 : AN_3 selected 1 0 0 : AN_4 selected 1 0 1 : AN_5 selected 1 1 0 : AN_6 selected 1 1 1 : AN_7 selected **(Note 2)**	Undefined	RW
1			Undefined	RW
2			Undefined	RW
3	A-D operation mode select bit	b4 b3 0 0 : One-shot mode 0 1 : Repeat mode 1 0 : Single sweep mode 1 1 : Repeat sweep mode	0	RW
4			0	RW
5	Trigger select bit	0 : Internal trigger 1 : External trigger	0	RW
6	A-D conversion start bit	0 : Stop A-D conversion 1 : Start A-D conversion	0	RW
7	A-D conversion frequency (ϕ_{AD}) select bit	0 : f_2 divided by 4 1 : f_2 divided by 2	0	RW

Notes 1: These bits are invalid in the single sweep and repeat sweep mode. (They may be either "0" or "1.")
2: When selecting an external trigger, the AN_7 pin cannot be used as an analog input pin.
3: Writing to each bit (except bit 6) of the A-D control register must be performed while the A-D converter halts.

A-D sweep pin select register



A-D sweep pin select register (Address $1F_{16}$)

Bit	Bit name	Functions	At reset	RW
0	A-D sweep pin select bits (Valid in single sweep and repeat sweep modes) **(Note 1)**	b1 b0 0 0 : AN_0, AN_1 (2 pins) 0 1 : AN_0 to AN_3 (4 pins) 1 0 : AN_0 to AN_5 (6 pins) 1 1 : AN_0 to AN_7 (8 pins) **(Note 2)**	1	RW
1			1	RW
7 to 2	Nothing is assigned.		Undefined	—

Notes 1: These bits are invalid in the one-shot and repeat modes. (They may be either "0" or "1.")
2: When selecting an external trigger, the AN_7 pin cannot be used as an analog input pin.
3: Writing to each bit of the A-D sweep pin select register must be performed while the A-D converter halts.

A-D register i

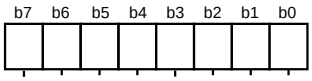


A-D register i (i = 0 to 7) (Addresses 20_{16}, 22_{16}, 24_{16}, 26_{16}, 28_{16}, $2A_{16}$, $2C_{16}$, $2E_{16}$)

Bit	Functions	At reset	RW
7 to 0	Reads an A-D conversion result.	Undefined	RO

EOL announced

UARTi transmit/receive mode register

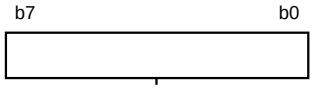


UART0 transmit/receive mode register (Address 30_{16})
UART1 transmit/receive mode register (Address 38_{16})

Bit	Bit name	Functions	At reset	RW
0	Serial I/O mode select bits	b2 b1 b0 0 0 0 : Serial I/O disabled (P8 functions as a programmable I/O port.) 0 0 1 : Clock synchronous serial I/O mode 0 1 0 : Do not select. 0 1 1 : Do not select. 1 0 0 : UART mode (Transfer data length = 7 bits) 1 0 1 : UART mode (Transfer data length = 8 bits) 1 1 0 : UART mode (Transfer data length = 9 bits) 1 1 1 : Do not select.	0	RW
1			0	RW
2			0	RW
3	Internal/External clock select bit	0 : Internal clock 1 : External clock	0	RW
4	Stop bit length select bit (Valid in UART mode) (**Note**)	0 : One stop bit 1 : Two stop bits	0	RW
5	Odd/Even parity select bit (Valid in UART mode when parity enable bit is "1") (**Note**)	0 : Odd parity 1 : Even parity	0	RW
6	Parity enable bit (Valid in UART mode) (**Note**)	0 : Parity disabled 1 : Parity enabled	0	RW
7	Sleep select bit (Valid in UART mode) (**Note**)	0 : Sleep mode terminated (Invalid) 1 : Sleep mode selected	0	RW

Note: Bits 4 to 6 are invalid in the clock synchronous serial I/O mode. (They may be either "0" or "1.") Additionally, fix bit 7 to "0."

UARTi baud rate register



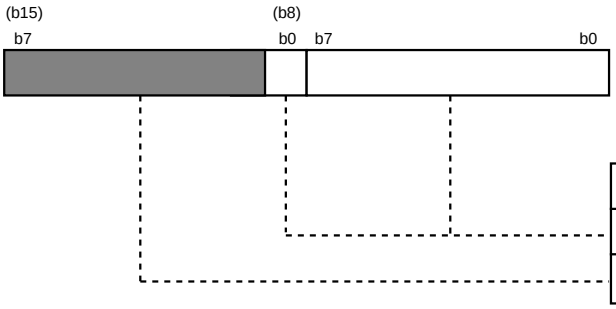
UART0 baud rate register (Address 31_{16})
UART1 baud rate register (Address 39_{16})

Bit	Functions	At reset	RW
7 to 0	Can be set to "00_{16}" to "FF_{16}." Assuming that the set value = n, BRGi divides the count source frequency by (n + 1).	Undefined	WO

Note: Writing to this register must be performed while the transmission/reception halts. Use the **LDM** or **STA** instruction for writing to this register.

EOL announced

UARTi transmit buffer register

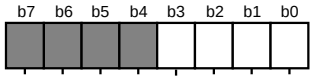


UART0 transmit buffer register (Addresses 33_{16}, 32_{16})
UART1 transmit buffer register (Addresses $3B_{16}$, $3A_{16}$)

Bit	Functions	At reset	RW
8 to 0	Transmit data is set.	Undefined	WO
15 to 9	Nothing is assigned.	Undefined	—

Note: Use the **LDM** or **STA** instruction for writing to this register.

UARTi transmit/receive control register 0

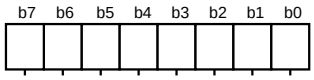


UART0 transmit/receive control register 0 (Address 34_{16})
UART1 transmit/receive control register 0 (Address $3C_{16}$)

Bit	Bit name	Functions	At reset	RW
0	BRG count source select bits	b1 b0 0 0 : f_2 0 1 : f_{16} 1 0 : f_{64} 1 1 : f_{512}	0	RW
1			0	RW
2	$\overline{CTS}$/$\overline{RTS}$ select bit	0 : $\overline{CTS}$ function selected 1 : $\overline{RTS}$ function selected	0	RW
3	Transmit register empty flag	0 : Data present in transmit register (During transmission) 1 : No data present in transmit register (Transmission completed)	1	RO
7 to 4	Nothing is assigned.		Undefined	—

EOL announced

UARTi transmit/receive control register 1



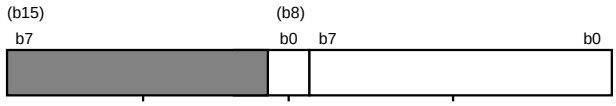
UART0 transmit/receive control register 1 (Address 35_{16})
UART1 transmit/receive control register 1 (Address $3D_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Transmit enable bit	0 : Transmission disabled 1 : Transmission enabled	0	RW
1	Transmit buffer empty flag	0 : Data present in transmit buffer register 1 : No data present in transmit buffer register	1	RO
2	Receive enable bit	0 : Reception disabled 1 : Reception enabled	0	RW
3	Receive complete flag	0 : No data present in receive buffer register 1 : Data present in receive buffer register	0	RO
4	Overrun error flag **(Note 1)**	0 : No overrun error 1 : Overrun error detected	0	RO
5	Framing error flag **(Notes 1, 2)** (Valid in UART mode)	0 : No framing error 1 : Framing error detected	0	RO
6	Parity error flag **(Notes 1, 2)** (Valid in UART mode)	0 : No parity error 1 : Parity error detected	0	RO
7	Error sum flag **(Notes 1, 2)** (Valid in UART mode)	0 : No error 1 : Error detected	0	RO

Notes 1: Bit 4 is cleared to "0" when the receive enable bit is cleared to "0" or when the serial I/O mode select bits (bits 2 to 0 at addresses 30_{16}, 38_{16}) are cleared to "000_2."
Bits 5 and 6 are cleared to "0" when one of the following is performed:
•Clearing the receive enable bit to "0"
•Reading the low-order byte of the UARTi receive buffer register (addresses 36_{16}, $3E_{16}$) out
•Clearing the serial I/O mode select bits (bits 2 to 0 at addresses 30_{16}, 38_{16}) to "000_2"
Bit 7 is cleared to "0" when all of bits 4 to 6 become "0."
2: Bits 5 to 7 are invalid in the clock synchronous serial I/O mode.

FOR Announced

UARTi receive buffer register



UART0 receive buffer register (Addresses 37_{16}, 36_{16})
UART1 receive buffer register (Addresses $3F_{16}$, $3E_{16}$)

Bit	Functions	At reset	RW
8 to 0	Receive data is read out from here.	Undefined	RO
15 to 9	Nothing is assigned. The value is "0" at reading.	0	–

APPENDIX

Appendix 3. Control registers

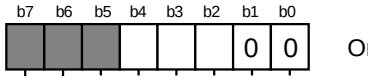
Count start register



Count start register (Address 40_{16})

Bit	Bit name	Functions	At reset	RW
0	Timer A0 count start bit	0 : Stop counting 1 : Start counting	0	RW
1	Timer A1 count start bit		0	RW
2	Timer A2 count start bit		0	RW
3	Timer A3 count start bit		0	RW
4	Timer A4 count start bit		0	RW
5	Timer B0 count start bit		0	RW
6	Timer B1 count start bit		0	RW
7	Timer B2 count start bit		0	RW

One-shot start register

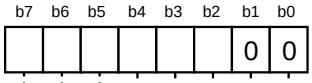


One-shot start register (Address 42_{16})

Bit	Bit name	Functions	At reset	RW
0	Fix these bits to "0." The value is "0" at reading.		0	WO
1			0	WO
2	Timer A2 one-shot start bit	1 : Start outputting one-shot pulse (valid when internal trigger is selected.) The value is "0" at reading.	0	WO
3	Timer A3 one-shot start bit		0	WO
4	Timer A4 one-shot start bit		0	WO
7 to 5	Nothing is assigned.		Undefined	—

Not announced

Up-down register



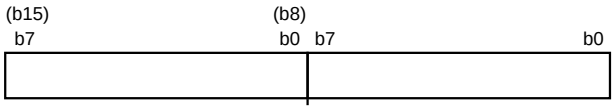
Up-down register (Address 44_{16})

Bit	Bit name	Functions	At reset	RW
0	Fix these bits to "0."		0	RW
1			0	RW
2	Timer A2 up-down bit	0 : Countdown 1 : Countup This function is valid when the contents of the up-down register is selected as the up-down switching factor.	0	RW
3	Timer A3 up-down bit		0	RW
4	Timer A4 up-down bit		0	RW
5	Timer A2 two-phase pulse signal processing select bit **(Note)**	0 : Two-phase pulse signal processing function disabled 1 : Two-phase pulse signal processing function enabled When not using the two-phase pulse signal processing function, set the bit to "0." The value is "0" at reading.	0	WO
6	Timer A3 two-phase pulse signal processing select bit **(Note)**		0	WO
7	Timer A4 two-phase pulse signal processing select bit **(Note)**		0	WO

Note: Use the **LDM** or **STA** instruction for writing to bits 5 to 7.

EOL announced

Timer Ai register

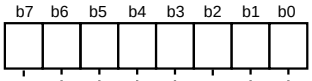


Timer A0 register (Addresses 47_{16}, 46_{16})
Timer A1 register (Addresses 49_{16}, 48_{16})
Timer A2 register (Addresses $4B_{16}$, $4A_{16}$)
Timer A3 register (Addresses $4D_{16}$, $4C_{16}$)
Timer A4 register (Addresses $4F_{16}$, $4E_{16}$)

Bit	Functions	At reset	RW
15 to 0	These bits have different functions according to the operating mode.	Undefined	RW **(Note 1)**

Notes 1: The access characteristics for the timer A2 register, timer A3 register, and timer A4 register differ according to Timer A's operating mode.
2: Read from or write to this register in a unit of 16 bits.

Timer Ai mode register

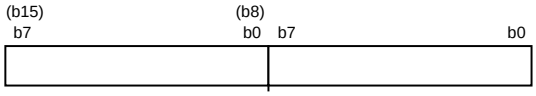


Timer Ai mode register (i = 0 to 4) (Addresses 56_{16} to $5A_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 0 0 : Timer mode 0 1 : Event counter mode 1 0 : One-shot pulse mode 1 1 : Pulse width modulation (PWM) mode	0	RW
1			0	RW
2	These bits have different functions according to the operating mode.		0	RW
3			0	RW
4			0	RW
5			0	RW
6			0	RW
7			0	RW

EOL announced

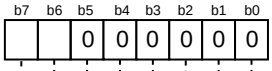
Timer Mode



Timer A0 register (Addresses 47_{16}, 46_{16})
Timer A1 register (Addresses 49_{16}, 48_{16})
Timer A2 register (Addresses $4B_{16}$, $4A_{16}$)
Timer A3 register (Addresses $4D_{16}$, $4C_{16}$)
Timer A4 register (Addresses $4F_{16}$, $4E_{16}$)

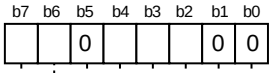
Bit	Functions	At reset	RW
15 to 0	These bits can be set to "0000_{16}" to "$FFFF_{16}$." Assuming that the set value = n, the counter divides the count source frequency by (n + 1). When reading, the register indicates the counter value.	Undefined	RW

Note: Read from or write to this register in a unit of 16 bits.



Timer A0 mode register (Address 56_{16})
Timer A1 mode register (Address 57_{16})

Bit	Bit name	Functions	At reset	RW
0	Fix these bits to "0."		0	RW
1			0	RW
2			0	RW
3			0	RW
4			0	RW
5			0	RW
6	Count source select bits	b7 b6 0 0 : f_2 0 1 : f_{16} 1 0 : f_{64} 1 1 : f_{512}	0	RW
7			0	RW

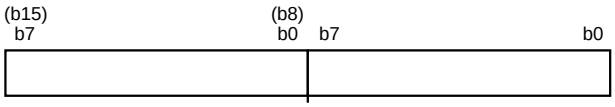


Timer Aj mode register (j = 2 to 4) (Addresses 58_{16} to $5A_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 0 0 : Timer mode	0	RW
1			0	RW
2	Pulse output function select bit	0 : No pulse output (TAj$_{OUT}$ pin functions as a programmable I/O port.) 1 : Pulse output (TAj$_{OUT}$ pin functions as a pulse output pin.)	0	RW
3	Gate function select bits	b4 b3 0 0 : No gate function 0 1 : (TAj$_{IN}$ pin functions as a programmable I/O port.) 1 0 : Counter counts only while TAj$_{IN}$ pin's input signal is at "L" level. 1 1 : Counter counts only while TAj$_{IN}$ pin's input signal is at "H" level.	0	RW
4			0	RW
5	Fix this bit to "0" in timer mode.		0	RW
6	Count source select bits	b7 b6 0 0 : f_2 0 1 : f_{16} 1 0 : f_{64} 1 1 : f_{512}	0	RW
7			0	RW

EOL announced

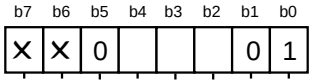
Event counter mode



Timer A2 register (Addresses $4B_{16}$, $4A_{16}$)
Timer A3 register (Addresses $4D_{16}$, $4C_{16}$)
Timer A4 register (Addresses $4F_{16}$, $4E_{16}$)

Bit	Functions	At reset	RW
15 to 0	These bits can be set to "0000_{16}" to "$FFFF_{16}$." Assuming that the set value = n, the counter divides the count source frequency by (n + 1) during countdown, or by ($FFFF_{16}$ – n + 1) during countup. When reading, the register indicates the counter value.	Undefined	RW

Note: Read from or write to this register in a unit of 16 bits.

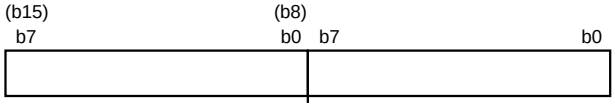


Timer Aj mode register (j = 2 to 4) (Addresses 58_{16} to $5A_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 0 1 : Event counter mode	0	RW
1			0	RW
2	Pulse output function select bit	0 : No pulse output (TAj$_{OUT}$ pin functions as a programmable I/O port.) 1 : Pulse output (TAj$_{OUT}$ pin functions as a pulse output pin.)	0	RW
3	Count polarity select bit	0 : Counts at falling edge of external signal 1 : Counts at rising edge of external signal	0	RW
4	Up-down switching factor select bit	0 : Contents of up-down register 1 : Input signal to TAj$_{OUT}$ pin	0	RW
5	Fix this bit to "0" in event counter mode.		0	RW
6	These bits are invalid in event counter mode.		0	RW
7			0	RW

EOL announced

One-shot pulse mode

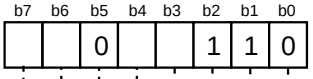


Timer A2 register (Addresses $4B_{16}$, $4A_{16}$)
Timer A3 register (Addresses $4D_{16}$, $4C_{16}$)
Timer A4 register (Addresses $4F_{16}$, $4E_{16}$)

Bit	Functions	At reset	RW
15 to 0	These bits can be set to "0001_{16}" to "$FFFF_{16}$." Assuming that the set value = n, the "H" level width of the one-shot pulse output from the TAj_{OUT} pin is expressed as follows : $\frac{n}{f_i}$	Undefined	WO

f_i: Frequency of count source (f_2 , f_{16}, f_{64}, or f_{512})

Note: Use the **LDM** or **STA** instruction for writing to this register.
Read from or write to this register in a unit of 16 bits.



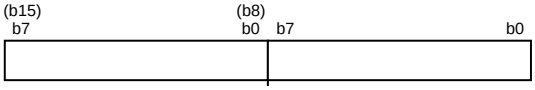
Timer Aj mode register (j = 2 to 4) (Addresses 58_{16} to $5A_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 1 0 : One-shot pulse mode	0	RW
1			0	RW
2	Fix this bit to "1" in one-shot pulse mode.		0	RW
3	Trigger select bits	b4 b3 0 0 : } Writing "1" to one-shot start register 0 1 : } (TAj_{IN} pin functions as a programmable I/O port.) 1 0 : Falling edge of TAj_{IN} pin's input signal 1 1 : Rising edge of TAj_{IN} pin's input signal	0	RW
4			0	RW
5	Fix this bit to "0" in one-shot pulse mode.		0	RW
6	Count source select bits	b7 b6 0 0 : f_2 0 1 : f_{16} 1 0 : f_{64} 1 1 : f_{512}	0	RW
7			0	RW

EOL announced

Pulse width modulation (PWM) mode

<When operating as a 16-bit pulse width modulator>

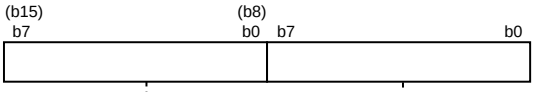


Timer A2 register (Addresses $4B_{16}$, $4A_{16}$)
Timer A3 register (Addresses $4D_{16}$, $4C_{16}$)
Timer A4 register (Addresses $4F_{16}$, $4E_{16}$)

Bit	Functions	At reset	RW
15 to 0	These bits can be set to "0000_{16}" to "$FFFE_{16}$." Assuming that the set value = n, the "H" level width of the PWM pulse output from the TAj$_{OUT}$ pin is expressed as follows: $\frac{n}{f_i}$ (PWM pulse period = $\frac{n^{16}-1}{f_i}$)	Undefined	WO

f_i: Frequency of count source (f_2, f_{16}, f_{64}, or f_{512})
Note: Use the **LDM** or **STA** instruction for writing to this register.
Read from or write to this register in a unit of 16 bits.

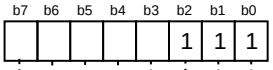
<When operating as an 8-bit pulse width modulator>



Timer A2 register (Addresses $4B_{16}$, $4A_{16}$)
Timer A3 register (Addresses $4D_{16}$, $4C_{16}$)
Timer A4 register (Addresses $4F_{16}$, $4E_{16}$)

Bit	Functions	At reset	RW
7 to 0	These bits can be set to "00_{16}" to "FF_{16}." Assuming that the set value = m, PWM pulse's period output from the TAj$_{OUT}$ pin is expressed as follows: $\frac{(m+1)(2^8-1)}{f_i}$	Undefined	WO
15 to 8	These bits can be set to "00_{16}" to "FE_{16}." Assuming that the set value = n, the "H" level width of the PWM pulse output from the TAj$_{OUT}$ pin is expressed as follows: $\frac{n(m+1)}{f_i}$	Undefined	WO

f_i: Frequency of count source (f_2, f_{16}, f_{64}, or f_{512})
Note: Use the **LDM** or **STA** instruction for writing to this register.
Read from or write to this register in a unit of 16 bits.

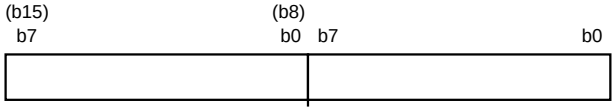


Timer Aj mode register (j = 2 to 4) (Addresses 58_{16} to $5A_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 1 1 : PWM mode	0	RW
1			0	RW
2	Fix this bit to "1" in PWM mode.		0	RW
3	Trigger select bits	b4 b3 0 0 : } Writing "1" to count start register 0 1 : } (TAj$_{IN}$ pin functions as a programmable I/O port.) 1 0 : Falling edge of TAj$_{IN}$ pin's input signal 1 1 : Rising edge of TAj$_{IN}$ pin's input signal	0	RW
4			0	RW
5	16/8-bit PWM mode select bit	0 : 16-bit pulse width modulator 1 : 8-bit pulse width modulator	0	RW
6	Count source select bits	b7 b6 0 0 : f_2 0 1 : f_{16} 1 0 : f_{64} 1 1 : f_{512}	0	RW
7			0	RW

EOL announced

Timer Bi register

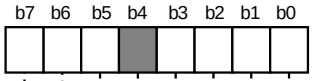


Timer B0 register (Addresses 51_{16}, 50_{16})
Timer B1 register (Addresses 53_{16}, 52_{16})
Timer B2 register (Addresses 55_{16}, 54_{16})

Bit	Functions	At reset	RW
15 to 0	These bits have different functions according to the operating mode.	Undefined	RW **(Note 1)**

Notes 1: The access characteristics for the timer B0 register and timer B1 register differ according to Timer B's operating mode.
2: Read from or write to this register in a unit of 16 bits.

Timer Bi mode register



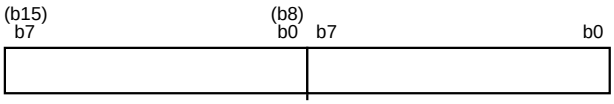
Timer Bi mode register (i = 0 to 2) (Addresses $5B_{16}$ to $5D_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 0 0 : Timer mode 0 1 : Event counter mode 1 0 : Pulse period/Pulse width measurement mode 1 1 : Do not select.	0	RW
1			0	RW
2	These bits have different functions according to the operating mode.		0	RW
3			0	RW
4	Nothing is assigned.		Undefined	—
5	These bits have different functions according to the operating mode.		Undefined	RO **(Note)**
6			0	RW
7			0	RW

Note: Bit 5 is invalid in the timer and event counter modes; its value is undefined at reading.

EOL announced

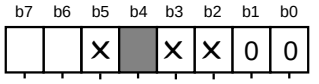
Timer mode



Timer B0 register (Addresses 51_{16}, 50_{16})
Timer B1 register (Addresses 53_{16}, 52_{16})
Timer B2 register (Addresses 55_{16}, 54_{16})

Bit	Functions	At reset	RW
15 to 0	These bits can be set to "0000_{16}" to "$FFFF_{16}$." Assuming that the set value = n, the counter divides the count source frequency by (n + 1). When reading, the register indicates the counter value.	Undefined	RW

Note: Read from or write to this register in a unit of 16 bits.

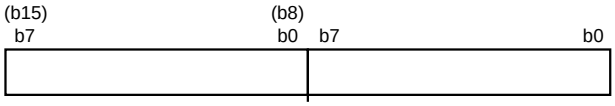


Timer Bi mode register (i = 0 to 2) (Addresses $5B_{16}$ to $5D_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 0 0 : Timer mode	0	RW
1			0	RW
2	These bits are invalid in timer mode.		0	RW
3			0	RW
4	Nothing is assigned.		Undefined	—
5	This bit is invalid in timer mode; its value is undefined at reading.		Undefined	RO
6	Count source select bits	b7 b6 0 0 : f_2 0 1 : f_{16} 1 0 : f_{64} 1 1 : f_{512}	0	RW
7			0	RW

EOL announced

Event counter mode



Timer B0 register (Addresses 51_{16}, 50_{16})
Timer B1 register (Addresses 53_{16}, 52_{16})

Bit	Functions	At reset	RW
15 to 0	These bits can be set to "0000_{16}" to "$FFFF_{16}$." Assuming that the set value = n, the counter divides the count source frequency by (n + 1). When reading, the register indicates the counter value.	Undefined	RW

Note: Read from or write to this register in a unit of 16 bits.

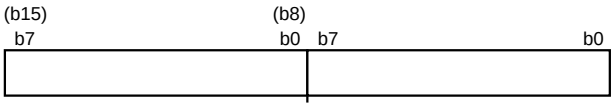


Timer Bj mode register (j = 0, 1) (Addresses $5B_{16}$, $5C_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 0 1 : Event counter mode	0	RW
1			0	RW
2	Count polarity select bits	b3 b2 0 0 : Count at falling edge of external signal 0 1 : Count at rising edge of external signal 1 0 : Counts at both falling and rising edges of external signal 1 1 : Do not select.	0	RW
3			0	RW
4	Nothing is assigned.		Undefined	—
5	This bit is invalid in event counter mode; its value is undefined at reading.		Undefined	RO
6	These bits are invalid in event counter mode.		0	RW
7			0	RW

EOL announced

Pulse period/pulse width measurement mode



Timer B0 register (Addresses 51_{16}, 50_{16})
Timer B1 register (Addresses 53_{16}, 52_{16})

Bit	Functions	At reset	RW
15 to 0	The measurement result of pulse period or pulse width is read out.	Undefined	RO

Note: Read from this register in a unit of 16 bits.



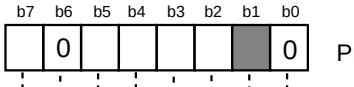
Timer Bj mode register (j = 0, 1) (Addresses $5B_{16}$, $5C_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Operating mode select bits	b1 b0 1 0 : Pulse period/Pulse width measurement mode	0	RW
1			0	RW
2	Measurement mode select bits	b3 b2 0 0 : Pulse period measurement (Interval between falling edges of measurement pulse) 0 1 : Pulse period measurement (Interval between rising edges of measurement pulse) 1 0 : Pulse width measurement (Interval from a falling edge to a rising edge, and from a rising edge to a falling edge of measurement pulse) 1 1 : Do not select.	0	RW
3			0	RW
4	Nothing is assigned.		Undefined	–
5	Timer Bj overflow flag **(Note)**	0 : No overflow 1 : Overflowed	Undefined	RO
6	Count source select bits	b7 b6 0 0 : f_2 0 1 : f_{16} 1 0 : f_{64} 1 1 : f_{512}	0	RW
7			0	RW

Note: The timer Bj overflow flag is cleared to “0” at the next count timing of the count source when a value is written to the timer Bj mode register with the count start bit = “1.”

EOL announced

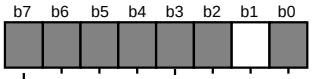
Processor mode register 0



Processor mode register 0 (Address 5E16)

Bit	Bit name	Functions	At reset	RW
0	Fix this bit to "0."		0	RW
1	Nothing is assigned. The value is "1" at reading.		1	–
2	Wait bit	0 : Software Wait is inserted when accessing external area. 1 : No software Wait is inserted when accessing external area.	0	RW
3	Software reset bit	The microcomputer is reset by writing "1" to this bit. The value is "0" at reading.	0	WO
4	Interrupt priority detection time select bits	b5 b4 0 0 : 7 cycles of ϕ 0 1 : 4 cycles of ϕ 1 0 : 2 cycles of ϕ 1 1 : Do not select.	0	RW
5			0	RW
6	Fix this bit to "0."		0	RW
7	Stack bank select bit	0 : Bank 016 1 : Bank FF16	0	RW

Processor mode register 1



Processor mode register 1 (Address 5F16)

Bit	Bit name	Functions	At reset	RW
0	Nothing is assigned.		Undefined	–
1	Internal RAM area select bit **(Notes 1, 2)**	0 : 512 bytes (addresses 8016 to 27F16) 1 : 1024 bytes (addresses 8016 to 47F16)	0	RW
7 to 2	Nothing is assigned.		Undefined	–

Notes 1: For the M37721S1BFP, fix bit 1 to "0."
2: For the M37721S2BFP, set bit 1 before setting the stack pointer.

EOL announced

Appendix 3. Control registers

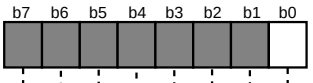
Watchdog timer register



Watchdog timer register (Address 60_{16})

Bit	Functions	At reset	RW
7 to 0	Initializes Watchdog timer. When dummy data is written to this register, Watchdog timer's value is initialized to "FFF_{16}." (Dummy data: 00_{16} to FF_{16})	Undefined	–

Watchdog timer frequency select register

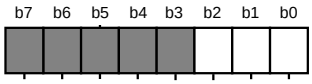


Watchdog timer frequency select register (Address 61_{16})

Bit	Bit name	Functions	At reset	RW
0	Watchdog timer frequency select bit	0 : f_{512} 1 : f_{32}	0	RW
7 to 1	Nothing is assigned.		Undefined	–

EOL announced

Real-time output control register



Real-time output control register (Address 62_{16})

Bit	Bit name	Functions	At reset	RW
0	Waveform output select bits	See the following Table.	0	RW
1			0	RW
2	Pulse output mode select bit	0 : Pulse mode 0 1 : Pulse mode 1	0	RW
7 to 3	Nothing is assigned. The value is "0" at reading.		Undefined	—

Note: When using the $P6_0$–$P6_7$ pins as the pulse output pins for real-time output, set the corresponding bits of the port P6 direction register (address 10_{16}) to "1."

b1 b0	00	01	10	11
When pulse mode 0 is selected	$P6_7$/RTP1$_3$ $P6_6$/RTP1$_2$ $P6_5$/RTP1$_1$ $P6_4$/RTP1$_0$ — Port $P6_3$/RTP0$_3$ $P6_2$/RTP0$_2$ $P6_1$/RTP0$_1$ $P6_0$/RTP0$_0$ — Port	$P6_7$/RTP1$_3$ $P6_6$/RTP1$_2$ $P6_5$/RTP1$_1$ $P6_4$/RTP1$_0$ — Port $P6_3$/RTP0$_3$ $P6_2$/RTP0$_2$ $P6_1$/RTP0$_1$ $P6_0$/RTP0$_0$ — RTP	$P6_7$/RTP1$_3$ $P6_6$/RTP1$_2$ $P6_5$/RTP1$_1$ $P6_4$/RTP1$_0$ — RTP $P6_3$/RTP0$_3$ $P6_2$/RTP0$_2$ $P6_1$/RTP0$_1$ $P6_0$/RTP0$_0$ — Port	$P6_7$/RTP1$_3$ $P6_6$/RTP1$_2$ $P6_5$/RTP1$_1$ $P6_4$/RTP1$_0$ — RTP $P6_3$/RTP0$_3$ $P6_2$/RTP0$_2$ $P6_1$/RTP0$_1$ $P6_0$/RTP0$_0$ — RTP
When pulse mode 1 is selected	$P6_7$/RTP1$_3$ $P6_6$/RTP1$_2$ $P6_5$/RTP1$_1$ $P6_4$/RTP1$_0$ $P6_3$/RTP0$_3$ $P6_2$/RTP0$_2$ — Port $P6_1$/RTP0$_1$ $P6_0$/RTP0$_0$ — Port	$P6_7$/RTP1$_3$ $P6_6$/RTP1$_2$ $P6_5$/RTP1$_1$ $P6_4$/RTP1$_0$ $P6_3$/RTP0$_3$ $P6_2$/RTP0$_2$ — Port $P6_1$/RTP0$_1$ $P6_0$/RTP0$_0$ — RTP	$P6_7$/RTP1$_3$ $P6_6$/RTP1$_2$ $P6_5$/RTP1$_1$ $P6_4$/RTP1$_0$ $P6_3$/RTP0$_3$ $P6_2$/RTP0$_2$ — RTP $P6_1$/RTP0$_1$ $P6_0$/RTP0$_0$ — Port	$P6_7$/RTP1$_3$ $P6_6$/RTP1$_2$ $P6_5$/RTP1$_1$ $P6_4$/RTP1$_0$ $P6_3$/RTP0$_3$ $P6_2$/RTP0$_2$ — RTP $P6_1$/RTP0$_1$ $P6_0$/RTP0$_0$ — RTP

Port : This functions as a programmable I/O port.
RTP : This functions as a pulse output pin.

FOLmnou.ed

DRAM control register

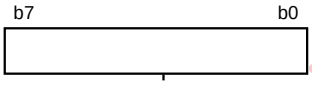


DRAM control register (Address 64_{16})

Bit	Bit name	Functions	At reset	RW
0	DRAM area select bits	b3 b2 b1 b0 0 0 0 0 : No DRAM area 0 0 0 1 : $F00000_{16}$–$FFFFFF_{16}$ (1 Mbyte) 0 0 1 0 : $E00000_{16}$–$FFFFFF_{16}$ (2 Mbytes) 0 0 1 1 : $D00000_{16}$–$FFFFFF_{16}$ (3 Mbytes) 0 1 0 0 : $C00000_{16}$–$FFFFFF_{16}$ (4 Mbytes) 0 1 0 1 : $B00000_{16}$–$FFFFFF_{16}$ (5 Mbytes) 0 1 1 0 : $A00000_{16}$–$FFFFFF_{16}$ (6 Mbytes) 0 1 1 1 : 900000_{16}–$FFFFFF_{16}$ (7 Mbytes) 1 0 0 0 : 800000_{16}–$FFFFFF_{16}$ (8 Mbytes) 1 0 0 1 : 700000_{16}–$FFFFFF_{16}$ (9 Mbytes) 1 0 1 0 : 600000_{16}–$FFFFFF_{16}$ (10 Mbytes) 1 0 1 1 : 500000_{16}–$FFFFFF_{16}$ (11 Mbytes) 1 1 0 0 : 400000_{16}–$FFFFFF_{16}$ (12 Mbytes) 1 1 0 1 : 300000_{16}–$FFFFFF_{16}$ (13 Mbytes) 1 1 1 0 : 200000_{16}–$FFFFFF_{16}$ (14 Mbytes) 1 1 1 1 : 100000_{16}–$FFFFFF_{16}$ (15 Mbytes)	0	RW
1			0	RW
2			0	RW
3			0	RW
6 to 4	Nothing is assigned. The value is “0” at reading.		0	–
7	DRAM validity bit **(Note)**	0 : Invalid ($P10_4$–$P10_7$ pins function as programmable input ports. A_0–A_7 pins function as address output pins. Refresh timer stops counting.) 1 : Valid ($P10_4$–$P10_7$ pins function as $\overline{CAS}$, $\overline{RAS}$, MA_8, and MA_9. A_0–A_7 function as MA_0–MA_7. Refresh timer starts counting.)	0	RW

Note: Set the refresh timer (address 66_{16}) before setting this bit to “1.”

Refresh timer



Refresh timer (Address 66_{16})

Bit	Functions	At reset	RW
7 to 0	These bits can be set to “01_{16}” to “FF_{16}.” Assuming that the set value = n, this register divides f_{16} by (n + 1).	Undefined	WO

Note: Use the **LDM** or **STA** instruction for writing to this register.
Do not set this register to “00_{16}.”

EOL announced

DMAC control register L



DMAC control register L (Address 68_{16})

Bit	Bit name	Functions	At reset	RW
0	Priority select bit	0 : Fixed 1 : Rotating	0	RW
1	$\overline{TC}$ pin validity bit	0 : Invalid ($P10_3$ pin functions as a programmable I/O port (CMOS).) 1 : Valid ($P10_3$ pin functions as $\overline{TC}$ pin (N-channel open-drain).)	0	RW
3, 2	Nothing is assigned.		Undefined	–
4	DMA0 request bit	0 : No request 1 : Requested **(Note 1)**	0	RW
5	DMA1 request bit		0	RW
6	DMA2 request bit		0	RW
7	DMA3 request bit		0	RW

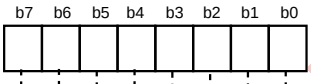
Notes 1. The state of bits 4 to 7 are not changed when writing "1" to these bits.

2. •When writing to this register while any of DMAi enable bits (bits 4 to 7 at address 69_{16}) is "1," set m flag to "1" and use the **LDM** or **STA** instruction. When DMAi request bit (bits 4 to 7 at address 68_{16}) must not be changed, set DMAi request bit to "1."

•When writing to this register while all of DMAi enable bits (bits 4 to 7 at address 69_{16}) are "0," m flag may be "0" or "1." Use the **LDM** or **STA** instruction for writing to this register. When DMAi request bit (bits 4 to 7 at address 68_{16}) must not be changed, set DMAi request bit to "1."

EOL announced

DMAC control register H

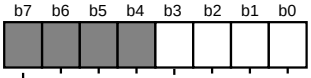


DMAC control register H (Address 69_{16})

Bit	Bit name	Functions	At reset	RW
0	Software DMA0 request bit	1 : DMA request (Valid when software DMA source is selected.) The value is "0" at reading.	0	WO
1	Software DMA1 request bit		0	WO
2	Software DMA2 request bit		0	WO
3	Software DMA3 request bit		0	WO
4	DMA0 enable bit	0 : Disabled 1 : Enabled	0	RW
5	DMA1 enable bit		0	RW
6	DMA2 enable bit		0	RW
7	DMA3 enable bit		0	RW

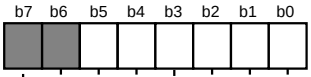
Note: When any of bits 4 to 7 is set to "1," use the **CLB** or **SEB** instruction for writing to this register.

Interrupt control register



DMA0 to DMA3, A-D conversion, UART0 and 1 transmit, UART0 and 1 receive, timers A0 to A4, timers B0 to B2 interrupt control registers (Addresses $6C_{16}$ to $7C_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Interrupt priority level select bits	b2 b1 b0 0 0 0 : Level 0 (Interrupt disabled) 0 0 1 : Level 1 0 1 0 : Level 2 0 1 1 : Level 3 1 0 0 : Level 4 1 0 1 : Level 5 1 1 0 : Level 6 1 1 1 : Level 7	0	RW
1			0	RW
2			0	RW
3	Interrupt request bit	0 : No interrupt requested 1 : Interrupt requested	0	RW
7 to 4	Nothing is assigned.		Undefined	—



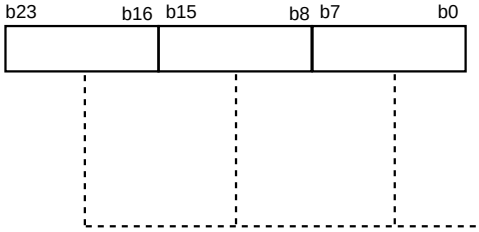
$\overline{INT_0}$ to $\overline{INT_2}$ interrupt control registers (Addresses $7D_{16}$ to $7F_{16}$)

Bit	Bit name	Functions	At reset	RW
0	Interrupt priority level select bits	b2 b1 b0 0 0 0 : Level 0 (Interrupt disabled) 0 0 1 : Level 1 0 1 0 : Level 2 0 1 1 : Level 3 1 0 0 : Level 4 1 0 1 : Level 5 1 1 0 : Level 6 1 1 1 : Level 7	0	RW
1			0	RW
2			0	RW
3	Interrupt request bit **(Note)**	0 : No interrupt requested 1 : Interrupt requested	0	RW
4	Polarity select bit	0 : Interrupt request bit is set to “1” at “H” level when level sense is selected; this bit is set to “1” at falling edge when edge sense is selected. 1 : Interrupt request bit is set to “1” at “L” level when level sense is selected; this bit is set to “1” at rising edge when edge sense is selected.	0	RW
5	Level sense/Edge sense select bit	0 : Edge sense 1 : Level sense	0	RW
7, 6	Nothing is assigned.		Undefined	—

Note: The interrupt request bits of $\overline{INT_0}$ to $\overline{INT_2}$ interrupts are invalid when the level sense is selected.

EOL announced

Source address register i

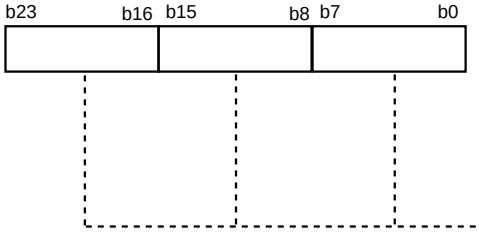


Source address register 0 (Addresses $1FC2_{16}$ to $1FC0_{16}$)
Source address register 1 (Addresses $1FD2_{16}$ to $1FD0_{16}$)
Source address register 2 (Addresses $1FE2_{16}$ to $1FE0_{16}$)
Source address register 3 (Addresses $1FF2_{16}$ to $1FF0_{16}$)

Bit	Functions	At reset	RW
23 to 0	These bits have different functions according to the operating mode.	Undefined	RW

Note: When writing to this register, write to all 24 bits.

Destination address register i

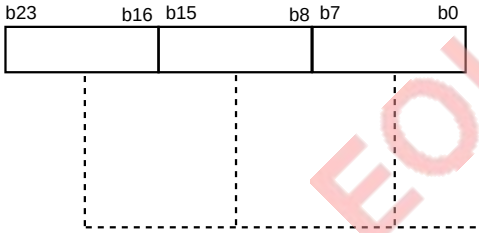


Destination address register 0 (Addresses $1FC6_{16}$ to $1FC4_{16}$)
Destination address register 1 (Addresses $1FD6_{16}$ to $1FD4_{16}$)
Destination address register 2 (Addresses $1FE6_{16}$ to $1FE4_{16}$)
Destination address register 3 (Addresses $1FF6_{16}$ to $1FF4_{16}$)

Bit	Functions	At reset	RW
23 to 0	These bits have different functions according to the operating mode.	Undefined	RW

Note: When writing to this register, write to all 24 bits.

Transfer counter register i



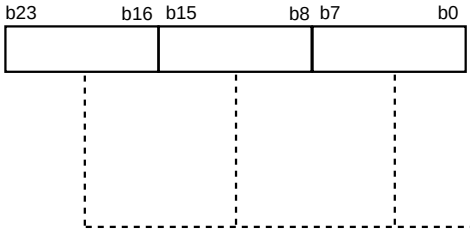
Transfer counter register 0 (Addresses $1FCA_{16}$ to $1FC8_{16}$)
Transfer counter register 1 (Addresses $1FDA_{16}$ to $1FD8_{16}$)
Transfer counter register 2 (Addresses $1FEA_{16}$ to $1FE8_{16}$)
Transfer counter register 3 (Addresses $1FFA_{16}$ to $1FF8_{16}$)

Bit	Functions	At reset	RW
23 to 0	These bits have different functions according to the operating mode.	Undefined	RW

Note: When writing to this register, write to all 24 bits.
Do not write "000000_{16}" to this register.

EOL announced

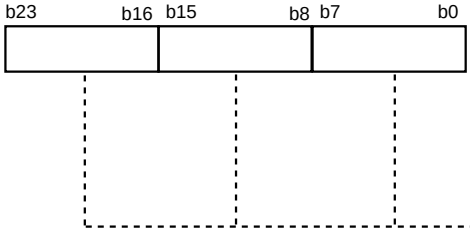
Single transfer mode



Source address register 0 (Addresses $1FC2_{16}$ to $1FC0_{16}$)
Source address register 1 (Addresses $1FD2_{16}$ to $1FD0_{16}$)
Source address register 2 (Addresses $1FE2_{16}$ to $1FE0_{16}$)
Source address register 3 (Addresses $1FF2_{16}$ to $1FF0_{16}$)

Bit	Functions	At reset	RW
23 to 0	[Write] Set the transfer start address of the source. These bits can be set to "000000_{16}" to "$FFFFFF_{16}$." [Read] The read value indicates the source address of data which is next transferred.	Undefined	RW

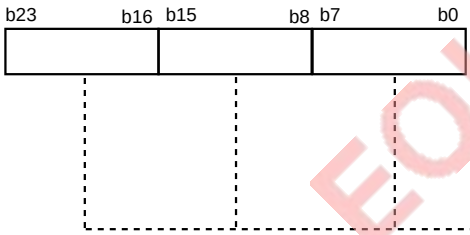
Note: When writing to this register, write to all 24 bits.



Destination address register 0 (Addresses $1FC6_{16}$ to $1FC4_{16}$)
Destination address register 1 (Addresses $1FD6_{16}$ to $1FD4_{16}$)
Destination address register 2 (Addresses $1FE6_{16}$ to $1FE4_{16}$)
Destination address register 3 (Addresses $1FF6_{16}$ to $1FF4_{16}$)

Bit	Functions	At reset	RW
23 to 0	[Write] Set the transfer start address of the destination. These bits can be set to "000000_{16}" to "$FFFFFF_{16}$." [Read] The read value indicates the destination address of data which is next transferred.	Undefined	RW

Note: When writing to this register, write to all 24 bits.



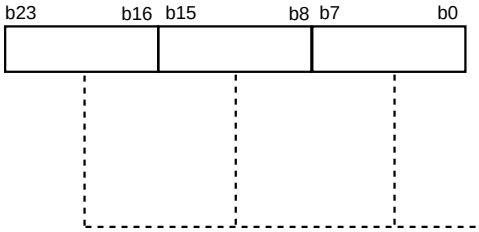
Transfer counter register 0 (Addresses $1FCA_{16}$ to $1FC8_{16}$)
Transfer counter register 1 (Addresses $1FDA_{16}$ to $1FD8_{16}$)
Transfer counter register 2 (Addresses $1FEA_{16}$ to $1FE8_{16}$)
Transfer counter register 3 (Addresses $1FFA_{16}$ to $1FF8_{16}$)

Bit	Functions	At reset	RW
23 to 0	[Write] Set the byte number of the transfer data. These bits can be set to "000001_{16}" to "$FFFFFF_{16}$." [Read] The read value indicates remaining byte number of the transfer data.	Undefined	RW

Note: When writing to this register, write to all 24 bits.
Do not set this register to "000000_{16}."

EOL announced

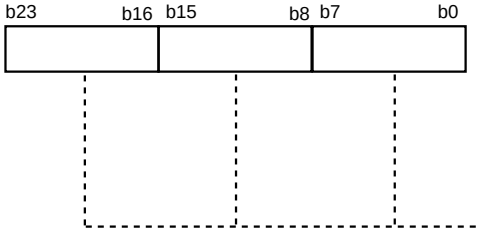
Repeat transfer mode



Source address register 0 (Addresses $1FC2_{16}$ to $1FC0_{16}$)
Source address register 1 (Addresses $1FD2_{16}$ to $1FD0_{16}$)
Source address register 2 (Addresses $1FE2_{16}$ to $1FE0_{16}$)
Source address register 3 (Addresses $1FF2_{16}$ to $1FF0_{16}$)

Bit	Functions	At reset	RW
23 to 0	[Write] Set the transfer start address of the source. These bits can be set to "000000_{16}" to "$FFFFFF_{16}$." [Read] The read value indicates the source address of data which is next transferred.	Undefined	RW

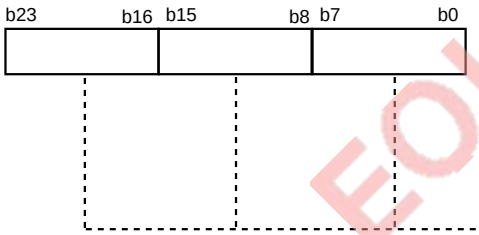
Note: When writing to this register, write to all 24 bits.



Destination address register 0 (Addresses $1FC6_{16}$ to $1FC4_{16}$)
Destination address register 1 (Addresses $1FD6_{16}$ to $1FD4_{16}$)
Destination address register 2 (Addresses $1FE6_{16}$ to $1FE4_{16}$)
Destination address register 3 (Addresses $1FF6_{16}$ to $1FF4_{16}$)

Bit	Functions	At reset	RW
23 to 0	[Write] Set the transfer start address of the destination. These bits can be set to "000000_{16}" to "$FFFFFF_{16}$." [Read] The read value indicates the destination address of data which is next transferred.	Undefined	RW

Note: When writing to this register, write to all 24 bits.



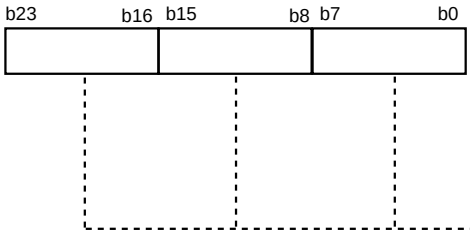
Transfer counter register 0 (Addresses $1FCA_{16}$ to $1FC8_{16}$)
Transfer counter register 1 (Addresses $1FDA_{16}$ to $1FD8_{16}$)
Transfer counter register 2 (Addresses $1FEA_{16}$ to $1FE8_{16}$)
Transfer counter register 3 (Addresses $1FFA_{16}$ to $1FF8_{16}$)

Bit	Functions	At reset	RW
23 to 0	[Write] Set the byte number of the transfer data. These bits can be set to "000001_{16}" to "$FFFFFF_{16}$." [Read] The read value indicates the remaining byte number of the block which is being transferred.	Undefined	RW

Note: When writing to this register, write to all 24 bits.
Do not write "000000_{16}" to this register.

EOL announced

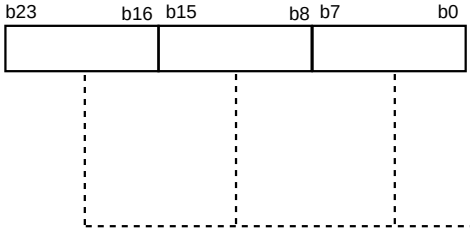
Array chain transfer mode



Source address register 0 (Addresses $1FC2_{16}$ to $1FC0_{16}$)
Source address register 1 (Addresses $1FD2_{16}$ to $1FD0_{16}$)
Source address register 2 (Addresses $1FE2_{16}$ to $1FE0_{16}$)
Source address register 3 (Addresses $1FF2_{16}$ to $1FF0_{16}$)

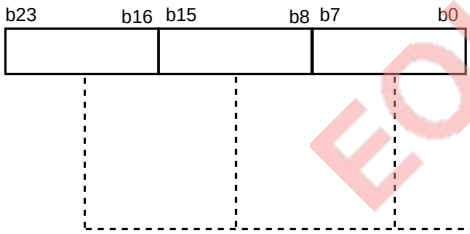
Bit	Functions	At reset	RW
23 to 0	[Write] Set the start address of transfer parameter memory. These bits can be set to "000000_{16}" to "$FFFFFF_{16}$." [Read] •After a value is written to this register and until transfer starts, the read value indicates the written value (the start address of the transfer parameter memory). •After tranfer starts, the read value indicates the source address of data which is next transferred.	Undefined	RW

Note: When writing to this register, write to all 24 bits.



Destination address register 0 (Addresses $1FC6_{16}$ to $1FC4_{16}$)
Destination address register 1 (Addresses $1FD6_{16}$ to $1FD4_{16}$)
Destination address register 2 (Addresses $1FE6_{16}$ to $1FE4_{16}$)
Destination address register 3 (Addresses $1FF6_{16}$ to $1FF4_{16}$)

Bit	Functions	At reset	RW
23 to 0	Need not to set. [Read] After transfer starts, the read value indicates the destination address of data which is next transferred.	Undefined	RW



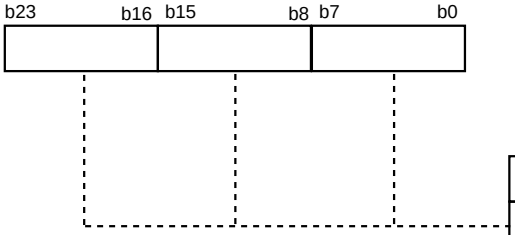
Transfer counter register 0 (Addresses $1FCA_{16}$ to $1FC8_{16}$)
Transfer counter register 1 (Addresses $1FDA_{16}$ to $1FD8_{16}$)
Transfer counter register 2 (Addresses $1FEA_{16}$ to $1FE8_{16}$)
Transfer counter register 3 (Addresses $1FFA_{16}$ to $1FF8_{16}$)

Bit	Functions	At reset	RW
23 to 0	[Write] Set the number of transfer blocks. These bits can be set to "000001_{16}" to "$FFFFFF_{16}$." [Read] •After a value is written to this register and until transfer starts, the read value indicates the written value (the transfer block number) . •After transfer starts, the read value indicates the remaining byte number of the block which is being transferred.	Undefined	RW

Note: When writing to this register, write to all 24 bits.
Do not write "000000_{16}" to this register.

EOL announced

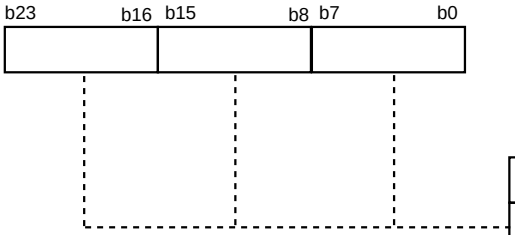
Link array chain transfer mode



Source address register 0 (Addresses $1FC2_{16}$ to $1FC0_{16}$)
Source address register 1 (Addresses $1FD2_{16}$ to $1FD0_{16}$)
Source address register 2 (Addresses $1FE2_{16}$ to $1FE0_{16}$)
Source address register 3 (Addresses $1FF2_{16}$ to $1FF0_{16}$)

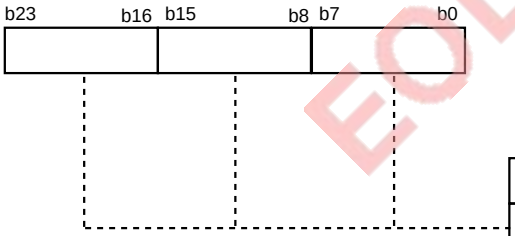
Bit	Functions	At reset	RW
23 to 0	[Write] Set the start address of transfer parameter memory of block which is first transferred. These bits can be set to "000000_{16}" to "$FFFFFF_{16}$." [Read] •After a value is written to this register and until transfer starts, the read value indicates the written value (the start address of the transfer parameter memory of block which is first transferred). •After transfer starts, the read value indicates the source address of data which is next transferred.	Undefined	RW

Note: When writing to this register, write to all 24 bits.



Destination address register 0 (Addresses $1FC6_{16}$ to $1FC4_{16}$)
Destination address register 1 (Addresses $1FD6_{16}$ to $1FD4_{16}$)
Destination address register 2 (Addresses $1FE6_{16}$ to $1FE4_{16}$)
Destination address register 3 (Addresses $1FF6_{16}$ to $1FF4_{16}$)

Bit	Functions	At reset	RW
23 to 0	Need not to set. [Read] After transfer starts, the read value indicates the destination address of data which is next transferred.	Undefined	RW



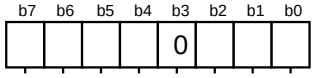
Transfer counter register 0 (Addresses $1FCA_{16}$ to $1FC8_{16}$)
Transfer counter register 1 (Addresses $1FDA_{16}$ to $1FD8_{16}$)
Transfer counter register 2 (Addresses $1FEA_{16}$ to $1FE8_{16}$)
Transfer counter register 3 (Addresses $1FFA_{16}$ to $1FF8_{16}$)

Bit	Functions	At reset	RW
23 to 0	[Write] Set the dummy data. These bits can be set to "000001_{16}" to "$FFFFFF_{16}$." [Read] •After a value is written to this register and until transfer starts, the read value indicates the written value (dummy data). •After transfer starts, the read value indicates the remaining byte number of the block which is being transferred.	Undefined	RW

Note: When writing to this register, write to all 24 bits.
Do not write "000000_{16}" to this register.

EOL announced

DMAi mode register L

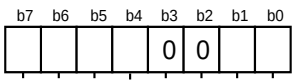


DMA0 mode register L (Address 1FCC16)
DMA1 mode register L (Address 1FDC16)
DMA2 mode register L (Address 1FEC16)
DMA3 mode register L (Address 1FFC16)

Bit	Bit name	Functions	At reset	RW
0	Number-of-unit-transfer-bits select bit **(Note)**	0 : 16 bits 1 : 8 bits	0	RW
1	Transfer method select bit	0 : 2-bus cycle transfer 1 : 1-bus cycle transfer	0	RW
2	Transfer mode select bit	0 : Burst transfer mode 1 : Cycle-steal transfer mode	0	RW
3	Fix this bit to "0."		0	RW
4	Transfer source address direction select bits	b5b4 0 0 : Fixed 0 1 : Forward 1 0 : Backward 1 1 : Do not select.	0	RW
5			0	RW
6	Transfer destination address direction select bits	b7b6 0 0 : Fixed 0 1 : Forward 1 0 : Backward 1 1 : Do not select.	0	RW
7			0	RW

Note: When the external data bus has a width of 8 bits and 1-bus cycle transfer is selected, set bit 0 to "1."

DMAi mode register H



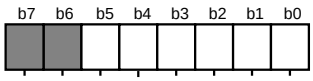
DMA0 mode register H (Address 1FCD16)
DMA1 mode register H (Address 1FDD16)
DMA2 mode register H (Address 1FED16)
DMA3 mode register H (Address 1FFD16)

Bit	Bit name	Functions	At reset	RW
0	Transfer direction select bit (Used in 1-bus cycle transfer) **(Note 1)**	0 : From memory to I/O 1 : From I/O to memory	0	RW
1	I/O connection select bit (Valid in 1-bus cycle transfer)	Refer to "**Fig.13.2.7.**"	0	RW
2	Fix these bits to "0."		0	RW
3			0	RW
4	Transfer source wait bit **(Note 2)**	0 : Wait 1 : No Wait	0	RW
5	Transfer destination wait bit **(Note 2)**		0	RW
6	Continuous transfer mode select bits	b7b6 0 0 : Single transfer 0 1 : Repeat transfer 1 0 : Array chain transfer 1 1 : Link array chain transfer	0	RW
7			0	RW

Notes 1: Set bit 0 to "0" in 2-bus cycle transfer.
2: Bits 4 and 5 are valid to the external and internal areas. However, DRAM area is always handled with "Wait" regardless of the contents of these bits. The wait bit (bit 2 at address 5E16) is invalid in DMA transfer.

EOL announced

DMAi control register



DMA0 control register (Address 1FCE16)
DMA1 control register (Address 1FDE16)
DMA2 control register (Address 1FEE16)
DMA3 control register (Address 1FFE16)

Bit	Bit name	Functions	At reset	RW
0	DMA request source select bits **(Note)**	b3b2b1b0 0 0 0 0 : Do not select. 0 0 0 1 : External source ($\overline{\text{DMAREQi}}$) 0 0 1 0 : Software DMA source 0 0 1 1 : Timer A0 0 1 0 0 : Timer A1 0 1 0 1 : Timer A2 0 1 1 0 : Timer A3 0 1 1 1 : Timer A4 1 0 0 0 : Timer B0 1 0 0 1 : Timer B1 1 0 1 0 : Timer B2 1 0 1 1 : UART0 receive 1 1 0 0 : UART0 transmit 1 1 0 1 : UART1 receive 1 1 1 0 : UART1 transmit 1 1 1 1 : A-D conversion	0	RW
1			0	RW
2			0	RW
3			0	RW
4	Edge sense/Level sense select bit (Used when external source and burst transfer mode are selected) **(Note)**	0 : Edge sense (Falling edge) 1 : Level sense (“L” level)	0	RW
5	$\overline{\text{DMAACKi}}$ validity bit	0 : Invalid (The pin functions as a programmable I/O port.) 1 : Valid ($\overline{\text{DMAACKi}}$.)	0	RW
7, 6	Nothing is assigned.		Undefined	–

Note: When a certain source other than an external source is selected by bits 0 to 3 or when the cycle-steal transfer mode is selected, set bit 4 to “0.”
Level sense can be selected only when both of the external source and the burst transfer mode are selected.

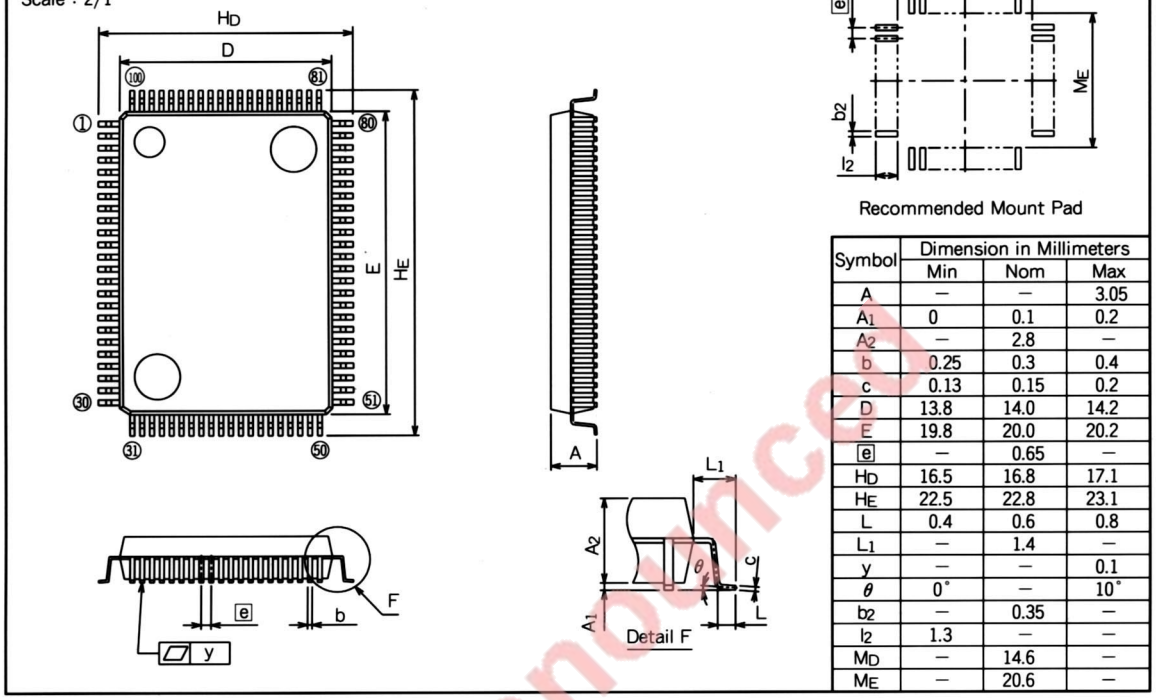
EO announced

Appendix 4. Package outline

100P6S–A Plastic 100pin 14×20mm body QFP

EIAJ Package Code	JEDEC Code	Weight(g)	Lead Material
QFP100-P-1420-0.65	–	1.58	Alloy 42

Scale : 2/1



Symbol	Dimension in Millimeters		
	Min	Nom	Max
A	–	–	3.05
A_1	0	0.1	0.2
A_2	–	2.8	–
b	0.25	0.3	0.4
c	0.13	0.15	0.2
D	13.8	14.0	14.2
E	19.8	20.0	20.2
e	–	0.65	–
H_D	16.5	16.8	17.1
H_E	22.5	22.8	23.1
L	0.4	0.6	0.8
L_1	–	1.4	–
y	–	–	0.1
θ	0°	–	10°
b_2	–	0.35	–
l_2	1.3	–	–
M_D	–	14.6	–
M_E	–	20.6	–

EOL announced

Appendix 5. Examples of handling unused pins

Examples of handling unused pins are described below. These descriptions are just examples. The user shall modify them according to the actual application and test them.

Table 1 Examples of handling unused pins

Pins	Handling example
$P4_3$ to $P4_7$, P5 to P10	Connect these pins to the Vcc or Vss pin via resistors after these pins are set to the input mode, or leave these pins open after they are set to the output mode **(Notes 1, 2)**.
$\overline{BLE}$, $\overline{BHE}$, ALE, ϕ_1, ST0, ST1	Leave this pin open.
X_{OUT} **(Note 3)**	Leave this pin open.
$\overline{HOLD}$, $\overline{RDY}$	Connect these pins to the Vcc pin via resistors (These pins are pulled high.) **(Note 2)**
CNVss	Connect this pin to the Vcc pin or Vss pin.
AVcc	Connect this pin to the Vcc pin.
AVss, V_{REF}	Connect these pins to the Vss pin.

Notes 1: When leaving these pins open after they are set to the output mode, note the following: these pins function as input ports from reset until they are switched to the output mode by software. Therefore, voltage levels of these pins are undefined and the power source current may increase while these pins function as input ports. Accordingly, set these ports to the output mode immediately after reset. Software reliability can be enhanced when the contents of the above ports' direction registers are set periodically. This is because these contents may be changed by noise, a program runaway which occurs to noise, etc.

2: For unused pins, use the shortest possible wiring (within 20 mm from the microcomputer's pins).

3: This applies when a clock externally generated is input to the X_{IN} pin.

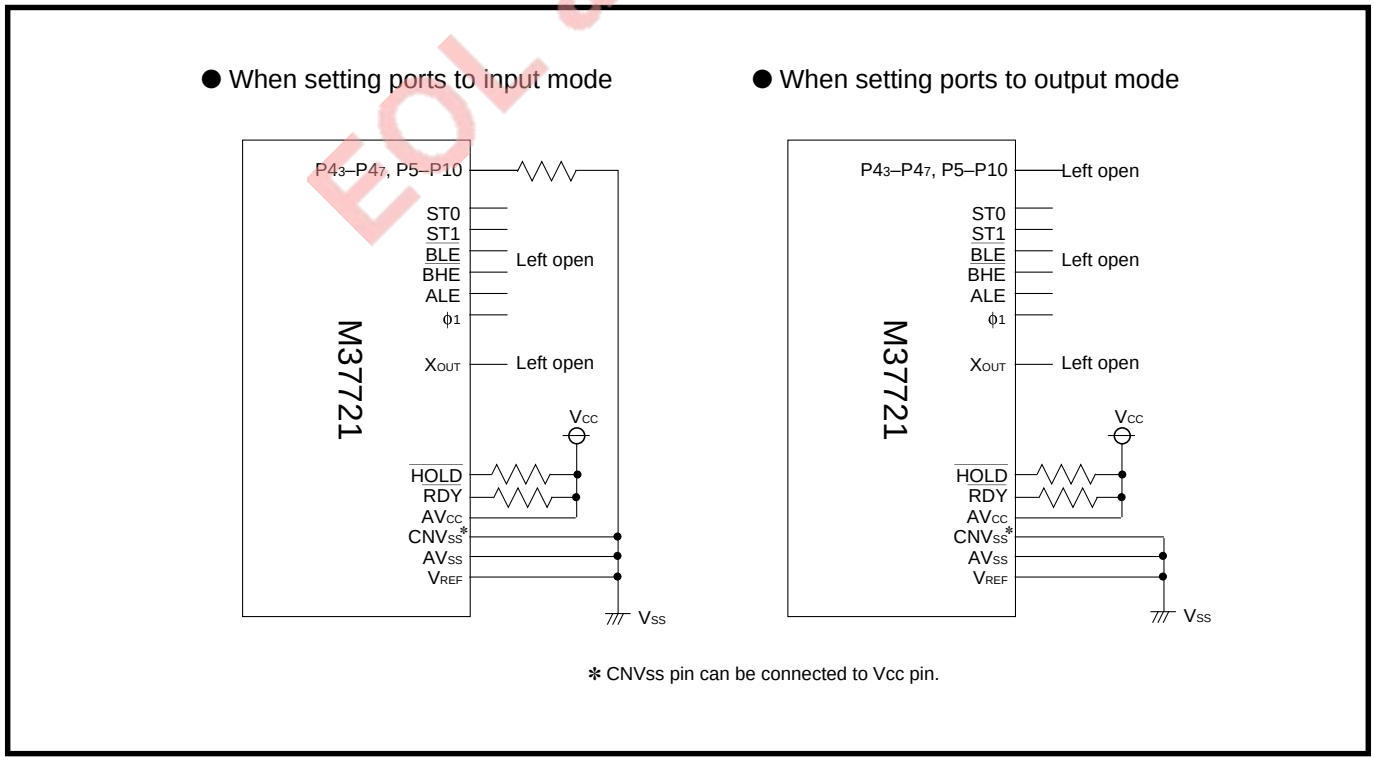


Fig. 2 Examples of handling unused pins

EOL announced

Appendix 6. Machine instructions

Symbol	Functions	Details	IMP op	IMP n	IMP #	IMM op	IMM n	IMM #	A op	A n	A #	DIR op	DIR n	DIR #	DIR,b op	DIR,b n	DIR,b #	DIR,X op	DIR,X n	DIR,X #	DIR,Y op	DIR,Y n	DIR,Y #	(DIR) op	(DIR) n	(DIR) #	(DIR,X) op	(DIR,X) n	(DIR,X) #	(DIR),Y op	(DIR),Y n	(DIR),Y #
ADC **(Notes 1,2)**	Acc,C←Acc+M+C	Adds the carry, the accumulator and the memory contents.The result is entered into the accumulator. When the D flag is "0," binary additions is done, and when the D flag is "1," decimal addition is done.				69 42 69	2 4	2 3				65 42 65	4 6	2 3				75 42 75	5 7	2 3				72 42 72	6 8	2 3	61 42 61	7 9	2 3	71 42 71	8 10	2 3
AND **(Notes 1,2)**	Acc←Acc∧M	Obtains the logical product of the contents of the accumulator and the contents of the memory . The result is entered into the accumulator.				29 42 29	2 4	2 3				25 42 25	4 6	2 3				35 42 35	5 7	2 3				32 42 32	6 8	2 3	21 42 21	7 9	2 3	31 42 31	8 10	2 3
ASL **(Note 1)**	m=0 C ← b_{15} ··· b_0 ←0 m=1 C ← b_7 ··· b_0 ←0	Shifts the accumulator or the memory contents one bit to the left. "0" is entered into bit 0 of the accumulator or the memory. The contents of bit 15 (bit 7 when the m flag is "1") of the accumulator or memory before shift is entered into the C flag.							0A 42 0A	2 4	1 2	06	7	2				16	7	2												
BBC **(Notes 3,5)**	Mb=0?	Tests the specified bit of the memory. Branches when all the contents of the specified bit is "0."																														
BBS **(Notes 3,5)**	Mb=1?	Tests the specified bit of the memory. Branches when all the contents of the specified bit is "1."																														
BCC **(Note 3)**	C=0?	Branches when the contents of the C flag is "0."																														
BCS **(Note 3)**	C=1?	Branches when the contents of the C flag is "1."																														
BEQ **(Note 3)**	Z=1?	Branches when the contents of the Z flag is "1."																														
BMI **(Note 3)**	N=1?	Branches when the contents of the N flag is "1."																														
BNE **(Note 3)**	Z=0?	Branches when the contents of the Z flag is "0."																														
BPL **(Note 3)**	N=0?	Branches when the contents of the N flag is "0."																														
BRA **(Note 4)**	PC←PC±offset PG←PG+1 (when carry occurs) PG←PG–1 (when borrow occurs)	Jumps to the address indicated by the program counter plus the offset value.																														
BRK	PC←PC+2 M(S)←PG S←S–1 M(S)←PC_H S←S–1 M(S)←PC_L S←S–1 M(S)←PS_H S←S–1 M(S)←PS_L S←S–1 I←1 PC_L←AD_L PC_H←AD_H PG←00_{16}	Executes software interruption.	00	15	2																											
BVC (Note 3)	V=0?	Branches when the contents of the V flag is "0."																														
BVS **(Note 3)**	V=1?	Branches when the contents of the V flag is "1."																														
CLB (Note 5)	Mb←0	Makes the contents of the specified bit in the memory "0."													14	8	3															
CLC	C←0	Makes the contents of the C flag "0."	18	2	1																											
CLI	I←0	Makes the contents of the I flag "0."	58	2	1																											
CLM	m←0	Makes the contents of the m flag "0."	D8	2	1																											
CLP	PSb←0	Specifies the bit position in the processor status register by the bit pattern of the second byte in the instruction, and sets "0" in that bit.				C2	4	2																								
CLV	V←0	Makes the contents of the V flag "0."	B8	2	1																											
CMP **(Notes 1,2)**	Acc–M	Compares the contents of the accumulator with the contents of the memory.				C9 42 C9	2 4	2 3				C5 42 C5	4 6	2 3				D5 42 D5	5 7	2 3				D2 42 D2	6 8	2 3	C1 42 C1	7 9	2 3	D1 42 D1	8 10	2 3

EOL announced

Addressing modes																																																						Processor status register										
L(DIR)			L(DIR),Y			ABS			ABS,b			ABS,X			ABS,Y			ABL			ABL,X			(ABS)			L(ABS)			(ABS,X)			STK			REL			DIR,b,R			ABS,b,R			SR			(SR),Y			BLK			10	9	8	7	6	5	4	3	2	1	0
op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	IPL			N	V	m	x	D	I	Z	C
67	10	2	77	11	2	6D	4	3				7D	6	3	79	6	3	6F	6	4	7F	7	4																						63	5	2	73	8	2				•	•	•	N	V	•	•	•	•	Z	C
42 67	12	3	42 77	13	3	42 6D	6	4				42 7D	8	4	42 79	8	4	42 6F	8	5	42 7F	9	5																						42 63	7	3	42 73	10	3														
27	10	2	37	11	2	2D	4	3				3D	6	3	39	6	3	2F	6	4	3F	7	4																						23	5	2	33	8	2				•	•	•	N	•	•	•	•	•	Z	•
42 27	12	3	42 37	13	3	42 2D	6	4				42 3D	8	4	42 39	8	4	42 2F	8	5	42 3F	9	5																						42 23	7	3	42 33	10	3														
						0E	7	3				1E	8	3																																								•	•	•	N	•	•	•	•	•	Z	C
																																							34	7	4	3C	8	5										•	•	•	•	•	•	•	•	•	•	•
																																							24	7	4	2C	8	5										•	•	•	•	•	•	•	•	•	•	•
																																				90	4	2																•	•	•	•	•	•	•	•	•	•	•
																																				B0	4	2																•	•	•	•	•	•	•	•	•	•	•
																																				F0	4	2																•	•	•	•	•	•	•	•	•	•	•
																																				30	4	2																•	•	•	•	•	•	•	•	•	•	•
																																				D0	4	2																•	•	•	•	•	•	•	•	•	•	•
																																				10	4	2																•	•	•	•	•	•	•	•	•	•	•
																																				80	4	2																•	•	•	•	•	•	•	•	•	•	•
																																				82	4	3																										
																																																						•	•	•	•	•	•	•	•	•	•	•
																																				50	4	2																•	•	•	•	•	•	•	•	•	•	•
																																				70	4	2																•	•	•	•	•	•	•	•	•	•	•
									IC	9	4																																											•	•	•	•	•	•	•	•	•	•	•
																																																						•	•	•	•	•	•	•	•	•	•	0
																																																						•	•	•	•	•	•	•	•	0	•	•
																																																						•	•	•	•	•	0	•	•	•	•	•
																																																						•	•	•	Specified flag becomes “0.”							
																																																						•	•	•	•	0	•	•	•	•	•	•
C7	10	2	D7	11	2	CD	4	3				DD	6	3	D9	6	3	CF	6	4	DF	7	4																						C3	5	2	D3	8	2				•	•	•	N	•	•	•	•	•	Z	C
42 C7	12	3	42 D7	13	3	42 CD	6	4				42 DD	8	4	42 D9	8	4	42 CF	8	5	42 DF	9	5																						42 C3	7	3	42 D3	10	3														

EOL announced

APPENDIX

Appendix 6. Machine instructions

Symbol	Functions	Details	Addressing modes																															
			IMP			IMM			A			DIR			DIR,b			DIR,X			DIR,Y			(DIR)			(DIR,X)			(DIR),Y		
			op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#
CPX **(Note 2)**	X–M	Compares the contents of the index register X with the contents of the memory.				E0	2	2				E4	4	2																				
CPY **(Note 2)**	Y–M	Compares the contents of the index register Y with the contents of the memory.				C0	2	2				C4	4	2																				
DEC **(Note 1)**	Acc←Acc–1 or M←M–1	Decrements the contents of the accumlator or memory by 1.							1A	2	1	C6	7	2				D6	7	2														
									42 1A	4	2																							
DEX	X←X–1	Decrements the contents of the index register X by 1.	CA	2	1																													
DEY	Y←Y–1	Decrements the contents of the index register Y by 1.	88	2	1																													
DIV **(Notes 2,10)**	A(quotient)←B,A/M B(remainder)	The numeral that places the contents of accumlator B to the higher order and the contents of accumulator A to the lower order is divided by the contents of the memory. The quotient is entered into accumulator A and the remainder into accumulator B.				89 29	27	3				89 25	29	3				89 35	30	3				89 32	31	3	89 21	32	3	89 31	33	3
EOR **(Notes 1,2)**	Acc←Acc▽M	Logical exclusive sum is obtained of the contents of the accumulator and the contents of the memory. The result is placed into the accumulator.				49	2	2				45	4	2				55	5	2				52	6	2	41	7	2	51	8	2
						42 49	4	3				42 45	6	3				42 55	7	3				42 52	8	3	42 41	9	3	42 51	10	3
INC **(Note 1)**	Acc←Acc+1 or M←M+1	Increments the contents of the accumulator or memory by 1.							3A	2	1	E6	7	2				F6	7	2														
									42 3A	4	2																							
INX	X←X+1	Increments the contents of the index register X by 1.	E8	2	1																													
INY	Y←Y+1	Increments the contents of the index register Y by 1.	C8	2	1																													
JMP	ABS PCL←ADL PCH←ADH ABL PCL←ADL PCH←ADH PG←ADG (ABS) PCL←(ADH, ADL) PCH←(ADH,ADL+1) L(ABS) PCL←(ADH, ADL) PCH←(ADH, ADL+1) PG←(ADH, ADL+2) (ABS, X) PCL←(ADH, ADL+X) PCH←(ADH, ADL+X +1)	Places a new address into the program counter and jumps to that new address.																															
JSR	ABS M(S)←PCH S←S–1 M(S)←PCL S←S–1 PCL←ADL PCH←ADH ABL M(S)←PG S←S–1 M(S)←PCH S←S–1 M(S)←PCL S←S–1 PCL←ADL PCH←ADH PG←ADG (ABS, X) M(S)←PCH S←S–1 M(S)←PCL S←S–1 PCL←(ADH, ADL+X) PCH←(ADH, ADL+X +1)	Saves the contents of the program counter (also the contents of the program bank register for ABL) into the stack, and jumps to the new address.																															

EOL announced

Addressing modes																																																						Processor status register										
L(DIR)			L(DIR),Y			ABS			ABS,b			ABS,X			ABS,Y			ABL			ABL,X			(ABS)			L(ABS)			(ABS,X)			STK			REL			DIR,b,R			ABS,b,R			SR			(SR),Y			BLK			10	9	8	7	6	5	4	3	2	1	0
op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	IPL			N	V	m	x	D	I	Z	C
						EC	4	3																																														•	•	•	N	•	•	•	•	•	Z	C
						CC	4	3																																														•	•	•	N	•	•	•	•	•	Z	C
						CE	7	3				DE	8	3																																								•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
89 27	35	3	89 37	36	3	89 2D	29	4				89 3D	31	4	89 39	31	4	89 2F	31	5	89 3F	32	5																							89 23	30	3	89 33	33	3				•	•	•	N	V	•	•	•	•	Z	C
47	10	2	57	11	2	4D	4	3				5D	6	3	59	6	3	4F	6	4	5F	7	4																							43	5	2	53	8	2				•	•	•	N	•	•	•	•	•	Z	•
42 47	12	3	42 57	13	3	42 4D	6	4				42 5D	8	4	42 59	8	4	42 4F	8	5	42 5F	9	5																							42 43	7	3	42 53	10	3														
						EE	7	3				FE	8	3																																								•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
						4C	2	3										5C	4	4				6C	4	3	DC	8	3	7C	6	3																						•	•	•	•	•	•	•	•	•	•	•
						20	6	3										22	8	4										FC	8	3																						•	•	•	•	•	•	•	•	•	•	•

EOL announced

APPENDIX

Appendix 6. Machine instructions

Symbol	Functions	Details	Addressing modes																																
			IMP			IMM			A			DIR			DIR,b			DIR,X			DIR,Y			(DIR)			(DIR,X)			(DIR),Y					
			op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#
LDA **(Notes 1,2)**	Acc←M	Enters the contents of the memory into the accummulator.				A9	2	2				A5	4	2				B5	5	2				B2	6	2	A1	7	2	B1	8	2			
						42 A9	4	3				42 A5	6	3				42 B5	7	3				42 B2	8	3	42 A1	9	3	42 B1	10	3			
LDM **(Note 5)**	M←IMM	Enters the immediate vaiue into the memory.										64	4	3				74	5	3															
LDT	DT←IMM	Enters the immediate value into the data bank regiater.				89 C2	5	3																											
LDX **(Note 2)**	X←M	Enters the contents of the memory into index register X.				A2	2	2				A6	4	2							B6	5	2												
LDY **(Note 2)**	Y←M	Enters the contents of the memory into index register Y.				A0	2	2				A4	4	2				B4	5	2															
LSR **(Note 1)**	m=0 0 → b15 ··· b0 →C m=1 0 → b7 ··· b0 →C	Shifts the contents of the accumulator or the contents of the memory one bit to the right. The bit 0 of the accumulator or the memory is entered into the C flag. "0" is entered into bit 15 (bit 7 when the m flag is "1.")							4A	2	1	46	7	2				56	7	2															
									42 4A	4	2																								
MPY **(Notes 2,11)**	B, A←A*M	Multiplies the contents of accumulator A and the contents of the memory. The higher order of the result of operation are entered into accumulator B, and the lower order into accumulator A.				89 09	16	3				89 05	18	3				89 15	19	3				89 12	20	3	89 01	21	3	89 11	22	3			
MVN **(Note 8)**	Mn+i←Mm+i	Transmits the data block. The transmission is done from the lower order address of the block.																																	
MVP **(Note 9)**	Mn–i←Mm–i	Transmits the data block. Transmission is done form the higher order address of the data block.																																	
NOP	PC←PC+1	Advances the program counter, but pertorms nothing else.	EA	2	1																														
ORA **(Notes 1,2)**	Acc←AccVM	Logical sum per bit of the contents of the accumulator and the contents of the memory is obtained. The result is entered into the accumulator.				09	2	2				05	4	2				15	5	2				12	6	2	01	7	2	11	8	2			
						42 09	4	3				42 05	6	3				42 15	7	3				42 12	8	3	42 01	9	3	42 11	10	3			
PEA	M(S)←IMM2 S←S–1 M(S)←IMM1 S←S–1	The 3rd and the 2nd bytes of the instruction are saved into the stack, in this order.																																	
PEI	M(S)←M((DPR)+IMM +1) S←S–1 M(S)←M((DPR)+IMM) S←S–1	Specifies 2 sequential bytes in the direct page in the 2nd byte of the instruction, and saves the contents into the stack.																																	
PER	EAR←PC+IMM2,IMM1 M(S)←EARH S←S–1 M(S)←EARL S←S–1	Regards the 2nd and 3rd bytes of the instruction as 16-bit numerals, adds them to the program counter, and saves the result into the stack.																																	
PHA	m=0 M(S)←AH S←S–1 M(S)←AL S←S–1 m=1 M(S)←AL S←S–1	Saves the contents of accumulator A into the stack.																																	
PHB	m=0 M(S)←BH S←S–1 M(S)←BL S←S–1 m=1 M(S)←BL S←S–1	Saves the contents of accumuator B into the stack.																																	

EOL announced

Addressing modes																																																						Processor status register										
L(DIR)			L(DIR),Y			ABS			ABS,b			ABS,X			ABS,Y			ABL			ABL,X			(ABS)			L(ABS)			(ABS,X)			STK			REL			DIR,b,R			ABS,b,R			SR			(SR),Y			BLK			10	9	8	7	6	5	4	3	2	1	0
op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	IPL			N	V	m	x	D	I	Z	C
A7	10	2	B7	11	2	AD	4	3				BD	6	3	B9	6	3	AF	6	4	BF	7	4																						A3	5	2	B3	8	2				•	•	•	N	•	•	•	•	•	Z	•
42 A7	12	3	42 B7	13	3	42 AD	6	4				42 BD	8	4	42 B9	8	4	42 AF	8	5	42 BF	9	5																						42 A3	7	3	42 B3	10	3														
						9C	5	4				9E	6	4																																								•	•	•	•	•	•	•	•	•	•	•
																																																					•	•	•	•	•	•	•	•	•	•	•
						AE	4	3							BE	6	3																																					•	•	•	N	•	•	•	•	•	Z	•
						AC	4	3				BC	6	3																																								•	•	•	N	•	•	•	•	•	Z	•
						4E	7	3				5E	8	3																																								•	•	•	0	•	•	•	•	•	Z	C
89 07	24	3	89 17	25	3	89 0D	18	4				89 1D	20	4	89 19	20	4	89 0F	20	5	89 1F	21	5																						89 03	19	3	89 13	22	3				•	•	•	N	•	•	•	•	•	Z	0
																																																			54	$7 + \frac{i}{2} \times 7$	3	•	•	•	•	•	•	•	•	•	•	•
																																																			44	$9 + \frac{i}{2} \times 7$	3	•	•	•	•	•	•	•	•	•	•	•
																																																						•	•	•	•	•	•	•	•	•	•	•
07	10	2	17	11	2	0D	4	3				1D	6	3	19	6	3	0F	6	4	1F	7	4																						03	5	2	13	8	2				•	•	•	N	•	•	•	•	•	Z	•
42 07	12	3	42 17	13	3	42 0D	6	4				42 1D	8	4	42 19	8	4	42 0F	8	5	42 1F	9	5																						42 03	7	3	42 13	10	3														
																																	F4	5	3																			•	•	•	•	•	•	•	•	•	•	•
																																	D4	6	2																			•	•	•	•	•	•	•	•	•	•	•
																																	62	5	3																			•	•	•	•	•	•	•	•	•	•	•
																																	48	4	1																			•	•	•	•	•	•	•	•	•	•	•
																																	42 48	6	2																			•	•	•	•	•	•	•	•	•	•	•

EOL announced

APPENDIX

Appendix 6. Machine instructions

Symbol	Functions	Details	Addressing modes																															
			IMP			IMM			A			DIR			DIR,b			DIR,X			DIR,Y			(DIR)			(DIR,X)			(DIR),Y					
			op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#			
PHD	M(S)←DPR$_H$ S←S–1 M(S)←DPR$_L$ S←S–1	Saves the contents of the direct page register into the stack.																																
PHG	M(S)←PG S←S–1	Saves the contents of the program bank register into the stack.																																
PHP	M(S)←PS$_H$ S←S–1 M(S)←PS$_L$ S←S–1	Saves the contents of the program status register into the stack.																																
PHT	M(S)←DT S←S–1	Saves the contents of the data bank register into the stack.																																
PHX	x=0 M(S)←X$_H$ S←S–1 M(S)←X$_L$ S←S–1 x=1 M(S)←X$_L$ S←S–1	Saves the contents of the index register X into the stack.																																
PHY	x=0 M(S)←Y$_H$ S←S–1 M(S)←Y$_L$ S←S–1 x=1 M(S)←Y$_L$ S←S–1	Saves the contents of the index register Y into the stack.																																
PLA	m=0 S←S+1 A$_L$←M(S) S←S+1 A$_H$←M(S) m=1 S←S+1 A$_L$←M(S)	Restores the contents of the stack on the accumulator A.																																
PLB	m=0 S←S+1 B$_L$←M(S) S←S+1 B$_H$←M(S) m=1 S←S+1 B$_L$←M(S)	Restores the contents of the stack on the accumulator B.																																
PLD	S←S+1 DPR$_L$←M(S) S←S+1 DPR$_H$←M(S)	Restores the contents of the stack on the direct page register.																																
PLP	S←S+1 PS$_L$←M(S) S←S+1 PS$_H$←M(S)	Restores the contents of the stack on the processor status register.																																
PLT	S←S+1 DT←M(S)	Restores the contents of the stack on the data bank register.																																
PLX	x=0 S←S+1 X$_L$←M(S) S←S+1 X$_H$←M(S) x=1 S←S+1 X$_L$←M(S)	Restores the contents of the stack on the index register X.																																

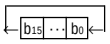
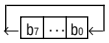
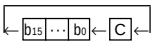
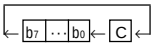
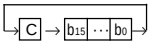
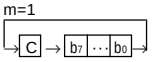
EOL announced

Addressing modes																																																						Processor status register										
L(DIR)			L(DIR),Y			ABS			ABS,b			ABS,X			ABS,Y			ABL			ABL,X			(ABS)			L(ABS)			(ABS,X)			STK			REL			DIR,b,R			ABS,b,R			SR			(SR),Y			BLK			10	9	8	7	6	5	4	3	2	1	0
op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	IPL			N	V	m	x	D	I	Z	C
																																	0B	4	1																			•	•	•	•	•	•	•	•	•	•	•
																																	4B	3	1																			•	•	•	•	•	•	•	•	•	•	•
																																	08	4	1																			•	•	•	•	•	•	•	•	•	•	•
																																	8B	3	1																			•	•	•	•	•	•	•	•	•	•	•
																																	DA	4	1																			•	•	•	•	•	•	•	•	•	•	•
																																	5A	4	1																			•	•	•	•	•	•	•	•	•	•	•
																																	68	5	1																			•	•	•	N	•	•	•	•	•	Z	•
																																	42 68	7	2																			•	•	•	N	•	•	•	•	•	Z	•
																																	2B	5	1																			•	•	•	•	•	•	•	•	•	•	•
																																	28	6	1																			Value saved in stack.										
																																	AB	6	1																			•	•	•	N	•	•	•	•	•	Z	•
																																	FA	5	1																			•	•	•	N	•	•	•	•	•	Z	•

EOL announced

APPENDIX

Appendix 6. Machine instructions

Symbol	Functions	Details	Addressing modes																																		
			IMP			IMM			A			DIR			DIR,b			DIR,X			DIR,Y			(DIR)			(DIR,X)			(DIR),Y								
			op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#						
PLY	x=0 S←S+1 YL←M(S) S←S+1 YH←M(S) x=1 S←S+1 YL←M(S)	Restores the contents of the stack on the index register Y.																																			
PSH **(Note 6)**	M(S)←A, B, X···	Saves the registers among accumulator, index register, direct page register, data bank register, program bank register, or processor status register, specified by the bit pattern of the second byte of the instruction into the stack.																																			
PUL **(Note 7)**	A, B, X··←M(S)	Restores the contents of the stack to the registers among accumulator, index register, direct page register, data bank register, or processor status register, specified by the bit pattern of the second byte of the instruction.																																			
RLA **(Note 13)**	m=0 n bit rotate left  m=1 n bit rotate left 	Rotates the contents of the accumulator A, n bits to the left.				89 49	6 + i	3																															
ROL **(Note 1)**	m=0  m=1 	Links the accumulator or the memory to C flag, and rotates result to the left by 1 bit.							2A 42 2A	2 4	1 2	26	7	2				36	7	2																			
ROR **(Note 1)**	m=0  m=1 	Links the accumulator or the memory to C flag, and rotates result to the right by 1 bit.							6A 42 6A	2 4	1 2	66	7	2				76	7	2																			
RTI	S←S+1 PSL←M(S) S←S+1 PSH←M(S) S←S+1 PCL←M(S) S←S+1 PCH←M(S) S←S+1 PG←M(S)	Returns from the interruption routine.	40	11	1																																
RTL	S←S+1 PCL←M(S) S←S+1 PCH←M(S) S←S+1 PG←M(S)	Returns from the subroutine. The contents of the program bank register are also restored.	6B	8	1																																
RTS	S←S+1 PCL←M(S) S←S+1 PCH←M(S)	Returns from the subroutine. The contents of the program bank register are not restored.	60	5	1																																
SBC **(Notes 1,2)**	ACC, C←ACC–M–$\overline{C}$	Subtracts the contents of the memory and the borrow from the contents of the accumulator.				E9 42 E9	2 4	2 3				E5 42 E5	4 6	2 3				F5 42 F5	5 7	2 3				F2 42 F2	6 8	2 3	E1 42 E1	7 9	2 3	F1 42 F1	8 10	2 3						

EOL announced

Addressing modes																																																						Processor status register										
L(DIR)			L(DIR),Y			ABS			ABS,b			ABS,X			ABS,Y			ABL			ABL,X			(ABS)			L(ABS)			(ABS,X)			STK			REL			DIR,b,R			ABS,b,R			SR			(SR),Y			BLK			10	9	8	7	6	5	4	3	2	1	0
op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	IPL			N	V	m	x	D	I	Z	C
																																	7A	5	1																			•	•	•	N	•	•	•	•	•	Z	•
																																	EB	12 + 2i1+i2	2																			•	•	•	•	•	•	•	•	•	•	•
																																	FB	14 + 3i1+4i2	2																			If restored the contents of PS, it becomes its value. And the other cases are no change.										
																																																					•	•	•	•	•	•	•	•	•	•	•
						2E	7	3				3E	8	3																																								•	•	•	N	•	•	•	•	•	Z	C
						6E	7	3				7E	8	3																																								•	•	•	N	•	•	•	•	•	Z	C
																																																						Value saved in stack.										
																																																					•	•	•	•	•	•	•	•	•	•	•
																																																					•	•	•	•	•	•	•	•	•	•	•
E7	10	2	F7	11	2	ED	4	3				FD	6	3	F9	6	3	EF	6	4	FF	7	4																						E3	5	2	F3	8	2				•	•	•	N	V	•	•	•	•	Z	C
42 E7	12	3	42 F7	13	3	42 ED	6	4				42 FD	8	4	42 F9	8	4	42 EF	8	5	42 FF	9	5																						42 E3	7	3	42 F3	10	3														

EOL announced

APPENDIX

Appendix 6. Machine instructions

Symbol	Functions	Details	Addressing modes																																
			IMP			IMM			A			DIR			DIR,b			DIR,X			DIR,Y			(DIR)			(DIR,X)			(DIR),Y					
			op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#			
SEB **(Note 5)**	Mb←1	Makes the contents of the specified bit in the memory "1."													04	8	3																		
SEC	C←1	Makes the contents of the C flag "1."	38	2	1																														
SEI	I←1	Makes the contents of the I flag "1."	78	2	1																														
SEM	m←1	Makes the contents of the m flag "1."	F8	2	1																														
SEP	PSb←1	Set the specified bit of the processor status register's lower byte (PS_L) to "1."				E2	3	2																											
STA **(Note 1)**	M←Acc	Stores the contents of the accumulator into the memory.										85	4	2				95	5	2				92	7	2	81	7	2	91	7	2			
												42 85	6	3				42 95	7	3				42 92	9	3	42 81	9	3	42 91	9	3			
STP		Stops the oscillation of the oscillator.	DB	3	1																														
STX	M←X	Stores the contents of the index register X into the memory.										86	4	2							96	5	2												
STY	M←Y	Stores the contents of the index register Y into the memory.										84	4	2				94	5	2															
TAD	DPR←A	Transmits the contents of the accumulator A to the direct page register.	5B	2	1																														
TAS	S←A	Transmits the contents of the accumulator A to the stack pointer.	1B	2	1																														
TAX	X←A	Transmits the contents of the accumulator A to the index register X.	AA	2	1																														
TAY	Y←A	Transmits the contents of the accumulator A to the index register Y.	A8	2	1																														
TBD	DPR←B	Transmits the contents of the accumulator B to the direct page register.	42 5B	4	2																														
TBS	S←B	Transmits the contents of the accumulator B to the stack pointer.	42 1B	4	2																														
TBX	X←B	Transmits the contents of the accumulator B to the index register X.	42 AA	4	2																														
TBY	Y←B	Transmits the contents of the accumulator B to the index register Y.	42 A8	4	2																														
TDA	A←DPR	Transmits the contents of the direct page register to the accumulator A.	7B	2	1																														
TDB	B←DPR	Transmits the contents of the direct page register to the accumulator B.	42 7B	4	2																														
TSA	A←S	Transmits the contents of the stack pointer to the accumulator A.	3B	2	1																														
TSB	B←S	Transmits the contents of the stack pointer to the accumulator B.	42 3B	4	2																														
TSX	X←S	Transmits the contents of the stack pointer to the index register X.	BA	2	1																														
TXA	A←X	Transmits the contents of the index register X to the accumulator A.	8A	2	1																														
TXB	B←X	Transmits the contents of the index register X to the accumulator B.	42 8A	4	2																														
TXS	S←X	Transmits the contents of the index register X to the stack pointer.	9A	2	1																														
TXY	Y←X	Transmits the contents of the index register X to the index register Y.	9B	2	1																														
TYA	A←Y	Transmits the contents of the index register Y to the accumulator A.	98	2	1																														
TYB	B←Y	Transmits the contents of the index register Y to the accumulator B.	42 98	4	2																														
TYX	X←Y	Transmits the contents of the index register Y to the index register X.	BB	2	1																														
WIT		Stops the internal clock.	CB	3	1																														
XAB	A⇆B	Exchanges the contents of the accumulator A and the contents of the accumulator B.	89 28	6	2																														

EOL announced

Addressing modes																																																						Processor status register										
L(DIR)			L(DIR),Y			ABS			ABS,b			ABS,X			ABS,Y			ABL			ABL,X			(ABS)			L(ABS)			(ABS,X)			STK			REL			DIR,b,R			ABS,b,R			SR			(SR),Y			BLK			10	9	8	7	6	5	4	3	2	1	0
op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	op	n	#	IPL			N	V	m	x	D	I	Z	C
									0C	9	4																																											•	•	•	•	•	•	•	•	•	•	•
																																																						•	•	•	•	•	•	•	•	•	•	1
																																																						•	•	•	•	•	•	•	•	1	•	•
																																																						•	•	•	•	•	1	•	•	•	•	•
																																																						•	•	•	Specified flag becomes “1.”							
87	10	2	97	11	2	8D	5	3				9D	5	3	99	5	3	8F	6	4	9F	7	4																						83	5	2	93	8	2				•	•	•	•	•	•	•	•	•	•	•
42 87	12	3	42 97	13	3	42 8D	7	4				42 9D	7	4	42 99	7	4	42 8F	8	5	42 9F	9	5																						42 83	7	3	42 93	10	3														
																																																						•	•	•	•	•	•	•	•	•	•	•
						8E	5	3																																														•	•	•	•	•	•	•	•	•	•	•
						8C	5	3																																														•	•	•	•	•	•	•	•	•	•	•
																																																						•	•	•	•	•	•	•	•	•	•	•
																																																						•	•	•	•	•	•	•	•	•	•	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	•	•	•	•	•	•	•	•
																																																						•	•	•	•	•	•	•	•	•	•	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	•	•	•	•	•	•	•	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	N	•	•	•	•	•	Z	•
																																																						•	•	•	•	•	•	•	•	•	•	•
																																																						•	•	•	N	•	•	•	•	•	Z	•

EOL announced

APPENDIX

Appendix 6. Machine instructions

The number of cycles shown in the table is described in the case of the fastest mode for each instruction. The number of cycles shown in the table is calculated for DPR_L=0. The number of cycles in the addressing mode concerning the DPR when DPR_L 0 must be incremented by 1.
The number of cycles shown in the table differs according to the bytes fetched into the instruction queue buffer, or according to whether the memory read/write address is odd or even. It also differs when the external region memory is accessed by BYTE="H."

Notes 1. The operation code at the upper row is used for accumulator A, and the operation at the lower row is used for accumulator B.

2. When setting flag m=0 to handle the data as 16-bit data in the immediate addressing mode, the number of bytes increments by 1.

3. The number of cycles increments by 2 when branching.

4. The operation code on the upper row is used for branching in the range of –128 to +127, and the operation code on the lower row is used for branching in the range of –32768 to +32767.

5. When handling 16-bit data with flag m=0, the byte in the table is incremented by 1.

6.

Type of register	A	B	X	Y	DPR	DT	PG	PS
Number of cycles	2	2	2	2	2	1	1	2

The number of cycles corresponding to the register to be pushed are added. The number of cycles when no pushing is done is 12. i_1 indicates the number of registers among A, B, X, Y, DPR, and PS to be saved, while i_2 indicates the number of registers among DT and PG to be saved.

7.

Type of register	A	B	X	Y	DPR	DT	PS
Number of cycles	3	3	3	3	4	3	3

The number of cycles corresponding to the register to be pulled are added. The number of cycles when no pulling is done is 14. i_1 indicates the number of registers among A, B, X, Y, DT, and PS to be restored, while i_2=1 when DPR is to be restored.

8. The number of cycles is the case when the number of bytes to be transferred is even.
When the number of bytes to be transferred is odd, the number is calculated as;

$$7 + (i/2) \times 7 + 4$$

Note that, (i/2) shows the integer part when i is divided by 2.

9. The number of cycles is the case when the number of bytes to be transferred is even.
When the number of bytes to be transferred is odd, the number is calculated as;

$$9 + (i/2) \times 7 + 5$$

Note that, (i/2) shows the integer part when i is divided by 2.

10. The number of cycles is the case in the 16-bit ÷ 8-bit operation. The number of cycles is incremented by 16 for 32-bit ÷ 16-bit operation.

11. The number of cycles is the case in the 8-bit × 8-bit operation. The number of cycles is incremented by 8 for 16-bit × 16-bit operation.

12. When setting flag x=0 to handle the data as 16-bit data in the immediate addressing mode, the number of bytes increments by 1.

13. When flag m is 0, the byte in the table is incremented by 1.

EOL announced

Symbols in machine instructions table

Symbol	Description	Symbol	Description
IMP	Implied addressing mode	∀	Exclusive OR
IMM	Immediate addressing mode	–	Negation
A	Accumulator addressing mode	←	Movement to the arrow direction
DIR	Direct addressing mode	ACC	Accumulator
DIR, b	Direct bit addressing mode	ACCH	Accumulator's upper 8 bits
DIR, X	Direct indexed X addressing mode	ACCL	Accumulator's lower 8 bits
DIR, Y	Direct indexed Y addressing mode	A	Accumulator A
(DIR)	Direct indirect addressing mode	AH	Accumulator A's upper 8 bits
(DIR,X)	Direct indexed X indirect addressing mode	AL	Accumulator A's lower 8 bits
(DIR), Y	Direct indirect indexed Y addressing mode	B	Accumulator B
L (DIR)	Direct indirect long addressing mode	BH	Accumulator B's upper 8 bits
L (DIR),Y	Direct indirect long indexed Y addressing mode	BL	Accumulator B's lower 8 bits
ABS	Absolute addressing mode	X	Index register X
ABS, b	Absolute bit addressing mode	XH	Index register X's upper 8 bits
ABS, X	Absolute indexed X addressing mode	XL	Index register X's lower 8 bits
ABS, Y	Absolute indexed Y addressing mode	Y	Index register Y
ABL	Absolute long addressing mode	YH	Index register Y's upper 8 bits
ABL, X	Absolute long indexed X addressing mode	YL	Index register Y's lower 8 bits
(ABS)	Absolute indirect addressing mode	S	Stack pointer
L (ABS)	Absolute indirect long addressing mode	PC	Program counter
(ABS, X)	Absolute indexed X indirect addressing mode	PCH	Program counter's upper 8 bits
STK	Stack addressing mode	PCL	Program counter's lower 8 bits
REL	Relative addressing mode	PG	Program bank register
DIR, b, REL	Direct bit relative addressing mode	DT	Data bank register
ABS, b, REL	Absolute bit relative addressing mode	DPR	Direct page register
SR	Stack pointer relative addressing mode	DPRH	Direct page register's upper 8 bits
(SR), Y	Stack pointer relative indirect indexed Y addressing mode	DPRL	Direct page register's lower 8 bits
		PS	Processor status register
BLK	Block transfer addressing mode	PSH	Processor status register's upper 8 bits
C	Carry flag	PSL	Processor status register's lower 8 bits
Z	Zero flag	PSb	Processor status register's b-th bit
I	Interrupt disable flag	M(S)	Contents of memory at address indicated by stack pointer
D	Decimal operation mode flag		
x	Index register length selection flag	Mb	b-th memory location
m	Data length selection flag	ADG	Value of 24-bit address's upper 8-bit (A_{23}–A_{16})
V	Overflow flag	ADH	Value of 24-bit address's middle 8-bit (A_{15}–A_8)
N	Negative flag	ADL	Value of 24-bit address's lower 8-bit (A_7–A_0)
IPL	Processor interrupt priority level	op	Operation code
+	Addition	n	Number of cycle
–	Subtraction	#	Number of byte
*	Multiplication	i	Number of transfer byte or rotation
/	Division	i_1, i_2	Number of registers pushed or pulled
∧	Logical AND		
∨	Logical OR		

EOL announced

Appendix 7. Hexadecimal instruction code table

INSTRUCTION CODE TABLE-1

D7–D4 \ D3–D0 / Hexadecimal notation		0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0	BRK	ORA A,(DIR,X)		ORA A,SR	SEB DIR,b	ORA A,DIR	ASL DIR	ORA A,L(DIR)	PHP	ORA A,IMM	ASL A	PHD	SEB ABS,b	ORA A,ABS	ASL ABS	ORA A,ABL
0001	1	BPL	ORA A,(DIR),Y	ORA A,(DIR)	ORA A,(SR),Y	CLB DIR,b	ORA A,DIR,X	ASL DIR,X	ORA A,L(DIR),Y	CLC	ORA A,ABS,Y	DEC A	TAS	CLB ABS,b	ORA A,ABS,X	ASL ABS,X	ORA A,ABL,X
0010	2	JSR ABS	AND A,(DIR,X)	JSR ABL	AND A,SR	BBS DIR,b,R	AND A,DIR	ROL DIR	AND A,L(DIR)	PLP	AND A,IMM	ROL A	PLD	BBS ABS,b,R	AND A,ABS	ROL ABS	AND A,ABL
0011	3	BMI	AND A,(DIR),Y	AND A,(DIR)	AND A,(SR),Y	BBC DIR,b,R	AND A,DIR,X	ROL DIR,X	AND A,L(DIR),Y	SEC	AND A,ABS,Y	INC A	TSA	BBC ABS,b,R	AND A,ABS,X	ROL ABS,X	AND A,ABL,X
0100	4	RTI	EOR A,(DIR,X)	**Note 1**	EOR A,SR	MVP	EOR A,DIR	LSR DIR	EOR A,L(DIR)	PHA	EOR A,IMM	LSR A	PHG	JMP ABS	EOR A,ABS	LSR ABS	EOR A,ABL
0101	5	BVC	EOR A,(DIR),Y	EOR A,(DIR)	EOR A,(SR),Y	MVN	EOR A,DIR,X	LSR DIR,X	EOR A,L(DIR),Y	CLI	EOR A,ABS,Y	PHY	TAD	JMP ABL	EOR A,ABS,X	LSR ABS,X	EOR A,ABL,X
0110	6	RTS	ADC A,(DIR,X)	PER	ADC A,SR	LDM DIR	ADC A,DIR	ROR DIR	ADC A,L(DIR)	PLA	ADC A,IMM	ROR A	RTL	JMP (ABS)	ADC A,ABS	ROR ABS	ADC A,ABL
0111	7	BVS	ADC A,(DIR),Y	ADC A,(DIR)	ADC A,(SR),Y	LDM DIR,X	ADC A,DIR,X	ROR DIR,X	ADC A,L(DIR),Y	SEI	ADC A,ABS,Y	PLY	TDA	JMP (ABS,X)	ADC A,ABS,X	ROR ABS,X	ADC A,ABL,X
1000	8	BRA REL	STA A,(DIR,X)	BRA REL	STA A,SR	STY DIR	STA A,DIR	STX DIR	STA A,L(DIR)	DEY	**Note 2**	TXA	PHT	STY ABS	STA A,ABS	STX ABS	STA A,ABL
1001	9	BCC	STA A,(DIR),Y	STA A,(DIR)	STA A,(SR),Y	STY DIR,X	STA A,DIR,X	STX DIR,Y	STA A,L(DIR),Y	TYA	STA A,ABS,Y	TXS	TXY	LDM ABS	STA A,ABS,X	LDM ABS,X	STA A,ABL,X
1010	A	LDY IMM	LDA A,(DIR,X)	LDX IMM	LDA A,SR	LDY DIR	LDA A,DIR	LDX DIR	LDA A,L(DIR)	TAY	LDA A,IMM	TAX	PLT	LDY ABS	LDA A,ABS	LDX ABS	LDA A,ABL
1011	B	BCS	LDA A,(DIR),Y	LDA A,(DIR)	LDA A,(SR),Y	LDY DIR,X	LDA A,DIR,X	LDX DIR,Y	LDA A,L(DIR),Y	CLV	LDA A,ABS,Y	TSX	TYX	LDY ABS,X	LDA A,ABS,X	LDX ABS,Y	LDA A,ABL,X
1100	C	CPY IMM	CMP A,(DIR,X)	CLP IMM	CMP A,SR	CPY DIR	CMP A,DIR	DEC DIR	CMP A,L(DIR)	INY	CMP A,IMM	DEX	WIT	CPY ABS	CMP A,ABS	DEC ABS	CMP A,ABL
1101	D	BNE	CMP A,(DIR),Y	CMP A,(DIR)	CMP A,(SR),Y	PEI	CMP A,DIR,X	DEC DIR,X	CMP A,L(DIR),Y	CLM	CMP A,ABS,Y	PHX	STP	JMP L(ABS)	CMP A,ABS,X	DEC ABS,X	CMP A,ABL,X
1110	E	CPX IMM	SBC A,(DIR,X)	SEP IMM	SBC A,SR	CPX DIR	SBC A,DIR	INC DIR	SBC A,L(DIR)	INX	SBC A,IMM	NOP	PSH	CPX ABS	SBC A,ABS	INC ABS	SBC A,ABL
1111	F	BEQ	SBC A,(DIR),Y	SBC A,(DIR)	SBC A,(SR),Y	PEA	SBC A,DIR,X	INC DIR,X	SBC A,L(DIR),Y	SEM	SBC A,ABS,Y	PLX	PUL	JSR (ABS,X)	SBC A,ABS,X	INC ABS,X	SBC A,ABL,X

EOL announced

Notes 1: 42_{16} specifies the contents of the INSTRUCTION CODE TABLE-2.
About the second word's codes, refer to the INSTRUCTION CODE TABLE-2.
2: 89_{16} specifies the contents of the INSTRUCTION CODE TABLE-3.
About the second word's codes, refer to the INSTRUCTION CODE TABLE-2.

Appendix 7. Hexadecimal instruction code table

INSTRUCTION CODE TABLE-2 (The first word's code of each instruction is 42_{16})

D7–D4 \ D3–D0	Hexadecimal notation	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0		ORA B,(DIR,X)		ORA B,SR		ORA B,DIR		ORA B,L(DIR)		ORA B,IMM	ASL B			ORA B,ABS		ORA B,ABL
0001	1		ORA B,(DIR),Y	ORA B,(DIR)	ORA B,(SR),Y		ORA B,DIR,X		ORA B,L(DIR),Y		ORA B,ABS,Y	DEC B	TBS		ORA B,ABS,X		ORA B,ABL,X
0010	2		AND B,(DIR,X)		AND B,SR		AND B,DIR		AND B,L(DIR)		AND B,IMM	ROL B			AND B,ABS		AND B,ABL
0011	3		AND B,(DIR),Y	AND B,(DIR)	AND B,(SR),Y		AND B,DIR,X		AND B,L(DIR),Y		AND B,ABS,Y	INC B	TSB		AND B,ABS,X		AND B,ABL,X
0100	4		EOR B,(DIR,X)		EOR B,SR		EOR B,DIR		EOR B,L(DIR)	PHB	EOR B,IMM	LSR B			EOR B,ABS		EOR B,ABL
0101	5		EOR B,(DIR),Y	EOR B,(DIR)	EOR B,(SR),Y		EOR B,DIR,X		EOR B,L(DIR),Y		EOR B,ABS,Y		TBD		EOR B,ABS,X		EOR B,ABL,X
0110	6		ADC B,(DIR,X)		ADC B,SR		ADC B,DIR		ADC B,L(DIR)	PLB	ADC B,IMM	ROR B			ADC B,ABS		ADC B,ABL
0111	7		ADC B,(DIR),Y	ADC B,(DIR)	ADC B,(SR),Y		ADC B,DIR,X		ADC B,L(DIR),Y		ADC B,ABS,Y		TDB		ADC B,ABS,X		ADC B,ABL,X
1000	8		STA B,(DIR,X)		STA B,SR		STA B,DIR		STA B,L(DIR)			TXB			STA B,ABS		STA B,ABL
1001	9		STA B,(DIR),Y	STA B,(DIR)	STA B,(SR),Y		STA B,DIR,X		STA B,L(DIR),Y	TYB	STA B,ABS,Y				STA B,ABS,X		STA B,ABL,X
1010	A		LDA B,(DIR,X)		LDA B,SR		LDA B,DIR		LDA B,L(DIR)	TBY	LDA B,IMM	TBX			LDA B,ABS		LDA B,ABL
1011	B		LDA B,(DIR),Y	LDA B,(DIR)	LDA B,(SR),Y		LDA B,DIR,X		LDA B,L(DIR),Y		LDA B,ABS,Y				LDA B,ABS,X		LDA B,ABL,X
1100	C		CMP B,(DIR,X)		CMP B,SR		CMP B,DIR		CMP B,L(DIR)		CMP B,IMM				CMP B,ABS		CMP B,ABL
1101	D		CMP B,(DIR),Y	CMP B,(DIR)	CMP B,(SR),Y		CMP B,DIR,X		CMP B,L(DIR),Y		CMP B,ABS,Y				CMP B,ABS,X		CMP B,ABL,X
1110	E		SBC B,(DIR,X)		SBC B,SR		SBC B,DIR		SBC B,L(DIR)		SBC B,IMM				SBC B,ABS		SBC B,ABL
1111	F		SBC B,(DIR),Y	SBC B,(DIR)	SBC B,(SR),Y		SBC B,DIR,X		SBC B,L(DIR),Y		SBC B,ABS,Y				SBC B,ABS,X		SBC B,ABL,X

EOL announced

APPENDIX

Appendix 7. Hexadecimal instruction code table

INSTRUCTION CODE TABLE-3 (The first word's code of each instruction is 89_{16})

D7–D4 \ D3–D0	Hexadecimal notation	0000	0001	0010	0011	0100	0101	0110	0111	1000	1001	1010	1011	1100	1101	1110	1111
		0	1	2	3	4	5	6	7	8	9	A	B	C	D	E	F
0000	0		MPY (DIR,X)		MPY SR		MPY DIR		MPY L(DIR)		MPY IMM				MPY ABS		MPY ABL
0001	1		MPY (DIR),Y	MPY (DIR)	MPY (SR),Y		MPY DIR,X		MPY L(DIR),Y		MPY ABS,Y				MPY ABS,X		MPY ABL,X
0010	2		DIV (DIR,X)		DIV SR		DIV DIR		DIV L(DIR)	XAB	DIV IMM				DIV ABS		DIV ABL
0011	3		DIV (DIR),Y	DIV (DIR)	DIV (SR),Y		DIV DIR,X		DIV L(DIR),Y		DIV ABS,Y				DIV ABS,X		DIV ABL,X
0100	4										RLA IMM						
0101	5																
0110	6																
0111	7																
1000	8																
1001	9																
1010	A																
1011	B																
1100	C			LDT IMM													
1101	D																
1110	E																
1111	F																

EOL announced

Appendix 8. Countermeasure against noise

General countermeasure examples against noise are described below. Although the effect of these countermeasure depends on each system, refer to the following when an noise-related problem occurs.

1. Short wiring length

The wiring on a printed circuit board may function as an antenna which feeds noise into the microcomputer. The shorter the total wiring length (by mm unit), the less possibility of noise insertion into the microcomputer.

(1) Wiring for $\overline{\text{RESET}}$ pin

Make the length of wiring connected to the $\overline{\text{RESET}}$ pin as short as possible.
In particular, connect a capacitor between the $\overline{\text{RESET}}$ pin and the Vss pin with the shortest possible wiring (within 20 mm).

Reason: If noise is input to the $\overline{\text{RESET}}$ pin, the microcomputer restarts operation before the internal state of the microcomputer is completely initialized. This may cause a program runaway.

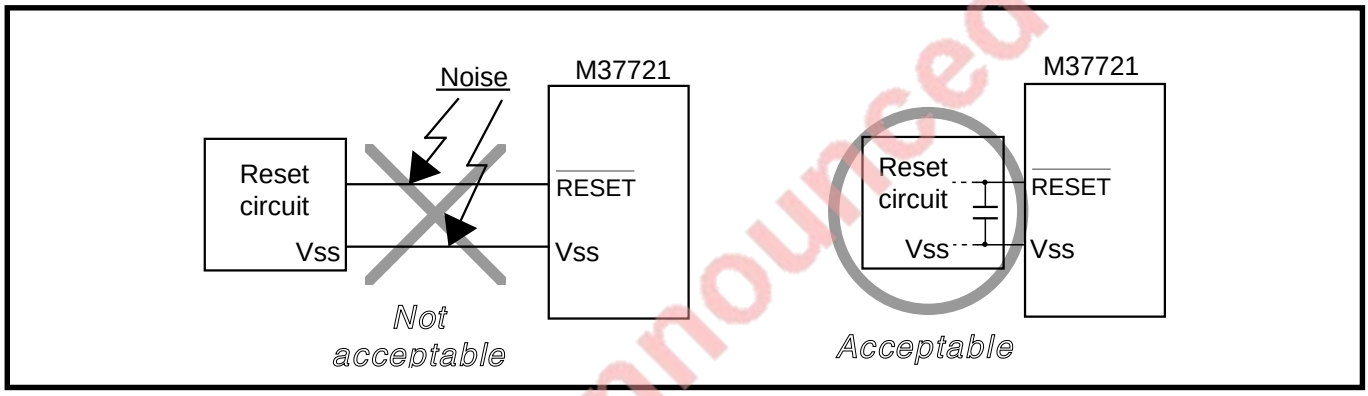


Fig. 3 Wiring for $\overline{\text{RESET}}$ pin

(2) Wiring for clock input/output pins

- Make the length of wiring connected to the clock input/output pins as short as possible.
- Make the length of wiring between the grounding lead of the capacitor, which is connected to the oscillator, and the Vss pin of the microcomputer, as short as possible (within 20 mm).
- Separate the Vss pattern for oscillation from all other Vss patterns. (Refer to **"Figure 11."**)

Reason: The microcomputer's operation synchronizes with a clock generated by the oscillation circuit.
If noise enters clock I/O pins, clock waveforms may be deformed. This may cause a malfunction or a program runaway.
Also, if the noise causes a potential difference between the Vss level of the microcomputer and the Vss level of an oscillator, the correct clock will not be input in the microcomputer.

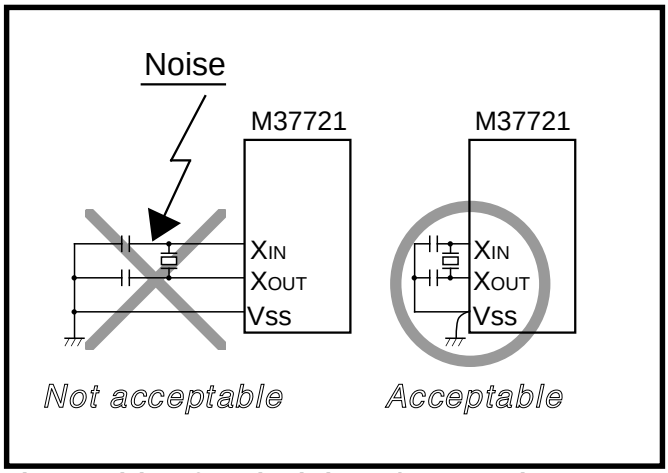


Fig. 4 Wiring for clock input/output pins

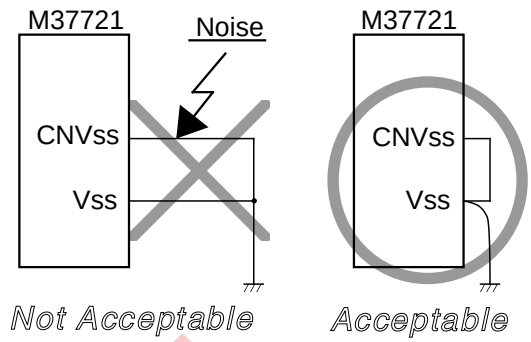
EOL announced

Appendix 8. Countermeasure against noise

(3) Wiring for CNVss pin

Connect CNVss pin to the Vss pin with the shortest possible wiring.

Reason: The processor mode of the microcomputer is influenced by a potential at the CNVss pin when the CNVss pin and the Vcc or Vss pin are connected.
If the noise causes a potential difference between the CNVss pin and the Vss or Vcc pin, the processor mode may become unstable. This may cause a microcomputer malfunction or a program runaway.



When connecting the CNVss and Vcc pins, connect them in the shortest possible distance, also.

Fig. 5 Wiring for CNVss pin

2. Connection of bypass capacitor between Vss and Vcc lines

Connect an approximate 0.1 μF bypass capacitor as follows:

- Connect a bypass capacitor between the Vss and Vcc pins, at equal lengths.
- The wiring connecting the bypass capacitor between the Vss and Vcc pins should be as short as possible.
- Use thicker wiring for the Vss and Vcc lines than that for the other signal lines.

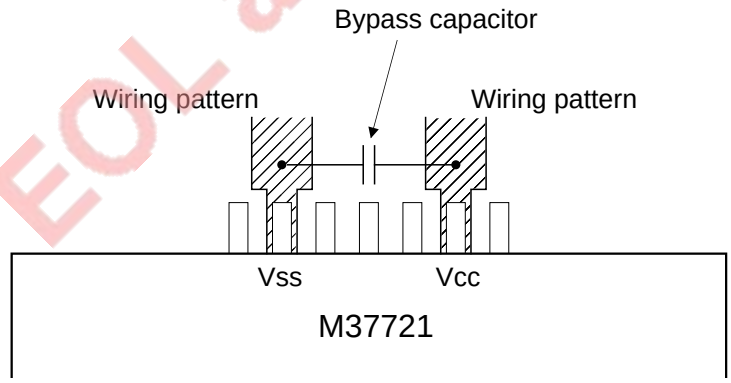


Fig. 6 Bypass capacitor between Vss and Vcc lines

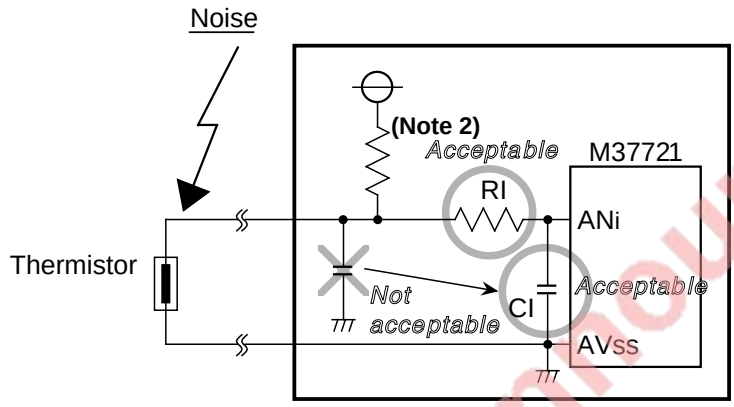
EOL announced

3. Wiring for analog input pins, analog power source pins, etc.

(1) Processing for analog input pins

- Connect a resistor to the analog signal line, which is connected to an analog input pin, in series. Additionally, connect the resistor to the microcomputer as close as possible.
- Connect a capacitor between the analog input pin and the AVss pin, as close to the AVss pin as possible.

Reason: A signal which is input to the analog input pin is usually an output signal from a sensor. The sensor, which detects changes in status, is installed far from the microcomputer's printed circuit board. Therefore, this long wiring between them becomes an antenna which picks up noise and feeds it into the microcomputer's analog input pin.
If a capacitor between an analog input pin and the AVss pin is grounded far away from the AVss pin, noise on the GND line may enter the microcomputer through the capacitor.



Reference values
RI : Approximate 100 Ω to 1000 Ω
CI : Approximate 100 pF to 1000 pF

Notes 1 : Design an external circuit for the ANi pin so that charge/discharge is available within 1 cycle of ϕ_{AD}.
2 : This resistor and thermistor are used to divide resistance.

Fig. 7 Countermeasure example against noise for analog input pin using thermistor

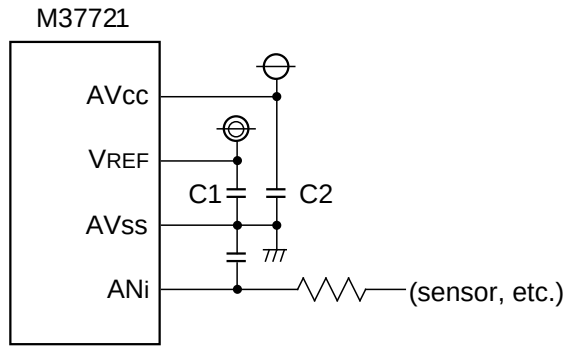
EOL announced

Appendix 8. Countermeasure against noise

(2) Processing for analog power source pins, etc.

- Use independent power sources for the Vcc, AVcc and V_{REF} pins.
- Insert capacitors between the AVcc and AVss pins, and between the V_{REF} and AVss pins.

Reasons: Prevents the A-D converter from noise on the Vcc line.



Reference values
C1 ≥ 0.47 µF
C2 ≥ 0.47 µF

Note : Connect capacitors using the thickest, shortest wiring possible.

Fig. 8 Processing for analog power source pins, etc.

EOL announced

4. Oscillator protection

The oscillator, which generates the basic clock for the microcomputer operations, must be protected from the affect of other signals.

(1) Distance oscillator from signal lines with large current flows

Install the microcomputer, especially the oscillator, as far as possible from signal lines which handle currents larger than the microcomputer current value tolerance.

Reason: The microcomputer is used in systems which contain signal lines for controlling motors, LEDs, thermal heads, etc. Noise occurs due to mutual inductance when a large current flows through the signal lines.

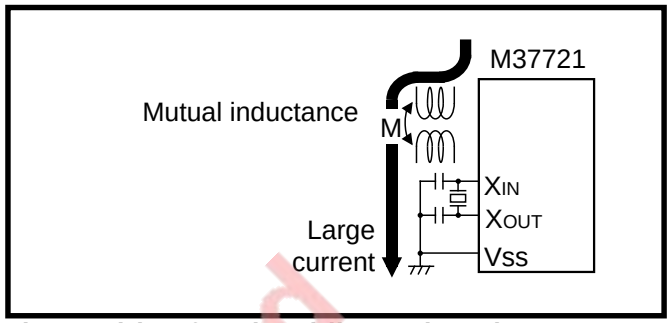


Fig. 9 Wiring for signal lines where large current flows

(2) Distance oscillator from signal lines with frequent potential level changes

- Install an oscillator and its wiring pattern away from signal lines where potential levels change frequently.
- Do not cross these signal lines over the clock-related or noise-sensitive signal lines.

Reason: Signal lines with frequently changing potential levels may affect other signal lines at a rising or falling edge. In particular, if the lines cross over a clock-related signal line, clock waveforms may be deformed, which causes a microcomputer malfunction or a program runaway.

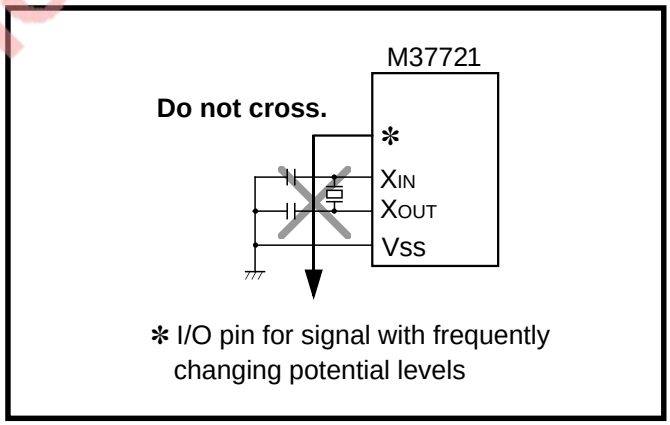


Fig. 10 Wiring for signal lines where potential levels frequently change

(3) Oscillator protection using Vss pattern

Print a Vss pattern on the bottom (soldering side) of a double-sided printed circuit board, under the oscillator mount position.

Connect the Vss pattern to the Vss pin of the microcomputer with the shortest possible wiring, separating it from other Vss patterns.

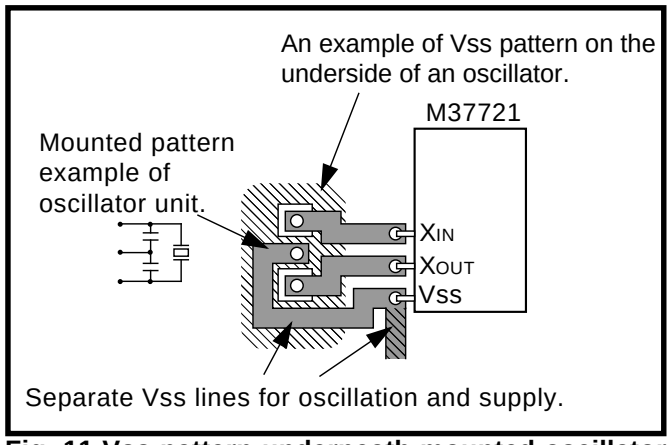


Fig. 11 Vss pattern underneath mounted oscillator

EOL announced

Appendix 8. Countermeasure against noise

5. Setup for I/O ports

Setup I/O ports by hardware and software as follows:

<Hardware protection>

- Connect a resistor of 100 Ω or more to an I/O port in series.

<Software protection>

- Read the data of an input port several times to confirm that input levels are equal.
- Since the output data may reverse because of noise, rewrite data to the output port's Pi register periodically.
- Rewrite data to port Pi direction registers periodically.

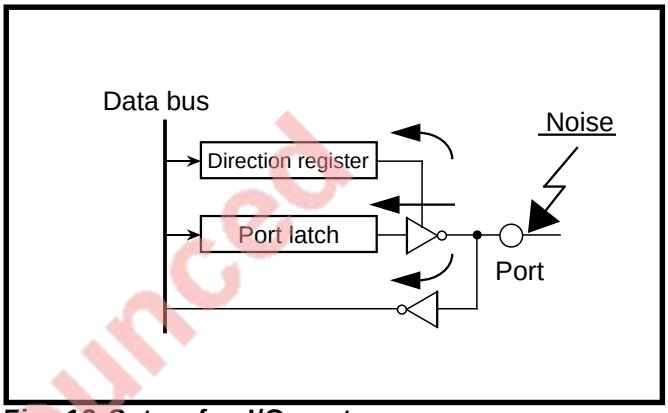


Fig. 12 Setup for I/O ports

6. Reinforcement of the power source line

- For the Vss and Vcc lines, use thicker wiring than that of other signal lines.
- When using a multilayer printed circuit board, the Vss and Vcc patterns must each be one of the middle layers.
- The following is necessary for double-sided printed circuit boards:
  - •On one side, the microcomputer is installed at the center, and the Vss line is looped or meshed around it. The vacant area is filled with the Vss line.
  - •On the opposite side, the Vcc line is wired the same as the Vss line.
  - •The power source lines of external devices which are connected by bus to the microcomputer must be connected to the microcomputer's power source lines with the shortest possible wiring.

Reasons: With external devices connected to the microcomputer, the levels of many of the signal lines (total external address buses: 24 bits) may change simultaneously, causing noise on the power source line.

EOL announced

Appendix 9. 7721 Group Q & A

Information which may be helpful in fully utilizing the 7721 Group is provided in Q & A format.
In Q & A, as a rule, one question and its answer are summarized within one page. The upper box on each page is a question, and a box below the question is its answer. (If a question or an answer extends to two or more pages, there is a page number at the lower right corner.)
At the upper right corner of each page, the main function related to the contents of description in that page is listed.

EOL announced

APPENDIX

Appendix 9. 7721 Group Q & A

Q

Is there any SFR to which a certain instruction cannot be used for writing ?

A

(1) Use the **LDM** or **STA** instruction to write to the registers or the bits listed below. Do not use read-modify-write instructions (i.e., **CLB, SEB, ASL, ASR, DEC, INC, LSR, ROL,** and **ROR**).

Pulse output data register 0, 1 (addresses $1A_{16}$, $1C_{16}$)
UART0, 1 baud rate register (addresses 31_{16}, 39_{16})
UART0, 1 transmit buffer register (addresses 33_{16}, 32_{16}, $3B_{16}$, $3A_{16}$)
Timer A2–A4 two-phase pulse signal processing select bit (bits 5–7 at address 44_{16})
Timer A2–A4 register (addresses $4A_{16}$–$4F_{16}$; one-shot pulse mode or pulse width modulation mode)
Refresh timer (address 66_{16})

(2) Use the **SEB** or **CLB** instruction to write to the following register.

DMAC control register H (address 69_{16} ; when any of bits 4 to 7 = “1”)

EOL announced

Reset, STP instruction, WIT instruction

Q

Is it possible to distinguish power-on reset from hardware reset for terminating the stop or wait mode ?

A

The contents of the internal RAM is undefined after power-on reset. On the other hand, the contents of the internal RAM are retained when performing hardware reset in the stop or wait mode with Vcc ≥ 2 V.
Accordingly, write a certain data to the internal RAM before executing **STP** or **WIT** instruction, and judge by checking the contents of the internal RAM after hardware reset.

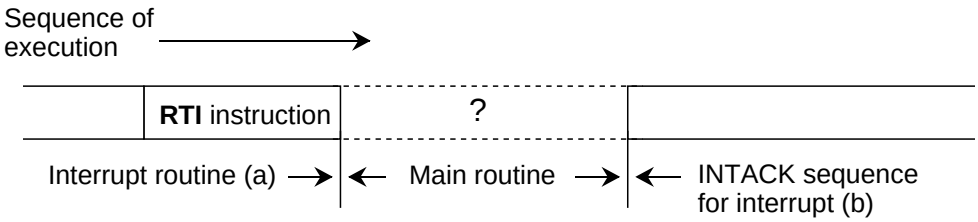
EOL announced

APPENDIX

Appendix 9. 7721 Group Q & A

Q

If an interrupt request (b) occurs while executing an interrupt routine (a), is it true that the main routine is not executed at all from when the execution of the interrupt routine (a) is completed until the execution of the INTACK sequence for the next interrupt (b) starts?



Conditions:

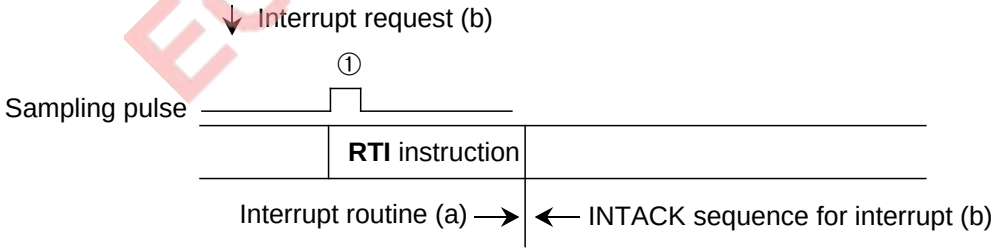
- I is cleared to "0" by executing the **RTI** instruction.
- Iinterrupt priority level of interrupt (b) is higher than IPL of main routine.
- Interrupt priority detection time is 2 cycles of ϕ.

EOL announced

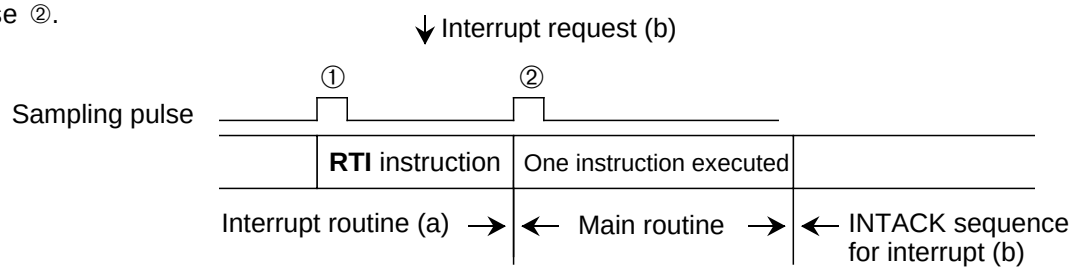
A

Sampling for interrupt requests is performed by sampling pulses generated synchronously with the CPU's op-code fetch cycles.

(1) If the next interrupt request (b) occurs before sampling pulse ① for the **RTI** instruction is generated, the microcomputer executes the INTACK sequence for (b) without executing the main routine (not even one instruction). It is because that sampling is completed while executing the **RTI** instruction.



(2) If the next interrupt request (b) occurs immediately after sampling pulse ① is generated, the microcomputer executes one instruction of the main routine before executing the INTACK sequence for (b). It is because that the interrupt request is sampled by the next sampling pulse ②.



Interrupt

Q

Suppose that there is a routine which should not accept one certain interrupt request. (The other interrupt request are acceptable).
Although when the interrupt priority level select bits for the above interrupt are set to "000_2," in other words, when this interrupt is set to be disabled, this interrupt request is actually accepted immediately after change of the priority level. Why did this occur and what should I do about it?

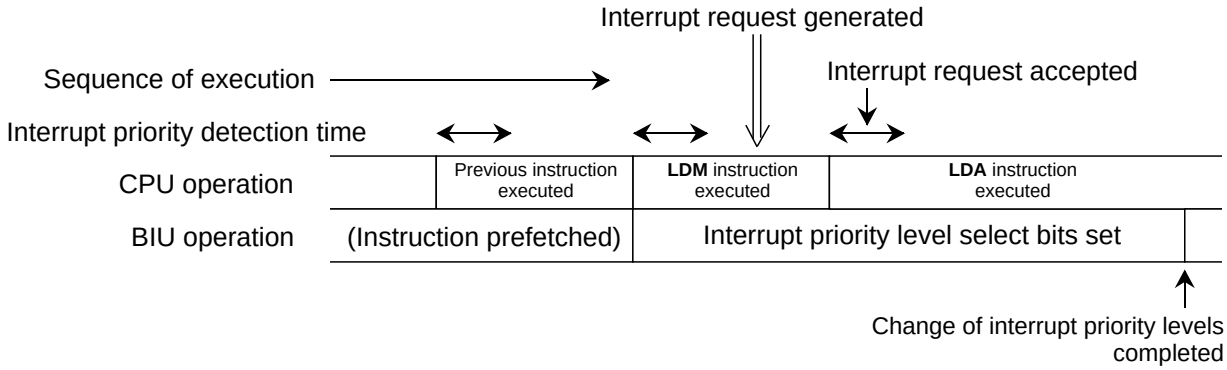
```
                :
Interrupt request is accepted in this interval →
                LDM #00H, XXXIC ; Writes "0002" to interrupt priority level select bits.
                                ; Clears interrupt request bit to "0."
                LDA A,DATA      ; Instruction at the beginning of the routine which
                                  should not accept one certain interrupt request.
                :               ;
```

A

As for the change of the interrupt priority level, when the following are met, the microcomputer may pretend to accept an interrupt request immediately after this interrupt is set to be disabled:
- The next instruction (in the above example, it is the **LDA** instruction) is already stored into a instruction queue buffer for the BIU.
- Conditions for accepting the instruction which should not be accepted are satisfied immediately before the next instruction in the instruction queue buffer is executed.

When writing to a memory or an I/O, the CPU passes the address and data to the BIU. Then, the CPU executes the next instruction in the instruction queue buffer while the BIU is writing data into the actual address. Detection of interrupt priority level is performed at the beginning of each instruction.

In the above case, the CPU executes the next instruction before the BIU completes the change of the interrupt priority level. Therefore, when the interrupt priority level is detected synchronously with the execution of the next instruction, the interrupt priority level before the change is detected and its interrupt request is accepted.



(1/2)

FOE announced

Interrupt

A

To prevent this problem, after change of the interrupt priority level is completed, use software to execute the routine that should not accept a certain interrupt request. The following shows a sample program.

[Sample program]

After an instruction which writes "000_2" to the interrupt priority level select bits, fill the instruction queue buffer with the **NOP** instruction to make the next instruction not to be executed before the writing is completed.

```
    :
LDM  #00H, XXXIC  ; Sets the interrupt priority level select bits to "0002."
NOP               ;
NOP               ;
NOP               ;
LDA  A,DATA       ; Instruction at the beginning of the routine that should not accept a certain
  :                 interrupt request
```

(2/2)

EOL announced

Interrupt

Q

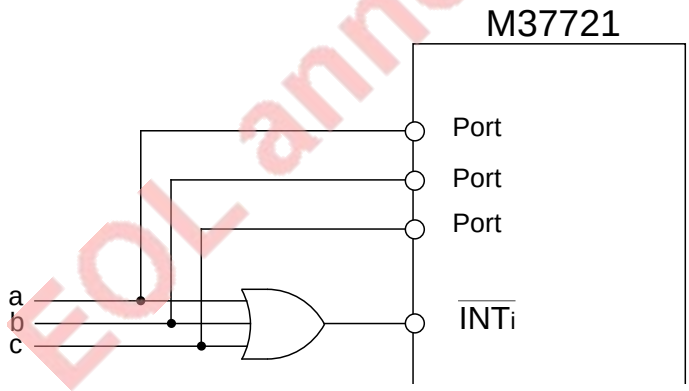
(1) Which timing of clock ϕ_1 is the external interrupts (input signals to the $\overline{INT_i}$ pin) detected?
(2) How can four or more external interrupt input pins ($\overline{INT_i}$) be used?

A

(1) In both the edge sense and level sense, external interrupt requests occur when the input signal to the $\overline{INT_i}$ pin changes its level. This is independent of clock ϕ_1.
In the edge sense, the interrupt request bit is set to "1" at this time.

(2) There are two methods: one uses external interrupt's level sense, and the other uses the timer's event counter mode.

① **Method using external interrupt's level sense**
As for hardware, input a logical sum of multiple interrupt signals (e.g., 'a', 'b', and 'c') to the $\overline{INT_i}$ pin, and input each signal to each corresponding port.
As for software, check the ports' input levels in the $\overline{INT_i}$ interrupt routine in order to detect which signal ('a', 'b', or 'c') was input.



② **Method using timer's event counter mode**
As for hardware, input interrupt signals to the TAi_{IN} pins or TBi_{IN} pins.
As for software, set the timer's operating mode to the event counter mode. Then, set a value "0000_{16}" into the timer register and select the valid edge.
The timer's interrupt request occurs when an interrupt signal (selected valid edge) is input.

EOL announced

APPENDIX

Appendix 9. 7721 Group Q & A

Stack, DRAM

Q

What are there the stack bank select bit (bit 7 at address $5E_{16}$) for?

A

It is supposed that DRAM is used as the stack area.

When connecting DRAM, the stack pointer addressing mode or stack operation instruction etc. can be used. It is because all of 64 Kbytes can be used as the stack area when bank FF_{16} which is assigned to DRAM is set as the stack area.
(The internal RAM also functions as the temporary area or the register file which is accessed frequently because the internal RAM can be accessed with no Wait. Accordingly, it is expected that the capacity will lack to be used as the stack area. As for the M37721, DRAM area can be set as the stack area because cheap DRAM can be connected.)
Use bank 0 which is assigned to the internal RAM area as the stack area when DRAM is not connected or the internal RAM is sufficient to be used as the stack area.

EOL announced

DRAM, WIT instruction

Q

Are there methods to refresh DRAM in the wait mode?

EOL announced

A

In the wait mode, DRAM refresh function does not operate, but the watchdog timer, timer A, and timer B operate. Accordingly, DRAM can be refreshed by using these timers and ports.

(1) Method using watchdog timer
Return from the wait mode by the watchdog timer interrupt. Control ports $P10_4$, $P10_5$ by software and perform the $\overline{CAS}$-before-$\overline{RAS}$-refresh.

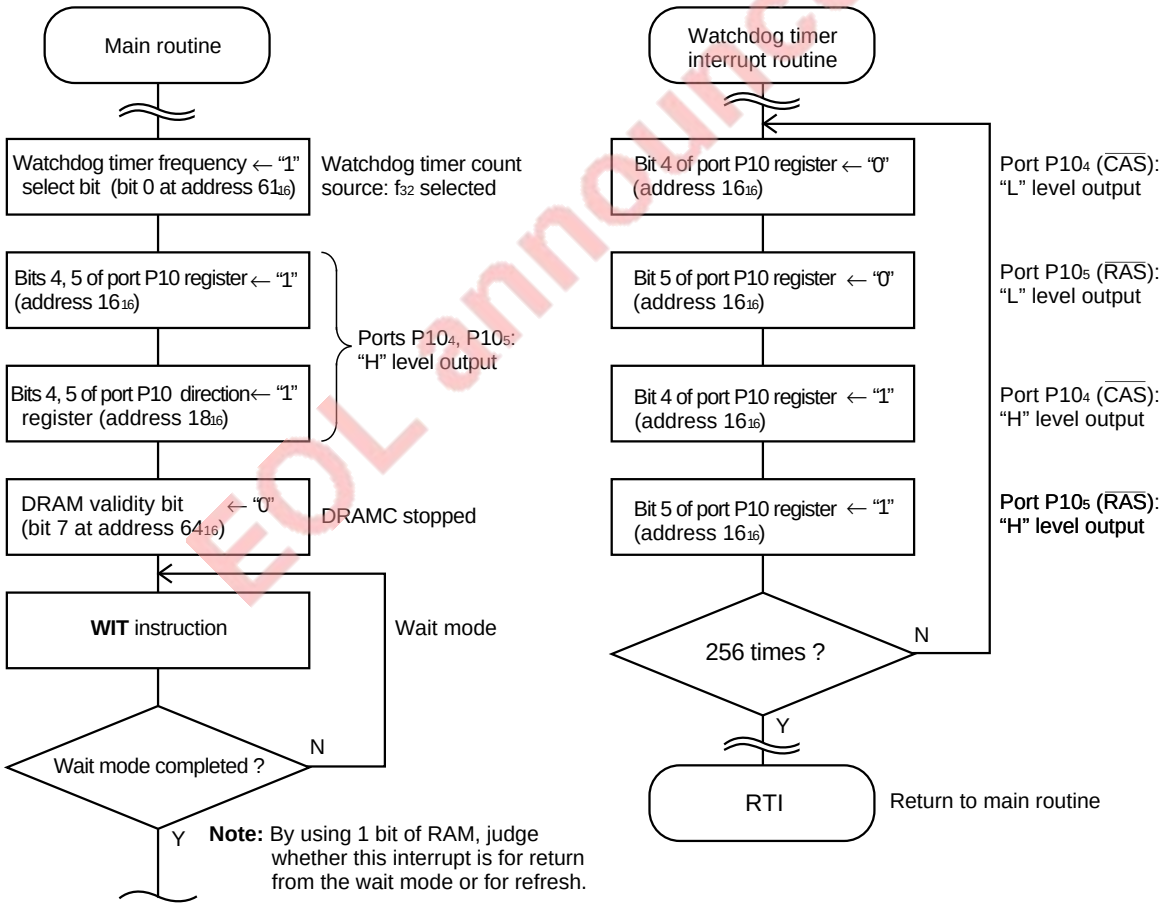
Interval of watchdog timer interrupt

$f(X_{IN})$	f_{32} selected	f_{512} selected
25 MHz	2.621 ms	41.943 ms
16 MHz	4.096 ms	65.536 ms

Example 1: A case in 1024 refresh cycles, every 16.4 ms, $f(X_{IN})$ = 16 MHz, watchdog timer count source = f_{32}

•DRAM refresh is performed 256 times. This refresh is performed by every watchdog timer interrupt. (See flow chart ①.)

Flow chart ①



EOL announced

(1/2)

A

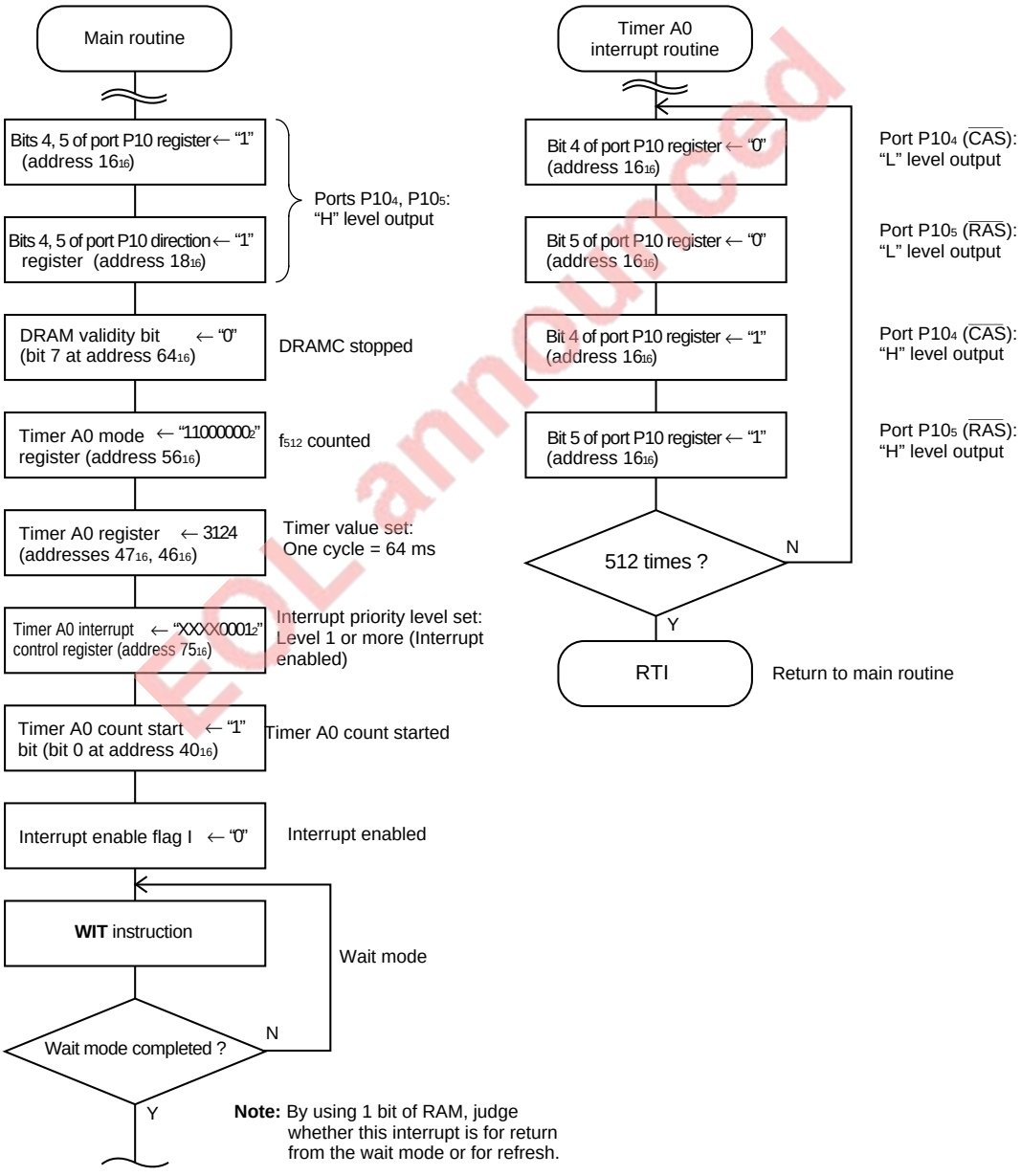
(2) Method using timer A or timer B

Return from the wait mode by a timer A (or timer B) interrupt every definite time. Control ports $P10_4$, $P10_5$ by software and perform the $\overline{CAS}$-before-$\overline{RAS}$-refresh.

Example 2: A case in 512 refresh cycles, every 64 ms, $f(X_{IN})$ = 25 MHz, timer A0 used

•DRAM refresh is performed 512 times by timer A0 interrupts. This interrupt occurs every 64 ms. (See flow chart ②.)

Flow chart ②



Note: By using 1 bit of RAM, judge whether this interrupt is for return from the wait mode or for refresh.

EOL announced

(2/2)

DRAM

Q

How is the program execution time affected when using DRAM ?

A

When the M37721 uses DRAM, the execution time is affected as follows:
•CPU stops and DRAM refresh cycle is inserted.
•1-bus cycle becomes 3ϕ when accessing DRAM.

(1) Refresh method of the M37721's DRAMC is the dispersion refresh and 5 cycles of ϕ are necessary for one refresh. The rate occupied by the DRAM refresh cycle during the program execution time is described below.

Rate occupied by DRAM refresh cycle during program execution time

Refresh interval	Rate occupied by DRAM refresh cycle	
	$f(X_{IN})$ = 25 MHz	$f(X_{IN})$ = 16 MHz
15.625 μs (Case of 512 refresh cycles, every 8 ms)	2.6 %	4.2 %
125 μs (Case of 512 refresh cycles, every 64 ms)	0.3 %	0.5 %

(2) The comparison results of two sample programs' execution times are listed below; one is for the case where SRAM is used and the other is for the case where DRAM is used.

Use conditions : Execution program — Sample program B (See (2/2))
$f(X_{IN})$ — 16 MHz
External data bus width — 16 bits
Refresh interval — 13 μs

Memory used as work area	Software wait valid area	Execution time	Speed comparison
SRAM	Nothing	3.4 ms	1.00
SRAM	ROM and RAM	5.0 ms	1.47
DRAM (bank FF_{16})	Nothing	3.9 ms	1.15
DRAM (bank FF_{16})	ROM	5.2 ms	1.53

EOL announced

(1/2)

DRAM

A

●Sample program B

```
                SEP     X
                CLM
                .DATA   16
                .INDEX  8
                LDY     #69
LOOP0:          LDX     #69
LOOP1:          ASL     SOUR, X
                SEM
                .DATA   8
                ROL     SOUR+2, X
                ROL     B
                CLM
                .DATA   16
                ROR     A
                DEX
                DEX
                DEX
                BNE     LOOP1
                STA     A, DEST, Y
                SEM
                .DATA   8
                STA     B, DEST+2, Y
                CLM
                .DATA   16
                DEY
                DEY
                DEY
                BNE     LOOP0
```

✻ SOUR, DEST : Work areas

EOL announced

(2/2)

APPENDIX

Appendix 9. 7721 Group Q & A

Q

When detecting the software runaway by the watchdog timer, if the same value as the contents of the reset vector address is set to the watchdog timer interrupt vector address, not performing software reset, how does it result in?
When branching to the reset branch address within the watchdog timer interrupt routine, how does it result in?

A

The CPU registers and the SFR are not initialized in the above-mentioned way. Accordingly, the user must initialize all of them by software.
Note that the processor interrupt priority level (IPL) retains “7” of the watchdog timer interrupt priority level and is not initialized. Consequently, all interrupt requests cannot be accepted. When rewriting the IPL by software, save once the 16-bit immediate value to the stack area and then restore that 16-bit immediate value to all bits of the processor status register (PS).

When a software runaway occurs, we recommend to use software reset in order to initialize the microcomputer.

EOL announced

Appendix 10. Differences between 7721 Group and 7720 Group

Table 2 Differences between M37721S2BFP and M37720S1AFP

Item		M37721S2BFP	M37720S1AFP
Internal RAM size		1024 bytes **(Note)**	512 bytes
External clock input frequency		25 MHz (maximum)	16 MHz (maximum)
Instruction execution time (minimum)		160 ns	250 ns
Real-time output	Bit configuration of real–time output channel	4 bits × 2 channels, or 6 bits × 1channel and 2 bits × 1channel	4 bits × 2 channels
	Port latch state after using real-time output	Retains the value before using real-time output	Undefined
Limitation for instruction used when writing to interrupt control register		Nothing (**LDM**, **STA** instructions can be used.)	Exists (**LDM**, **STA** instructions cannot be used.)
Serial I/O	Timing when overrun error flag becomes “0”	One of the following: •When setting the receive enable bit to “0” •When setting the serial I/O mode select bits to “000_2”	One of the following: •When setting the receive enable bit to “0” •When setting the serial I/O mode select bits to “000_2” •When reading the receive buffer register
	Conditions for outputting “L” of $\overline{\text{RTS}}$ signal in clock synchronous serial I/O mode	When all of the following are satisfied: •Receive enable bit = “1” •Reception is stopped. •Dummy data is present in the transmit buffer register	When all of the following are satisfied: •Receive enable bit = “1” •Reception is stopped.
DMA shortest transfer rate (At 1-bus cycle transfer)		12.5 Mbytes/sec	8 Mbytes/sec

Note: 512 bytes can be selected by software. For the M37721S1BFP, its internal RAM size is 512 bytes.

EOL announced

Appendix 11. Electrical characteristics

The electrical characteristics of the M37721S2BFP are described below.
For the latest data, inquire of addresses described last (☞**"CONTACT ADDRESSES FOR FURTHER INFORMATION")**.

Absolute maximum ratings

Symbol	Parameter		Conditions	Ratings	Unit
V_{CC}	Power source voltage			–0.3 to 7	V
AV_{CC}	Analog power source voltage			–0.3 to 7	V
V_I	Input voltage	$\overline{RESET}$, CNV_{SS}, BYTE		–0.3 to 12	V
V_I	Input voltage	A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7, $P4_3$–$P4_7$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_7$, $P8_0$–$P8_7$, $P9_0$–$P9_7$, $P10_0$–$P10_7$, $\overline{RDY}$, $\overline{HOLD}$, X_{IN}, V_{REF}		–0.3 to V_{CC}+0.3	V
V_O	Output voltage	A_0/MA_0–A_7/MA_7, A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7, $P4_3$–$P4_7$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_7$, $P8_0$–$P8_7$, $P9_0$–$P9_7$, $P10_0$–$P10_7$, ϕ_1, $\overline{RESET_{OUT}}$, X_{OUT}, $\overline{E}$, ST0, ST1, ALE, $\overline{BLE}$, $\overline{BHE}$, $R/\overline{W}$		–0.3 to V_{CC}+0.3	V
P_d	Power dissipation		Ta = 25 °C	300	mW
T_{opr}	Operating temperature			–20 to 85	°C
T_{stg}	Storage temperature			–40 to 150	°C

EOL announced

Recommended operating conditions (V_{CC} = 5 V ± 10 %, Ta = –20 to 85 °C, unless otherwise noted)

Symbol	Parameter		Limits			Unit
			Min.	Typ.	Max.	
Vcc	Power source voltage		4.5	5.0	5.5	V
AVcc	Analog power source voltage			Vcc		V
Vss	Power source voltage			0		V
AVss	Analog power source voltage			0		V
V_{IH}	High-level input voltage	$P4_3$–$P4_7$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_7$, $P8_0$–$P8_7$, $P9_0$–$P9_7$, $P10_0$–$P10_7$, $\overline{RDY}$, $\overline{HOLD}$, BYTE, CNVss, $\overline{RESET}$, X_{IN}, V_{REF}	0.8 Vcc		Vcc	V
V_{IH}	High-level input voltage	A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7	0.5 Vcc		Vcc	V
V_{IL}	Low-level input voltage	$P4_3$–$P4_7$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_7$, $P8_0$–$P8_7$, $P9_0$–$P9_7$, $P10_0$–$P10_7$, $\overline{RDY}$, $\overline{HOLD}$, BYTE, CNVSS, $\overline{RESET}$, X_{IN}, V_{REF}	0		0.2 Vcc	V
V_{IL}	Low-level input voltage	A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7	0		0.16 Vcc	V
$I_{OH\ (peak)}$	High-level peak output current	A_0/MA_0–A_7/MA_7, A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7, $P4_3$–$P4_7$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_7$, $P8_0$–$P8_7$, $P9_0$–$P9_7$, $P10_0$–$P10_7$, ϕ_1, $\overline{RESET_{OUT}}$, ST0, ST1, ALE, $\overline{BLE}$, $\overline{BHE}$, R/$\overline{W}$			–10	mA
$I_{OH\ (avg)}$	High-level average output current	A_0/MA_0–A_7/MA_7, A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7, $P4_3$–$P4_7$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_7$, $P8_0$–$P8_7$, $P9_0$–$P9_7$, $P10_0$–$P10_7$, ϕ_1, $\overline{RESET_{OUT}}$, ST0, ST1, ALE, $\overline{BLE}$, $\overline{BHE}$, R/$\overline{W}$			–5	mA
$I_{OL\ (peak)}$	Low-level peak output current	A_0/MA_0–A_7/MA_7, A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7, $P4_3$–$P4_7$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_7$, $P8_0$–$P8_7$, $P9_0$–$P9_7$, $P10_0$–$P10_7$, ϕ_1, $\overline{RESET_{OUT}}$, ST0, ST1, ALE, $\overline{BLE}$, $\overline{BHE}$, R/$\overline{W}$			10	mA
$I_{OL\ (avg)}$	Low-level average output current	A_0/MA_0–A_7/MA_7, A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7, $P4_3$–$P4_7$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_7$, $P8_0$–$P8_7$, $P9_0$–$P9_7$, $P10_0$–$P10_7$, ϕ_1, $\overline{RESET_{OUT}}$, ST0, ST1, ALE, $\overline{BLE}$, $\overline{BHE}$, R/$\overline{W}$			5	mA
f(X_{IN})	External clock input frequency				25	MHz

EOL announced

Notes 1: Average output current is the average value of a 100 ms interval.

2: The sum of $I_{OL(peak)}$ for P8, P9, A_0/MA_0–A_7/MA_7, A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7, ST0, ST1, ALE, $\overline{BLE}$, $\overline{BHE}$, and R/$\overline{W}$ must be 80 mA or less; the sum of $I_{OH(peak)}$ for P8, P9, A_0/MA_0–A_7/MA_7, A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7, ST0, ST1, ALE, $\overline{BLE}$, $\overline{BHE}$, and R/$\overline{W}$ must be 80 mA or less; the sum of $I_{OL(peak)}$ for P4, P5, P6, P7, P10, and ϕ_1 must be 80 mA or less; the sum of $I_{OH(peak)}$ for P4, P5, P6, P7, P10, and ϕ_1 must be 80 mA or less.

APPENDIX

Appendix 11. Electrical characteristics

Electrical characteristics (V_{CC} = 5 V, V_{SS} = 0 V, Ta = –20 to 85 °C, f(X_{IN}) = 25 MHz, unless otherwise noted)

Symbol	Parameter		Test conditions	Limits			Unit
				Min.	Typ.	Max.	
V_{OH}	High-level output voltage	A_0/MA_0–A_7/MA_7, A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7, $P4_3$–$P4_7$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_7$, $P8_0$–$P8_7$, $P9_0$–$P9_7$, $P10_0$–$P10_7$, ϕ_1, $\overline{RESET_{OUT}}$, ST0, ST1, ALE, $\overline{BLE}$, $\overline{BHE}$, R/$\overline{W}$	I_{OH} = –10 mA	3			V
V_{OH}	High-level output voltage	A_0/MA_0–A_7/MA_7, A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7, MA_8, MA_9, $\overline{RAS}$, $\overline{CAS}$, ϕ_1, ST0, ST1, $\overline{BLE}$, $\overline{BHE}$, R/$\overline{W}$	I_{OH} = –400 μA	4.7			V
V_{OH}	High-level output voltage	ALE	I_{OH} = –10 mA	3.1			V
			I_{OH} = –400 μA	4.8			
V_{OH}	High-level output voltage	$\overline{E}$	I_{OH} = –10 mA	3.4			V
			I_{OH} = –400 μA	4.8			
V_{OL}	Low-level output voltage	A_0/MA_0–A_7/MA_7, A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7, $P4_3$–$P4_7$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_7$, $P8_0$–$P8_7$, $P9_0$–$P9_7$, $P10_0$–$P10_7$, ϕ_1, $\overline{RESET_{OUT}}$, ST0, ST1, ALE, $\overline{BLE}$, $\overline{BHE}$, R/$\overline{W}$	I_{OL} = 10 mA			2	V
V_{OL}	Low-level output voltage	A_0/MA_0–A_7/MA_7, A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7, MA_8, MA_9, $\overline{RAS}$, $\overline{CAS}$, ϕ_1, ST0, ST1, $\overline{BLE}$, $\overline{BHE}$, R/$\overline{W}$	I_{OL} = 2 mA			0.45	V
V_{OL}	Low-level output voltage	ALE	I_{OL} = 10 mA			1.9	V
			I_{OL} = 2 mA			0.43	
V_{OL}	Low-level output voltage	$\overline{E}$	I_{OL} = 10 mA			1.6	V
			I_{OL} = 2 mA			0.4	
V_{T+}–V_{T-}	Hysteresis	$\overline{HOLD}$, $\overline{RDY}$, $TA2_{IN}$–$TA4_{IN}$, $TB0_{IN}$, $TB1_{IN}$, $\overline{INT_0}$–$\overline{INT_2}$, $\overline{AD_{TRG}}$, $\overline{CTS_0}$, $\overline{CTS_1}$, CLK_0, CLK_1, $\overline{DMAREQ0}$–$\overline{DMAREQ3}$, $\overline{TC}$		0.4		1	V
V_{T+}–V_{T-}	Hysteresis	$\overline{RESET}$		0.2		0.5	V
V_{T+}–V_{T-}	Hysteresis	X_{IN}		0.1		0.3	V
I_{IH}	High-level input current	A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7, $P4_3$–$P4_7$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_7$, $P8_0$–$P8_7$, $P9_0$–$P9_7$, $P10_0$–$P10_7$, $\overline{RDY}$, $\overline{HOLD}$, BYTE, CNVss, X_{IN}, $\overline{RESET}$	V_I = 5 V			5	μA
I_{IL}	Low-level input current	A_8/D_8–A_{15}/D_{15}, A_{16}/D_0–A_{23}/D_7, $P4_3$–$P4_7$, $P5_0$–$P5_7$, $P6_0$–$P6_7$, $P7_0$–$P7_7$, $P8_0$–$P8_7$, $P9_0$–$P9_7$, $P10_0$–$P10_7$, $\overline{RDY}$, $\overline{HOLD}$, BYTE, CNVss, X_{IN}, $\overline{RESET}$	V_I = 0 V			–5	μA
V_{RAM}	RAM hold voltage		When clock is stopped.	2			V
Icc	Power source current		f(X_{IN}) = 25 MHz (Square waveform)		27	54	mA
			Ta = 25 °C (when clock is stopped)			1	μA
			Ta = 85 °C (when clock is stopped)			20	μA

A-D CONVERTER CHARACTERISTICS (V_{CC} = 5 V, V_{SS} = 0 V, Ta = –20 to 85 °C, f(X_{IN}) = 25 MHz, unless otherwise noted)

Symbol	Parameter	Test conditions	Limits			Unit
			Min.	Typ.	Max.	
—	Resolution	V_{REF} = V_{CC}			8	Bits
—	Absolute accuracy	V_{REF} = V_{CC}			±3	LSB
R_{LADDER}	Ladder resistance	V_{REF} = V_{CC}	2		10	kΩ
t_{CONV}	Conversion time		9.12			μs
V_{REF}	Reference voltage		2		V_{CC}	V
V_{IA}	Analog input voltage		0		V_{REF}	V

Not announced

Internal peripheral devices' timing requirements (V_{CC} = 5 V ± 10 %, V_{SS} = 0 V, Ta = –20 to 85 °C, $f(X_{IN})$ = 25 MHz, unless otherwise noted)
Note: The limits depend on $f(X_{IN})$. Table 3 lists calculation formulas for the limits.

Timer A input (Count input in event counter mode)

Symbol	Parameter	Limits		Unit
		Min.	Max.	
$t_{c(TA)}$	TAj_{IN} input cycle time	80		ns
$t_{w(TAH)}$	TAj_{IN} input high-level pulse width	40		ns
$t_{w(TAL)}$	TAj_{IN} input low-level pulse width	40		ns

Timer A input (Gating input in timer mode)

Symbol	Parameter	Limits		Unit
		Min.	Max.	
$t_{c(TA)}$	TAj_{IN} input cycle time **(Note)**	320		ns
$t_{w(TAH)}$	TAj_{IN} input high-level pulse width **(Note)**	160		ns
$t_{w(TAL)}$	TAj_{IN} input low-level pulse width **(Note)**	160		ns

Timer A input (External trigger input in one-shot pulse mode)

Symbol	Parameter	Limits		Unit
		Min.	Max.	
$t_{c(TA)}$	TAj_{IN} input cycle time **(Note)**	160		ns
$t_{w(TAH)}$	TAj_{IN} input high-level pulse width	80		ns
$t_{w(TAL)}$	TAj_{IN} input low-level pulse width	80		ns

Timer A input (External trigger input in pulse width modulation mode)

Symbol	Parameter	Limits		Unit
		Min.	Max.	
$t_{w(TAH)}$	TAj_{IN} input high-level pulse width	80		ns
$t_{w(TAL)}$	TAj_{IN} input low-level pulse width	80		ns

Timer A input (Up-down input in event counter mode)

Symbol	Parameter	Limits		Unit
		Min.	Max.	
$t_{c(UP)}$	TAj_{OUT} input cycle time	2000		ns
$t_{w(UPH)}$	TAj_{OUT} input high-level pulse width	1000		ns
$t_{w(UPL)}$	TAj_{OUT} input low-level pulse width	1000		ns
$t_{su(UP\text{-}TIN)}$	TAj_{OUT} input setup time	400		ns
$t_{h(TIN\text{-}UP)}$	TAj_{OUT} input hold time	400		ns

Timer A input (Two-phase pulse input in event counter mode)

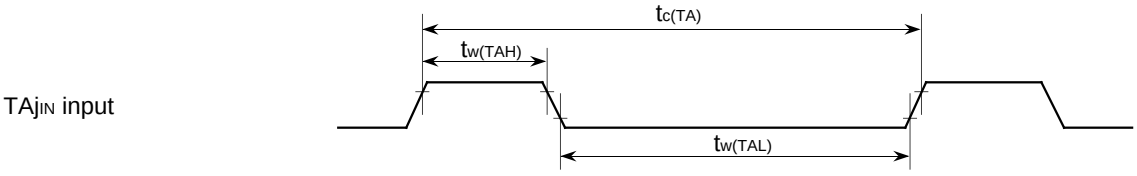
Symbol	Parameter	Limits		Unit
		Min.	Max.	
$t_{c(TA)}$	TAj_{IN} input cycle time	800		ns
$t_{su(TAjIN–TAjOUT)}$	TAj_{IN} input setup time	200		ns
$t_{su(TAjOUT–TAjIN)}$	TAj_{OUT} input setup time	200		ns

EOL announced

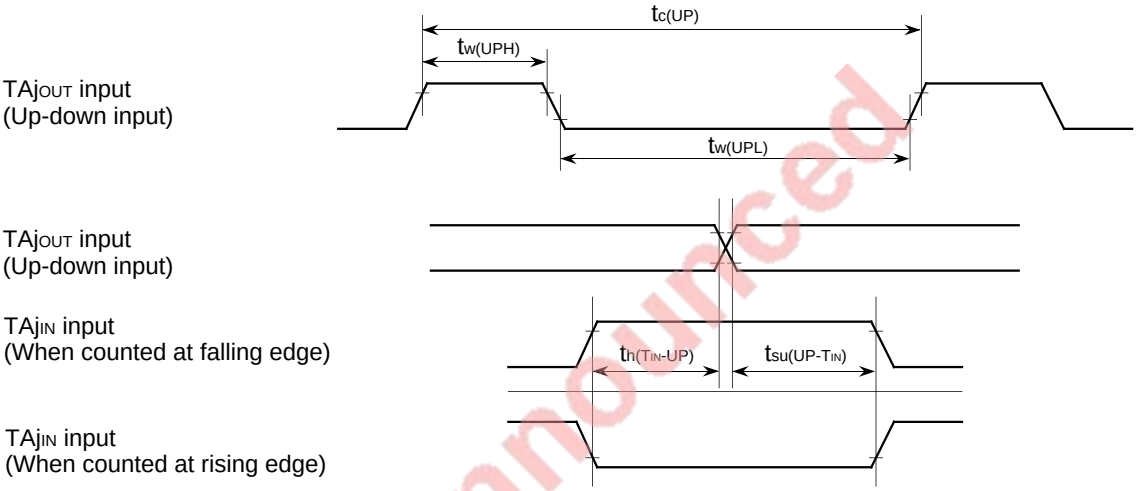
Appendix 11. Electrical characteristics

Internal peripheral devices

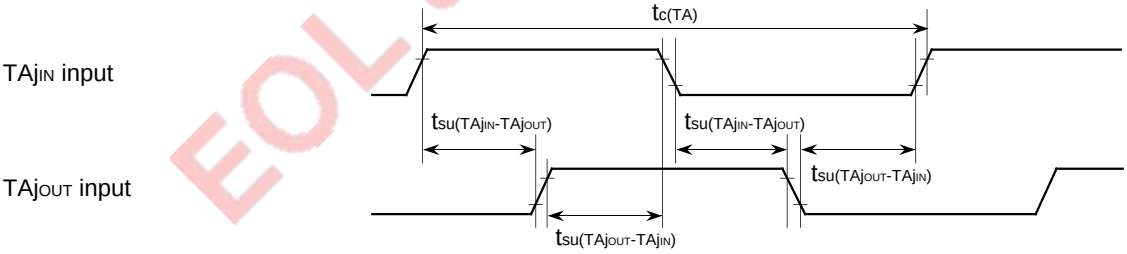
•Count input in event counter mode
•Gating input in timer mode
•External trigger input in one-shot pulse mode
•External trigger input in pulse width modulation mode



•Up-down input and count input in event counter mode



•Two-phase pulse input in event counter mode



Test conditions
•Vcc = 5 V ± 10 %
•Input timing voltage : V_{IL} = 1.0 V, V_{IH} = 4.0 V

EOL announced

Timer B input (Count input in event counter mode)

Symbol	Parameter	Limits		Unit
		Min.	Max.	
$t_{c(TB)}$	TBj_{IN} input cycle time (one edge count)	80		ns
$t_{w(TBH)}$	TBj_{IN} input high-level pulse width (one edge count)	40		ns
$t_{w(TBL)}$	TBj_{IN} input low-level pulse width (one edge count)	40		ns
$t_{c(TB)}$	TBj_{IN} input cycle time (both edges count)	160		ns
$t_{w(TBH)}$	TBj_{IN} input high-level pulse width (both edges count)	80		ns
$t_{w(TBL)}$	TBj_{IN} input low-level pulse width (both edges count)	80		ns

Timer B input (Pulse period measurement mode)

Symbol	Parameter		Limits		Unit
			Min.	Max.	
$t_{c(TB)}$	TBj_{IN} input cycle time	**(Note)**	320		ns
$t_{w(TBH)}$	TBj_{IN} input high-level pulse width	**(Note)**	160		ns
$t_{w(TBL)}$	TBj_{IN} input low-level pulse width	**(Note)**	160		ns

Timer B input (Pulse width measurement mode)

Symbol	Parameter		Limits		Unit
			Min.	Max.	
$t_{c(TB)}$	TBj_{IN} input cycle time	**(Note)**	320		ns
$t_{w(TBH)}$	TBj_{IN} input high-level pulse width	**(Note)**	160		ns
$t_{w(TBL)}$	TBj_{IN} input low-level pulse width	**(Note)**	160		ns

A-D trigger input

Symbol	Parameter	Limits		Unit
		Min.	Max.	
$t_{c(AD)}$	$\overline{AD_{TRG}}$ input cycle time (trigger enabled minimum)	1000		ns
$t_{w(ADL)}$	$\overline{AD_{TRG}}$ input low-level pulse width	125		ns

Serial I/O

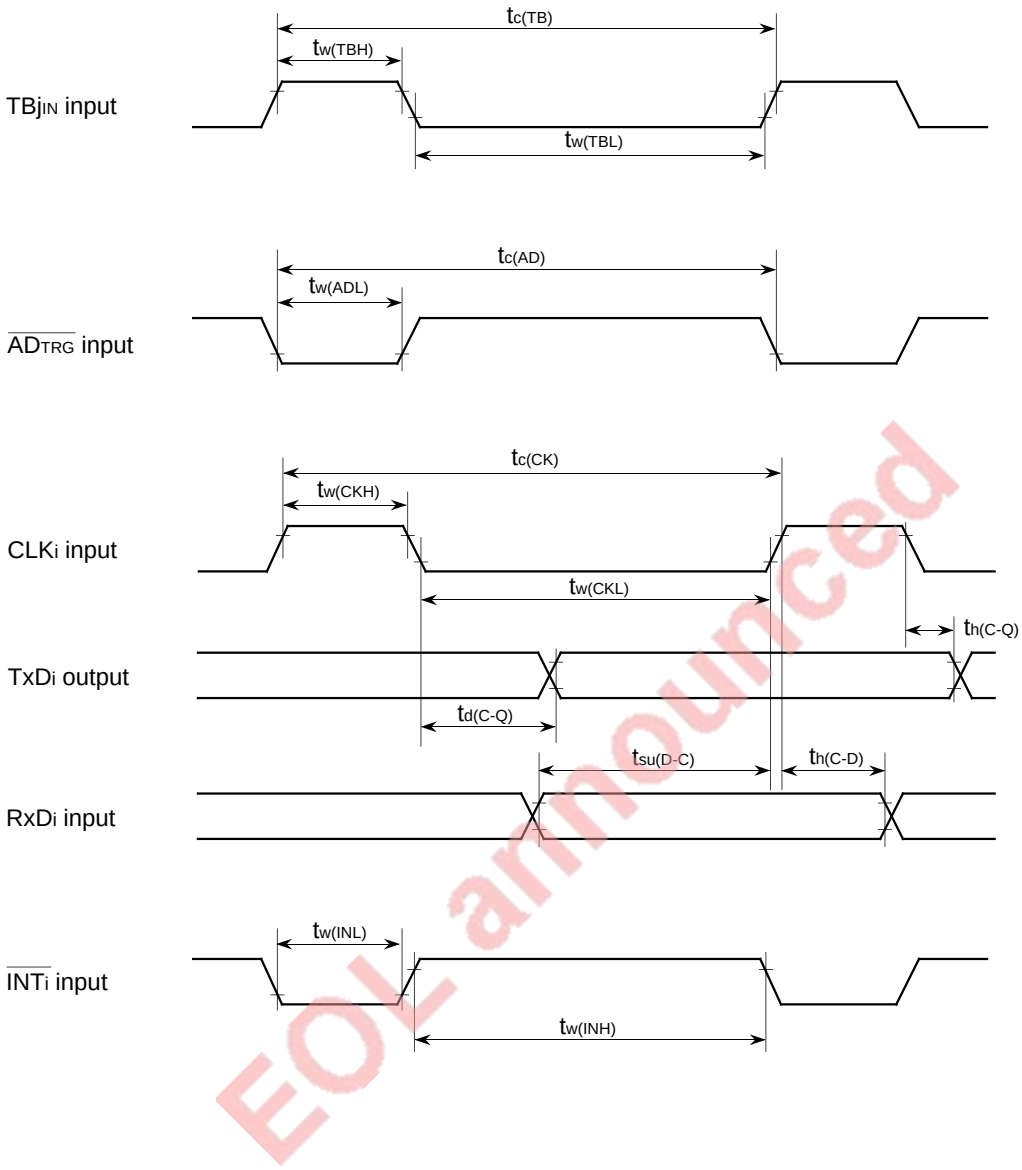
Symbol	Parameter	Limits		Unit
		Min.	Max.	
$t_{c(CK)}$	CLKi input cycle time	200		ns
$t_{w(CKH)}$	CLKi input high-level pulse width	100		ns
$t_{w(CKL)}$	CLKi input low-level pulse width	100		ns
$t_{d(C-Q)}$	TxDi output delay time		80	ns
$t_{h(C-Q)}$	TxDi hold time	0		ns
$t_{su(D-C)}$	RxDi input setup time	20		ns
$t_{h(C-D)}$	RxDi input hold time	90		ns

External interrupt $\overline{INTi}$ input

Symbol	Parameter	Limits		Unit
		Min.	Max.	
$t_{w(INH)}$	$\overline{INTi}$ input high-level pulse width	250		ns
$t_{w(INL)}$	$\overline{INTi}$ input low-level pulse width	250		ns

EOL announced

Internal peripheral devices



EOL announced

Test conditions
- Vcc = 5 V ± 10 %
- Input timing voltage : V_{IL} = 1.0 V, V_{IH} = 4.0 V
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V

Ready and Hold

Timing requirements (Vcc = 5 V ± 10 %, Vss = 0 V, Ta = –20 to 85 °C, f(X_{IN}) = 25 MHz, unless otherwise noted)

Symbol	Parameter	Limits		Unit
		Min.	Max.	
$t_{su(RDY-\phi1)}$	$\overline{RDY}$ input setup time	55		ns
$t_{su(HOLD-\phi1)}$	$\overline{HOLD}$ input setup time	55		ns
$t_{h(\phi1-RDY)}$	$\overline{RDY}$ input hold time	0		ns
$t_{h(\phi1-HOLD)}$	$\overline{HOLD}$ input hold time	0		ns

Switching characteristics (Vcc = 5 V ± 10 %, Vss = 0 V, Ta = –20 to 85 °C, f(X_{IN}) = 25 MHz, unless otherwise noted)

Symbol	Parameter	Limits		Unit
		Min.	Max.	
$t_{d(\phi1-STi)}$	ST0, ST1 output delay time		40	ns

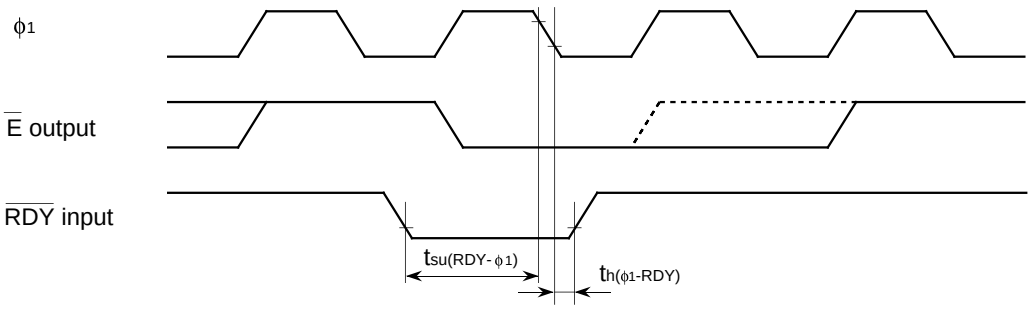
Note: Figure 13 shows the test circuit.

EOL announced

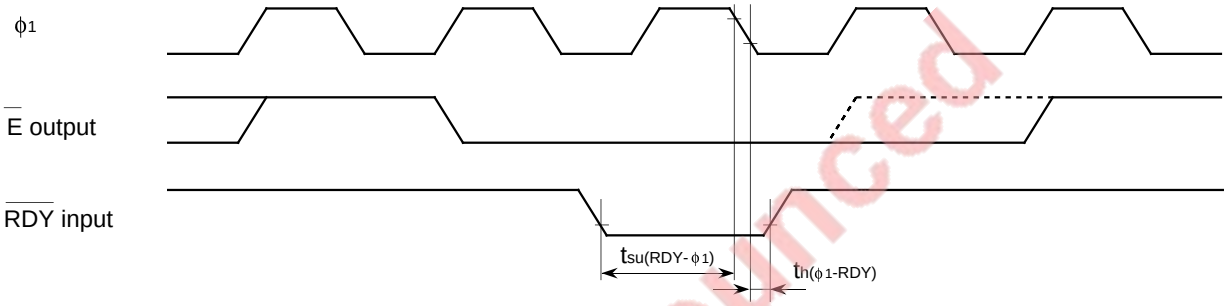
Appendix 11. Electrical characteristics

•Ready function

With no Wait

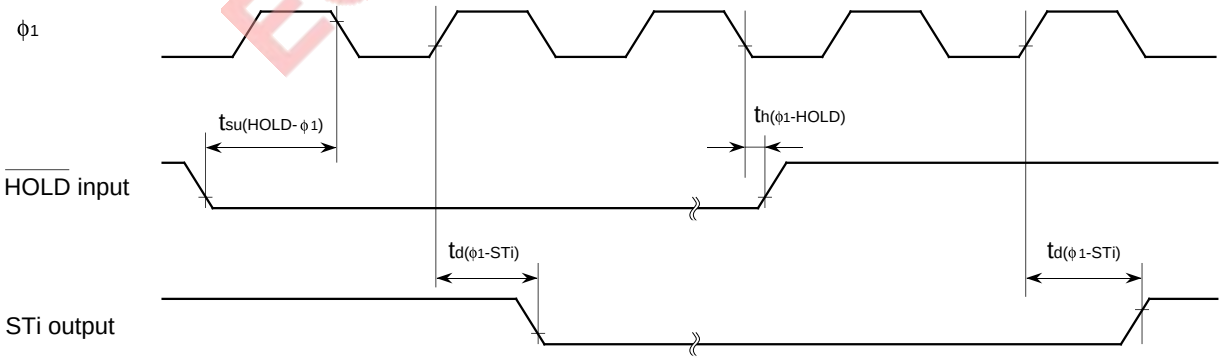


With Wait



Test conditions
- Vcc = 5 V ± 10 %
- Input timing voltage : V_{IL} = 1.0 V, V_{IH} = 4.0 V
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V

•Hold function



Test conditions
- Vcc = 5 V ± 10 %
- Input timing voltage : V_{IL} = 1.0 V, V_{IH} = 4.0 V
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V

EOL announced

Microprocessor mode : with no Wait

Note: The limits depend on f(X_{IN}). Table 4 lists calculation formulas for the limits.

Timing requirements (V_{CC} = 5 V ± 10 %, V_{SS} = 0 V, Ta = –20 to 85 °C, f(X_{IN}) = 25 MHz, unless otherwise noted)

Symbol	Parameter	Limits		Unit
		Min.	Max.	
t_c	External clock input cycle time	40		ns
$t_{w(H)}$	External clock input high-level pulse width	15		ns
$t_{w(L)}$	External clock input low-level pulse width	15		ns
t_r	External clock input rising time		8	ns
t_f	External clock input falling time		8	ns
$t_{su(PiD–E)}$	Port Pi input setup time (i = 4–10)	60		ns
$t_{h(E–PiD)}$	Port Pi input hold time (i = 4–10)	0		ns

Switching characteristics (V_{CC} = 5 V ± 10 %, V_{SS} = 0 V, Ta = –20 to 85 °C, f(X_{IN}) = 25 MHz, unless otherwise noted)

Symbol	Parameter	Limits		Unit
		Min.	Max.	
$t_{d(E-PiQ)}$	Port Pi data output delay time (i = 4–10)		80	ns
$t_{d(AL–E)}$	Address low-order output delay time **(Note)**	15		ns
$t_{d(E–DHQ)}$	Data high-order output delay time (BYTE = “L”)		35	ns
$t_{pxz(E–DHZ)}$	Data high-order floating start delay time (BYTE = “L”)		0	ns
$t_{d(AM–E)}$	Address middle-order output delay time **(Note)**	15		ns
$t_{d(AM–ALE)}$	Address middle-order output delay time **(Note)**	5		ns
$t_{d(E–DLQ)}$	Data low-order output delay time		35	ns
$t_{pxz(E–DLZ)}$	Data low-order floating start delay time		0	ns
$t_{d(AH–E)}$	Address high-order output delay time **(Note)**	15		ns
$t_{d(AH–ALE)}$	Address high-order output delay time **(Note)**	5		ns
$t_{d(ALE–E)}$	ALE output delay time	4		ns
$t_{w(ALE)}$	ALE pulse width **(Note)**	22		ns
$t_{d(BHE–E)}$	$\overline{BHE}$ output delay time **(Note)**	20		ns
$t_{d(BLE–E)}$	$\overline{BLE}$ output delay time **(Note)**	20		ns
$t_{d(R/W–E)}$	R/$\overline{W}$ output delay time **(Note)**	20		ns
$t_{d(E–\phi1)}$	ϕ_1 output delay time	0	18	ns
$t_{h(E–AL)}$	Address low-order hold time **(Note)**	18		ns
$t_{h(ALE–AM)}$	Address middle-order hold time (BYTE = “L”)	9		ns
$t_{h(E–DHQ)}$	Data high-order hold time (BYTE = “L”) **(Note)**	18		ns
$t_{pzx(E–DHZ)}$	Data high-order floating release delay time (BYTE = “L”) **(Note)**	20		ns
$t_{h(E–AM)}$	Address middle-order hold time (BYTE = “H”) **(Note)**	18		ns
$t_{h(ALE–AH)}$	Address high-order hold time	9		ns
$t_{h(E–DLQ)}$	Data low-order hold time **(Note)**	18		ns
$t_{pzx(E–DLZ)}$	Data low-order floating release delay time **(Note)**	20		ns
$t_{h(E–BHE)}$	$\overline{BHE}$ hold time **(Note)**	18		ns
$t_{h(E–BLE)}$	$\overline{BLE}$ hold time **(Note)**	18		ns
$t_{h(E–R/W)}$	R/$\overline{W}$ hold time **(Note)**	18		ns
$t_{w(EL)}$	$\overline{E}$ pulse width **(Note)**	55		ns
$t_{su(A–DL)}$	Data low-order setup time after address stabilization **(Note)**		50	ns
$t_{su(ALE–DL)}$	Data low-order setup time after rising of ALE **(Note)**		55	ns
$t_{su(A–DH)}$	Data high-order setup time after address stabilization **(Note)**		50	ns
$t_{su(ALE–DH)}$	Data high-order setup time after rising of ALE **(Note)**		55	ns

Note: Figure 13 shows the test circuit.

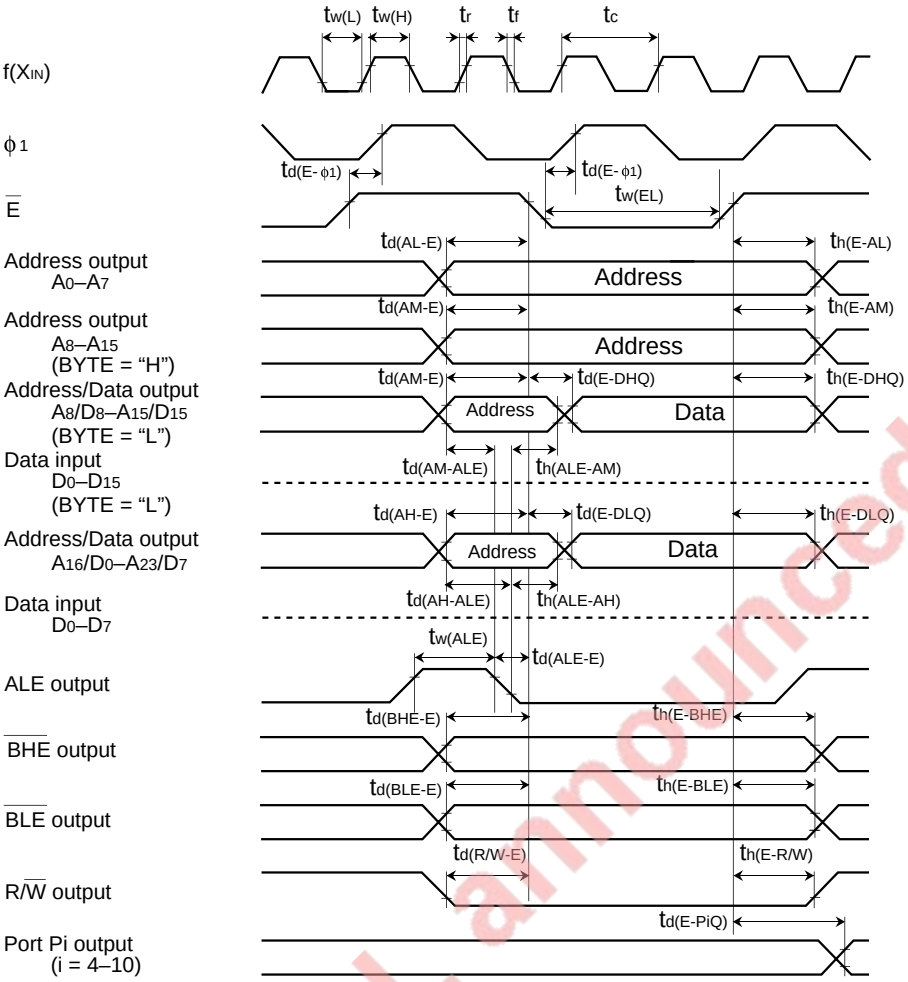
EOL announced

APPENDIX

Appendix 11. Electrical characteristics

Microprocessor mode : with no Wait

<Write>



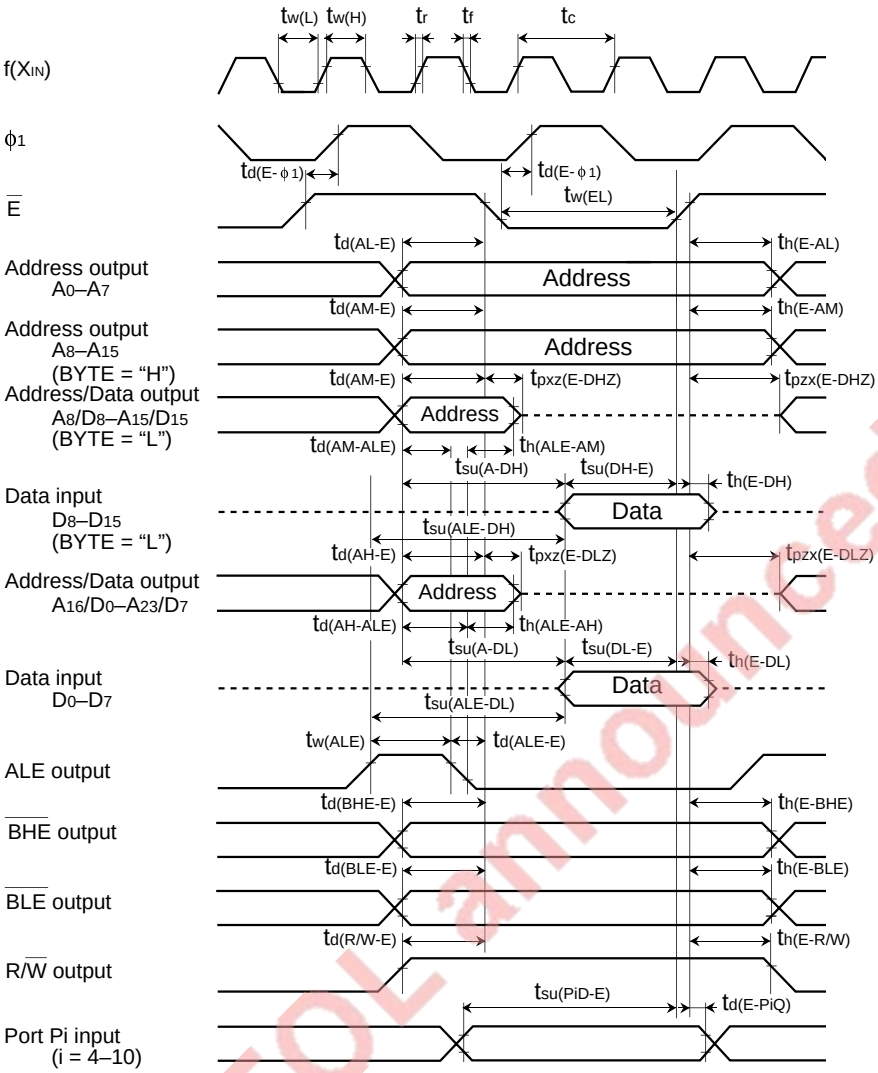
Test conditions (port Pi)
- Vcc = 5 V ± 10 %
- Input timing voltage : V_{IL} = 1.0 V, V_{IH} = 4.0 V
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V

Test conditions (except port Pi)
- Vcc = 5 V ± 10 %
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V
- Data input : V_{IL} = 0.8 V, V_{IH} = 2.5 V

EOL announced

Microprocessor mode : with no Wait

<Read>



EOL announced

Test conditions (port Pi)
- Vcc = 5 V ± 10 %
- Input timing voltage : V_{IL} = 1.0 V, V_{IH} = 4.0 V
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V

Test conditions (except port Pi)
- Vcc = 5 V ± 10 %
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V
- Data input : V_{IL} = 0.8 V, V_{IH} = 2.5 V

APPENDIX

Appendix 11. Electrical characteristics

Microprocessor mode : with Wait

Note: The limits depend on f(X_{IN}). Table 4 lists calculation formulas for the limits.

Timing requirements (V_{CC} = 5 V ± 10 %, V_{SS} = 0 V, Ta = –20 to 85 °C, f(X_{IN}) = 25 MHz, unless otherwise noted)

Symbol	Parameter	Limits		Unit
		Min.	Max.	
t_c	External clock input cycle time	40		ns
$t_{w(H)}$	External clock input high-level pulse width	15		ns
$t_{w(L)}$	External clock input low-level pulse width	15		ns
t_r	External clock input rising time		8	ns
t_f	External clock input falling time		8	ns
$t_{su(PiD–E)}$	Port Pi input setup time (i = 4–10)	60		ns
$t_{h(E–PiD)}$	Port Pi input hold time (i = 4–10)	0		ns

Switching characteristics (V_{CC} = 5 V ± 10 %, V_{SS} = 0 V, Ta = –20 to 85 °C, f(X_{IN}) = 25 MHz, unless otherwise noted)

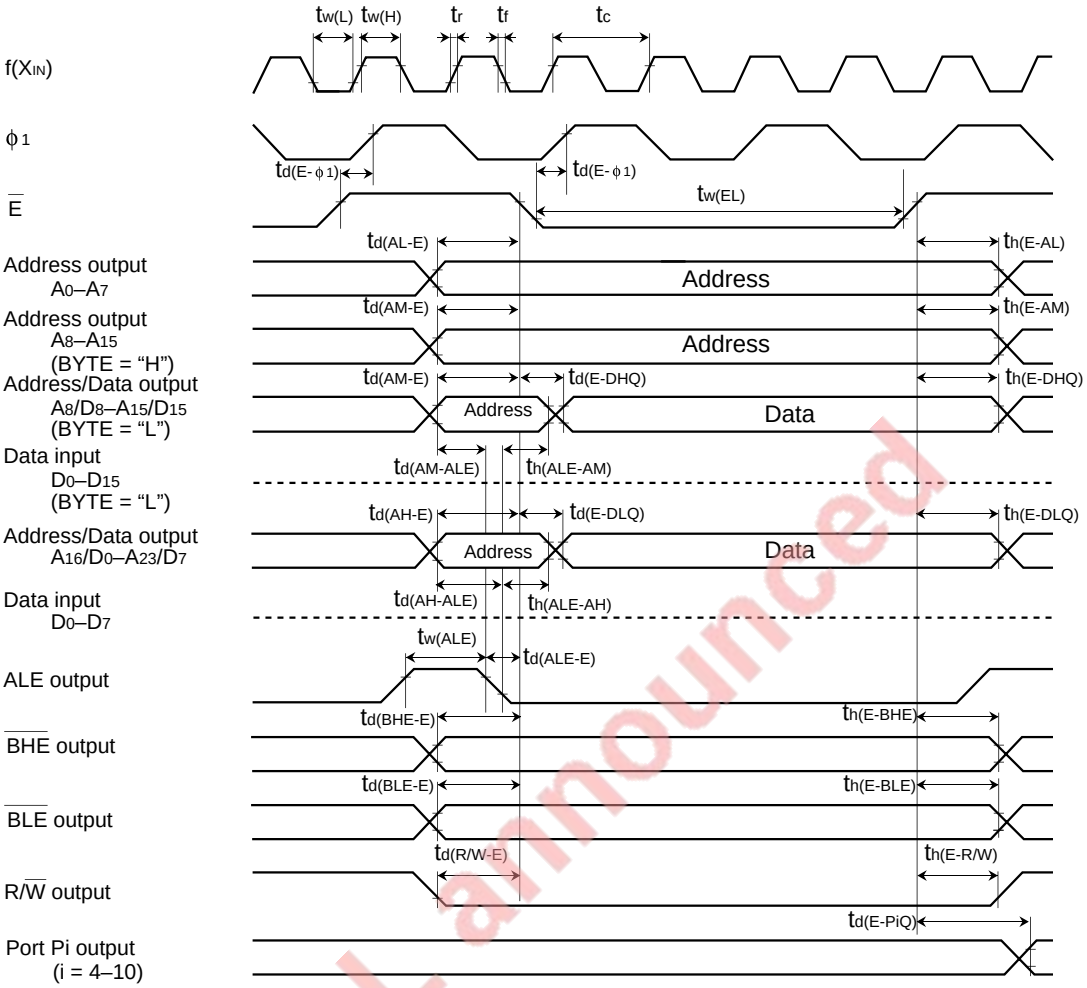
Symbol	Parameter		Limits		Unit
			Min.	Max.	
$t_{d(E-PiQ)}$	Port Pi data output delay time			80	ns
$t_{d(AL–E)}$	Address low-order output delay time	**(Note)**	15		ns
$t_{d(E–DHQ)}$	Data high-order output delay time (BYTE = “L”)			35	ns
$t_{pxz(E–DHZ)}$	Data high-order floating start delay time (BYTE = “L”)			0	ns
$t_{d(AM–E)}$	Address middle-order output delay time	**(Note)**	15		ns
$t_{d(AM–ALE)}$	Address middle-order output delay time	**(Note)**	5		ns
$t_{d(E–DLQ)}$	Data low-order output delay time			35	ns
$t_{pxz(E–DLZ)}$	Data low-order floating start delay time			0	ns
$t_{d(AH–E)}$	Address high-order output delay time	**(Note)**	15		ns
$t_{d(AH–ALE)}$	Address high-order output delay time	**(Note)**	5		ns
$t_{d(ALE–E)}$	ALE output delay time		4		ns
$t_{w(ALE)}$	ALE pulse width	**(Note)**	22		ns
$t_{d(BHE–E)}$	$\overline{BHE}$ output delay time	**(Note)**	20		ns
$t_{d(BLE–E)}$	$\overline{BLE}$ output delay time	**(Note)**	20		ns
$t_{d(R/W–E)}$	R/$\overline{W}$ output delay time	**(Note)**	20		ns
$t_{d(E–\phi1)}$	ϕ_1 output delay time		0	18	ns
$t_{h(E–AL)}$	Address low-order hold time	**(Note)**	18		ns
$t_{h(ALE–AM)}$	Address middle-order hold time (BYTE = “L”)		9		ns
$t_{h(E–DHQ)}$	Data high-order hold time (BYTE = “L”)	**(Note)**	18		ns
$t_{pzx(E–DHZ)}$	Data high-order floating release delay time (BYTE = “L”)	**(Note)**	20		ns
$t_{h(E–AM)}$	Address middle-order hold time (BYTE = “H”)	**(Note)**	18		ns
$t_{h(ALE–AH)}$	Address high-order hold time		9		ns
$t_{h(E–DLQ)}$	Data low-order hold time	**(Note)**	18		ns
$t_{pzx(E–DLZ)}$	Data low-order floating release delay time	**(Note)**	20		ns
$t_{h(E–BHE)}$	$\overline{BHE}$ hold time	**(Note)**	18		ns
$t_{h(E–BLE)}$	$\overline{BLE}$ hold time	**(Note)**	18		ns
$t_{h(E–R/W)}$	R/$\overline{W}$ hold time	**(Note)**	18		ns
$t_{w(EL)}$	$\overline{E}$ pulse width	**(Note)**	135		ns
$t_{su(A–DL)}$	Data low-order setup time after address stabilization	**(Note)**		130	ns
$t_{su(ALE–DL)}$	Data low-order setup time after rising of ALE	**(Note)**		135	ns
$t_{su(A–DH)}$	Data high-order setup time after address stabilization	**(Note)**		130	ns
$t_{su(ALE–DH)}$	Data high-order setup time after rising of ALE	**(Note)**		135	ns

Note: Figure 13 shows the test circuit.

EOL announced

Microprocessor mode : with Wait

<Write>



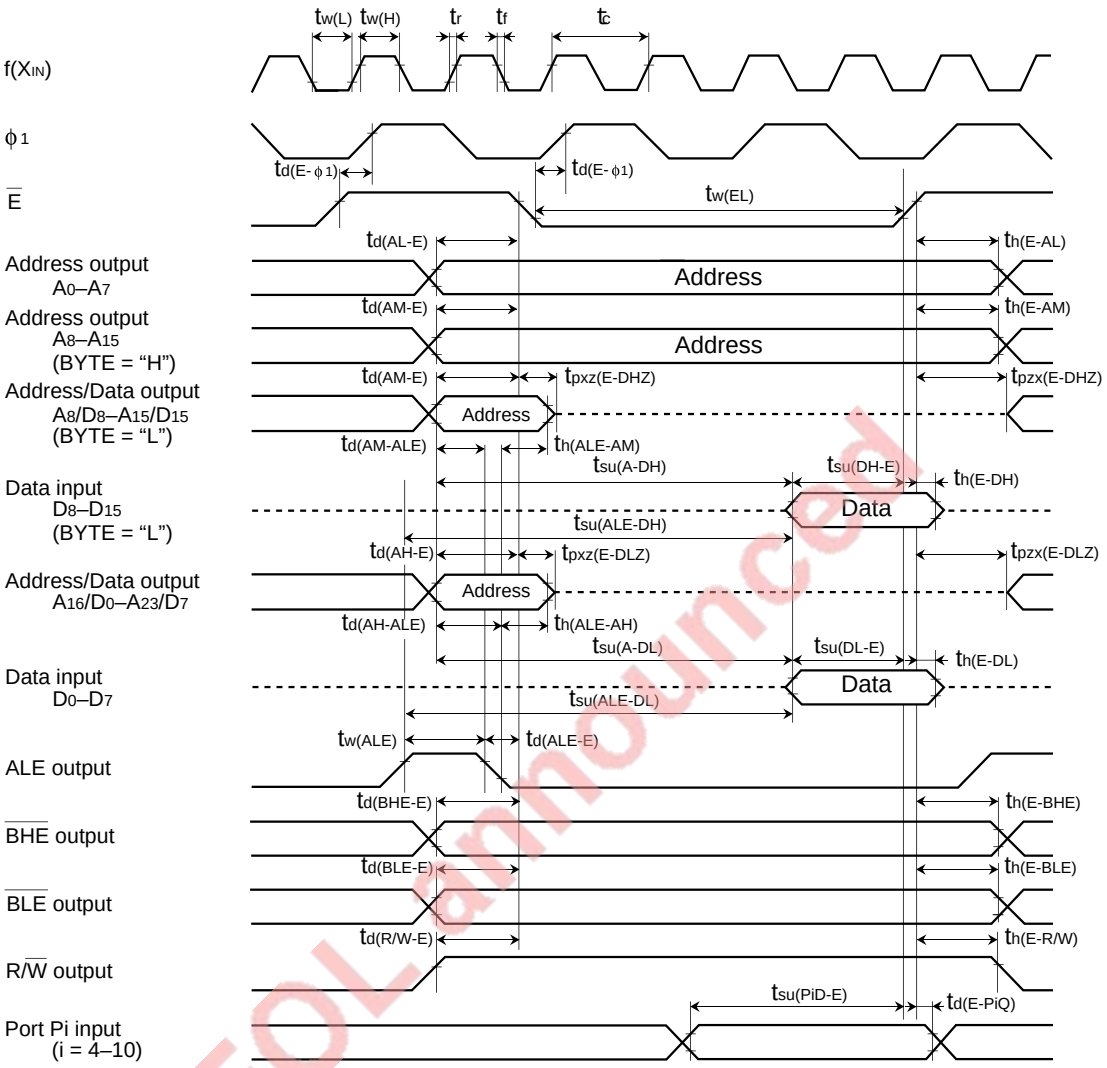
Test conditions (port Pi)
- Vcc = 5 V ± 10 %
- Input timing voltage : V_{IL} = 1.0 V, V_{IH} = 4.0 V
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V

Test conditions (except port Pi)
- Vcc = 5 V ± 10 %
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V
- Data input : V_{IL} = 0.8 V, V_{IH} = 2.5 V

EOL announced

Microprocessor mode : with Wait

<Read>



EOL announced

Test conditions (port Pi)
- Vcc = 5 V ± 10 %
- Input timing voltage : V_{IL} = 1.0 V, V_{IH} = 4.0 V
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V

Test conditions (except port Pi)
- Vcc = 5 V ± 10 %
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V
- Data input : V_{IL} = 0.8 V, V_{IH} = 2.5 V

DRAM control switching characteristics (V_{CC} = 5 V ± 10 %, V_{SS} = 0 V, Ta = –20 to 85 °C, f(X_{IN}) = 25 MHz, unless otherwise noted)

Note: The limits depend on f(X_{IN}). Table 5 lists calculation formulas for the limits.

Read

Symbol	Parameter		Limits Min.	Limits Max.	Unit
$t_{w(RASL)}$	$\overline{RAS}$ low–level pulse width	**(Note)**	120		ns
$t_{w(CASL)}$	$\overline{CAS}$ low–level pulse width	**(Note)**	92.5		ns
$t_{w(RASH)}$	$\overline{CAS}$ high–level pulse width	**(Note)**	60		ns
$t_{d(RAS–CAS)}$	$\overline{RAS}$–$\overline{CAS}$ delay time	**(Note)**	28		ns
$t_{d(RA–RAS)}$	Row address delay time before $\overline{RAS}$	**(Note)**	5		ns
$t_{h(RAS–RA)}$	Row address hold time after $\overline{RAS}$	**(Note)**	18		ns
$t_{d(CA–CAS)}$	Column address delay time before $\overline{CAS}$		5		ns
$t_{h(CAS–CA)}$	Column address hold time after $\overline{CAS}$	**(Note)**	100		ns
$t_{d(R/W–RAS)}$	R/$\overline{W}$ delay time before $\overline{RAS}$	**(Note)**	18		ns
$t_{h(CAS–R/W)}$	R/$\overline{W}$ hold time after $\overline{CAS}$	**(Note)**	18		ns
$t_{d(E–CA)}$	Column address delay time after $\overline{E}$'s low level	**(Note)**		65	ns
$t_{d(E–RASL)}$	$\overline{RAS}$ delay time after $\overline{E}$'s low level			30	ns
$t_{d(E–CASL)}$	$\overline{CAS}$ delay time after $\overline{E}$'s low level	**(Note)**		77.5	ns
$t_{d(E–RASH)}$	$\overline{RAS}$ delay time after $\overline{E}$'s high level		0	20	ns
$t_{d(E–CASH)}$	$\overline{CAS}$ delay time after $\overline{E}$'s high level		0	20	ns

Note: Figure 13 shows the test circuit.

Write

Symbol	Parameter		Limits Min.	Limits Max.	Unit
$t_{w(RASL)}$	$\overline{RAS}$ low–level pulse width	**(Note)**	120		ns
$t_{w(CASL)}$	$\overline{CAS}$ low–level pulse width	**(Note)**	55		ns
$t_{w(RASH)}$	$\overline{CAS}$ high–level pulse width	**(Note)**	60		ns
$t_{d(RAS–CAS)}$	$\overline{RAS}$–$\overline{CAS}$ delay time	**(Note)**	60		ns
$t_{d(RA–RAS)}$	Row address delay time before $\overline{RAS}$	**(Note)**	5		ns
$t_{h(RAS–RA)}$	Row address hold time after $\overline{RAS}$	**(Note)**	18		ns
$t_{d(CA–CAS)}$	Column address delay time before $\overline{CAS}$	**(Note)**	10		ns
$t_{h(CAS–CA)}$	Column address hold time afrer $\overline{CAS}$	**(Note)**	60		ns
$t_{d(R/W–RAS)}$	R/$\overline{W}$ delay time before $\overline{RAS}$	**(Note)**	18		ns
$t_{h(CAS–R/W)}$	R/$\overline{W}$ hold time after $\overline{CAS}$	**(Note)**	18		ns
$t_{d(E–RASL)}$	$\overline{RAS}$ delay time after $\overline{E}$'s low level			30	ns
$t_{d(E–CASL)}$	$\overline{CAS}$ delay time after $\overline{E}$'s low level	**(Note)**	80	115	ns
$t_{d(E–RASH)}$	$\overline{RAS}$ delay time after $\overline{E}$'s high level		0	20	ns
$t_{d(E–CASH)}$	$\overline{CAS}$ delay time after $\overline{E}$'s high level		0	20	ns

Note: Figure 13 shows the test circuit.

Refresh state

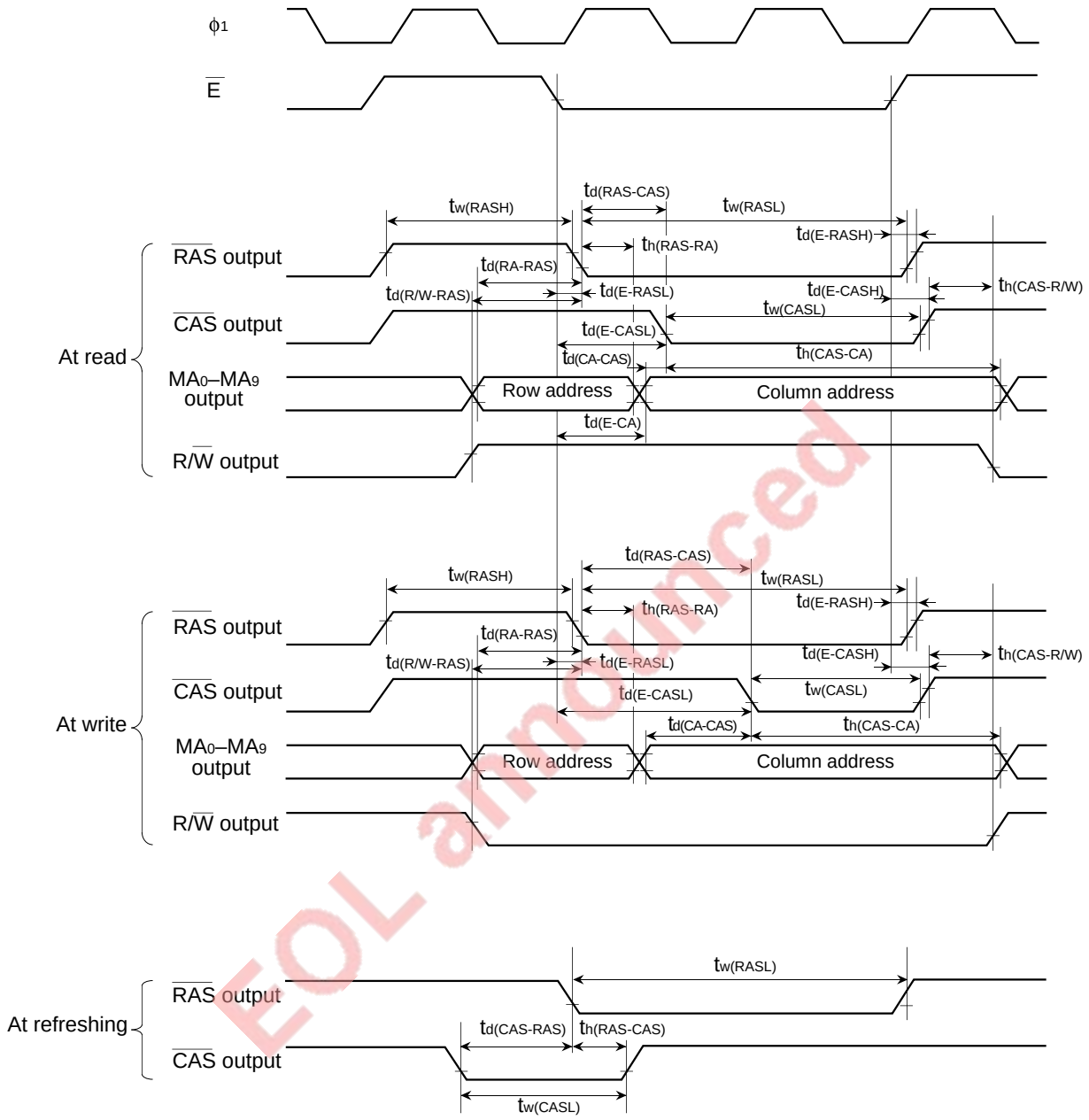
Symbol	Parameter		Limits Min.	Limits Max.	Unit
$t_{w(RASL)}$	$\overline{RAS}$ low–level pulse width	**(Note)**	120		ns
$t_{w(CASL)}$	$\overline{CAS}$ low–level pulse width	**(Note)**	55		ns
$t_{d(CAS–RAS)}$	$\overline{CAS}$–$\overline{RAS}$ delay time	**(Note)**	17.5		ns
$t_{h(RAS–CAS)}$	$\overline{CAS}$ hold time after $\overline{RAS}$	**(Note)**	17.5		ns

Note: Figure 13 shows the test circuit.

FOL announced

Appendix 11. Electrical characteristics

At DRAM control



EOL announced

Test conditions
- Vcc = 5 V ± 10 %
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V
- D_0–D_{15} input : V_{IL} = 0.8 V, V_{IH} = 2.5 V

DMAC switching characteristics (V_{CC} = 5 V ±10 %, V_{SS} = 0 V, Ta = –20 to 85 °C, f(X_{IN}) = 25 MHz, unless otherwise noted)

Note: The limits depend on f(X_{IN}). Table 6 lists calculation formulas for the limits.

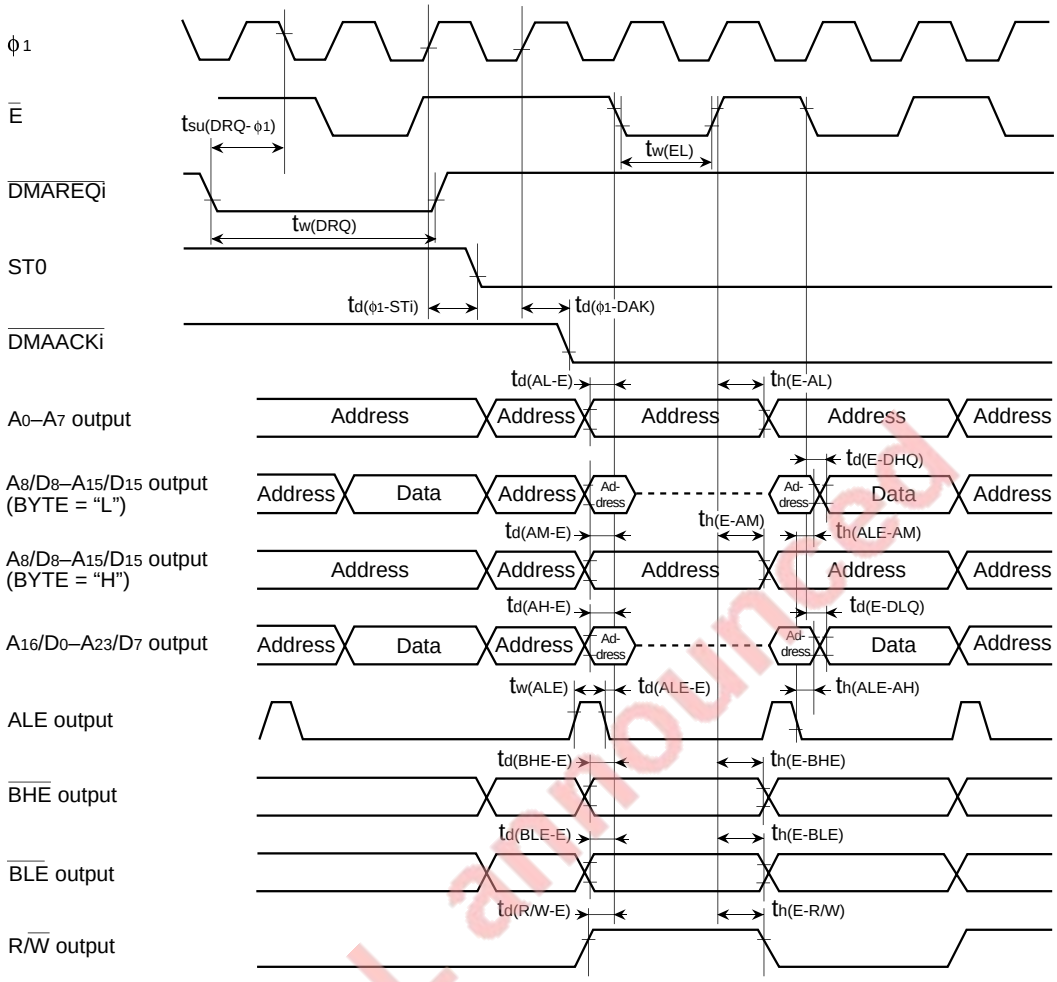
Symbol	Parameter	Limits		Unit
		Min.	Max.	
$t_{su(DRQ-\phi1)}$	$\overline{DMAREQ}$i input setup time	60		ns
$t_{w(DRQ)}$	$\overline{DMAREQ}$i input pulse width	80		ns
$t_{d(\phi1-STi)}$	ST0, ST1 output delay time		40	ns
$t_{d(\phi1-DAK)}$	$\overline{DMAACK}$i output delay time		60	ns
$t_{d(AL-E)}$	Address low-order output delay time **(Note)**	15		ns
$t_{d(E-DHQ)}$	Data high-order output delay time (BYTE = “L”)		35	ns
$t_{pxz(E-DHZ)}$	Data high-order floating start delay time (BYTE = “L”)		0	ns
$t_{d(AM-E)}$	Address middle-order output delay time **(Note)**			ns
$t_{d(E-DLQ)}$	Data low-order output delay time		35	ns
$t_{pxz(E-DLZ)}$	Data low-order floating start delay time		0	ns
$t_{d(AH-E)}$	Address high-order output delay time **(Note)**	15		ns
$t_{d(ALE-E)}$	ALE output delay time	4		ns
$t_{w(ALE)}$	ALE pulse width **(Note)**	22		ns
$t_{d(BHE-E)}$	$\overline{BHE}$ output delay time **(Note)**	20		ns
$t_{d(BLE-E)}$	$\overline{BLE}$ output delay time **(Note)**	20		ns
$t_{d(R/W-E)}$	R/$\overline{W}$ output delay time **(Note)**	20		ns
$t_{h(E-AL)}$	Address low-order hold time **(Note)**	18		ns
$t_{h(ALE-AM)}$	Address middle-order hold time (BYTE = “L”)	9		ns
$t_{h(E-DHQ)}$	Data high-order hold time (BYTE = “L”) **(Note)**	18		ns
$t_{pzx(E-DHZ)}$	Data high-order floating release delay time (BYTE = “L”) **(Note)**	20		ns
$t_{h(E-AM)}$	Address middle-order hold time (BYTE = “H”) **(Note)**	18		ns
$t_{h(ALE-AH)}$	Address high-order hold time	9		ns
$t_{h(E-DLQ)}$	Data low-order hold time **(Note)**	18		ns
$t_{pzx(E-DLZ)}$	Data low-order floating release delay time **(Note)**	20		ns
$t_{h(E-BHE)}$	$\overline{BHE}$ hold time **(Note)**	18		ns
$t_{h(E-BLE)}$	$\overline{BLE}$ hold time **(Note)**	18		ns
$t_{h(E-R/W)}$	R/$\overline{W}$ hold time **(Note)**	18		ns
$t_{w(EL)}$	$\overline{E}$ pulse width **(Note)**	55		ns
$t_{d(data)}$	Copy delay time	50		ns
$t_{d(\phi1-TC)}$	$\overline{TC}$ output delay time		50	ns
$t_{w(TC)}$	$\overline{TC}$ output pulse width **(Note)**	50		ns
$t_{su(TCIN)}$	$\overline{TC}$ input setup time	60		ns
$t_{w(TCIN)}$	$\overline{TC}$ input pulse width	80		ns

Note: Figures 13 and 14 show the test circuits.

EOL announced

At DMA transfer

•Burst transfer timing (External source $\overline{\text{DMAREQi}}$)

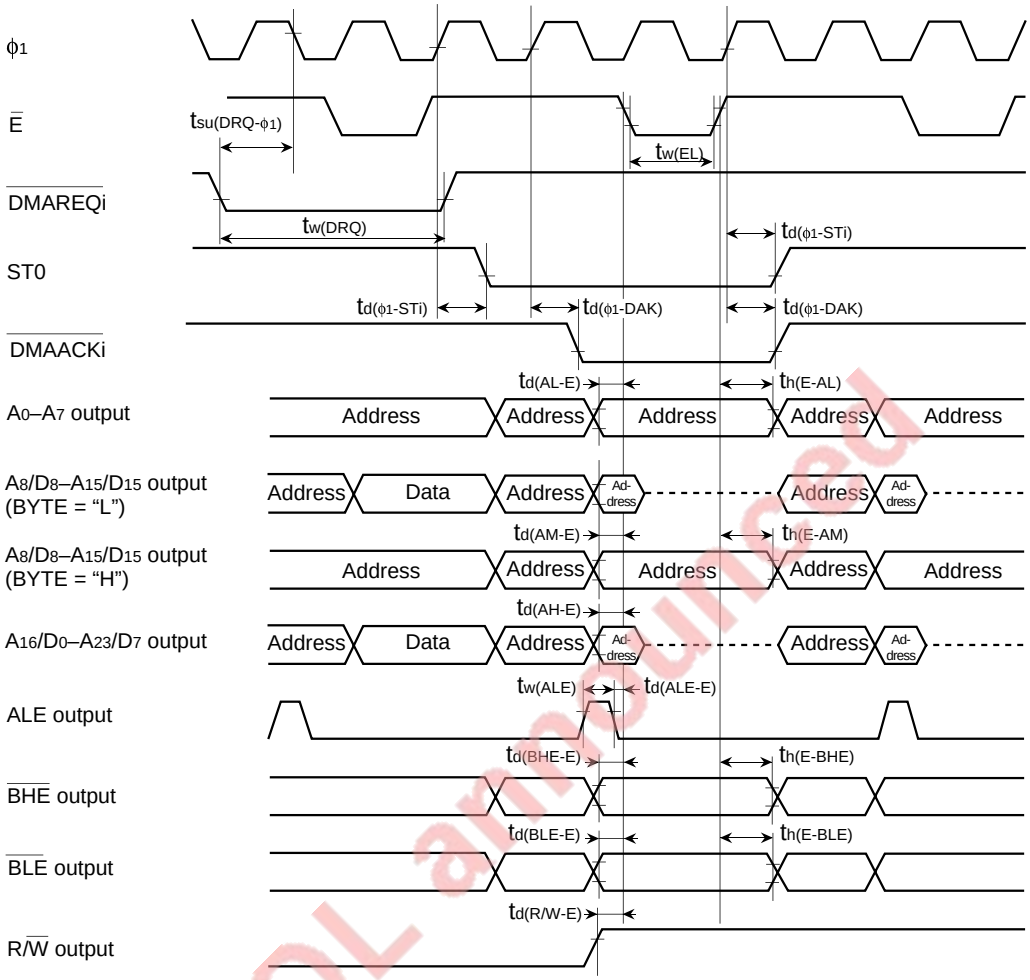


Test conditions
- Vcc = 5 V ± 10 %
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V
- D_0–D_{15} input : V_{IL} = 0.8 V, V_{IH} = 2.5 V
- $\overline{\text{DMAREQi}}$ input : V_{IL} = 0.8 V, V_{IH} = 2.5 V

FOL announced

At DMA transfer

•Cycle-steal transfer timing (External source $\overline{\text{DMAREQi}}$)



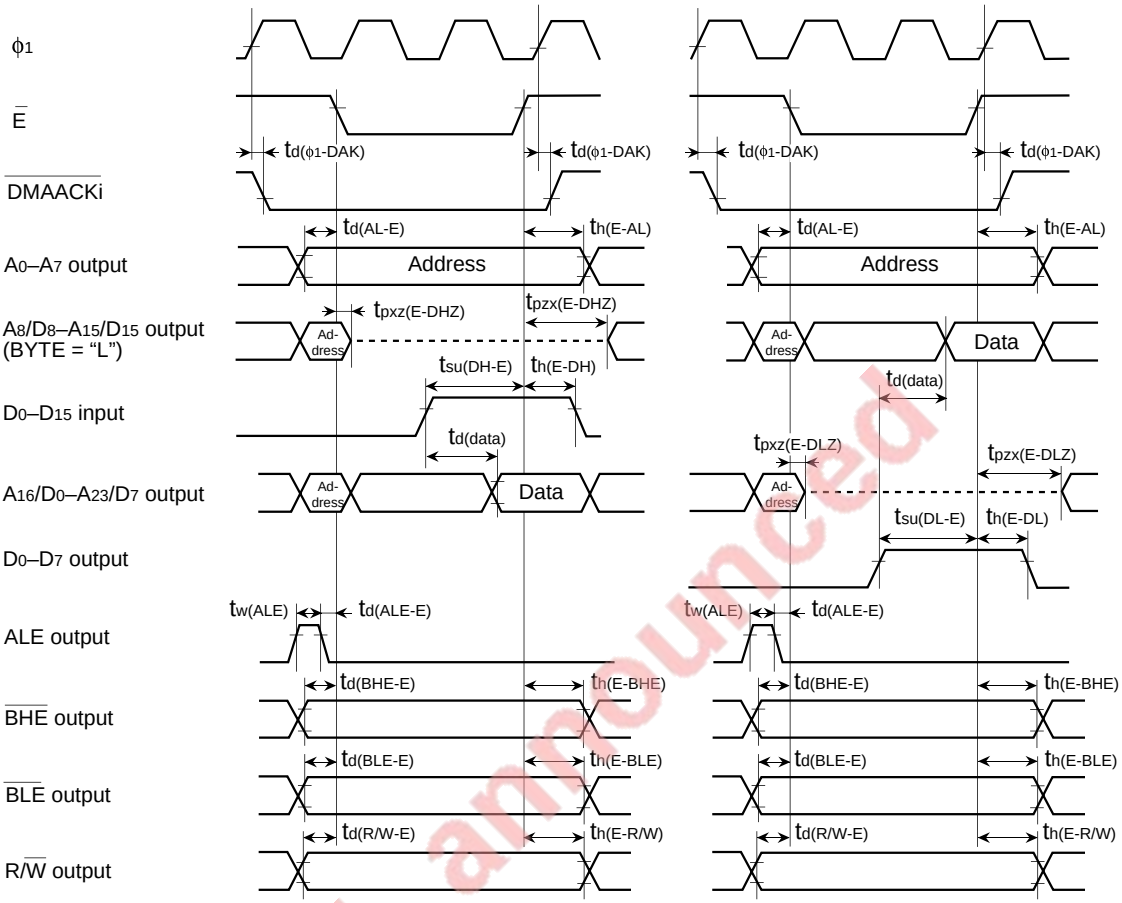
Test conditions
- Vcc = 5 V ± 10 %
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V
- D_0–D_{15} input : V_{IL} = 0.8 V, V_{IH} = 2.5 V
- DMAREQi input : V_{IL} = 0.8 V, V_{IH} = 2.5 V

EOL announced

Appendix 11. Electrical characteristics

At DMA transfer

•1-bus transfer timing

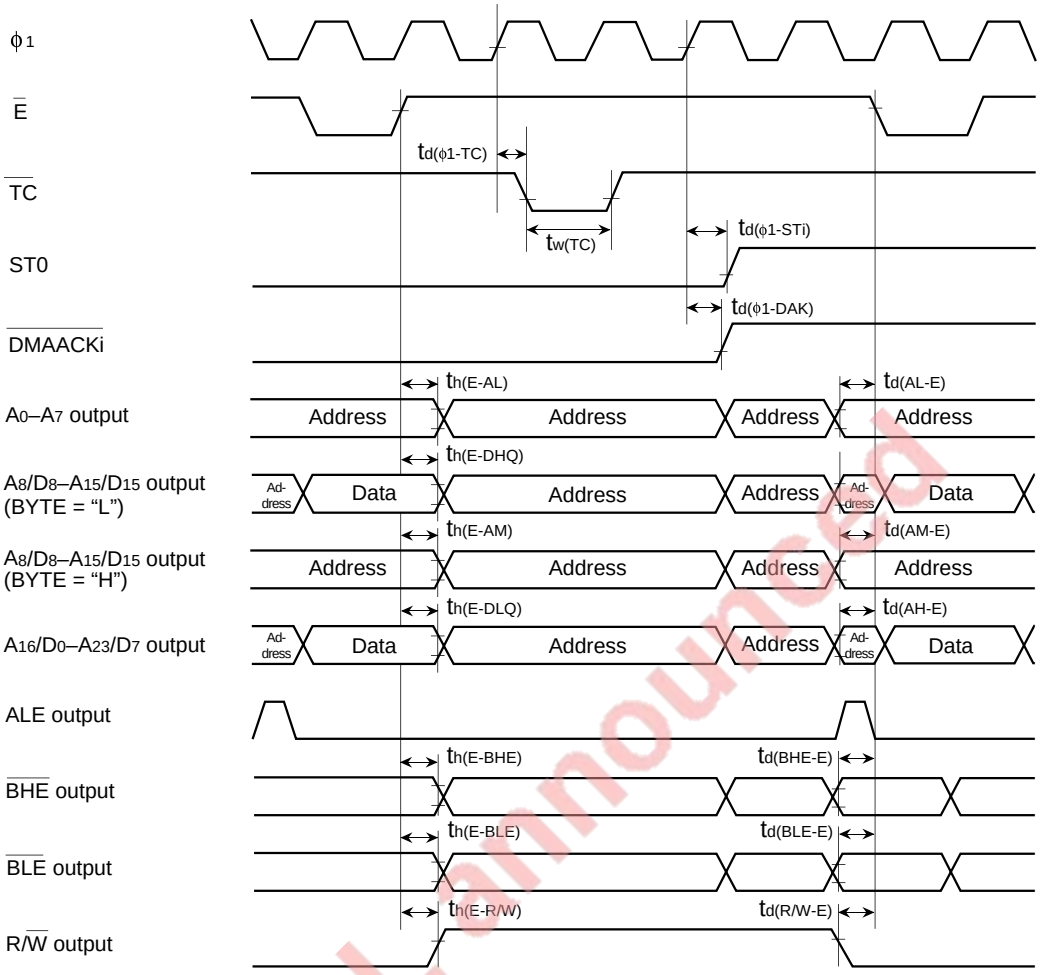


Test conditions
- V_{CC} = 5 V ± 10 %
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V
- D_0–D_{15} input : V_{IL} = 0.8 V, V_{IH} = 2.5 V

EOL announced

At DMA transfer

•Transfer complete timing



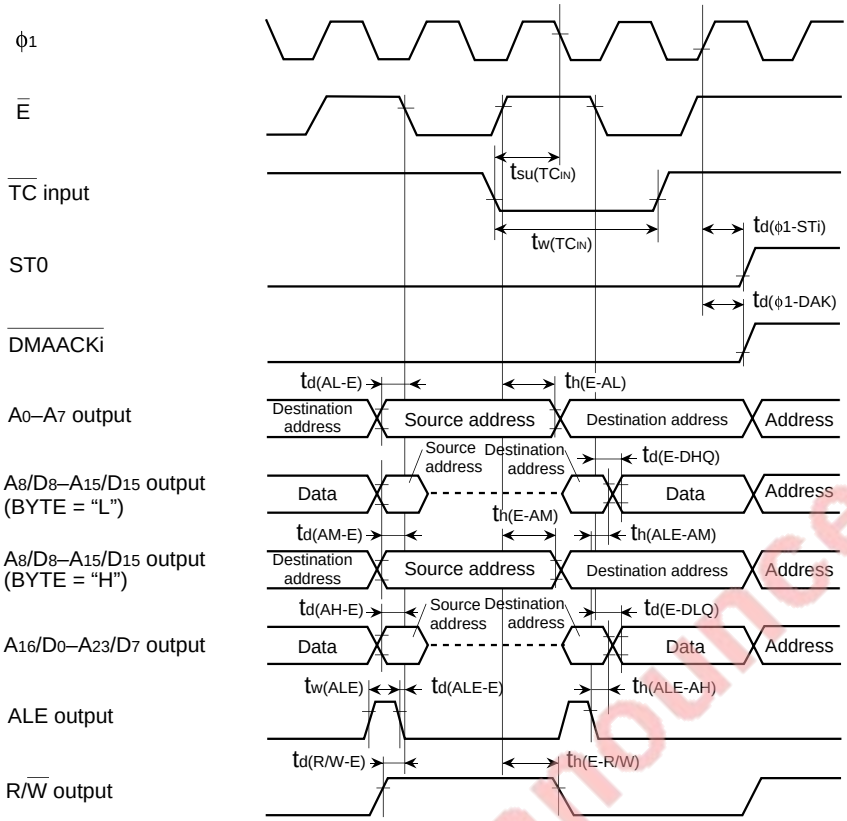
Test conditions

- Vcc = 5 V ± 10 %
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V
- D_0–D_{15} input : V_{IL} = 0.8 V, V_{IH} = 2.5 V

EOL announced

When DMA transfer is forcedly completed by $\overline{TC}$ input

•$\overline{TC}$ input timing



Test conditions
- Vcc = 5 V ± 10 %
- Output timing voltage : V_{OL} = 0.8 V, V_{OH} = 2.0 V
- D_0–D_{15} input : V_{IL} = 0.8 V, V_{IH} = 2.5 V
- $\overline{TC}$ input : V_{IL} = 0.8 V, V_{IH} = 2.5 V

EOL announced

Table 3 Calculation formulas for internal peripheral devices' input/output timing depending on f(X_{IN}) (Vcc = 5 V ± 10 %, Vss = 0 V, Ta = –20 to 85 °C)

Timer A input (Gating input in timer mode)

Symbol	Calculation formula	Unit
$t_{c(TA)}$	$\frac{8 \times 10^9}{f(X_{IN})}$	ns
$t_{w(TAH)}$	$\frac{4 \times 10^9}{f(X_{IN})}$	ns
$t_{w(TAL)}$	$\frac{4 \times 10^9}{f(X_{IN})}$	ns

Timer A input (External trigger input in one-shot pulse mode)

Symbol	Calculation formula	Unit
$t_{c(TA)}$	$\frac{4 \times 10^9}{f(X_{IN})}$	ns

Timer B input (Pulse period measurement mode)

Symbol	Calculation formula	Unit
$t_{c(TB)}$	$\frac{8 \times 10^9}{f(X_{IN})}$	ns
$t_{w(TBH)}$	$\frac{4 \times 10^9}{f(X_{IN})}$	ns
$t_{w(TBL)}$	$\frac{4 \times 10^9}{f(X_{IN})}$	ns

Timer B input (Pulse width measurement mode)

Symbol	Calculation formula	Unit
$t_{c(TB)}$	$\frac{8 \times 10^9}{f(X_{IN})}$	ns
$t_{w(TBH)}$	$\frac{4 \times 10^9}{f(X_{IN})}$	ns
$t_{w(TBL)}$	$\frac{4 \times 10^9}{f(X_{IN})}$	ns

EOL announced

APPENDIX

Appendix 11. Electrical characteristics

Table 4 Calculation formulas for bus timing depending on $f(X_{IN})$ (V_{CC} = 5 V ± 10 %, V_{SS} = 0 V, Ta = –20 to 85 °C)

Symbol		Calculation formula	Unit
$t_{d(AL-E)}$ $t_{d(AM-E)}$ $t_{d(AH-E)}$		$\frac{1 \times 10^9}{f(X_{IN})} - 25$	ns
$t_{d(AM-ALE)}$ $t_{d(AH-ALE)}$		$\frac{1 \times 10^9}{f(X_{IN})} - 35$	ns
$t_{w(ALE)}$		$\frac{1 \times 10^9}{f(X_{IN})} - 18$	ns
$t_{d(BLE-E)}$ $t_{d(BHE-E)}$ $t_{d(R/W-E)}$		$\frac{1 \times 10^9}{f(X_{IN})} - 20$	ns
$t_{h(E-AL)}$ $t_{h(E-AM)}$		$\frac{1 \times 10^9}{f(X_{IN})} - 22$	ns
$t_{h(E-DLQ)}$ $t_{h(E-DHQ)}$		$\frac{1 \times 10^9}{f(X_{IN})} - 22$	ns
$t_{pzx(E-DLZ)}$ $t_{pzx(E-DHZ)}$		$\frac{1 \times 10^9}{f(X_{IN})} - 20$	ns
$t_{h(E-BLE)}$ $t_{h(E-BHE)}$ $t_{h(E-R/W)}$		$\frac{1 \times 10^9}{f(X_{IN})} - 22$	ns
$t_{w(EL)}$	Wait bit = “1”	$\frac{2 \times 10^9}{f(X_{IN})} - 25$	ns
	Wait bit = “0”	$\frac{4 \times 10^9}{f(X_{IN})} - 25$	ns
$t_{su(A-DL)}$ $t_{su(A-DH)}$	Wait bit = “1”	$\frac{3 \times 10^9}{f(X_{IN})} - 70$	ns
	Wait bit = “0”	$\frac{5 \times 10^9}{f(X_{IN})} - 70$	ns
$t_{su(ALE-DL)}$ $t_{su(ALE-DH)}$	Wait bit = “1”	$\frac{3 \times 10^9}{f(X_{IN})} - 65$	ns
	Wait bit = “0”	$\frac{5 \times 10^9}{f(X_{IN})} - 65$	ns

EOL announced

Table 5 Calculation formulas for DRAM control bus timing depending of $f(X_{IN})$ (Vcc = 5 V ± 10 %, Vss = 0 V, Ta = –20 to 85 °C)

Read

Symbol	Calculation formula	Unit	Symbol	Calculation formula	Unit
$t_{w(RASL)}$	$\frac{4 \times 10^9}{f(X_{IN})} - 40$	ns	$t_{h(CAS-CA)}$	$\frac{4 \times 10^9}{f(X_{IN})} - 60$	ns
$t_{w(CASL)}$	$\frac{3 \times 10^9}{f(X_{IN})} - 27.5$	ns	$t_{d(R/W-RAS)}$	$\frac{1 \times 10^9}{f(X_{IN})} - 22$	ns
$t_{w(RASH)}$	$\frac{2 \times 10^9}{f(X_{IN})} - 20$	ns	$t_{h(CAS-R/W)}$	$\frac{1 \times 10^9}{f(X_{IN})} - 22$	ns
$t_{d(RAS-CAS)}$	$\frac{1 \times 10^9}{f(X_{IN})} - 12$	ns	$t_{d(E-CA)}$	$\frac{1 \times 10^9}{f(X_{IN})} + 25$	ns
$t_{d(RA-RAS)}$	$\frac{1 \times 10^9}{f(X_{IN})} - 35$	ns	$t_{d(E-CASL)}$	$\frac{1 \times 10^9}{f(X_{IN})} + 37.5$	ns
$t_{h(RAS-RA)}$	$\frac{1 \times 10^9}{f(X_{IN})} - 22$	ns			

Write

Symbol	Calculation formula	Unit	Symbol	Calculation formula	Unit
$t_{w(RASL)}$	$\frac{4 \times 10^9}{f(X_{IN})} - 40$	ns	$t_{d(CA-CAS)}$	$\frac{1 \times 10^9}{f(X_{IN})} - 30$	ns
$t_{w(CASL)}$	$\frac{2 \times 10^9}{f(X_{IN})} - 25$	ns	$t_{h(CAS-CA)}$	$\frac{3 \times 10^9}{f(X_{IN})} - 60$	ns
$t_{w(RASH)}$	$\frac{2 \times 10^9}{f(X_{IN})} - 20$	ns	$t_{d(R/W-RAS)}$	$\frac{1 \times 10^9}{f(X_{IN})} - 22$	ns
$t_{d(RAS-CAS)}$	$\frac{2 \times 10^9}{f(X_{IN})} - 20$	ns	$t_{h(CAS-R/W)}$	$\frac{1 \times 10^9}{f(X_{IN})} - 22$	ns
$t_{d(RA-RAS)}$	$\frac{1 \times 10^9}{f(X_{IN})} - 35$	ns	$t_{d(E-CASL)}$	$\frac{2 \times 10^9}{f(X_{IN})} + 35(0)^*$	ns
$t_{h(RAS-RA)}$	$\frac{1 \times 10^9}{f(X_{IN})} - 22$	ns			

✻ The value within ( ) is for the minimum value.

Refresh

Symbol	Calculation formula	Unit	Symbol	Calculation formula	Unit
$t_{w(RASL)}$	$\frac{4 \times 10^9}{f(X_{IN})} - 40$	ns	$t_{d(CAS-RAS)}$	$\frac{1 \times 10^9}{f(X_{IN})} - 22.5$	ns
$t_{w(CASL)}$	$\frac{2 \times 10^9}{f(X_{IN})} - 25$	ns	$t_{h(RAS-CAS)}$	$\frac{1 \times 10^9}{f(X_{IN})} - 22.5$	ns

EOL announced

APPENDIX

Appendix 11. Electrical characteristics

Table 6 Calculation formulas for DMA transfer bus timing depending on f(X_{IN}) (Vcc = 5 V ± 10 %, Vss = 0 V, Ta = –20 to 85 °C)

Symbol		Calculation formula	Unit
$t_{d(AL-E)}$ $t_{d(AM-E)}$ $t_{d(AH-E)}$		$\frac{1 \times 10^9}{f(X_{IN})} - 25$	ns
$t_{w(ALE)}$		$\frac{1 \times 10^9}{f(X_{IN})} - 18$	ns
$t_{d(BLE-E)}$ $t_{d(BHE-E)}$ $t_{d(R/W-E)}$		$\frac{1 \times 10^9}{f(X_{IN})} - 20$	ns
$t_{h(E-AL)}$ $t_{h(E-AM)}$		$\frac{1 \times 10^9}{f(X_{IN})} - 22$	ns
$t_{h(E-DLQ)}$ $t_{h(E-DHQ)}$		$\frac{1 \times 10^9}{f(X_{IN})} - 22$	ns
$t_{pzx(E-DLZ)}$ $t_{pzx(E-DHZ)}$		$\frac{1 \times 10^9}{f(X_{IN})} - 20$	ns
$t_{h(E-BLE)}$ $t_{h(E-BHE)}$ $t_{h(E-R/W)}$		$\frac{1 \times 10^9}{f(X_{IN})} - 22$	ns
$t_{w(EL)}$	Transfer source/Transfer destination wait bit = “1”	$\frac{2 \times 10^9}{f(X_{IN})} - 25$	ns
	Transfer source/Transfer destination wait bit = “0”	$\frac{4 \times 10^9}{f(X_{IN})} - 25$	ns
$t_{w(TC)}$		$\frac{2 \times 10^9}{f(X_{IN})} - 30$	ns

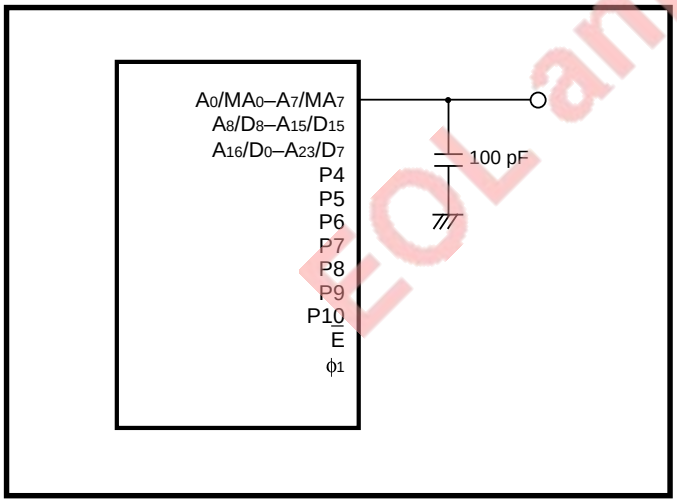


Fig. 13 Test circuit for each pin

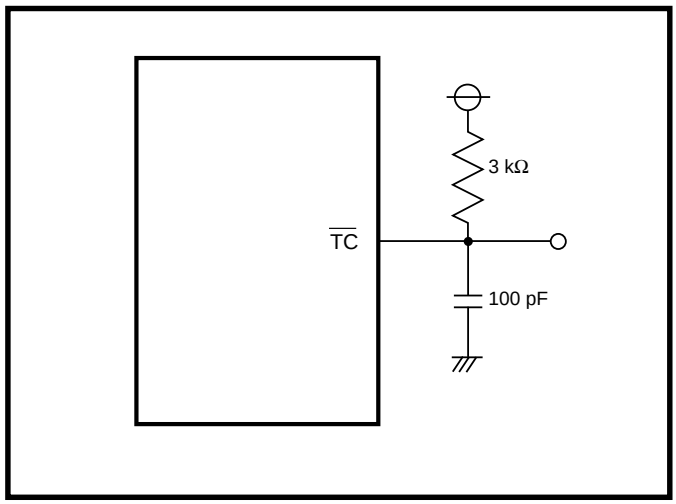


Fig. 14 Test circuit for $\overline{TC}$ output delay time and TC output pulse width

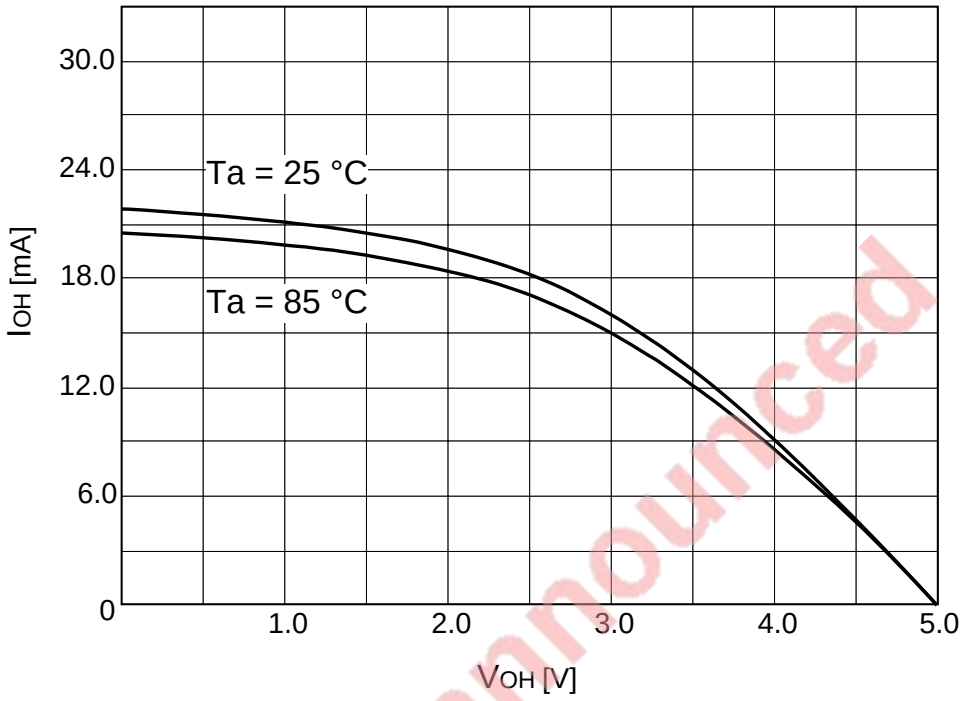
EOL announced

Appendix 12. Standard characteristics

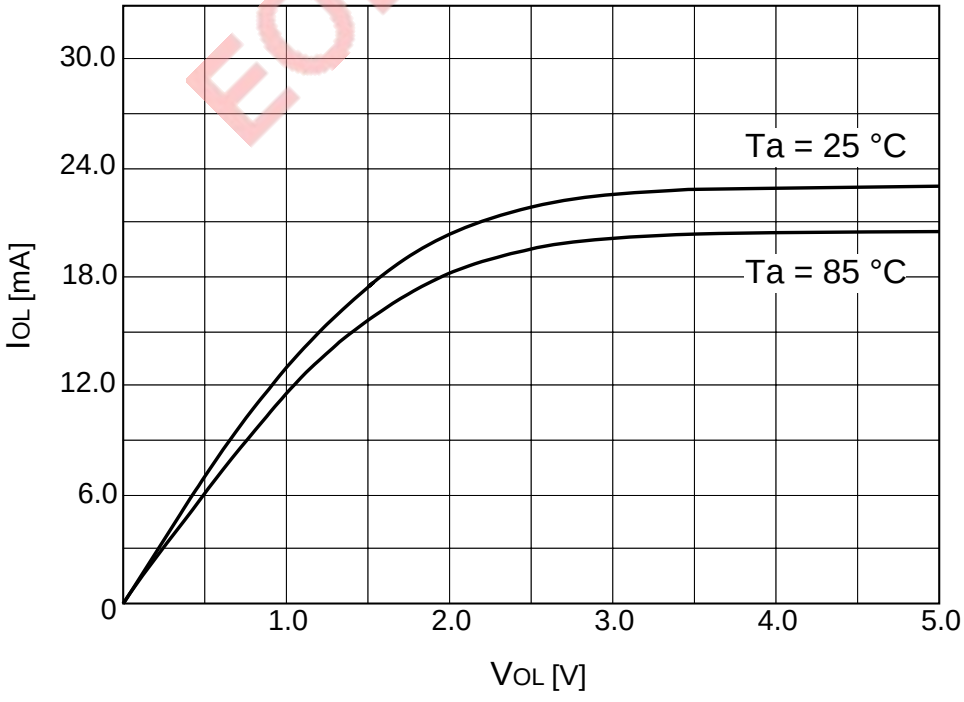
Standard characteristics described below are just examples of the M37721S2BFP's characteristics and are not guaranteed. For each parameter's limits, refer to section **"Appendix 11. Electrical characteristics."**

1. Programmable I/O port (CMOS output) standard characteristics

(1) P-channel I_{OH}–V_{OH} characteristics



(2) N-channel I_{OL}–V_{OL} characteristics



EOL announced

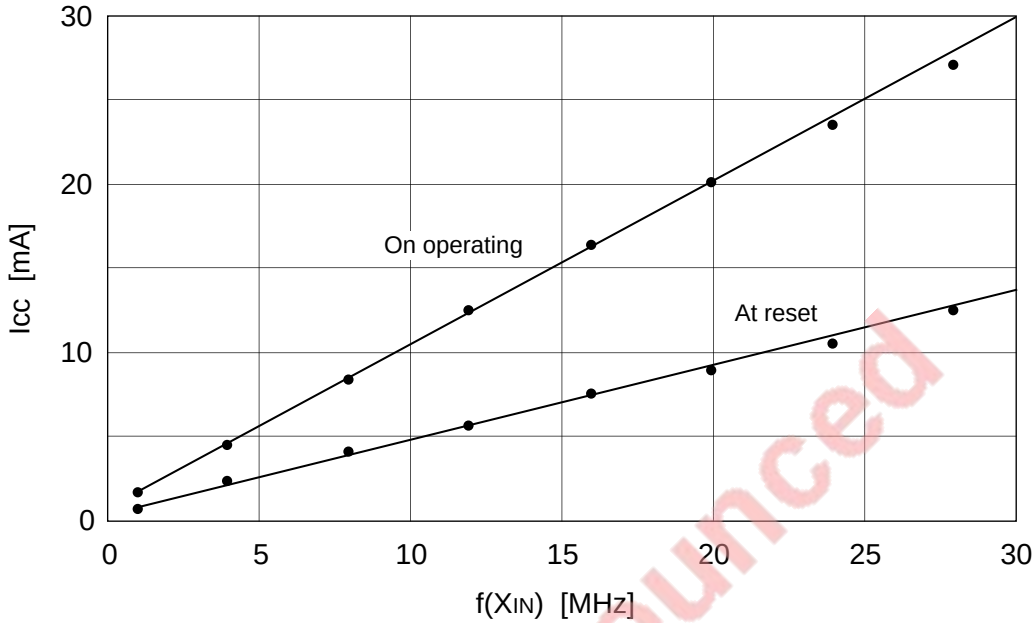
APPENDIX

Appendix 12. Standard characteristics

2. Icc–f(XIN) standard characteristics

(1) Icc–f(XIN) characteristics on operating and at reset

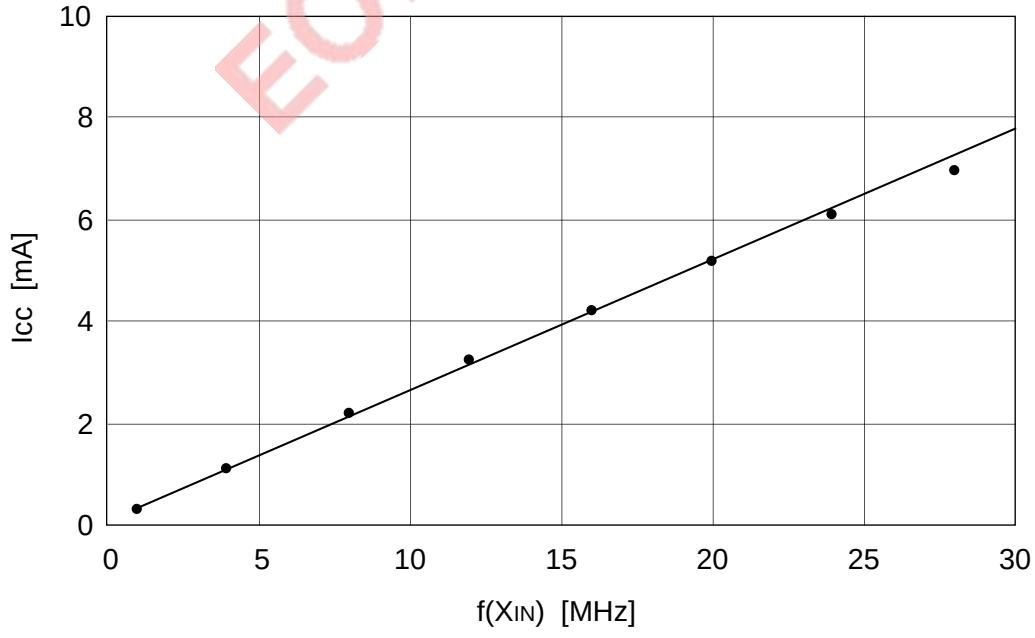
Measurement condition (Vcc = 5.0 V, Ta = 25 °C, f(XIN) : square waveform, microprocessor mode)



EOL announced

(2) Wait mode

Measurement condition (Vcc = 5.0 V, Ta = 25 °C, f(XIN) : square waveform, microprocessor mode)



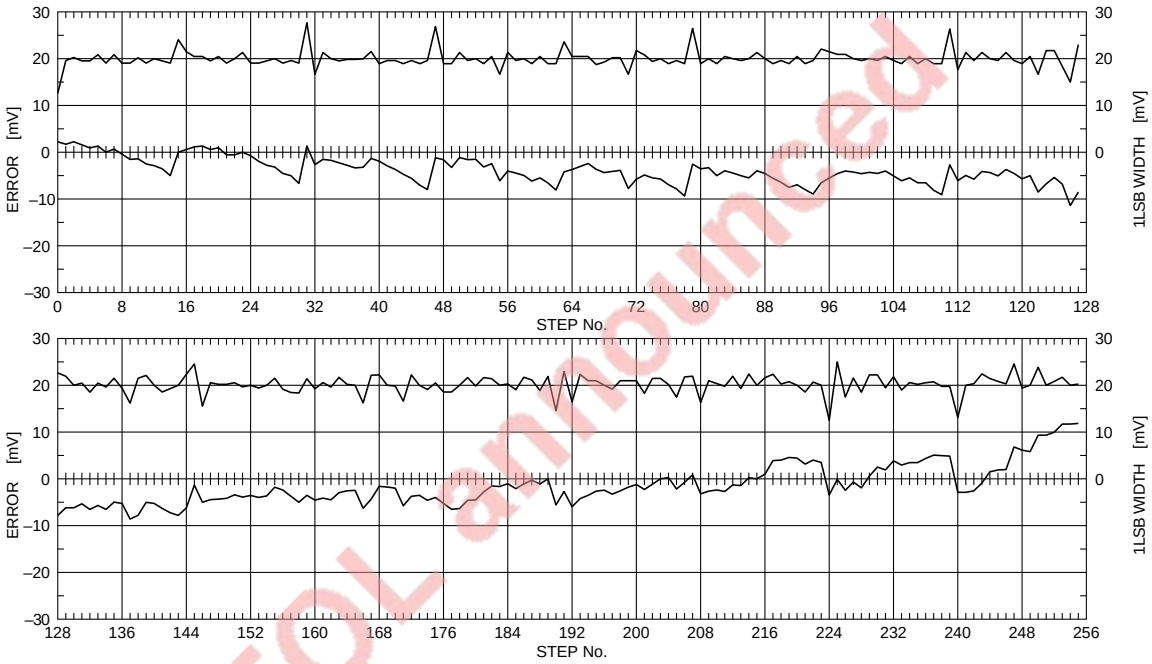
3. A-D converter standard characteristics

The lower lines of the graph indicate the absolute precision errors. These are expressed as the deviation from the ideal value when the output code changes. For example, the change in output code from 0 to 1 should occur at 10 mV, but the measured value is +2 mV. Accordingly, the measured point of change is 10 + 2 = 12 mV.

The upper lines of the graph indicate the input voltage width for which the output code is constant. For example, the measured input voltage width for which the output code is 15 is 24 mV, so that the differential non-linear error is 24 – 20 = 4 mV (0.2 LSB).

[Measurement conditions]

•Vcc = 5 V, V_{REF} = 5.12 V, f(X_{IN}) = 25 MHz, Ta = 25 °C, ϕ_{AD} = f_2 divided by 2



EOL announced

APPENDIX

Appendix 12. Standard characteristics

MEMORANDUM

EOL announced

GLOSSARY

EOL announced

GLOSSARY

This section briefly explains the terms used in this user's manual. The terms defined here apply to this manual only.

Term	Meaning	Relevant term
Access	Means performing read, write, or read and write. In DRAMC, also means performing DRAM refresh.	
Access space	An accessible memory space of up to 16 Mbytes.	Access
Access characteristics	Means whether accessible or not.	Access
Branch	Means moving the program's execution point (= address) to another location.	
Bus control signal	A generic name for ALE, $\overline{E}$, R/$\overline{W}$, $\overline{BLE}$, $\overline{BHE}$, $\overline{RDY}$, $\overline{HOLD}$, $\overline{HLDA}$, BYTE, ST0, and ST1 signals.	
Countdown	Means decreasing by 1 and counting.	Countup
Count source	A signal that is counted by timers A and B, the UARTi baud rate register (BRGi) and the watchdog timer. That is f_2, f_{16}, f_{64}, f_{512} selected by the count source select bits and others.	
Countup	Means increasing by 1 and counting.	Countdown
External area	An accessible area for external devices connected. It is up to 16-Mbyte external area.	Internal area
External bus	A generic name for the external address bus and the external data bus.	
External device	Devices connected externally to the microcomputer. A generic name for a memory, an I/O device and a peripheral IC.	
Internal area	An accessible internal area. A generic name for areas of the internal RAM and the SFR.	External area
Interrupt routine	A routine that is automatically executed when an interrupt request is accepted. Set the start address of this routine into the interrupt vector table.	
Overflow	A state where the countup resultant is greater than the counter resolution.	Underflow Countup
Read-modify-write instruction	An instruction that reads the memory contents, modifies them and writes back to the same address. Relevant instructions are the **ASL**, **ASR**, **CLB**, **DEC**, **INC**, **LSR**, **ROL**, **ROR**, **SEB** instructions.	
Signal required for access to external device	A generic name for bus control, address bus, and data bus signals.	Bus control signal
Stop mode	A state where the oscillation circuit halts and the program execution is stopped. By executing the **STP** instruction, the microcomputer enters the stop mode.	Wait mode
UART	Clock asynchronous serial I/O. When used to designate the name of a functional block, this term also means the serial I/O which can be switched to the cock synchronous serial I/O.	Clock synchronous serial I/O
Underflow	A state where the countdown resultant is greater than the counter resolution.	Overflow Countdown
Wait mode	A state where the oscillation circuit is operating, however, the program execution is stopped. By executing the **WIT** instruction, the microcomputer enters the wait mode.	Stop mode

EOL announced

EOL announced

MITSUBISHI SEMICONDUCTORS
USER'S MANUAL
7721 Group

Sep. First Edition 1997

Editioned by
Committee of editing of Mitsubishi Semiconductor USER'S MANUAL

Published by
Mitsubishi Electric Corp., Semiconductor Marketing Division

This book, or parts thereof, may not be reproduced in any form without permission of Mitsubishi Electric Corporation.
©1997 MITSUBISHI ELECTRIC CORPORATION

User's Manual
7721 Group

EOL announced

Renesas Technology Corp.
Nippon Bldg.,6-2,Otemachi 2-chome,Chiyoda-ku,Tokyo,100-0004 Japan

© 1997 MITSUBISHI ELECTRIC CORPORATION.

New publication, effective Sep. 1997.
Specifications subject to change without notice.