To our customers,

Old Company Name in Catalogs and Other Documents

On April 1st, 2010, NEC Electronics Corporation merged with Renesas Technology Corporation, and Renesas Electronics Corporation took over all the business of both companies. Therefore, although the old company name remains in this document, it is a valid Renesas Electronics document. We appreciate your understanding.

Renesas Electronics website: http://www.renesas.com

April 1st, 2010
Renesas Electronics Corporation

Issued by: Renesas Electronics Corporation (http://www.renesas.com)

Send any inquiries to http://www.renesas.com/inquiry.

Notice

1. All information included in this document is current as of the date this document is issued. Such information, however, is subject to change without any prior notice. Before purchasing or using any Renesas Electronics products listed herein, please confirm the latest product information with a Renesas Electronics sales office. Also, please pay regular and careful attention to additional and different information to be disclosed by Renesas Electronics such as that disclosed through our website.
2. Renesas Electronics does not assume any liability for infringement of patents, copyrights, or other intellectual property rights of third parties by or arising from the use of Renesas Electronics products or technical information described in this document. No license, express, implied or otherwise, is granted hereby under any patents, copyrights or other intellectual property rights of Renesas Electronics or others.
3. You should not alter, modify, copy, or otherwise misappropriate any Renesas Electronics product, whether in whole or in part.
4. Descriptions of circuits, software and other related information in this document are provided only to illustrate the operation of semiconductor products and application examples. You are fully responsible for the incorporation of these circuits, software, and information in the design of your equipment. Renesas Electronics assumes no responsibility for any losses incurred by you or third parties arising from the use of these circuits, software, or information.
5. When exporting the products or technology described in this document, you should comply with the applicable export control laws and regulations and follow the procedures required by such laws and regulations. You should not use Renesas Electronics products or the technology described in this document for any purpose relating to military applications or use by the military, including but not limited to the development of weapons of mass destruction. Renesas Electronics products and technology may not be used for or incorporated into any products or systems whose manufacture, use, or sale is prohibited under any applicable domestic or foreign laws or regulations.
6. Renesas Electronics has used reasonable care in preparing the information included in this document, but Renesas Electronics does not warrant that such information is error free. Renesas Electronics assumes no liability whatsoever for any damages incurred by you resulting from errors in or omissions from the information included herein.
7. Renesas Electronics products are classified according to the following three quality grades: "Standard", "High Quality", and "Specific". The recommended applications for each Renesas Electronics product depends on the product's quality grade, as indicated below. You must check the quality grade of each Renesas Electronics product before using it in a particular application. You may not use any Renesas Electronics product for any application categorized as "Specific" without the prior written consent of Renesas Electronics. Further, you may not use any Renesas Electronics product for any application for which it is not intended without the prior written consent of Renesas Electronics. Renesas Electronics shall not be in any way liable for any damages or losses incurred by you or third parties arising from the use of any Renesas Electronics product for an application categorized as "Specific" or for which the product is not intended where you have failed to obtain the prior written consent of Renesas Electronics. The quality grade of each Renesas Electronics product is "Standard" unless otherwise expressly specified in a Renesas Electronics data sheets or data books, etc.

   "Standard": Computers; office equipment; communications equipment; test and measurement equipment; audio and visual equipment; home electronic appliances; machine tools; personal electronic equipment; and industrial robots.

   "High Quality": Transportation equipment (automobiles, trains, ships, etc.); traffic control systems; anti-disaster systems; anti-crime systems; safety equipment; and medical equipment not specifically designed for life support.

   "Specific": Aircraft; aerospace equipment; submersible repeaters; nuclear reactor control systems; medical equipment or systems for life support (e.g. artificial life support devices or systems), surgical implantations, or healthcare intervention (e.g. excision, etc.), and any other applications or purposes that pose a direct threat to human life.
8. You should use the Renesas Electronics products described in this document within the range specified by Renesas Electronics, especially with respect to the maximum rating, operating supply voltage range, movement power voltage range, heat radiation characteristics, installation and other product characteristics. Renesas Electronics shall have no liability for malfunctions or damages arising out of the use of Renesas Electronics products beyond such specified ranges.
9. Although Renesas Electronics endeavors to improve the quality and reliability of its products, semiconductor products have specific characteristics such as the occurrence of failure at a certain rate and malfunctions under certain use conditions. Further, Renesas Electronics products are not subject to radiation resistance design. Please be sure to implement safety measures to guard them against the possibility of physical injury, and injury or damage caused by fire in the event of the failure of a Renesas Electronics product, such as safety design for hardware and software including but not limited to redundancy, fire control and malfunction prevention, appropriate treatment for aging degradation or any other appropriate measures. Because the evaluation of microcomputer software alone is very difficult, please evaluate the safety of the final products or system manufactured by you.
10. Please contact a Renesas Electronics sales office for details as to environmental matters such as the environmental compatibility of each Renesas Electronics product. Please use Renesas Electronics products in compliance with all applicable laws and regulations that regulate the inclusion or use of controlled substances, including without limitation, the EU RoHS Directive. Renesas Electronics assumes no liability for damages or losses occurring as a result of your noncompliance with applicable laws and regulations.
11. This document may not be reproduced or duplicated, in any form, in whole or in part, without prior written consent of Renesas Electronics.
12. Please contact a Renesas Electronics sales office if you have any questions regarding the information contained in this document or Renesas Electronics products, or if you have any other inquiries.

(Note 1) "Renesas Electronics" as used in this document means Renesas Electronics Corporation and also includes its majority-owned subsidiaries.

(Note 2) "Renesas Electronics product(s)" means any product developed or manufactured by or for Renesas Electronics.

4571 Group

SINGLE-CHIP 4-BIT CMOS MICROCOMPUTER

REJ03B0179-0102
Rev.1.02
May 25, 2007

DESCRIPTION

The 4571 Group is a 4-bit single-chip microcomputer designed with CMOS technology. Its CPU is that of the 4500 series using a simple, high-speed instruction set. The computer is equipped with three 8-bit timers (each timer has one or two reload registers), interrupts, and voltage drop detection circuit.
The various microcomputers in the 4571 Group include variations of the built-in memory size as shown in the table below.

FEATURES

- Minimum instruction execution time ..... 0.5 µs
  (at 6 MHz oscillation frequency, in through-mode)
- Supply voltage ..... 1.8 to 5.5 V
  (It depends on oscillation frequency and operation mode)
- Timers
  Timer 1 ..... 8-bit timer with a reload register and carrier wave output auto-control function
  Timer 2 ..... 8-bit timer with a reload register
  Timer 3 ..... 8-bit timer with two reload registers and carrier wave generation circuit
- Interrupt ..... 6 sources
- Key-on wakeup function pins 12
- I/O port 17
- Output port 1
- Input port 1
- Voltage drop detection circuit
  Reset occurrence ..... Typ. 1.65 V (Ta = 25 °C)
  Reset release ..... Typ. 1.75 V (Ta = 25 °C)
  Interrupt occurrence ..... Typ. 1.85 V (Ta = 25 °C)
- Watchdog timer
- Power-on reset circuit
- Clock generating circuit (ceramic resonator)

APPLICATION

Remote control transmitter

Table 1 Support Product

Part number	ROM size (× 10 bits)	RAM size (× 4 bits)	Package	ROM type
M34571G4FP (Note 1)	4096 words	128 words	PRSP0024GA-A	QzROM
M34571G4-XXXFP	4096 words	128 words	PRSP0024GA-A	QzROM
M34571G6FP (Note 1)	6144 words	128 words	PRSP0024GA-A	QzROM
M34571G6-XXXFP	6144 words	128 words	PRSP0024GA-A	QzROM
M34571GDFP (Note 1)	16384 words	128 words	PRSP0024GA-A	QzROM
M34571GD-XXXFP	16384 words	128 words	PRSP0024GA-A	QzROM

Note 1.Shipped in blank

PIN CONFIGURATION

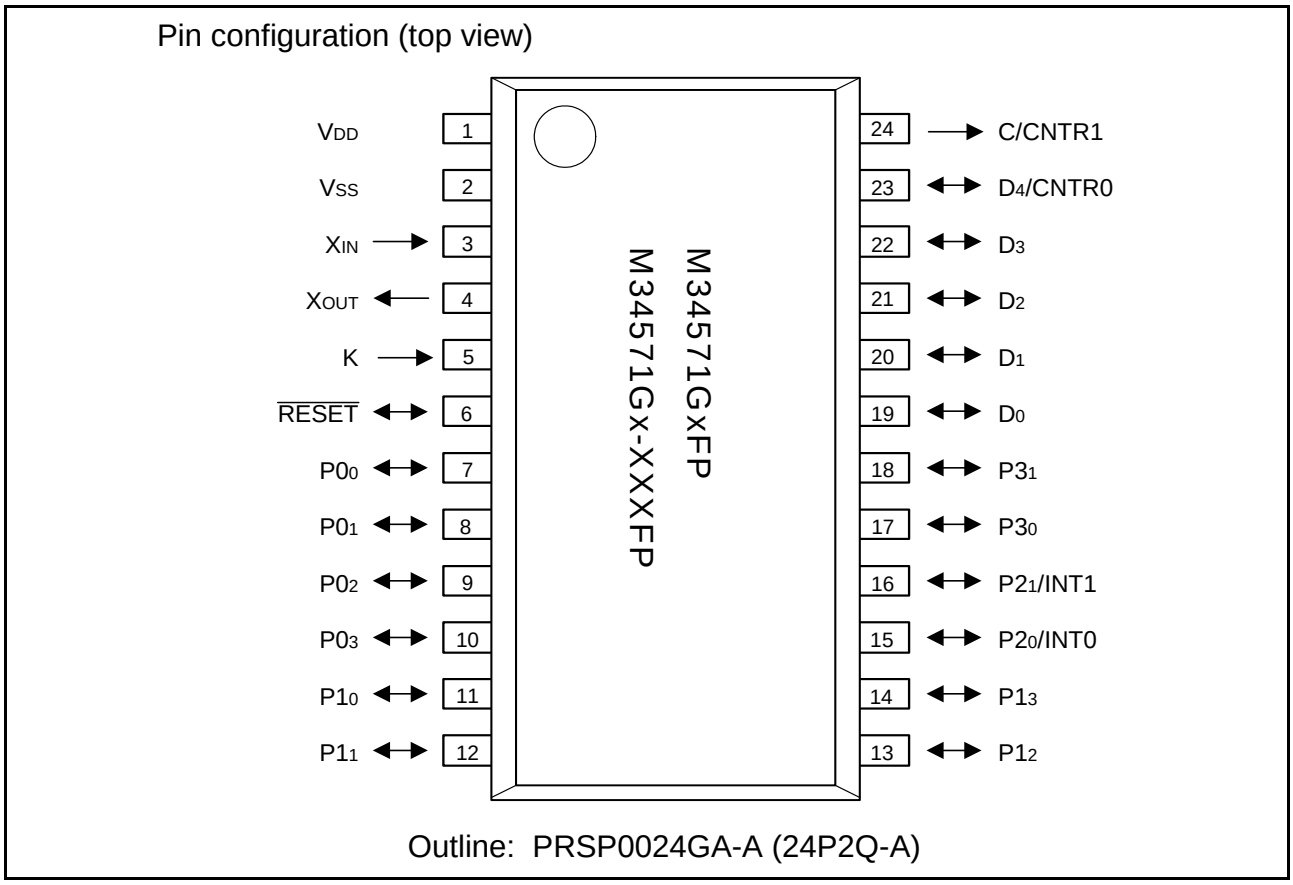


Fig 1. Pin configuration (PRSP0024GA-A type)

FUNCTIONAL BLOCK

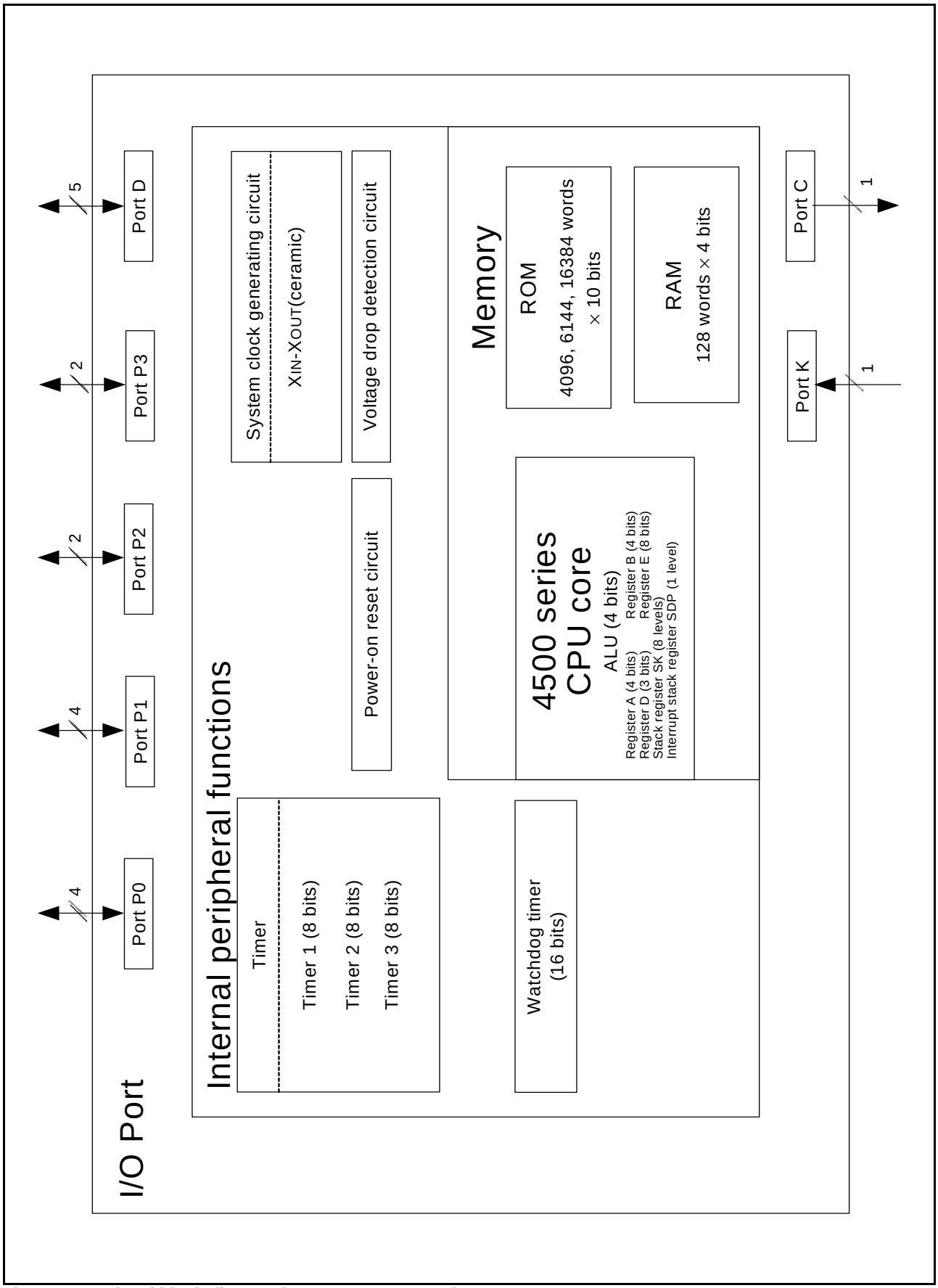


Fig 2. Functional block diagram (PRSP0024GA-A type)

PERFORMANCE OVERVIEW

Table 2 Performance overview

Parameter			Function
Number of basic instructions		M34571G4/G6	126
		M34571GD	128
Minimum instruction execution time			0.5 µs (Oscillation frequency 6 MHz: through mode)
Memory sizes	ROM	M34571G4	4096 words × 10 bits
		M34571G6	6144 words × 10 bits
		M34571GD	16384 words × 10 bits
	RAM		128 words × 4 bits
I/O port	D_0–D_4	I/O (Input is examined by skip decision)	Five independent I/O ports; The output structure of ports D_0–D_3 is switched by software. Port D_4 is also used as CNTR0, respectively.
	$P0_0$–$P0_3$	I/O	4-bit I/O port; a pull-up function and a key-on wakeup function can be switched by software.
	$P1_0$–$P1_3$	I/O	4-bit I/O port; a pull-up function and a key-on wakeup function can be switched by software.
	$P2_0$, $P2_1$	I/O	2-bit I/O port; a pull-up function and a key-on wakeup function can be switched by software. Ports $P2_0$ and $P2_1$ are also used as INT0 and INT1, respectively.
	$P3_0$, $P3_1$	I/O	2-bit I/O port ; the output structure is switched by software.
	C	Output	1-bit output port (CMOS output only); port C is also used as CNTR1 pin.
	K	Input	1-bit input port ; a key-on wakeup function can be switched by software.
	CNTR0	Timer I/O	1-bit I/O port ; CNTR0 pin is also used as port D_4.
	CNTR1	Timer output	1-bit output port ; CNTR1 pin is also used as port C.
	INT0, INT1	Interrupt input	1-bit input port ; INT0 and INT1 are also used as ports $P2_0$ and $P2_1$, respectively.
Timer	Timer 1		8-bit timer with a reload register and carrier wave output auto-control function, and has an event counter.
	Timer 2		8-bit timer with a reload register.
	Timer 3		8-bit timer with two reload registers and carrier wave generation function.
	Watchdog timer		16-bit timer, fixed dividing frequency (timer for monitor)
Power-on reset circuit			Built-in
Voltage drop detection circuit	Reset occurrence		Typ. 1.65 V (Ta=25 °C)
	Reset release		Typ. 1.75 V (Ta=25 °C)
	Interrupt occurrence		Typ. 1.85 V (Ta=25 °C)
Interrupt	Source		6 sources (two for external, three for timers, voltage drop detection circuit)
	Nesting		1 level
Subroutine nesting			8 levels
Device structure			CMOS sillicon gate
Package			24-pin plastic molded SSOP (PRSP0024GA-A)
Operating temperature range			-20 to 85 °C
Power source voltage			1.8 to 5.5 V (It depends on oscillation frequency and operation mode)
Power dissipation (Typ. value)	At active mode		0.3 mA (Ta = 25 °C, V_{DD} = 3.0 V, f(X_{IN})=4 MHz, f(STCK)=f(X_{IN})/8)
	At RAM back-up		0.1 µA (Ta = 25 °C, output transistor is cut-off state)

PIN DESCRIPTION

Table 3 Pin description

Pin	Name	Input/Output	Function
V_{DD}	Power source	–	Connected to a plus power supply.
V_{SS}	Power source	–	Connected to a 0 V power supply.
$\overline{RESET}$	Reset I/O	I/O	An N-channel open-drain I/O pin for a system reset. When the SRST instruction, watchdog timer, or the built-in power-on reset causes the system to be reset, the $\overline{RESET}$ pin outputs "L" level.
X_{IN}	Main clock input	Input	I/O pins of the main clock generating circuit. Connect a ceramic resonator between pins X_{IN} and X_{OUT}. A feedback resistor is built-in between them.
X_{OUT}	Main clock output	Output	
D_0–D_4	I/O port D (Input is examined by skip decision.)	I/O	Each pin of port D has an independent 1-bit wide I/O function. The output structure of ports D_0–D_3 can be switched to N-channel open-drain or CMOS by software. For input use, set the latch of the specified bit to "1" and select the N-channel open-drain. Port D_4 is also used as CNTR0 pin.
$P0_0$–$P0_3$	I/O port P0	I/O	Port P0 serves as a 4-bit I/O port. The output structure is N-channel open-drain. Port P0 has a key-on wakeup function and a pull-up function. Both functions can be switched by software.
$P1_0$–$P1_3$	I/O port P1	I/O	Port P1 serves as a 4-bit I/O port. The output structure is N-channel open-drain. Port P1 has a key-on wakeup function and a pull-up function. Both functions can be switched by software.
$P2_0$, $P2_1$	I/O port P2	I/O	Port P2 serves as a 2-bit I/O port. The output structure is N-channel open-drain. For input use, set the latch of the specified bit to "1". Ports $P2_0$ and $P2_1$ are also used as INT0 pin and INT1 pin, respectively.
$P3_0$, $P3_1$	I/O port P3	I/O	Port P3 serves as a 2-bit I/O port. The output structure can be switched to N-channel open-drain or CMOS by software. For input use, set the latch of the specified bit to "1".
C	Output port C	Output	Port C serves as a 1-bit output port. The output structure is CMOS. Port C is also used as CNTR1.
K	Input port K	Input	Port K serves as a 1-bit input port. It has the key-on wakeup function which can be switched by software. When port K is used for the input of key matrix, connect a pull-up resistor to it externally.
CNTR0, CNTR1	Timer I/O	I/O	CNTR0 pin has the function to input the clock for the timer 1 event counter, and to output the timer 1 or timer 2 underflow signal divided by 2. CNTR1 pin has the function to output the PWM signal generated by timer 3. CNTR0 pin and CNTR1 pin are also used as Ports D_4 and C, respectively.
INT0, INT1	Interrupt input	Input	INT0 pin and INT1 pin accept external interrupts. They have the key-on wakeup function which can be switched by software. INT0 pin and INT1 pin are also used as Ports $P2_0$ and $P2_1$, respectively.

MULTIFUNCTION

Table 4 Pin description

Pin	Multifunction	Pin	Multifunction	Pin	Multifunction	Pin	Multifunction
C	CNTR1	$P2_0$	INT0	CNTR1	C	INT0	$P2_0$
D_4	CNTR0	$P2_1$	INT1	CNTR0	D_4	INT1	$P2_1$

Note 1. Pins except above have just single function.
Note 2. The input of D_4 can be used even when CNTR0 (output) is selected.
The input/output of D_4 can be used even when CNTR0 (input) is selected.
Be careful when using inputs of both CNTR0 and D_4 since the input threshold value of CNTR0 pin is different from that of port D_4.
Note 3. "H" output function of port C can be used even when the CNTR1 (output) is used.
Note 4. The input/output of $P2_0$ can be used even when INT0 is used.
Be careful when using inputs of both INT0 and $P2_0$ since the input threshold value of INT0 pin is different from that of port $P2_0$.
Note 5. The input/output of $P2_1$ can be used even when INT1 is used.
Be careful when using inputs of both INT1 and $P2_1$ since the input threshold value of INT1 pin is different from that of port $P2_1$.

PORT FUNCTION

Table 5 Port function

Port	Pin	Input Output	Output structure	I/O unit	Control instructions	Control registers	Remark
Port D	D_0–D_3	I/O (5)	N-channel open-drain/ CMOS	1 bit	SD, RD SZD, CLD	FR1	Programmable output structure selection function
	D_4/CNTR0		N-channel open-drain			W1 W2 W5	–
Port P0	$P0_0$ $P0_1$ $P0_2$ $P0_3$	I/O (4)	N-channel open-drain	4 bits	OP0A IAP0	PU0 K0	Programmable pull-up and key-on wakeup function
Port P1	$P1_0$ $P1_1$ $P1_2$ $P1_3$	I/O (4)	N-channel open-drain	4 bits	OP1A IAP1	PU1 K1	Programmable pull-up and key-on wakeup function
Port P2	$P2_0$/INT0 $P2_1$/INT1	I/O (2)	N-channel open-drain	2 bits	OP2A IAP2	PU2 K2, I1, I2, L1	Programmable pull-up and key-on wakeup function
Port P3	$P3_0$ $P3_1$	I/O (2)	N-channel open-drain/ CMOS	2 bits	OP3A IAP3	FR0	Programmable output structure selection function
Port C	C/CNTR1	Output (1)	CMOS	1 bit	RCP SCP	W1, W3, W5	–
Port K	K	Input (1)	-	1 bit	IAK	K2	Programmable key-on wakeup function

DEFINITION OF CLOCK AND CYCLE

- Operation source clock

The operation source clock is the source clock to operate this product. In this product, the following clocks are used.

- Clock (f(X_{IN})) by the external ceramic resonator
- Clock (f(X_{IN})) by the external input

- System clock

The system clock is the basic clock for controlling this product. The system clock is selected by the register MR.

- Instruction clock

The instruction clock is a signal derived by dividing the system clock by 3. The one instruction clock cycle generates the one machine cycle.

- Machine cycle

The machine cycle is the standard cycle required to execute the instruction.

Table 6 Table Selection of system clock

Register MR		System clock	Operation mode
MR_3	MR_2		
1	1	f(STCK) = f(X_{IN})/8	Frequency divided by 8 mode
1	0	f(STCK) = f(X_{IN})/4	Frequency divided by 4 mode
0	1	f(STCK) = f(X_{IN})/2	Frequency divided by 2 mode
0	0	f(STCK) = f(X_{IN})	Frequency through mode

Note 1.The frequency divided by 8 is selected after system is released from reset.

CONNECTIONS OF UNUSED PINS

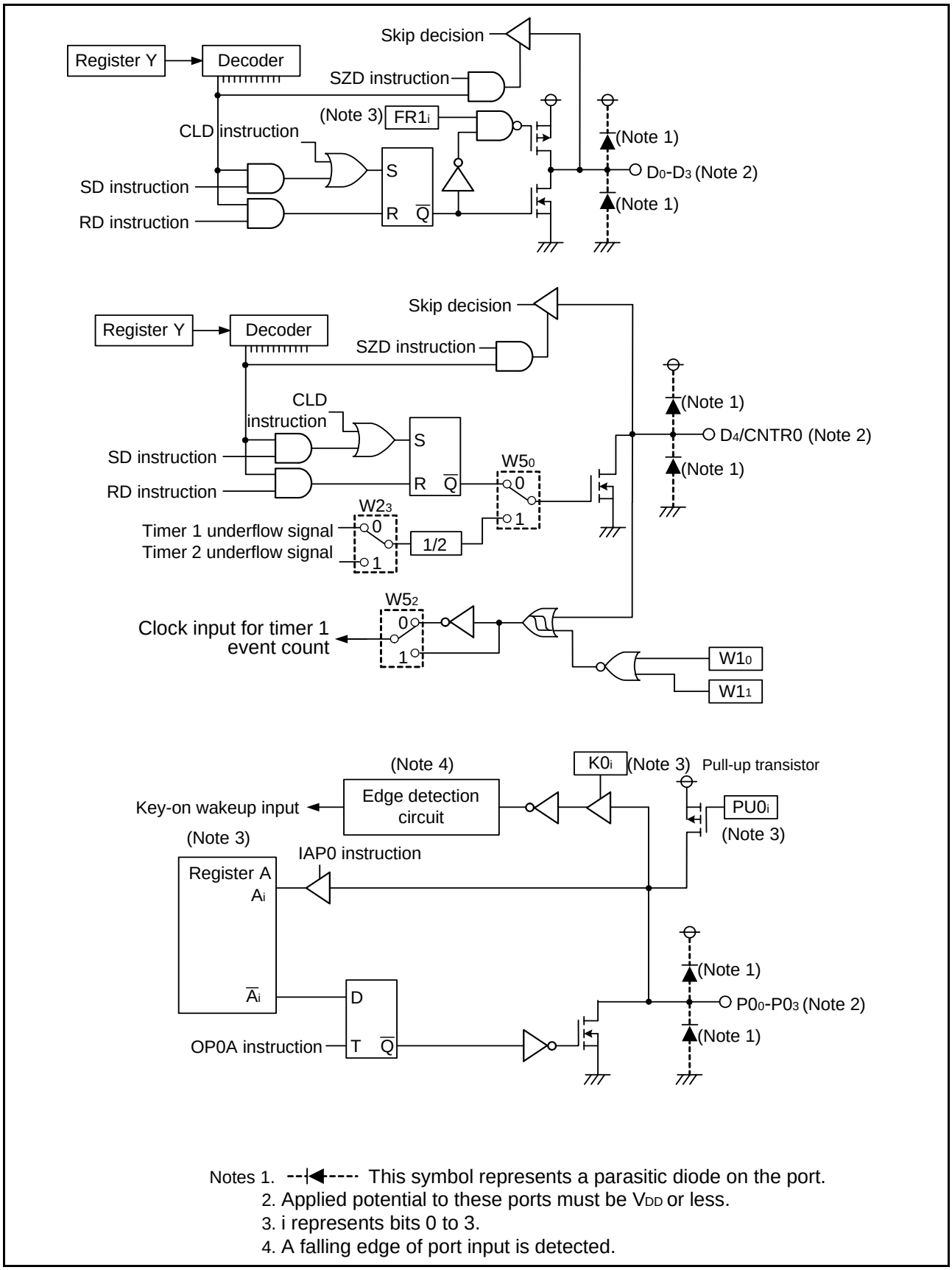
Table 7 Port function

Pin	Connection	Usage condition				
		Output structure	Pull-up transistor	Key-on wakeup	Value of output latch	Others
D_0–D_3 $P3_0$, $P3_1$	Open.	N-channel open-drain	–	–	0/1	(Note 1)
		CMOS	–	–	0/1	–
	Connect to V_{SS}.	N-channel open-drain	–	–	0/1	–
		CMOS	–	–	0	–
	Connect to V_{DD}.	N-channel open-drain	–	–	1	–
		CMOS	–	–	1	–
D_4/CNTR0	Open.	N-channel open-drain	–	–	0/1	(Notes 1, 2)
	Connect to V_{SS}.	N-channel open-drain	–	–	0/1	(Note 2)
	Connect to V_{DD}.	N-channel open-drain	–	–	1	(Note 2)
$P0_0$–$P0_3$, $P1_0$–$P1_3$	Open.	N-channel open-drain	OFF	Invalid	0/1	(Note 1)
			ON	Invalid	1	–
	Connect to V_{SS}.	N-channel open-drain	OFF	Invalid	0/1	–
	Connect to V_{DD}.	N-channel open-drain	ON/OFF	Valid/Invalid	1	–
$P2_0$/INT0 $P2_1$/INT1	Open.	N-channel open-drain	OFF	Invalid	0/1	(Notes 1, 3)
			ON	Invalid	1	(Note 3)
	Connect to V_{SS}.	N-channel open-drain	OFF	Invalid	0/1	(Note 3)
	Connect to V_{DD}.	N-channel open-drain	ON/OFF	Valid/Invalid	1	(Note 3)
C/CNTR1	Open.	CMOS	–	–	0/1	–
	Connect to V_{SS}.	CMOS	–	–	0	(Note 4)
K	Connect to V_{SS}.	–	–	Invalid	–	–
	Connect to V_{DD}.	–	–	Valid/Invalid	–	–

Note 1.If a port input instruction (SZD, IAP0, IAP1, IAP2, IAP3) is executed when the output latch is 1, the supply voltage may be increased in the instruction execution cycle by the through current.
Note 2.Do not select the CNTR0 input as the timer 1 count source. ($W11$ $W10 \neq 11$)
Note 3.Set the input of INT0 pin or INT1 pin to be disabled. ($I13=0$, $I23=0$)
Note 4.Set the output of the CNTR1 pin to be invalid. ($W33=0$)

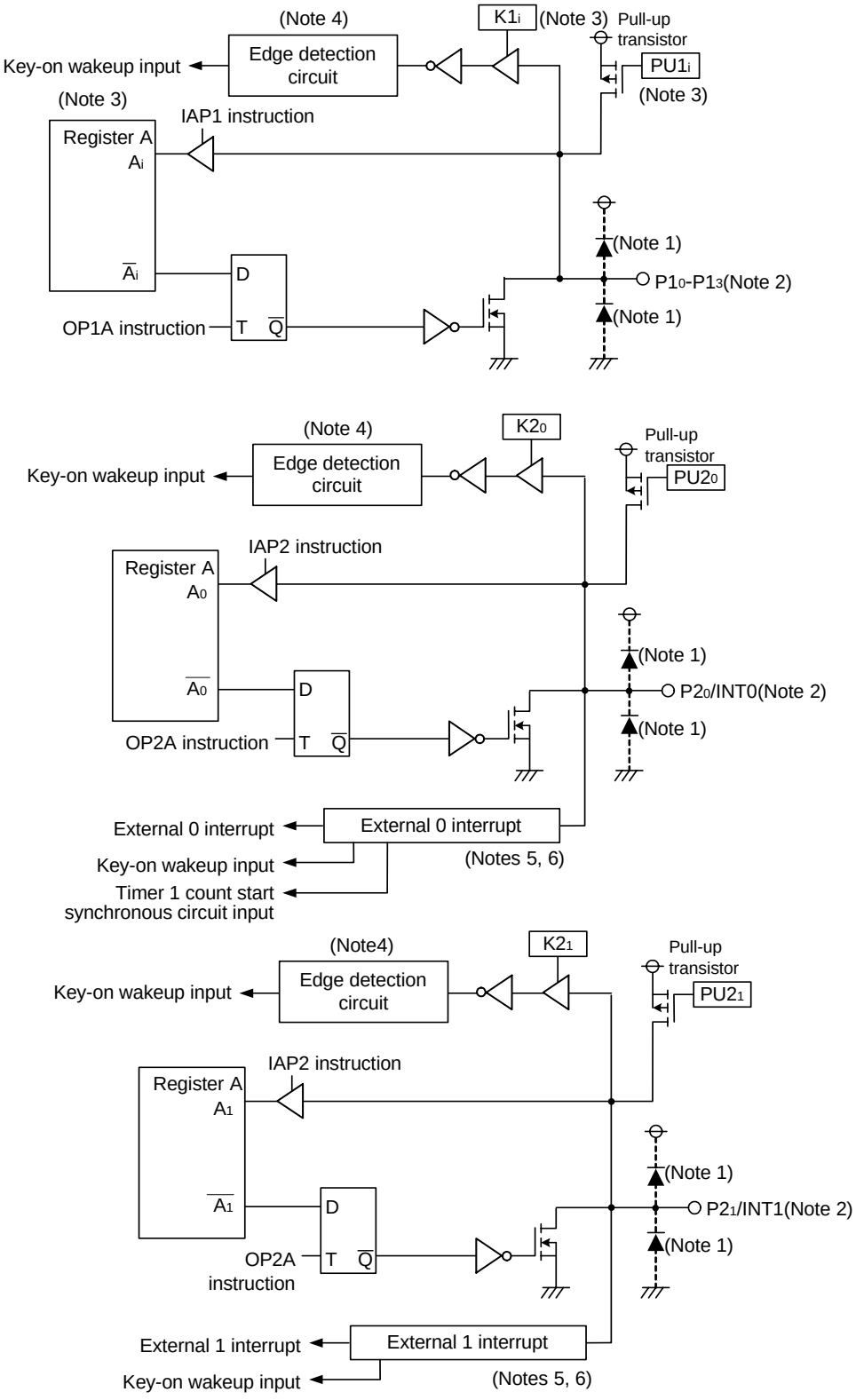
(Note when connecting to V_{SS} or V_{DD})
Connect the unused pins to V_{SS} using the thickest wire at the shortest distance against noise.

PORT BLOCK DIAGRAM



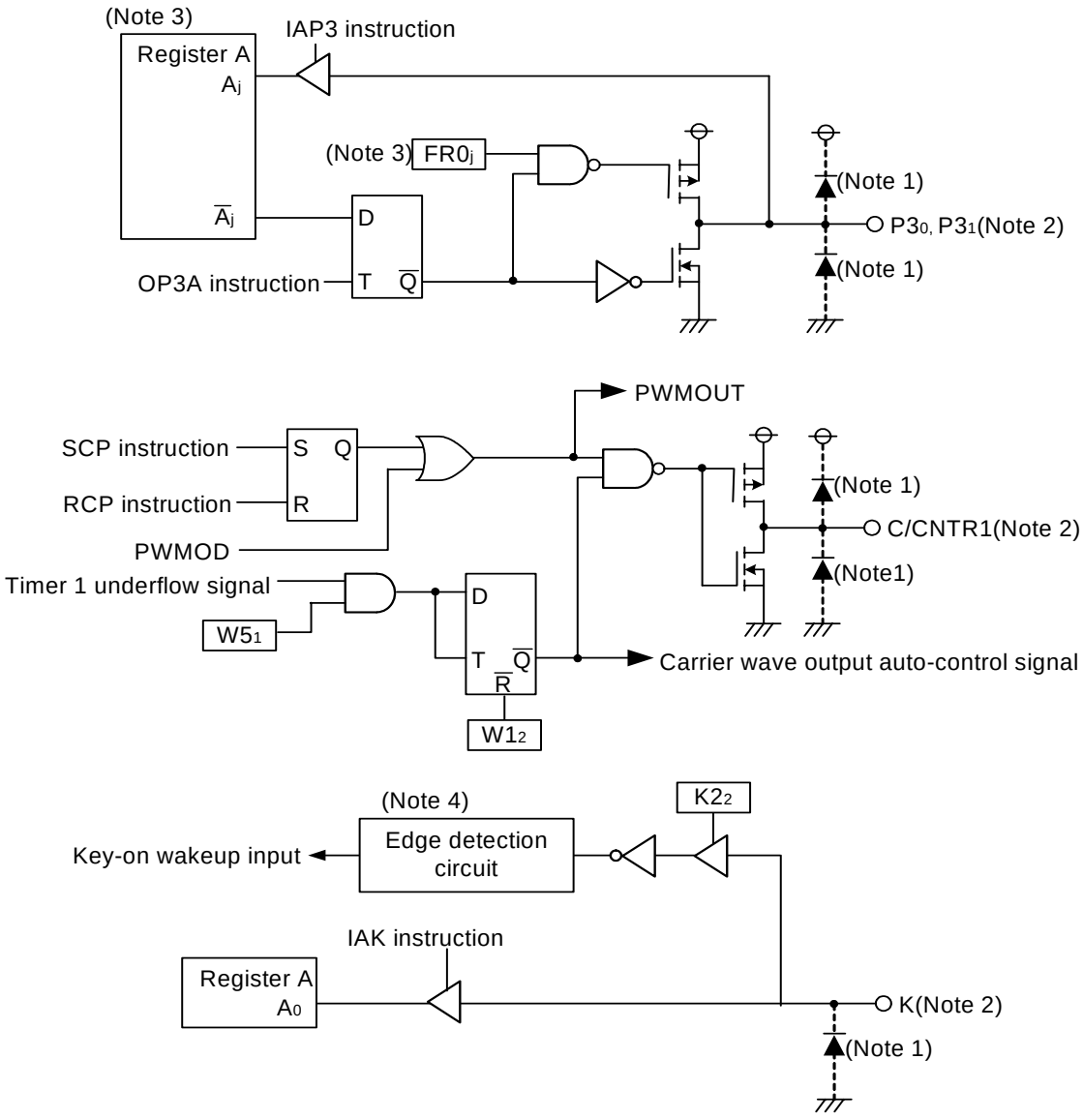
Notes 1. ---|◀--- This symbol represents a parasitic diode on the port.
2. Applied potential to these ports must be V_{DD} or less.
3. i represents bits 0 to 3.
4. A falling edge of port input is detected.

Fig 3. Port block diagram (1)



Notes 1. ---|◀----- This symbol represents a parasitic diode on the port.
2. Applied potential to these ports must be V_{DD} or less.
3. i represents bits 0 to 3.
4. Falling edge of port input is detected.
5. As for details, refer to the external interrupt structure.
6. The threshold value of port input is different from that of external interrupt input.

Fig 4. Port block diagram (2)



Notes 1. ⊣◄---- This symbol represents a parasitic diode on the port.
2. Applied potential to these ports must be V_{DD} or less.
3. j represents bits 0 or 1.
4. Falling edge of port input is detected.

Fig 5. Port block diagram (3)

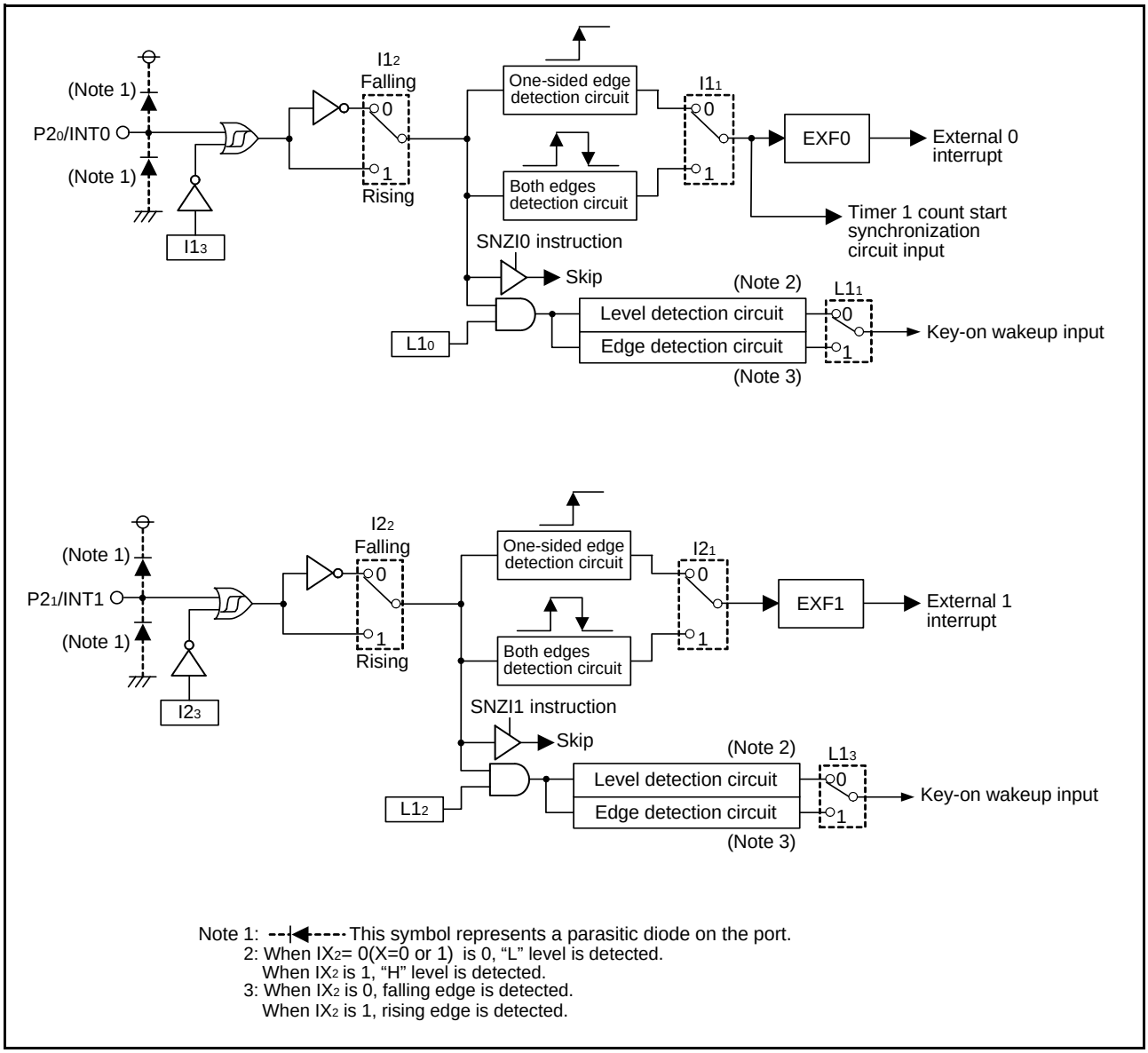


Fig 6. Port block diagram (4)

FUNCTION BLOCK OPERATIONS

CPU

(1) Arithmetic logic unit (ALU)

The arithmetic logic unit ALU performs 4-bit arithmetic such as 4-bit data addition, comparison, AND operation, OR operation, and bit manipulation.

(2) Register A and carry flag

Register A is a 4-bit register used for arithmetic, transfer, exchange, and I/O operation.
Carry flag CY is a 1-bit flag that is set to "1" when there is a carry with the AMC instruction (Figure 7).
It is unchanged with both A n instruction and AM instruction.
The value of A_0 is stored in carry flag CY with the RAR instruction (Figure 8).
Carry flag CY can be set to "1" with the SC instruction and cleared to "0" with the RC instruction.

(3) Registers B and E

Register B is a 4-bit register used for temporary storage of 4-bit data, and for 8-bit data transfer together with register A.
Register E is an 8-bit register. It can be used for 8-bit data transfer with register B used as the high-order 4 bits and register A as the low-order 4 bits (Figure 9).
Register E is undefined after system is released from reset and returned from the RAM back-up. Accordingly, set the initial value.

(4) Register D

Register D is a 3-bit register.
It is used to store a 7-bit ROM address together with register A and is used as a pointer within the specified page when the TABP p, BLA p, or BMLA p instruction is executed (Figure 10).
Also, when the TABP p instruction is executed at UPTF flag = "1", the high-order 2 bits of ROM reference data is stored to the low-order 2 bits of register D, the high-order 1 bit of register D is "0".
When the TABP p instruction is executed at UPTF flag = "0", the contents of register D remains unchanged. The UPTF flag is set to "1" with the SUPT instruction and cleared to "0" with the RUPT instruction.
The initial value of UPTF flag is "0".
Register D is undefined after system is released from reset and returned from the RAM back-up. Accordingly, set the initial value.

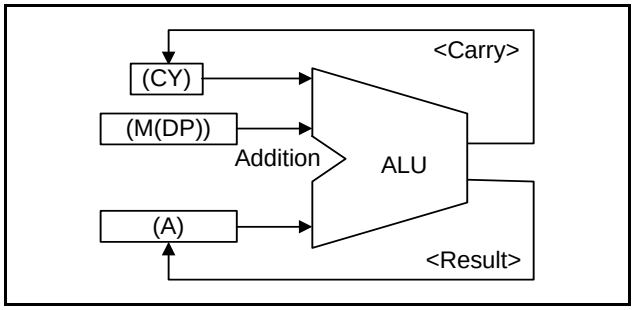


Fig 7. AMC instruction execution example

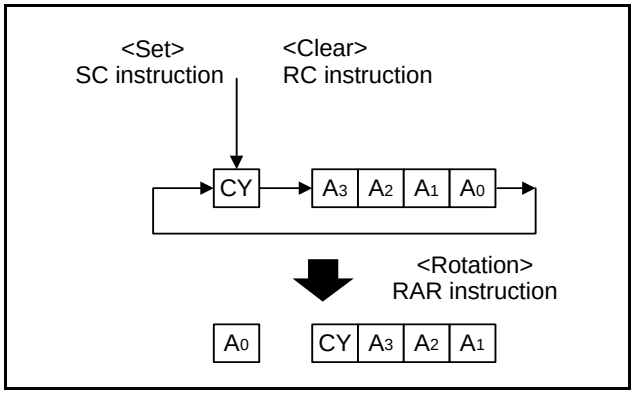


Fig 8. RAR instruction execution example

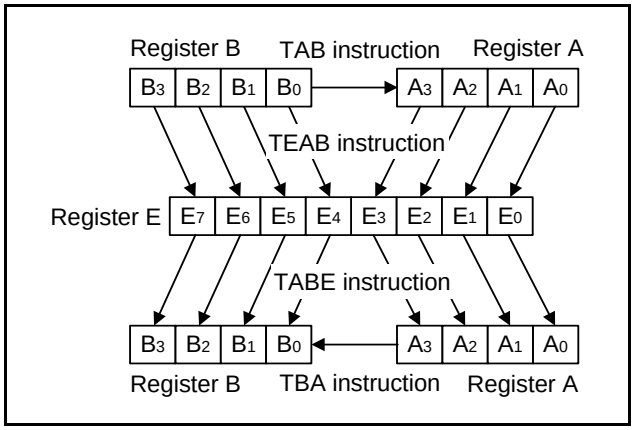


Fig 9. Registers A, B and register E

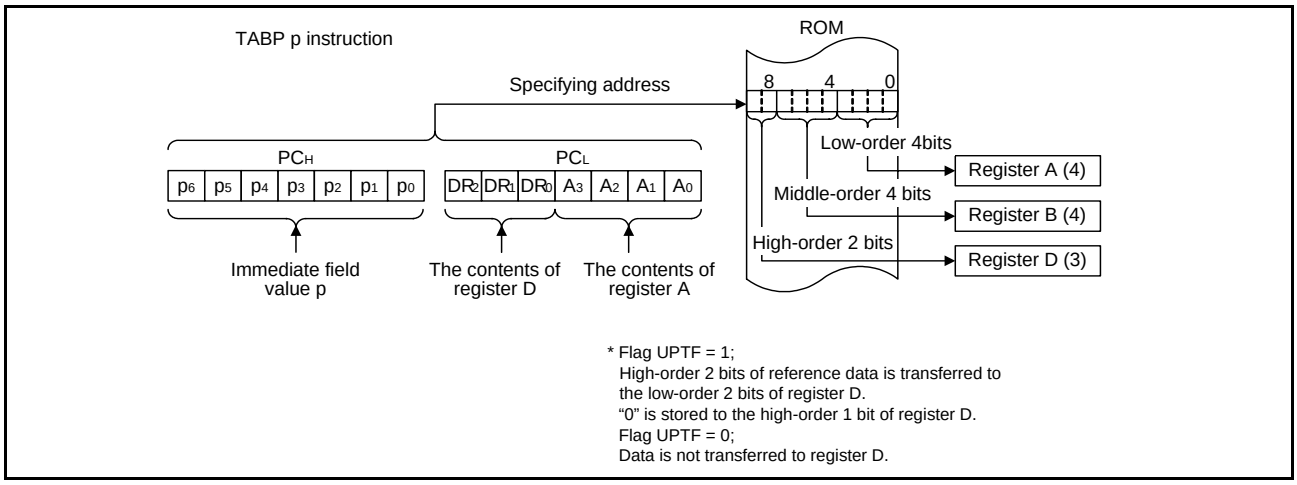


Fig 10. TABP p instruction execution example

(5) Stack registers (SKS) and stack pointer (SP)

Stack registers (SKs) are used to temporarily store the contents of program counter (PC) just before branching until returning to the original routine when;

• branching to an interrupt service routine (referred to as an interrupt service routine),
• performing a subroutine call, or
• executing the table reference instruction (TABP p).

Stack registers (SKs) are eight identical registers, so that subroutines can be nested up to 8 levels. However, one of stack registers is used respectively when using an interrupt service routine and when executing a table reference instruction. Accordingly, be careful not to over the stack when performing these operations together. The contents of registers SKs are destroyed when 8 levels are exceeded.

The register SK nesting level is pointed automatically by 3-bit stack pointer (SP). The contents of the stack pointer (SP) can be transferred to register A with the TASP instruction.

Figure 11 shows the stack registers (SKs) structure.

Figure 12 shows the example of operation at subroutine call.

(6) Interrupt stack register (SDP)

Interrupt stack register (SDP) is a 1-stage register. When an interrupt occurs, this register (SDP) is used to temporarily store the contents of data pointer, carry flag, skip flag, register A, and register B just before an interrupt until returning to the original routine.

Unlike the stack registers (SKs), this register (SDP) is not used when executing the subroutine call instruction and the table reference instruction.

(7) Skip flag

Skip flag controls skip decision for the conditional skip instructions and continuous described skip instructions. When an interrupt occurs, the contents of skip flag is stored automatically in the interrupt stack register (SDP) and the skip condition is retained.

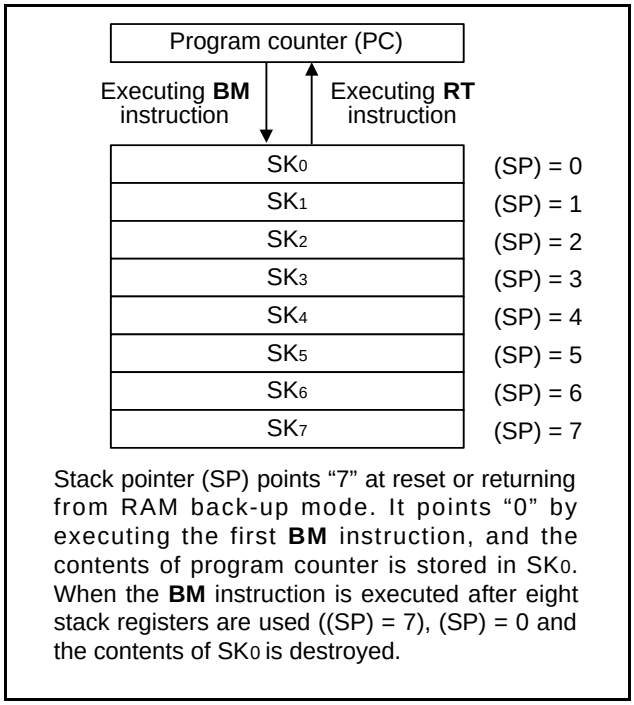


Fig 11. Stack registers (SKs) structure

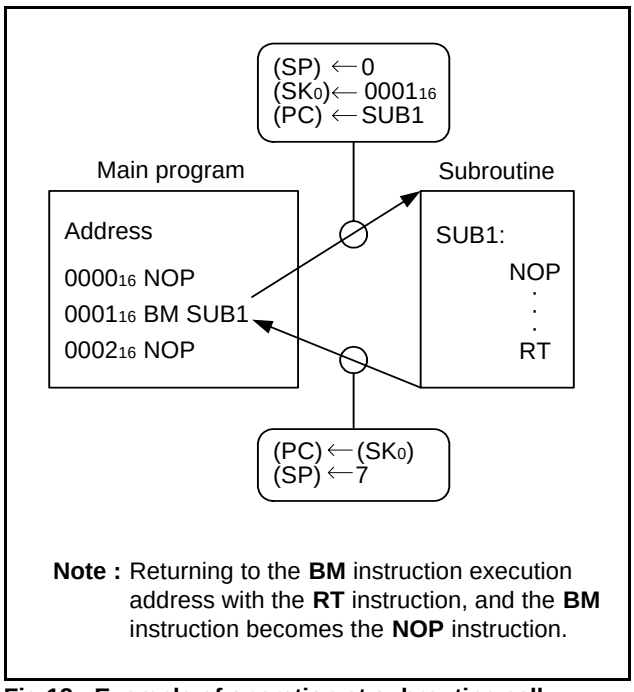


Fig 12. Example of operation at subroutine call

(8) Program counter (PC)

Program counter (PC) is used to specify a ROM address (page and address). It determines a sequence in which instructions stored in ROM are read. It is a binary counter that increments the number of instruction bytes each time an instruction is executed. However, the value changes to a specified address when branch instructions, subroutine call instructions, return instructions, or the table reference instruction (TABP p) is executed.
Program counter consists of PC_H (most significant bit to bit 7) which specifies to a ROM page and PC_L (bits 6 to 0) which specifies an address within a page. After it reaches the last address (address 127) of a page, it specifies address 0 of the next page (Figure 13).
Make sure that the PC_H does not specify after the last page of the built-in ROM.

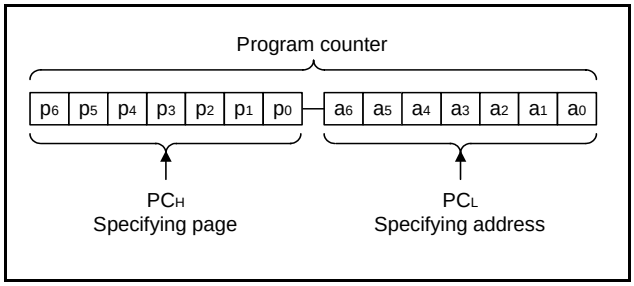


Fig 13. Program counter (PC) structure

(9) Data pointer (DP)

Data pointer (DP) is used to specify a RAM address and consists of registers Z, X, and Y. Register Z specifies a RAM file group, register X specifies a file, and register Y specifies a RAM digit (Figure 14).
Register Y is also used to specify the port D bit position.
When using port D, set the port D bit position to register Y certainly and execute the SD, RD, or SZD instruction (Figure 15).

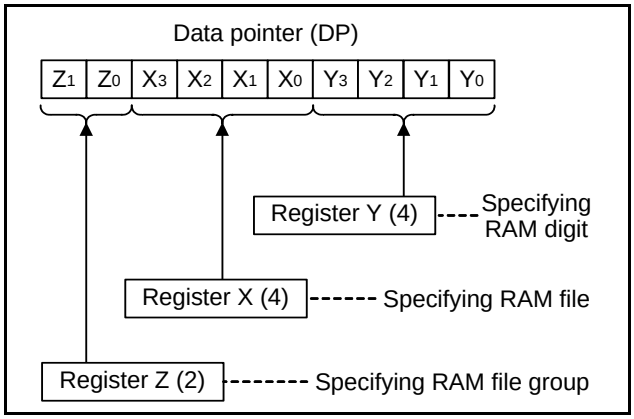


Fig 14. Data pointer (DP) structure

• Note

Register Z of data pointer is undefined after system is released from reset.
Also, registers Z, X and Y are undefined in the RAM back-up. After system is returned from the RAM back-up, set these registers.

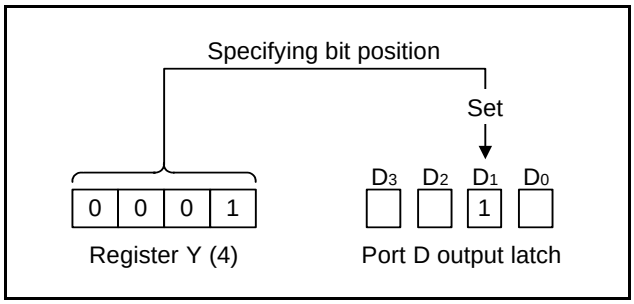


Fig 15. SD instruction execution example

PROGRAM MEMORY (ROM)

The program memory is a mask ROM. 1 word of ROM is composed of 10 bits. ROM is separated every 128 words by the unit of page (addresses 0 to 127). Table 1 shows the ROM size and pages. Figure 16 shows the ROM map of M34571G6.

Table 8 ROM size and pages

Part number	ROM (PROM) size (× 10 bits)	Pages
M34571G4	4096 words	32 (0 to 31)
M34571G6	6144 words	48 (0 to 47)
M34571GD	16384 words	128 (0 to 127)

Note 1. Data in pages 64 to 127 can be referred with the TABP p instruction after the SBK instruction is executed.
Data in pages 0 to 63 can be referred with the TABP p instruction after the RBK instruction is executed.

A part of page 1 (addresses 0080_{16} to $00FF_{16}$) is reserved for interrupt addresses (Figure 17). When an interrupt occurs, the address (interrupt address) corresponding to each interrupt is set in the program counter, and the instruction at the interrupt address is executed. When using an interrupt service routine, write the instruction generating the branch to that routine at an interrupt address.
Page 2 (addresses 0100_{16} to $017F_{16}$) is the special page for subroutine calls. Subroutines written in this page can be called from any page with the 1-word instruction (BM). Subroutines extending from page 2 to another page can also be called with the BM instruction when it starts on page 2.
ROM pattern (bits 9 to 0) of all addresses can be used as data areas with the TABP p instruction.

ROM Code Protect Address

When selecting the protect bit write by using a serial programmer or selecting protect enabled for writing shipment by Renesas Technology corp., reading or writing from/to QzROM is disabled by a serial programmer.
As for the QzROM product in blank, the ROM code is protected by selecting the protect bit write at ROM writing with a serial programmer.
As for the QzROM product shipped after writing, whether the ROM code protect is used or not can be selected as ROM option setup (“MASK option” written in the mask file converter) when ordering.

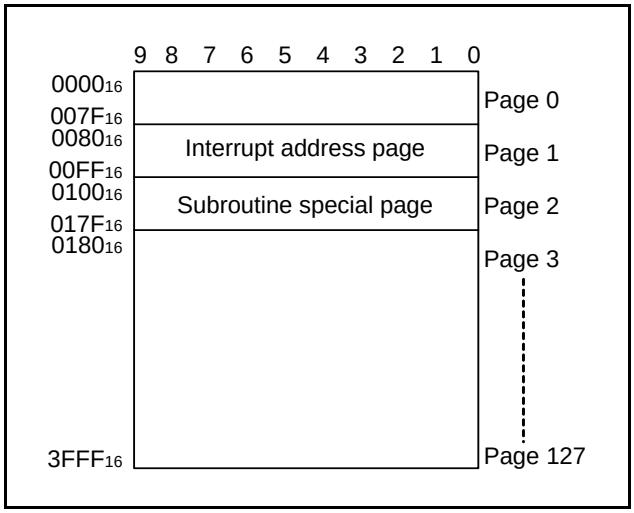


Fig 16. ROM map of M34571GD

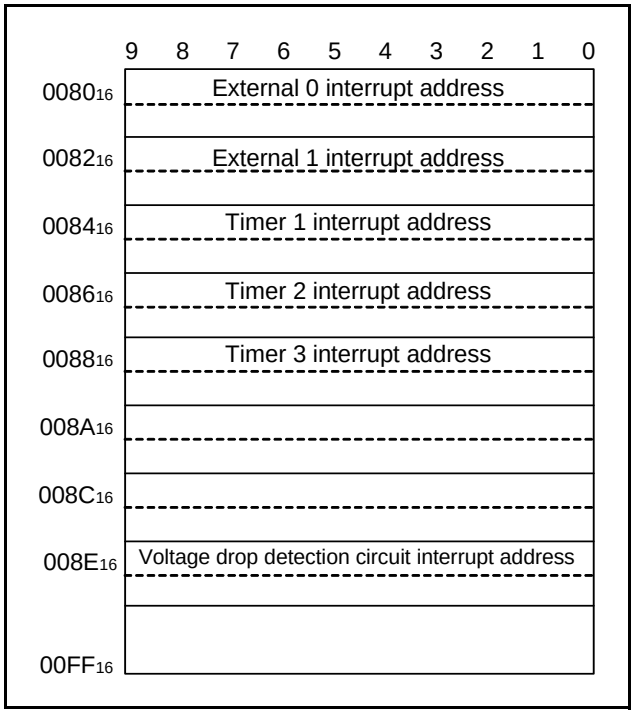


Fig 17. Page 1 (addresses 0080_{16} to $00FF_{16}$) structure

DATA MEMORY (RAM)

1 word of RAM is composed of 4 bits, but 1-bit manipulation (with the SB j, RB j, and SZB j instructions) is enabled for the entire memory area. A RAM address is specified by a data pointer. The data pointer consists of registers Z, X, and Y. Set a value to the data pointer certainly when executing an instruction to access RAM (also, set a value after system returns from RAM back-up).
Table 9 shows the RAM size. Figure 18 shows the RAM map.

• Note
Register Z of data pointer is undefined after system is released from reset.
Also, registers Z, X and Y are undefined in the RAM back-up. After system is returned from the RAM back-up, set these registers.

Table 9 RAM size and pages

Part number	RAM size
M34571G4	128 words × 4 bits (512 bits)
M34571G6	
M34571GD	

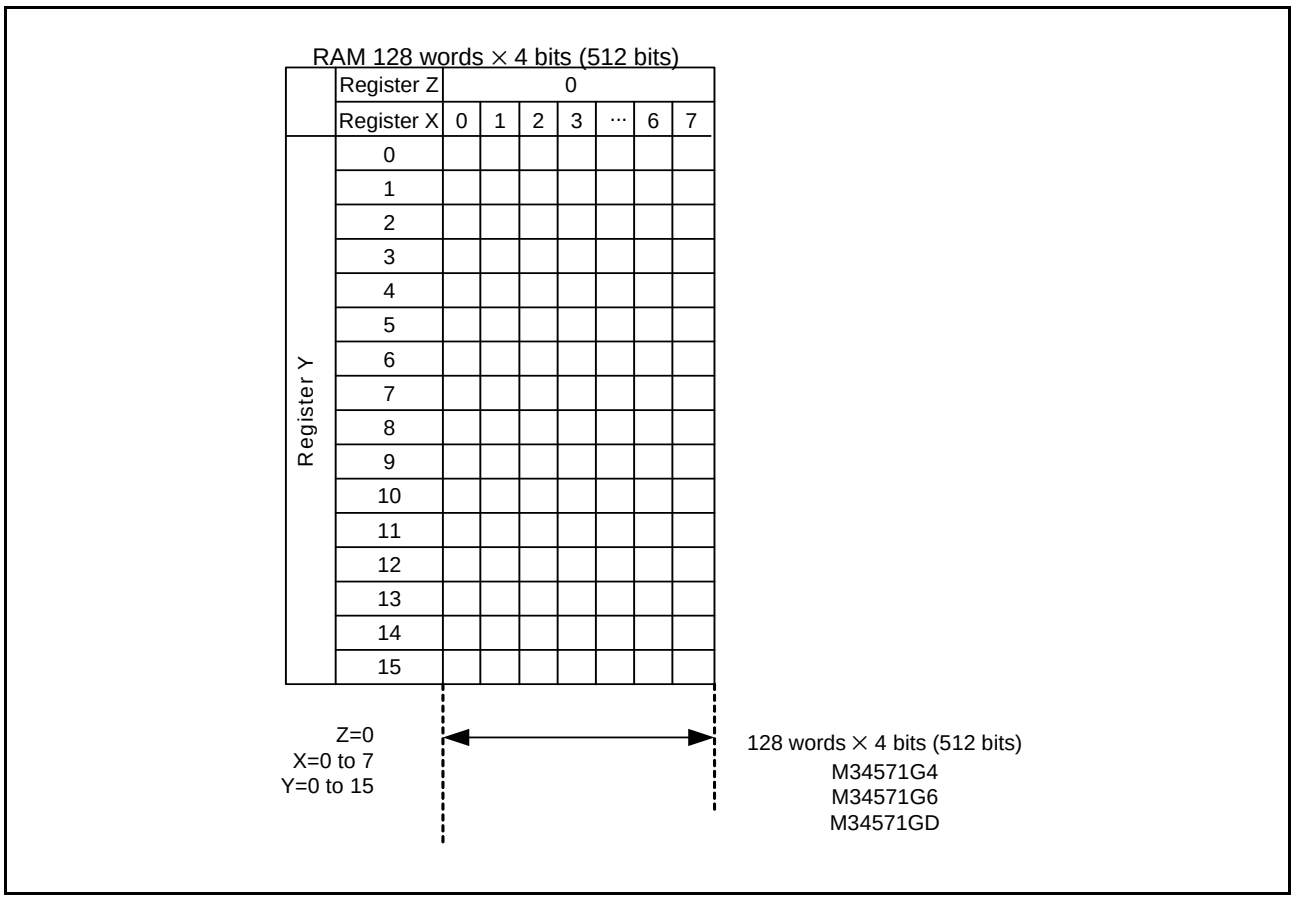


Fig 18. RAM map

INTERRUPT FUNCTION

The interrupt type is a vectored interrupt branching to an individual address (interrupt address) according to each interrupt source. An interrupt occurs when the following 3 conditions are satisfied.

- An interrupt activated condition is satisfied (request flag = "1")
- Interrupt enable bit is enabled ("1")
- Interrupt enable flag is enabled (INTE = "1")

Table 10 shows interrupt sources. (Refer to each interrupt request flag for details of activated conditions.)

(1) Interrupt enable flag (INTE)

The interrupt enable flag (INTE) controls whether the every interrupt enable/disable. Interrupts are enabled when INTE flag is set to "1" with the EI instruction and disabled when INTE flag is cleared to "0" with the DI instruction. When any interrupt occurs, the INTE flag is automatically cleared to "0," so that other interrupts are disabled until the EI instruction is executed.

(2) Interrupt enable bit

Use an interrupt enable bit of interrupt control registers V1 and V2 to select the corresponding interrupt or skip instruction.
Table 11 shows the interrupt request flag, interrupt enable bit and skip instruction.
Table 12 shows the interrupt enable bit function.

(3) Interrupt request flag

When the activated condition for each interrupt is satisfied, the corresponding interrupt request flag is set to "1." Each interrupt request flag except the voltage drop detection circuit interrupt request flag is cleared to "0" when either;

- an interrupt occurs, or
- the next instruction is skipped with a skip instruction.

The voltage drop detection circuit interrupt request flag cannot be cleared to "0" at the state that the activated condition is satisfied.
Each interrupt request flag is set when the activated condition is satisfied even if the interrupt is disabled by the INTE flag or its interrupt enable bit. Once set, the interrupt request flag retains set until a clear condition is satisfied.
Accordingly, an interrupt occurs when the interrupt disable state is released while the interrupt request flag is set.
If more than one interrupt request flag is set when the interrupt disable state is released, the interrupt priority level is as follows shown in Table 10.

Table 10 Interrupt sources

Priority level	Interrupt source		Interrupt address
	Interrupt name	Activated condition	
1	Voltage drop detection circuit interrupt	when supply voltage goes lower than specified value	Address E in page 1
2	External 0 interrupt	Level change of INT0 pin	Address 0 in page 1
3	External 1 interrupt	Level change of INT1 pin	Address 2 in page 1
4	Timer 1 interrupt	Timer 1 underflow	Address 4 in page 1
5	Timer 2 interrupt	Timer 2 underflow	Address 6 in page 1
6	Timer 3 interrupt	Timer 3 underflow	Address 8 in page 1

Table 11 Interrupt request flag, interrupt enable bit and skip instruction

Interrupt name	Interrupt request flag	Skip instruction	Interrupt enable bit
Voltage drop detection circuit interrupt	VDF	SNZVD	$V2_3$
External 0 interrupt	EXF0	SNZ0	$V1_0$
External 1 interrupt	EXF1	SNZ1	$V1_1$
Timer 1 interrupt	T1F	SNZT1	$V1_2$
Timer 2 interrupt	T2F	SNZT2	$V1_3$
Timer 3 interrupt	T3F	SNZT3	$V2_0$

Table 12 Interrupt enable bit function

Interrupt enable bit	Occurrence of interrupt	Skip instruction
1	Enabled	Invalid
0	Disabled	Valid

(4) Internal state during an interrupt

The internal state of the microcomputer during an interrupt is as follows (Figure 20).

- Program counter (PC)
  An interrupt address is set in program counter. The address to be executed when returning to the main routine is automatically stored in the stack register (SK).
- Interrupt enable flag (INTE)
  INTE flag is cleared to "0" so that interrupts are disabled.
- Interrupt request flag
  Only the request flag for the current interrupt source is cleared to "0" (the voltage drop detection circuit interrupt request flag is excluded)
- Data pointer, carry flag, skip flag, registers A and B
  The contents of these registers and flags are stored automatically in the interrupt stack register (SDP).

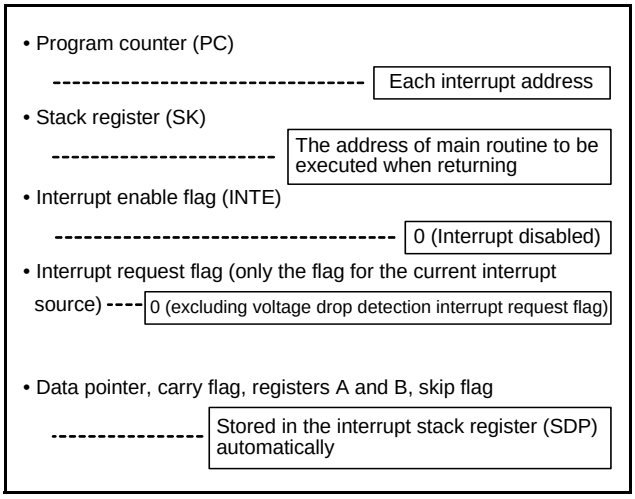


Fig 20. Internal state when interrupt occurs

(5) Interrupt processing

When an interrupt occurs, a program at an interrupt address is executed after branching a data store sequence to stack register. Write the branch instruction to an interrupt service routine at an interrupt address. Use the RTI instruction to return from an interrupt service routine.
Interrupt enabled by executing the EI instruction is performed after executing 1 instruction (just after the next instruction is executed). Accordingly, when the EI instruction is executed just before the RTI instruction, interrupts are enabled after returning the main routine. (Refer to Figure 19)

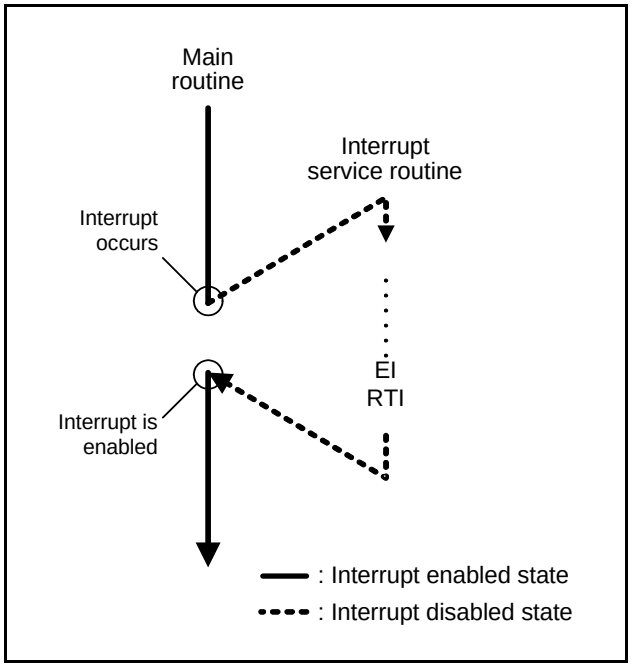


Fig 19. Program example of interrupt processing

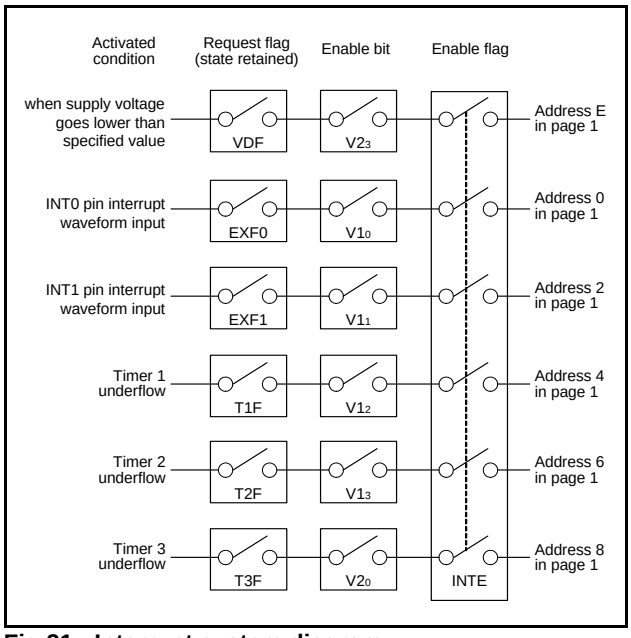


Fig 21. Interrupt system diagram

(6) Interrupt control registers

• Interrupt control register V1

Interrupt enable bits of external 0, external 1, timer 1 and timer 2 are assigned to register V1. Set the contents of this register through register A with the TV1A instruction. The TAV1 instruction can be used to transfer the contents of register V1 to register A.

• Interrupt control register V2

The voltage drop detection circuit interrupt enable bit and timer 3 interrupt enable bit are assigned to register V2. Set the contents of this register through register A with the TV2A instruction. The TAV2 instruction can be used to transfer the contents of register V2 to register A.

Table 13 Interrupt control registers

Interrupt control register V1		at reset : 0000_2	at RAM back-up : 0000_2	R/W TAV1/TV1A
$V1_3$	Timer 2 interrupt enable bit	0	Interrupt disabled (SNZT2 instruction is valid)	
		1	Interrupt enabled (SNZT2 instruction is invalid)	
$V1_2$	Timer 1 interrupt enable bit	0	Interrupt disabled (SNZT1 instruction is valid)	
		1	Interrupt enabled (SNZT1 instruction is invalid)	
$V1_1$	External 1 interrupt enable bit	0	Interrupt disabled (SNZ1 instruction is valid)	
		1	Interrupt enabled (SNZ1 instruction is invalid)	
$V1_0$	External 0 interrupt enable bit	0	Interrupt disabled (SNZ0 instruction is valid)	
		1	Interrupt enabled (SNZ0 instruction is invalid)	

Interrupt control register V2		at reset : 0000_2	at RAM back-up : 0000_2	R/W TAV2/TV2A
$V2_3$	Voltage drop detector interrupt enable bit	0	Interrupt disabled (SNZVD instruction is valid)	
		1	Interrupt enabled (SNZVD instruction is invalid)	
$V2_2$	Not used	0	This bit has no function, but read/write is enabled.	
		1		
$V2_1$	Not used	0	This bit has no function, but read/write is enabled.	
		1		
$V2_0$	Timer 3 interrupt enable bit	0	Interrupt disabled (SNZT3 instruction is valid)	
		1	Interrupt enabled (SNZT3 instruction is invalid)	

Note 1."R" represents read enabled, and "W" represents write enabled.

(7) Interrupt sequence

Interrupts only occur when the respective INTE flag, interrupt enable bits ($V1_0$–$V1_3$, $V2_0$, $V2_3$), and interrupt request flag are "1." The interrupt actually occurs 2 to 3 machine cycles after the cycle in which all three conditions are satisfied. The interrupt occurs after 3 machine cycles only when the three interrupt conditions are satisfied on execution of other than one-cycle instructions (Refer to Figure 22).

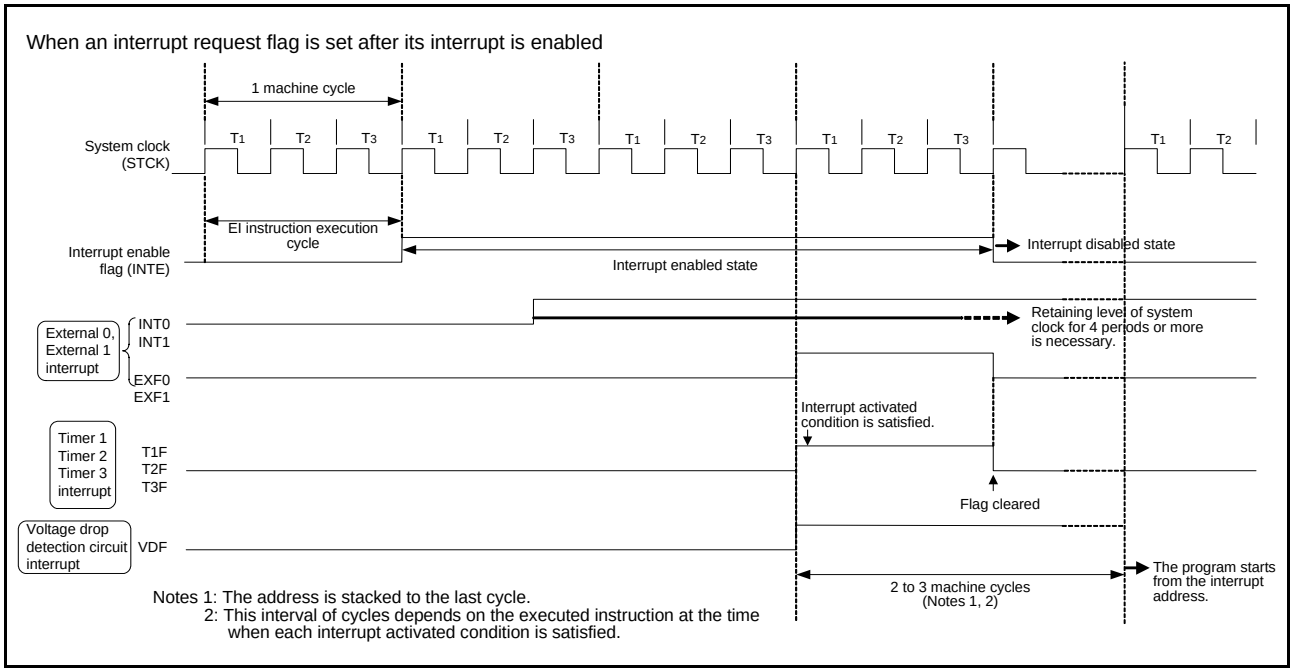


Fig 22. Interrupt sequence

EXTERNAL INTERRUPTS

The 4571 Group has the external 0 interrupt and external 1 interrupt. An external interrupt request occurs when a valid waveform is input to an interrupt input pin (edge detection).
The external interrupt can be controlled with the interrupt control registers I1 and I2.

Table 14 External interrupt activated conditions

Name	Input pin	Activated condition	Valid waveform selection bit
External 0 interrupt	$P2_0$/INT0	When the next waveform is input to $P2_0$/INT0 pin • Falling waveform ("H" → "L") • Rising waveform ("L" → "H") • Both rising and falling waveforms	$I1_1$ $I1_2$
External 1 interrupt	$P2_1$/INT1	When the next waveform is input to $P2_1$/INT1 pin • Falling waveform ("H" → "L") • Rising waveform ("L" → "H") • Both rising and falling waveforms	$I2_1$ $I2_2$

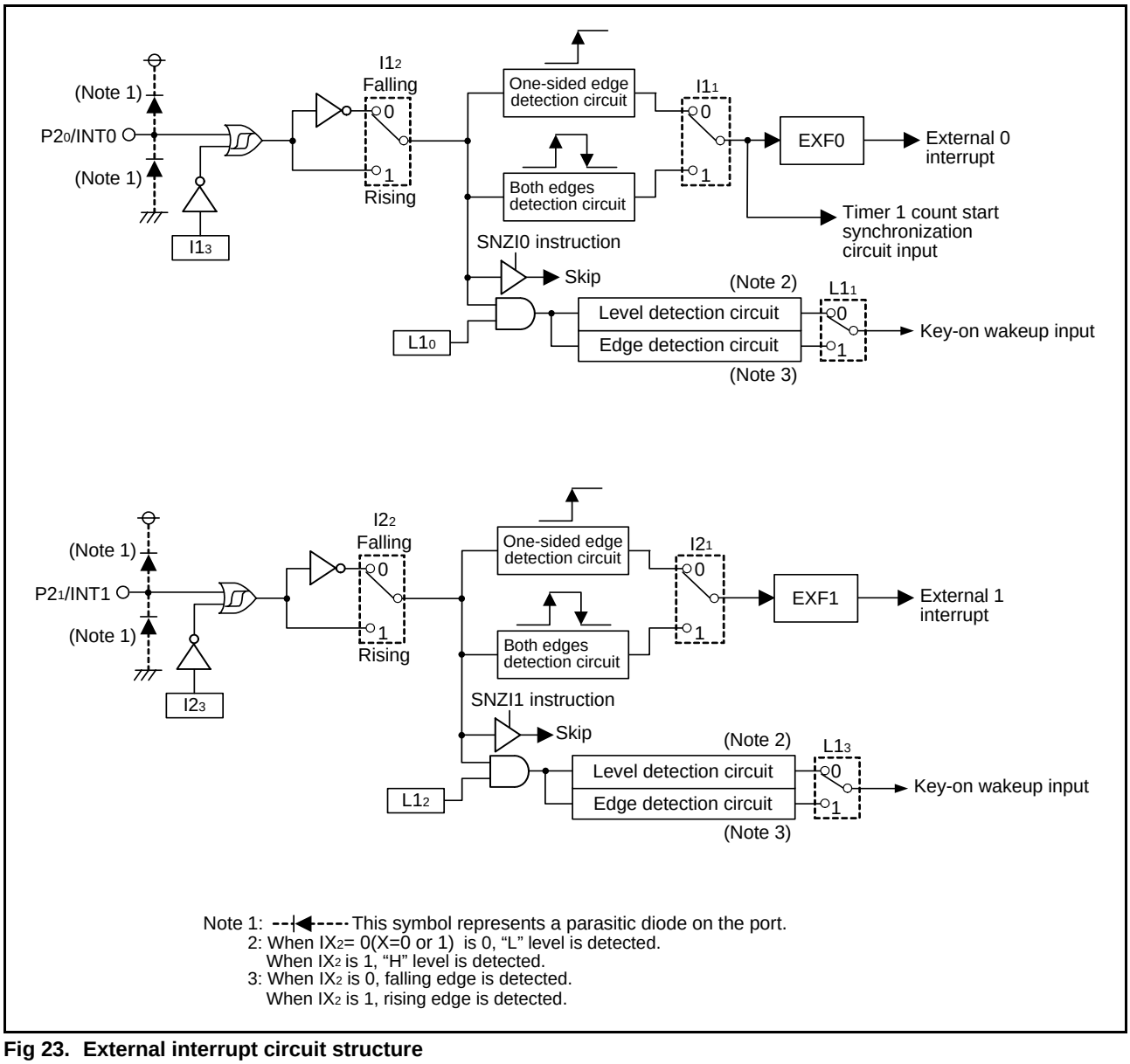


Fig 23. External interrupt circuit structure

(1) External 0 interrupt request flag (EXF0)

External 0 interrupt request flag (EXF0) is set to "1" when a valid waveform is input to P2$_0$/INT0 pin.
The valid waveforms causing the interrupt must be retained at their level for 4 clock cycles or more of the system clock (Refer to Figure 22).
The state of EXF0 flag can be examined with the skip instruction (SNZ0). Use the interrupt control register V1 to select the interrupt or the skip instruction. The EXF0 flag is cleared to "0" when an interrupt occurs or when the next instruction is skipped with the skip instruction.

- External 0 interrupt activated condition

  External 0 interrupt activated condition is satisfied when a valid waveform is input to P2$_0$/INT0 pin.

  The valid waveform can be selected from rising waveform, falling waveform or both rising and falling waveforms. An example of how to use the external 0 interrupt is as follows.

(1) Set the bit 3 of register I1 to "1" for the INT0 pin to be in the input enabled state.
(2) Select the valid waveform with the bits 1 and 2 of register I1.
(3) Clear the EXF0 flag to "0" with the SNZ0 instruction.
(4) Set the NOP instruction for the case when a skip is performed with the SNZ0 instruction.
(5) Set both the external 0 interrupt enable bit (V1$_0$) and the INTE flag to "1."

The external 0 interrupt is now enabled. Now when a valid waveform is input to the P2$_0$/INT0 pin, the EXF0 flag is set to "1" and the external 0 interrupt occurs.

(2) External 1 interrupt request flag (EXF1)

External 1 interrupt request flag (EXF1) is set to "1" when a valid waveform is input to P2$_1$/INT1 pin.
The valid waveforms causing the interrupt must be retained at their level for 4 clock cycles or more of the system clock (Refer to Figure 22).
The state of EXF1 flag can be examined with the skip instruction (SNZ1). Use the interrupt control register V1 to select the interrupt or the skip instruction. The EXF1 flag is cleared to "0" when an interrupt occurs or when the next instruction is skipped with the skip instruction.

- External 1 interrupt activated condition

  External 1 interrupt activated condition is satisfied when a valid waveform is input to P2$_1$/INT1 pin.

  The valid waveform can be selected from rising waveform, falling waveform or both rising and falling waveforms. An example of how to use the external 1 interrupt is as follows.

(1) Set the bit 3 of register I2 to "1" for the INT1 pin to be in the input enabled state.
(2) Select the valid waveform with the bits 1 and 2 of register I2.
(3) Clear the EXF1 flag to "0" with the SNZ1 instruction.
(4) Set the NOP instruction for the case when a skip is performed with the SNZ1 instruction.
(5) Set both the external 1 interrupt enable bit (V1$_1$) and the INTE flag to "1."

The external 1 interrupt is now enabled. Now when a valid waveform is input to the P2$_1$/INT1 pin, the EXF1 flag is set to "1" and the external 1 interrupt occurs.

(3) External interrupt control registers

(1) Interrupt control register I1
Register I1 controls the valid waveform for the external 0 interrupt. Set the contents of this register through register A with the TI1A instruction. The TAI1 instruction can be used to transfer the contents of register I1 to register A.

(2) Interrupt control register I2
Register I2 controls the valid waveform for the external 1 interrupt. Set the contents of this register through register A with the TI2A instruction. The TAI2 instruction can be used to transfer the contents of register I2 to register A.

Table 15 External interrupt control register

Interrupt control register I1		at reset : 0000_2	at RAM back-up : state retained	R/W TAI1/TI1A
$I1_3$	INT0 pin input control bit (Note 2)	0	INT0 pin input disabled	
		1	INT0 pin input enabled	
$I1_2$	Interrupt valid waveform for INT0 pin/ return level selection bit (Note 2)	0	Falling waveform (“L” level of INT0 pin is recognized with the SNZI0 instruction)/“L” level	
		1	Rising waveform (“H” level of INT0 pin is recognized with the SNZI0 instruction)/“H” level	
$I1_1$	INT0 pin edge detection circuit control bit	0	One-sided edge detected	
		1	Both edges detected	
$I1_0$	INT0 pin timer 1 control enable bit	0	Timer 1 disabled	
		1	Timer 1 enabled	

Interrupt control register I2		at reset : 0000_2	at RAM back-up : state retained	R/W TAI2/TI2A
$I2_3$	INT1 pin input control bit (Note 3)	0	INT0 pin input disabled	
		1	INT0 pin input enabled	
$I2_2$	Interrupt valid waveform for INT1 pin/ return level selection bit (Note 3)	0	Falling waveform (“L” level of INT0 pin is recognized with the SNZI1 instruction)/“L” level	
		1	Rising waveform (“H” level of INT0 pin is recognized with the SNZI1 instruction)/“H” level	
$I2_1$	INT1 pin edge detection circuit control bit	0	One-sided edge detected	
		1	Both edges detected	
$I2_0$	Not used	0	This bit has no function, but read/write is enabled.	
		1		

Note 1.“R” represents read enabled, and “W” represents write enabled.
Note 2.When the contents of $I1_2$ and $I1_3$ are changed, the external interrupt request flag EXF0 may be set.
Note 3.When the contents of $I2_2$ and $I2_3$ are changed, the external interrupt request flag EXF1 may be set.

(4) Notes on interrupts

(1) Bit 3 of register I1

When the input of the P20/INT0 pin is controlled with the bit 3 of register I1 in software, be careful about the following notes.

- Depending on the input state of the P20/INT0 pin, the external 0 interrupt request flag (EXF0) may be set when the bit 3 of register I1 is changed. In order to avoid the occurrence of an unexpected interrupt, clear the bit 0 of register V1 to "0" (refer to (1) in Figure 24) and then, change the bit 3 of register I1.
  In addition, execute the SNZ0 instruction to clear the EXF0 flag to "0" after executing at least one instruction (refer to (2) in Figure 24).
  Also, set the NOP instruction for the case when a skip is performed with the SNZ0 instruction (refer to (3) in Figure 24).

```
    ⋮
LA   4        ; (×××02)
TV1A          ; The SNZ0 instruction is valid ..... (1)
LA   8        ; (1×××2)
TI1A          ; Control of INT0 pin input is changed
NOP           ..................................................... (2)
SNZ0          ; The SNZ0 instruction is executed
                (EXF0 flag cleared)
NOP           ..................................................... (3)
    ⋮

×: these bits are not used here.
```

Fig 24. External 0 interrupt program example-1

(2) Bit 3 of register I1

When the bit 3 of register I1 is cleared to "0", the RAM back-up mode is selected and the input of INT0 pin is disabled, be careful about the following notes.

- When the INT0 pin input is disabled (register I13 = "0"), set the key-on wakeup of INT0 pin to be invalid (register L10 = "0") before system enters to the RAM back-up mode. (refer to (1) in Figure 25).

```
    ⋮
LA   0        ; (×××02)
TL1A          ; INT0 key-on wakeup disabled .....(1)
DI
EPOF
POF           ; RAM back-up
    ⋮

×: these bits are not used here.
```

Fig 25. External 0 interrupt program example-2

(3) Bit 2 of register I1

When the interrupt valid waveform of the P20/INT0 pin is changed with the bit 2 of register I1 in software, be careful about the following notes.

- Depending on the input state of the P20/INT0 pin, the external 1 interrupt request flag (EXF0) may be set when the bit 2 of register I1 is changed. In order to avoid the occurrence of an unexpected interrupt, clear the bit 0 of register V1 to "0" (refer to (1) in Figure 26) and then, change the bit 2 of register I1 is changed.
  In addition, execute the SNZ0 instruction to clear the EXF0 flag to "0" after executing at least one instruction (refer to (2) in Figure 26).
  Also, set the NOP instruction for the case when a skip is performed with the SNZ0 instruction (refer to (3) in Figure 26).

```
    ⋮
LA   4        ; (×××02)
TV1A          ; The SNZ0 instruction is valid  .....(1)
LA   12       ; (1×××2)
TI1A          ; Interrupt valid waveform is changed
NOP           .....(2)
SNZ0          ; The SNZ0 instruction is executed
                (EXF0 flag cleared)
NOP           .....(3)
    ⋮

×: these bits are not used here.
```

Fig 26. External 0 interrupt program example-3

(4) Bit 3 of register I2

When the input of the P21/INT1 pin is controlled with the bit 3 of register I2 in software, be careful about the following notes.

- Depending on the input state of the P21/INT1 pin, the external 1 interrupt request flag (EXF1) may be set when the bit 3 of register I2 is changed. In order to avoid the occurrence of an unexpected interrupt, clear the bit 1 of register V1 to "0" (refer to (1) in Figure 27) and then, change the bit 3 of register I2.
  In addition, execute the SNZ1 instruction to clear the EXF1 flag to "0" after executing at least one instruction (refer to (2) in Figure 27).
  Also, set the NOP instruction for the case when a skip is performed with the SNZ1 instruction (refer to (3) in Figure 27).

```
      :
      :
LA    4         ; (××0×2)
TV1A            ; The SNZ1 instruction is valid ..... (1)
LA    8         ; (1×××2)
TI1A            ; Control of INT1 pin input is changed
NOP             ..... (2)
SNZ0            ; The SNZ1 instruction is executed
                  (EXF1 flag cleared)
NOP             ..... (3)
      :
      :

×: these bits are not used here.
```

Fig 27. External 1 interrupt program example-1

(5) Bit 3 of register I2

When the bit 3 of register I2 is cleared to "0", the RAM back-up mode is selected and the input of INT1 pin is disabled, be careful about the following notes.

- When the INT1 pin input is disabled (register I23 = "0"), set the key-on wakeup of INT1 pin to be invalid (register L20 = "0") before system enters to the RAM back-up mode. (refer to (1) in Figure 28)

.

```
      :
      :
LA    0         ; (×0××2)
TL1A            ; INT1 key-on wakeup disabled .....(1)
DI
EPOF
POF             ; RAM back-up
      :
      :

×: these bits are not used here.
```

Fig 28. External 1 interrupt program example-2

(6) Bit 2 of register I2

When the interrupt valid waveform of the P21/INT1 pin is changed with the bit 2 of register I2 in software, be careful about the following notes.

- Depending on the input state of the P21/INT1 pin, the external 1 interrupt request flag (EXF1) may be set when the bit 2 of register I2 is changed. In order to avoid the occurrence of an unexpected interrupt, clear the bit 1 of register V1 to "0" (refer to (1) in Figure 29) and then, change the bit 2 of register I2 is changed.
  In addition, execute the SNZ1 instruction to clear the EXF1 flag to "0" after executing at least one instruction (refer to (2) in Figure 29).
  Also, set the NOP instruction for the case when a skip is performed with the SNZ1 instruction (refer to (3) in Figure 29).

```
      :
      :
LA    4         ; (××0×2)
TV1A            ; The SNZ1 instruction is valid  .....(1)
LA    12        ; (1×××2)
TI1A            ; Interrupt valid waveform is changed
NOP             .....(2)
SNZ0            ; The SNZ1 instruction is executed
                  (EXF1 flag cleared)
NOP             .....(3)
      :
      :

×: these bits are not used here.
```

Fig 29. External 1 interrupt program example-3

TIMERS

The 4571 Group has the following timers.

- Programmable timer
  The programmable timer has a reload register and enables the frequency dividing ratio to be set. It is decremented from a setting value n. When it underflows (count to n + 1), a timer interrupt request flag is set to "1," new data is loaded from the reload register, and count continues (auto-reload function).
- Fixed dividing frequency timer
  The fixed dividing frequency timer has the fixed frequency dividing ratio (n). An interrupt request flag is set to "1" after every n count of a count pulse.

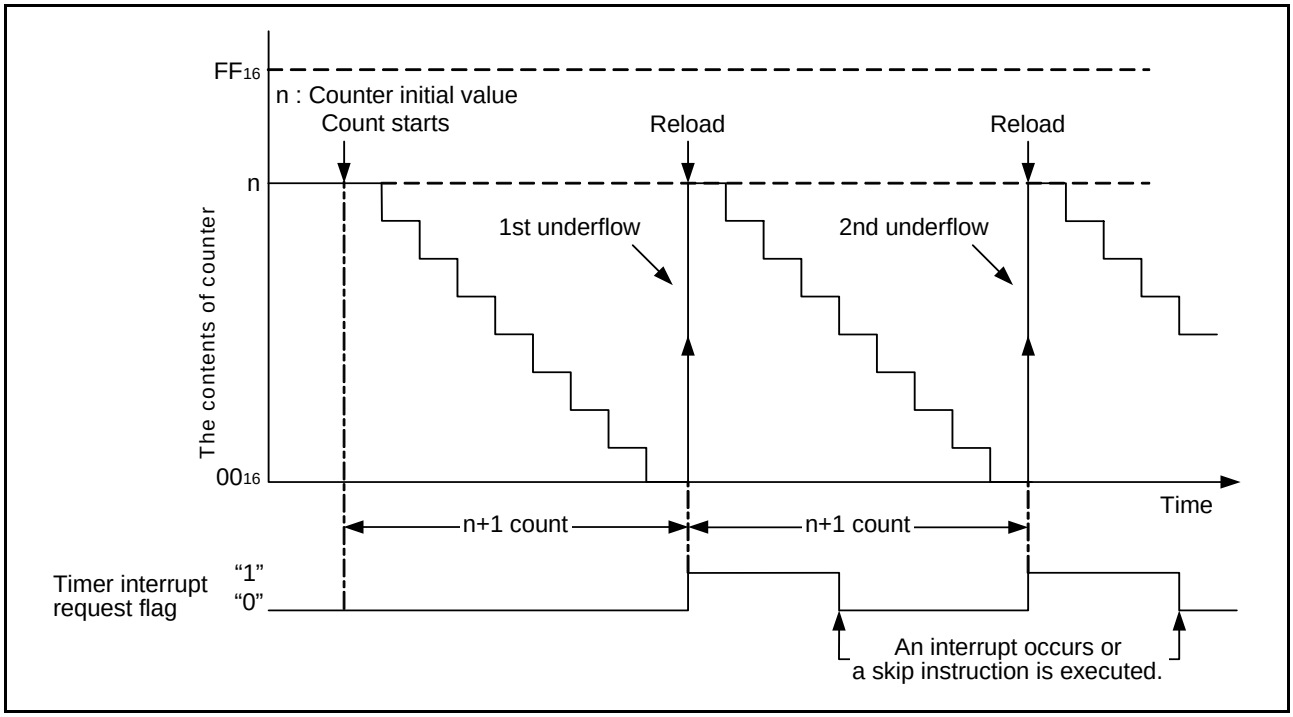


Fig 30. Auto-reload function

The 4571 Group timer consists of the following circuits.
- Prescaler : 8-bit programmable timer
- Timer 1 : 8-bit programmable timer
- Timer 2 : 8-bit programmable timer
- Timer 3 : 8-bit programmable timer
- Watchdog timer: 16-bit fixed frequency timer

(Timers 1, 2 and 3 have the interrupt function, respectively)

Prescaler, timer 1, timer 2 and timer 3 can be controlled with the timer control registers PA, W1, W2, W3 and W5. The watchdog timer is a free counter which is not controlled with the control register.
Each function is described below.

Table 16 Function related timers

Circuit	Structure	Count source	Frequency dividing ratio	Use of output signal	Control register
Prescaler	8-bit programmable binary down counter	• Instruction clock (INSTCK) • Instruction clock divided by 4 (INSTCK/4)	1 to 256	• Timer 1 count source • Timer 2 count source • Timer 3 count source	PA
Timer 1	8-bit programmable binary down counter (link to INT0 input) (carrier wave output auto-control function)	• PWM signal (PWMOUT) • Prescaler output (ORCLK) • CNTR0 input (CNTR0IN) • System clock (STCK)	1 to 256	• Timer 2 count source • CNTR0 output • Carrier wave output auto-control • Timer 1 interrupt	W1 W5
Timer 2	8-bit programmable binary down counter	• PWM signal (PWMOUT) • Timer 1 underflow (T1UDF) • Prescaler output (ORCLK) • System clock (STCK)	1 to 256	• CNTR0 output • Timer 2 interrupt	W2 W5
Timer 3	8-bit programmable binary down counter (with carrier wave generation function)	• X_{IN} input • Prescaler output divided by 2 (ORCLK/2)	1 to 256	• Timer 1 count source • Timer 2 count source • CNTR1 output • Timer 3 interrupt	W1 W3 W5
Watchdog timer	16-bit fixed dividing frequency	• Instruction clock (INSTCK)	65536	• System reset (counting twice) • Decision of flag WDF1	-

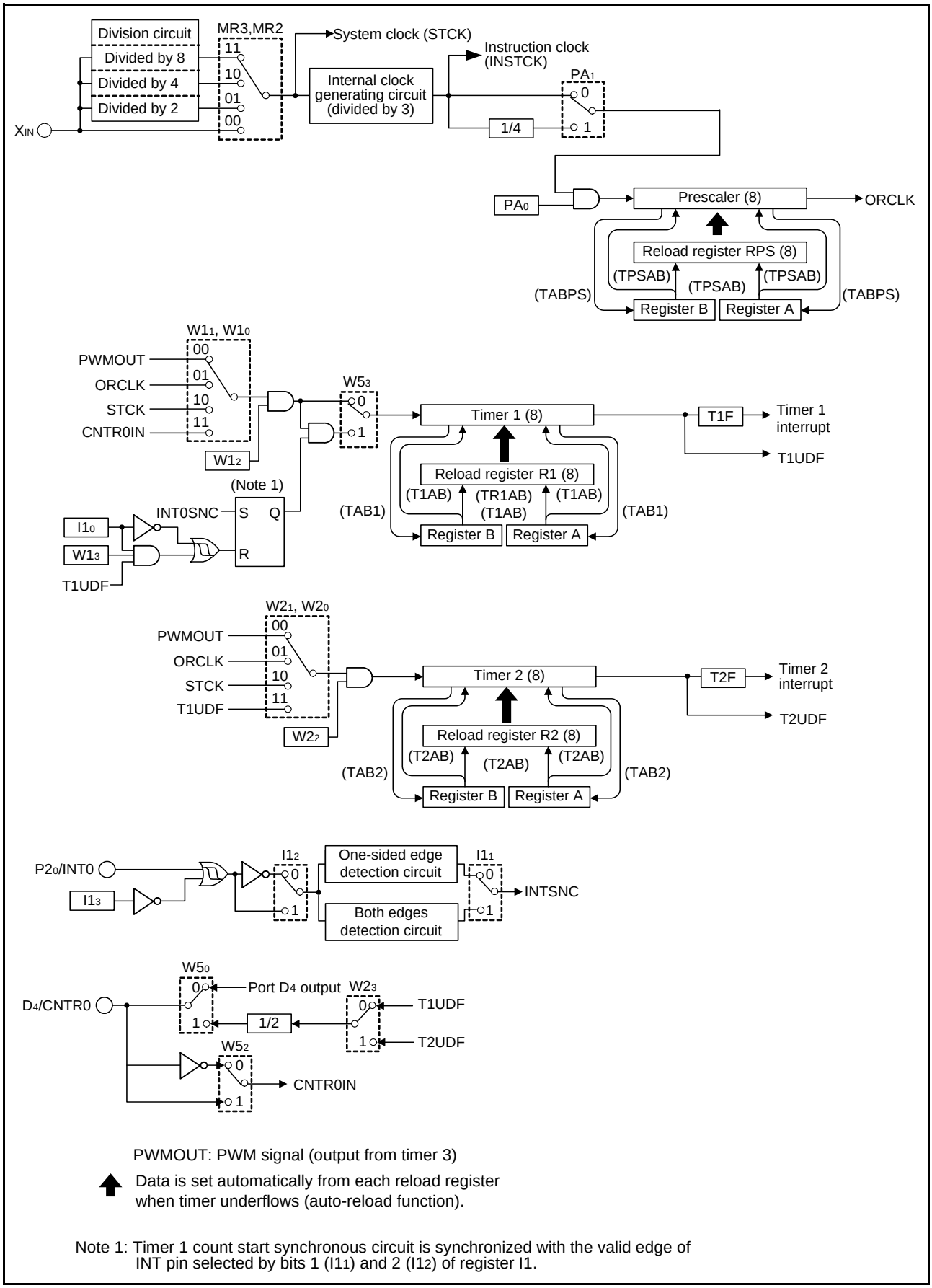
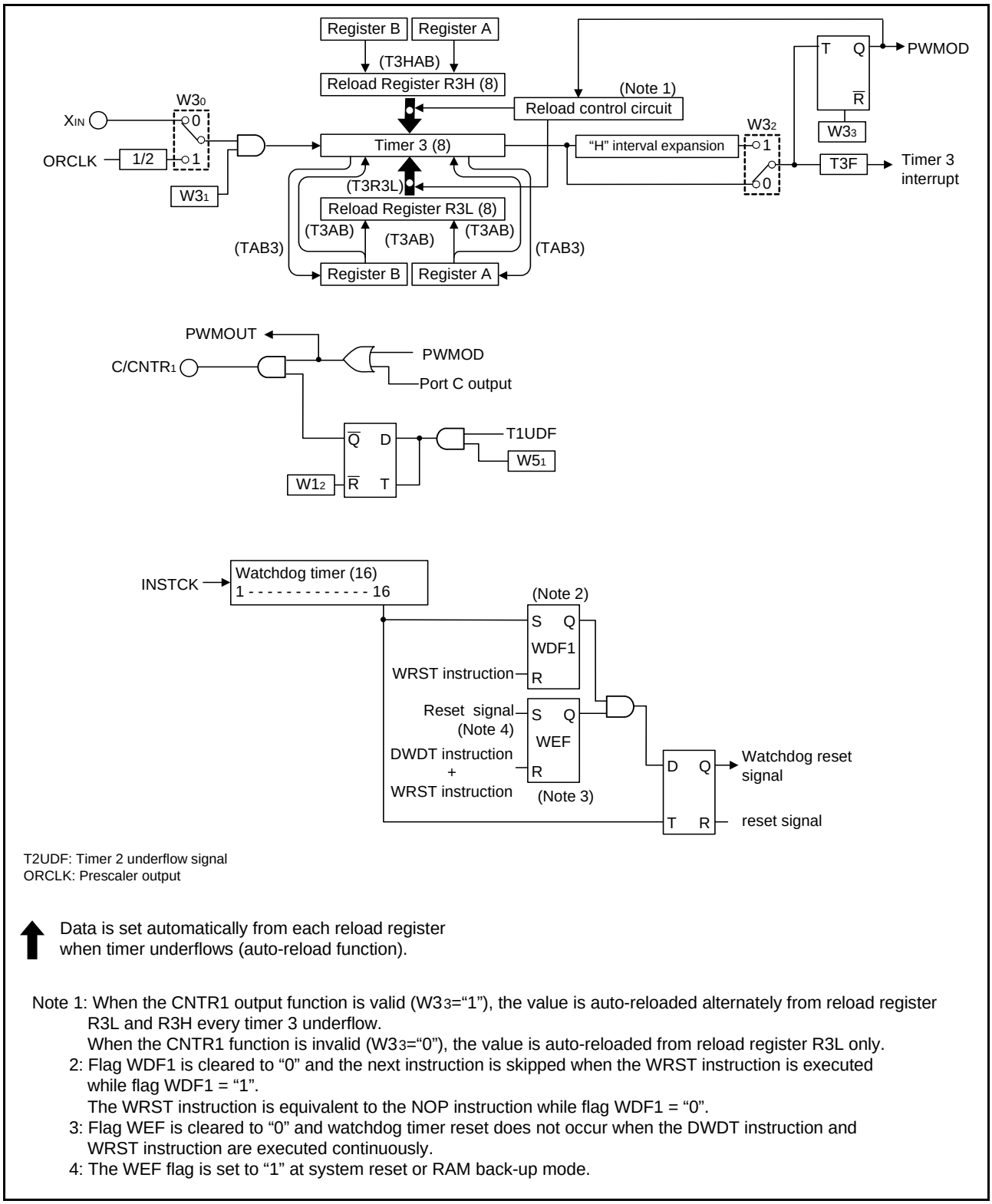


Fig 31. Timers structure (1)



T2UDF: Timer 2 underflow signal
ORCLK: Prescaler output

Data is set automatically from each reload register when timer underflows (auto-reload function).

Note 1: When the CNTR1 output function is valid (W3₃="1"), the value is auto-reloaded alternately from reload register R3L and R3H every timer 3 underflow.
When the CNTR1 function is invalid (W3₃="0"), the value is auto-reloaded from reload register R3L only.
2: Flag WDF1 is cleared to "0" and the next instruction is skipped when the WRST instruction is executed while flag WDF1 = "1".
The WRST instruction is equivalent to the NOP instruction while flag WDF1 = "0".
3: Flag WEF is cleared to "0" and watchdog timer reset does not occur when the DWDT instruction and WRST instruction are executed continuously.
4: The WEF flag is set to "1" at system reset or RAM back-up mode.

Fig 32. Timers structure (2)

Table 17 Timer control registers

Timer control register PA		at reset : 00_2	at RAM back-up : 00_2	W TPAA
PA_1	Prescaler count source selection bit	0	Instruction clock (INSTCK)	
		1	Instruction clock divided by 4 (INSTCK/4)	
PA_0	Prescaler control bit	0	Stop (state initialized)	
		1	Operating	

Timer control register W1		at reset : 0000_2		at RAM back-up : state retained	R/W TAW1/TW1A
$W1_3$	Timer 1 count auto-stop circuit selection bit (Note 2)	0	Timer 1 count auto-stop circuit not selected		
		1	Timer 1 count auto-stop circuit selected		
$W1_2$	Timer 1 control bit	0	Stop (state retained)		
		1	Operating		
$W1_1$	Timer 1 count source selection bits	$W1_1$	$W1_0$	Count source	
$W1_0$		0	0	PWM signal (PWMOUT)	
		0	1	Prescaler output (ORCLK)	
		1	0	System clock (STCK)	
		1	1	CNTR0 input	

Timer control register W2		at reset : 0000_2		at RAM back-up : state retained	R/W TAW2/TW2A
$W2_3$	CNTR0 pin function selection bit	0	Timer 1 underflow signal divided by 2 output		
		1	Timer 2 underflow signal divided by 2 output		
$W2_2$	Timer 2 control bit	0	Stop (state retained)		
		1	Operating		
$W2_1$	Timer 2 count source selection bits	$W2_1$	$W2_0$	Count source	
$W2_0$		0	0	PWM signal (PWMOUT)	
		0	1	Prescaler output (ORCLK)	
		1	0	System clock (STCK)	
		1	1	Timer 1 underflow signal (T1UDF)	

Timer control register W3		at reset : 0000_2	at RAM back-up : 0000_2	R/W TAW3/TW3A
$W3_3$	CNTR1 pin output control bit	0	CNTR1 pin output invalid	
		1	CNTR1 pin output valid	
$W3_2$	PWM signal “H” interval expansion function control bit	0	PWM signal “H” interval expansion function invalid	
		1	PWM signal “H” interval expansion function valid	
$W3_1$	Timer 3 control bit	0	Stop (state retained)	
		1	Operating	
$W3_0$	Timer 3 count source selection bit	0	X_{IN} input	
		1	Prescaler output (ORCLK)/2	

Timer control register W5		at reset : 0000_2	at RAM back-up : state retained	R/W TAW5/TW5A
$W5_3$	Timer 1 count start synchronous circuit selection bit (Note 3)	0	Count start synchronous circuit not selected	
		1	Count start synchronous circuit selected	
$W5_2$	CNTR0 pin input count edge selection bit	0	Falling edge	
		1	Rising edge	
$W5_1$	CNTR 1 pin output auto-control circuit selection bit	0	Output auto-control circuit not selected	
		1	Output auto-control circuit selected	
$W5_0$	D_4/CNTR0 pin function selection bit	0	D_4 (I/O) / CNTR0 (input)	
		1	D_4 (input) /CNTR0 (I/O)	

Note 1.“R” represents read enabled, and “W” represents write enabled.
Note 2.This function is valid only when the INT0 pin/timer 1 control is enabled ($I1_0$ =“1”) and the timer 1 count start synchronous circuit is selected ($W5_3$ =“1”).
Note 3.This function is valid only when the INT0 pin/timer 1 control is enabled ($I1_0$ =“1”).

(1) Timer control registers

- Timer control register PA
  Register PA controls the count operation and count source of prescaler. Set the contents of this register through register A with the TPAA instruction.
- Timer control register W1
  Register W1 controls the count operation and count source of timer 1, and timer 1 count auto-stop circuit. Set the contents of this register through register A with the TW1A instruction. The TAW1 instruction can be used to transfer the contents of register W1 to register A.
- Timer control register W2
  Register W2 controls the count operation and count source of timer 2, and CNTR0 pin output signal function. Set the contents of this register through register A with the TW2A instruction. The TAW2 instruction can be used to transfer the contents of register W2 to register A.
- Timer control register W3
  Register W3 controls timer 3 count source, timer 3 count operation, CNTR1 pin output and PWM signal "H" interval expansion function. Set the contents of this register through register A with the TW3A instruction. The TAW3 instruction can be used to transfer the contents of register W3 to register A.
- Timer control register W5
  Register W5 controls the input count edge of CNTR0 pin, the timer 1 count start synchronous circuit, CNTR1 pin output auto-control circuit and the D_4/CNTR1 pin function. Set the contents of this register through register A with the TW5A instruction. The TAW5 instruction can be used to transfer the contents of register W5 to register A.

(2) Prescaler

Prescaler is an 8-bit binary down counter with the prescaler reload register RPS. Data can be set simultaneously in prescaler and the reload register RPS with the TPSAB instruction. Data can be read from reload register RPS with the TABPS instruction.
Stop counting and then execute the TPSAB or TABPS instruction to read or set prescaler data.
Prescaler starts counting after the following process;
(1) set data in prescaler, and
(2) set the bit 0 of register PA to "1."
When a value set in reload register RPS is n, prescaler divides the count source signal by n + 1 (n = 0 to 255).
Count source for prescaler can be selected the instruction clock (INSTCK) or the instruction clock (INSTCK)/4.
Once count is started, when prescaler underflows (the next count pulse is input after the contents of prescaler becomes "0"), new data is loaded from reload register RPS, and count continues (auto-reload function).
The output signal (ORCLK) of prescaler can be used for timer 1 and 2 count sources.

(3) Timer 1 (interrupt function)

Timer 1 is an 8-bit binary down counter with a timer 1 reload register (R1). Data can be set simultaneously in timer 1 and the reload register R1 with the TR1AB instruction. Data can be read from timer 1 with the TAB1 instruction.
Stop counting and then execute the T1AB or TAB1 instruction to read or set timer 1 data.
When executing the TR1AB instruction to set data to reload register R1 while timer 1 is operating, avoid a timing when timer 1 underflows.
Timer 1 starts counting after the following process;
(1) set data in timer 1
(2) set count source by bits 0 and 1 of register W1, and
(3) set the bit 2 of register W1 to "1."

When a value set in reload register R1 is n, timer 1 divides the count source signal by n + 1 (n = 0 to 255).
Once count is started, when timer 1 underflows (the next count pulse is input after the contents of timer 1 becomes "0"), the timer 1 interrupt request flag (T1F) is set to "1," new data is loaded from reload register R1, and count continues (auto-reload function).
After timer 1 control by INT0 pin is enabled by setting the bit 0 of register I1 to "1", INT0 pin input can be used as the start trigger for timer 1 count operation by setting the bit 3 of register W5 to "1".
Also, in this time, the auto-stop function by timer 1 underflow can be performed by setting the bit 3 of register W1 to "1."
The timer 1 underflow signal divided by 2 can be output from the CNTR0 pin by setting the bit 0 of register W5 to "1" and bit 3 of register W2 to "0".

(4) Timer 2 (interrupt function)

Timer 2 is an 8-bit binary down counter with timer 2 reload register (R2). Data can be set simultaneously in timer 2 and the reload register R2 with the T2AB instruction. Data can be read from timer 2 with the TAB2 instruction.
Stop counting and then execute the T2AB or TAB2 instruction to read or set timer 2 data.
Timer 2 starts counting after the following process;
(1) set data in timer 2
(2) set count source by bits 0 and 1 of register W2, and
(3) set the bit 2 of register W2 to "1."
When a value set in reload register R2 is n, timer 2 divides the count source signal by n + 1 (n = 0 to 255).
Once count is started, when timer 2 underflows (the next count pulse is input after the contents of timer 2 becomes "0"), the timer 2 interrupt request flag (T2F) is set to "1," new data is loaded from reload register R2, and count continues (auto-reload function).
The timer 2 underflow signal divided by 2 can be output from the CNTR0 pin by setting the bit 0 of register W5 to "1" and bit 3 of register W2 to "1".

(5) Timer 3 (interrupt function)

Timer 3 is an 8-bit binary down counter with two timer 3 reload registers (R3L, R3H). Data can be set simultaneously in timer 3 and the reload register R3L with the T3AB instruction. Data can be set in the reload register R3H with the T3HAB instruction. The contents of reload register R3L set with the T3AB instruction can be set to timer 3 again with the T3R3L instruction. Data can be read from timer 3 with the TAB3 instruction.
Stop counting and then execute the T3AB or TAB3 instruction to read or set timer 3 data.
When executing the T3HAB instruction to set data to reload register R3H while timer 3 is operating, avoid a timing when timer 3 underflows.
Timer 3 starts counting after the following process;
(1) set data in timer 3
(2) set count source by bit 0 of register W3, and
(3) set the bit 1 of register W3 to "1."

When a value set in reload register R3L is n and a value set in reload register R3H is m, timer 3 divides the count source signal by n + 1 or m + 1 (n = 0 to 255, m = 0 to 255).
<Bit 3 of register W3 = "0" (CNTR1 pin output invalid)>
Once count is started, when timer 3 underflows (the next count pulse is input after the contents of timer 3 becomes "0"), the timer 3 interrupt request flag (T3F) is set to "1," new data is loaded from reload register R3L, and count continues (auto-reload function).
<Bit 3 of register W3 = "1" (CNTR1 pin output valid)>
Timer 3 generates the PWM signal of the "L" interval set as reload register R3L, and the "H" interval set as reload register R3H. The PWM (PWMOD) signal generated by timer 3 is output from CNTR1 pin.
When bit 2 of register W3 is set to "1" at this time, timer 3 extends the interval set to reload register R3H for a half period of count source. When a value set in reload register R3H is n, timer 3 divides the count source signal by m + 1.5 (m = 1 to 255).
When this function is used, set "1" or more to reload register R3H.
When bit 1 of register W5 is set to "1", the PWM signal output to CNTR1 pin is switched to valid/invalid each timer 1 underflow. However, when timer 3 is stopped, this function is canceled.
Even when bit 1 of a register W3 is cleared to "0" in the "H" interval of PWM signal, timer 3 does not stop until it next timer 3 underflow.
When bit 1 of register W3 is cleared to "0" in order to stop timer 3 while the PWM output is used, avoid a timing when timer 3 underflows.
If these timings overlap, a hazard may occur in a CNTR1 output waveform.

(6) Count start synchronization circuit (timer 1)

Timer 1 has the count start synchronous circuit which synchronizes the input of INT0 pin, and can start the timer count operation.
Timer 1 count start synchronous circuit function can be selected after timer 1 control by INT0 pin is enabled by setting the bit 0 of register I1 to "1" and its function is selected by setting the bit 3 of register W5 to "1".
When timer 1 count start synchronous circuit is used, the count start synchronous circuit is set, the count source is input to timer by inputting valid waveform to INT0 pin.
The valid waveform of INT0 pin to set the count start synchronous circuit is the same as the external interrupt activated condition.
Once set, the count start synchronous circuit is cleared by clearing the bit $I1_0$ to "0" or system reset.
However, when the count auto-stop circuit is selected, the count start synchronous circuit is cleared (auto-stop) at the timer 1 underflow.

(7) Count auto-stop circuit (timer 1)

Timer 1 has the count auto-stop circuit which is used to stop timer 1 automatically by the timer 1 underflow when the count start synchronous circuit is used.
The count auto-stop circuit is valid by setting the bit 3 of register W1 to "1". It is cleared by the timer 1 underflow and the count source to timer 1 is stopped.
This function is valid only when the timer 1 count start synchronous circuit is selected.

(8) Timer input/output pin (D_4/CNTR0)

CNTR0 pin is used to input the timer 1 count source and output the timer 1 or timer 2 underflow signal/2.
The D_4/CNTR0 pin function can be selected by bit 0 of register W5.
The output signal can be selected by bit 0 of register W2.
When the CNTR0 input is selected for timer 1 count source, timer 1 counts the falling or rising waveform of CNTR0 input. The count edge is selected by bit 2 of register W5.

(9) PWM signal output function (C/CNTR1, timer 1, timer 2)

The C/CNTR1 pin is also used to output the PWM signal generated by timer 3.
When the bit 3 of register W3 is set to "1", the PWM signal can be output from the C/CNTR1 pin. In this time, set the output latch of port C to "1."

(10)Timer interrupt request flags (T1F, T2F, T3F)

Each timer interrupt request flag is set to "1" when each timer underflows. The state of these flags can be examined with the skip instructions (SNZT1, SNZT2, SNZT3).
Use the interrupt control register V1, V2 to select an interrupt or a skip instruction.
An interrupt request flag is cleared to "0" when an interrupt occurs or when the next instruction is skipped with a skip instruction.

(11) Precautions

- Prescaler
  Stop prescaler counting and then execute the TABPS instruction to read its data.
  Stop prescaler counting and then execute the TPSAB instruction to write data to prescaler.
  Stop prescaler counting to change its count source.
- Timer count source
  Stop timer 1, 2 or 3 counting to change its count source.
- Reading the count value
  Stop timer 1, 2 or 3 counting and then execute the TAB1, TAB2 or TAB3 instruction to read its data.
- Writing to the timer
  Stop timer 1, 2 or 3 counting and then execute the T1AB, T2AB, T3AB or T3R3L instruction to write data to timer.
- Writing to reload register
  In order to write a data to the reload register R1 while the timer 1 is operating, execute the TR1AB instruction except a timing of the timer 1 underflow.
  In order to write a data to the reload register R3H while the timer 3 is operating, execute the T3HAB instruction except a timing of the timer 3 underflow.
- PWM signal
  If the timer 3 count stop timing and the timer 3 underflow timing overlap during output of the PWM signal, a hazard may occur in the PWM output waveform.
  When "H" interval expansion function of the PWM signal is used, set "1" or more to reload register R3H.
  Set the port C output latch to "0" to output the PWM signal from C/CNTR1 pin.
- Prescaler, timer 1, timer 2 and timer 3 count start timing and count time when operation starts
  Count starts from the first rising edge of the count source (2) in Figure 33 after prescaler and timer operations start (1) in Figure 33.
  Time to first underflow (3) in Figure 33 is shorter (for up to 1 period of the count source) than time among next underflow (4) in Figure 33 by the timing to start the timer and count source operations after count starts.
  When selecting CNTR0 input as the count source of timer 1, timer 1 operates synchronizing with the count edge (falling edge or rising edge) of CNTR0 input selected by software.

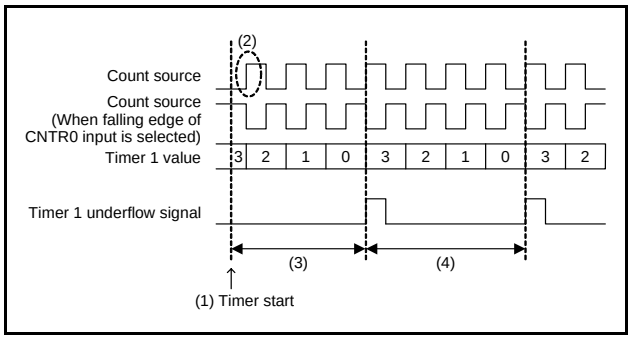


Fig 33. Timer count start timing and count time when operation starts

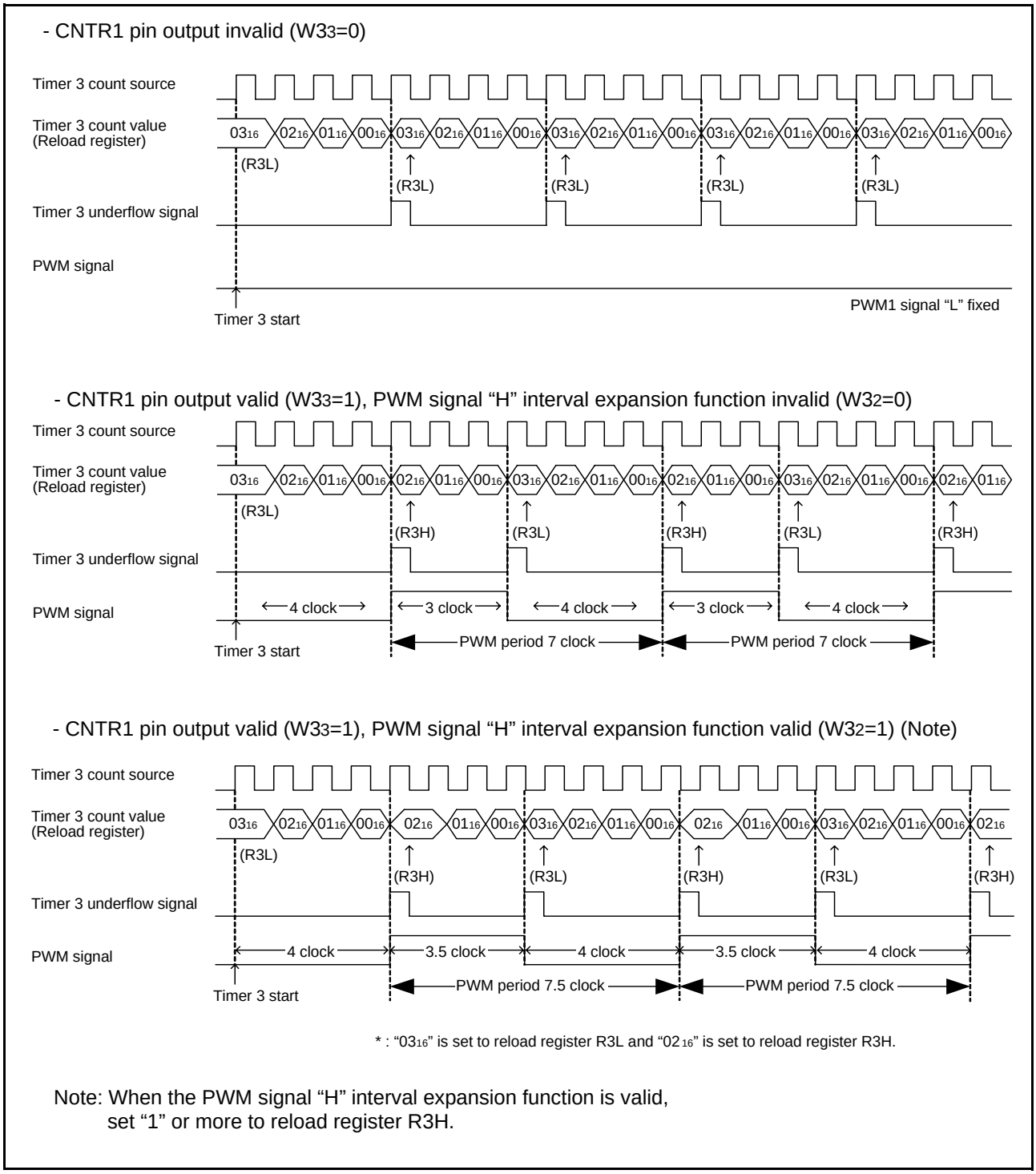
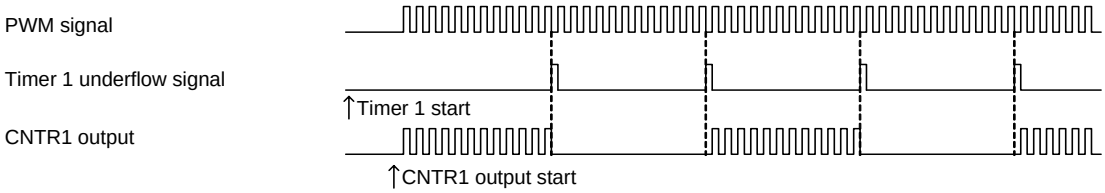


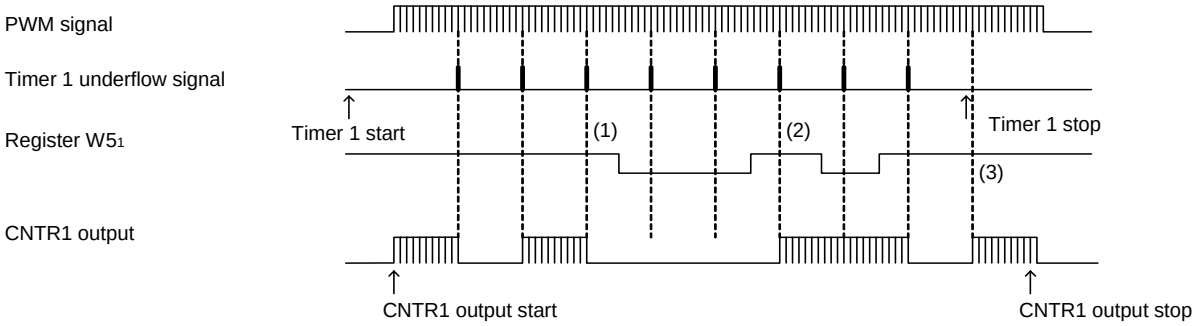
Fig 34. Timer 3 operation example

• CNTR1 output auto-control circuit operation example 1 ($W3_3$ = "1", $W5_1$ = "1")



* When the CNTR1 output auto-control circuit is selected, valid/invalid of CNTR1 output is repeated every timer 1 underflows.

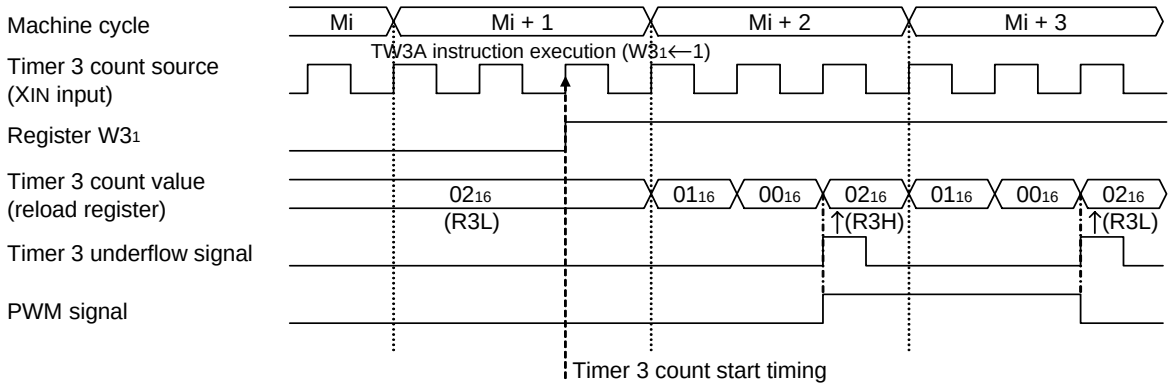
• CNTR1 output auto-control circuit operation example 2 ($W3_3$ = "1", $W5_1$ = "1")



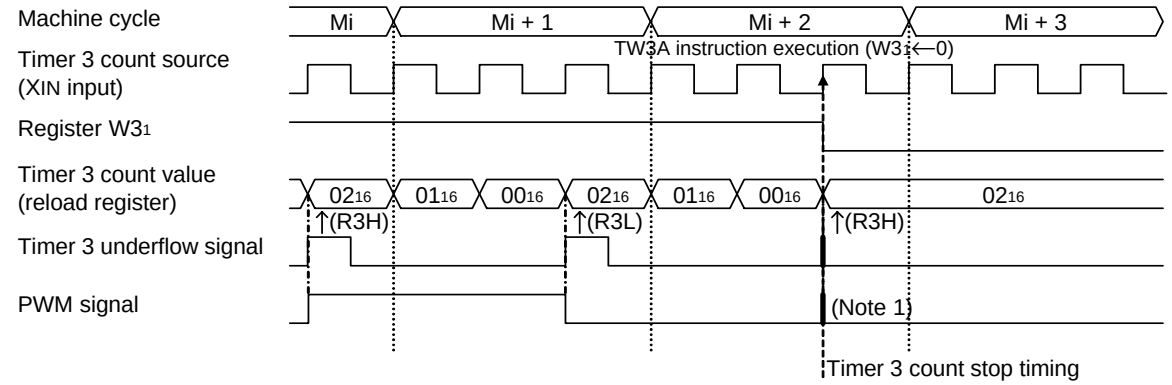
(1) When the CNTR1 output auto-control function is not selected while the CNTR1 output is invalid, CNTR1 output invalid state is retained.
(2) When the CNTR1 output auto-control function is not selected while the CNTR1 output is valid, CNTR1 output valid state is retained.
(3) When the timer 1 is stopped, the CNTR1 output auto-control function becomes invalid.

Fig 35. CNTR1 output auto-control function by timer 1

Timer 3 count start timing (R3L = "02$_{16}$", R3H = "02$_{16}$", W3$_3$ = "1")



Timer 3 count stop timing (R3L = "02$_{16}$", R3H = "02$_{16}$", W3$_3$ = "1")



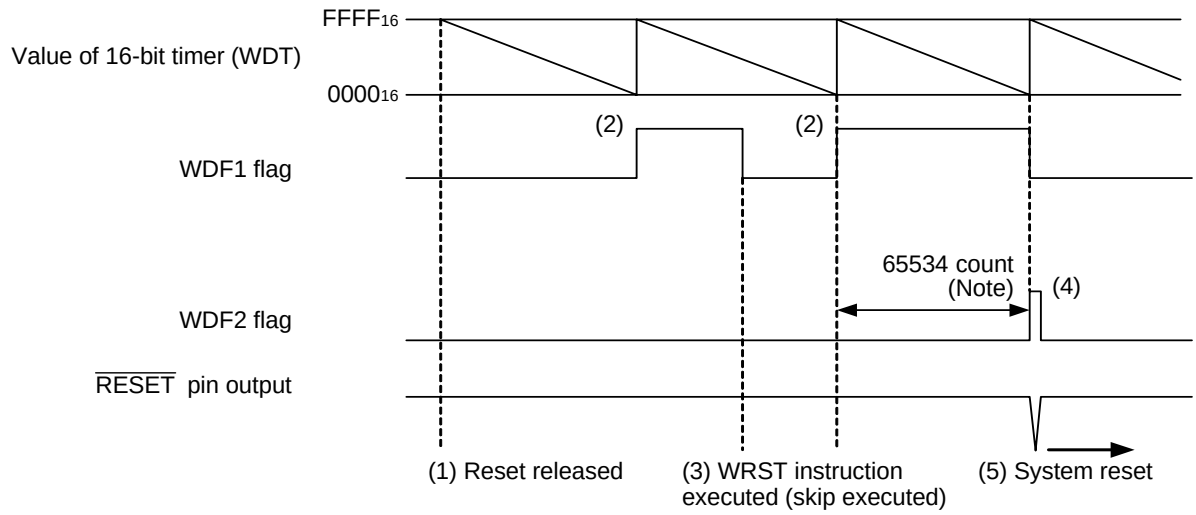
Notes 1: If the timer count stop timing and the timer underflow timing overlap while the CNTR1 pin output is valid (W33="1"), a hazard may occur in the PWM signal waveform.
2: When timer count is stopped during "H" interval of the PWM signal, timer is stopped after the end of the "H" output interval.

Fig 36. Timer count start/stop timing

WATCHDOG TIMER

Watchdog timer provides a method to reset the system when a program run-away occurs. Watchdog timer consists of timer WDT(16-bit binary counter), watchdog timer enable flag (WEF), and watchdog timer flags (WDF1, WDF2).
The timer WDT downcounts the instruction clocks as the count source from "FFFF16" after system is released from reset.
After the count is started, when the timer WDT underflow occurs (after the count value of timer WDT reaches "000016," the next count pulse is input), the WDF1 flag is set to "1." If the WRST instruction is never executed until the timer WDT underflow occurs (until timer WDT counts 65534), WDF2 flag is set to "1," and the $\overline{\text{RESET}}$ pin outputs "L" level to reset the microcomputer.
Execute the WRST instruction at each period of 65534 machine cycle or less by software when using watchdog timer to keep the microcomputer operating normally.

When the WEF flag is set to "1" after system is released from reset, the watchdog timer function is valid.
When the DWDT instruction and the WRST instruction are executed continuously, the WEF flag is cleared to "0" and the watchdog timer function is invalid.
The WEF flag is initialized to "1" at system reset or RAM back-up mode.
The WRST instruction has the skip function. When the WRST instruction is executed while the WDF1 flag is "1", the WDF1 flag is cleared to "0" and the next instruction is skipped.
When the WRST instruction is executed while the WDF1 flag is "0", the next instruction is not skipped.
The skip function of the WRST instruction can be used even when the watchdog timer function is invalid.



(1) After system is released from reset (= after program is started), timer WDT starts count down.
(2) When timer WDT underflow occurs, WDF1 flag is set to "1".
(3) When the WRST instruction is executed while the WDF1 flag is "1", WDF1 flag is cleared to "0", the next instruction is skipped.
(4) When timer WDT underflow occurs while WDF1 flag is "1", WDF2 flag is set to "1" and the watchdog reset signal is output.
(5) The output transistor of $\overline{\text{RESET}}$ pin is turned "ON" by the watchdog reset signal and system reset is executed.

Note: The number of count is equal to the number of machine cycle because the count source of watchdog timer is the instruction clock.

Fig 37. Watchdog timer function

When the watchdog timer is used, clear the WDF1 flag at the period of 65534 machine cycles or less with the WRST instruction.
When the watchdog timer is not used, execute the DWDT instruction and the WRST instruction continuously (refer to Figure 38).
The watchdog timer is not stopped with only the DWDT instruction.
The contents of WDF1 flag and timer WDT are initialized at the RAM back-up mode.
When using the watchdog timer and the RAM back-up mode, initialize the WDF1 flag with the WRST instruction just before the microcomputer enters the RAM back-up state. Also, set the NOP instruction after the WRST instruction, for the case when a skip is performed with the WRST instruction (refer to Figure 39).

```
    ⋮
WRST        ; WDF1 flag cleared
    ⋮
DI
DWDT        ; Watchdog timer function enabled/disabled
WRST        ; WEF and WDF1 flags cleared
    ⋮
```

Fig 38. Program example to start/stop watchdog timer

```
    ⋮
WRST        ; WDF1 flag cleared
NOP
DI          ; Interrupt disabled
EPOF        ; POF instruction enabled
POF         ; RAM back-up mode
    ↓
Oscillation stop
    ⋮
```

Fig 39. Program example when using the watchdog timer

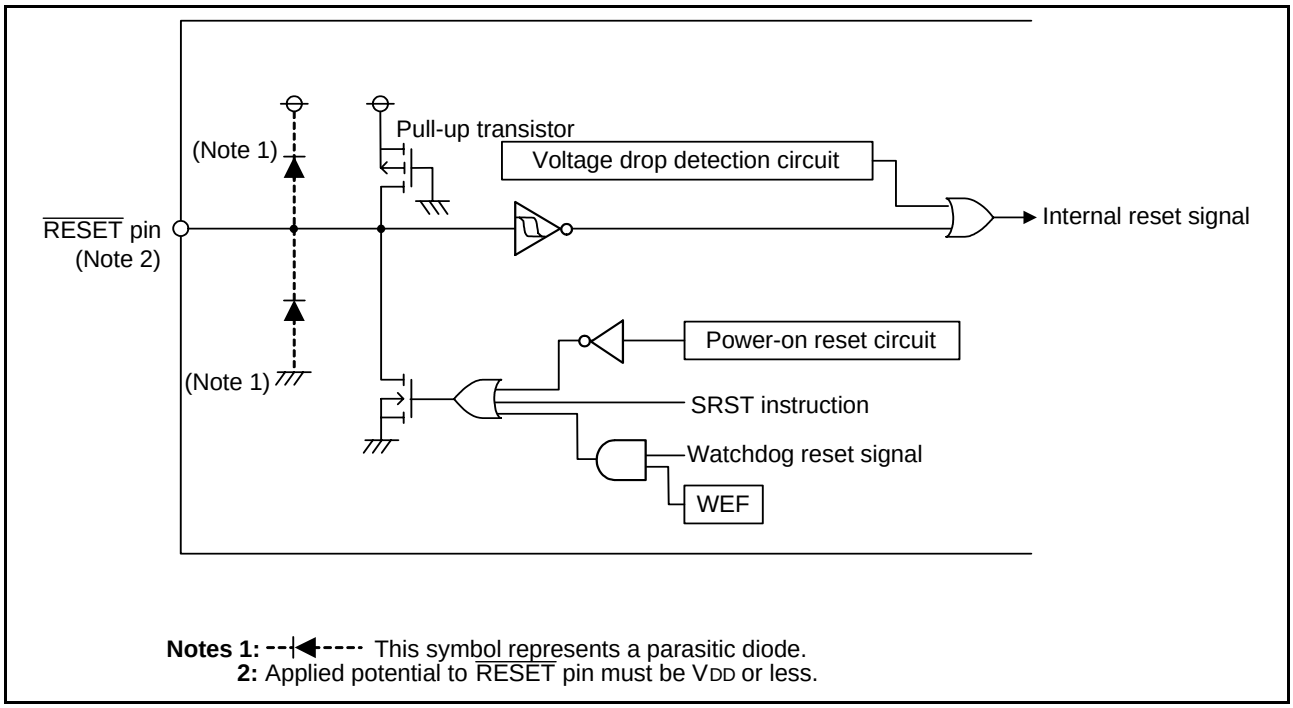
RESET FUNCTION

System reset is performed by the followings:
- "L" level is applied to the $\overline{\text{RESET}}$ pin externally,
- System reset instruction (SRST) is executed,
- Reset occurs by watchdog timer,
- Reset occurs by built-in power-on reset
- Reset occurs by voltage drop detection circuit

Then when "H" level is applied to $\overline{\text{RESET}}$ pin, software starts from address 0 in page 0.

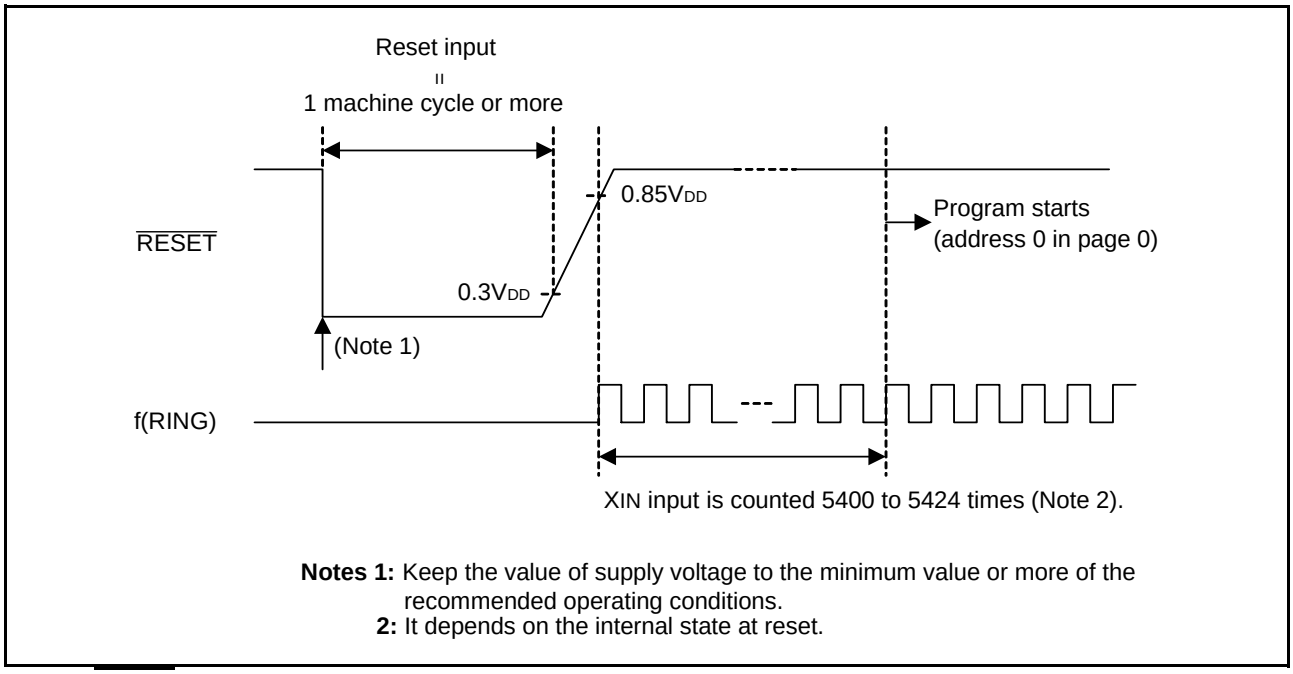
(1) $\overline{\text{RESET}}$ pin input

System reset is performed certainly by applying "L" level to $\overline{\text{RESET}}$ pin for 1 machine cycle or more when the following condition is satisfied;
the value of supply voltage is the minimum value or more of the recommended operating conditions.



Notes 1: ---◄--- This symbol represents a parasitic diode.
2: Applied potential to $\overline{\text{RESET}}$ pin must be V_{DD} or less.

Fig 40. Structure of reset pin and its peripherals



Notes 1: Keep the value of supply voltage to the minimum value or more of the recommended operating conditions.
2: It depends on the internal state at reset.

Fig 41. $\overline{\text{RESET}}$ pin input waveform and reset release timing

(2) Power-on reset

Reset can be automatically performed at power on (power-on reset) by the built-in power-on reset circuit. When the built-in power-on reset circuit is used, set the time for the supply voltage to rise from 0 V to the minimum voltage of recommended operating conditions to 100 μs or less.
If the rising time exceeds 100 μs, connect a capacitor between the $\overline{\text{RESET}}$ pin and Vss at the shortest distance, and input "L" level to $\overline{\text{RESET}}$ pin until the value of supply voltage reaches the minimum operating voltage.

(3) System reset instruction (SRST)

By executing the SRST instruction, "L" level is output to $\overline{\text{RESET}}$ pin and system reset is performed.

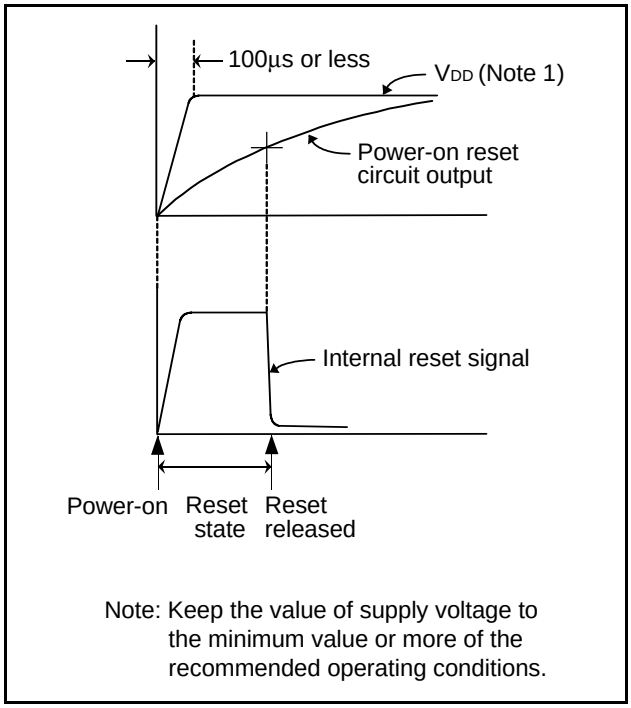


Fig 42. Power-on reset operation

Table 18 Port state at reset

Name	Function	State
D_0–D_3	D_0–D_3	High-impedance (Notes 1, 2)
D_4/CNTR0	D_4	High-impedance (Note 1)
$P0_0$–$P0_3$	$P0_0$–$P0_3$	High-impedance (Notes 1, 3)
$P1_0$–$P1_3$	$P1_0$–$P1_3$	High-impedance (Notes 1, 3)
$P2_0$/INT0, $P2_1$/INT1	$P2_0$, $P2_1$	High-impedance (Notes 1, 3)
$P3_0$, $P3_1$	$P3_0$, $P3_1$	High-impedance (Notes 1, 2)
C/CNTR1	C/CNTR1	(V_{SS})
K	K	High-impedance

Note 1.Output latch is set to "1."
Note 2.The output structure is N-channel open-drain.
Note 3.Pull-up transistor is turned OFF.

(4) Internal state at reset

Figure 43 shows internal state at reset (they are the same after system is released from reset). The contents of timers, registers, flags and RAM except shown in Figure 43 are undefined, so set the initial value to them.

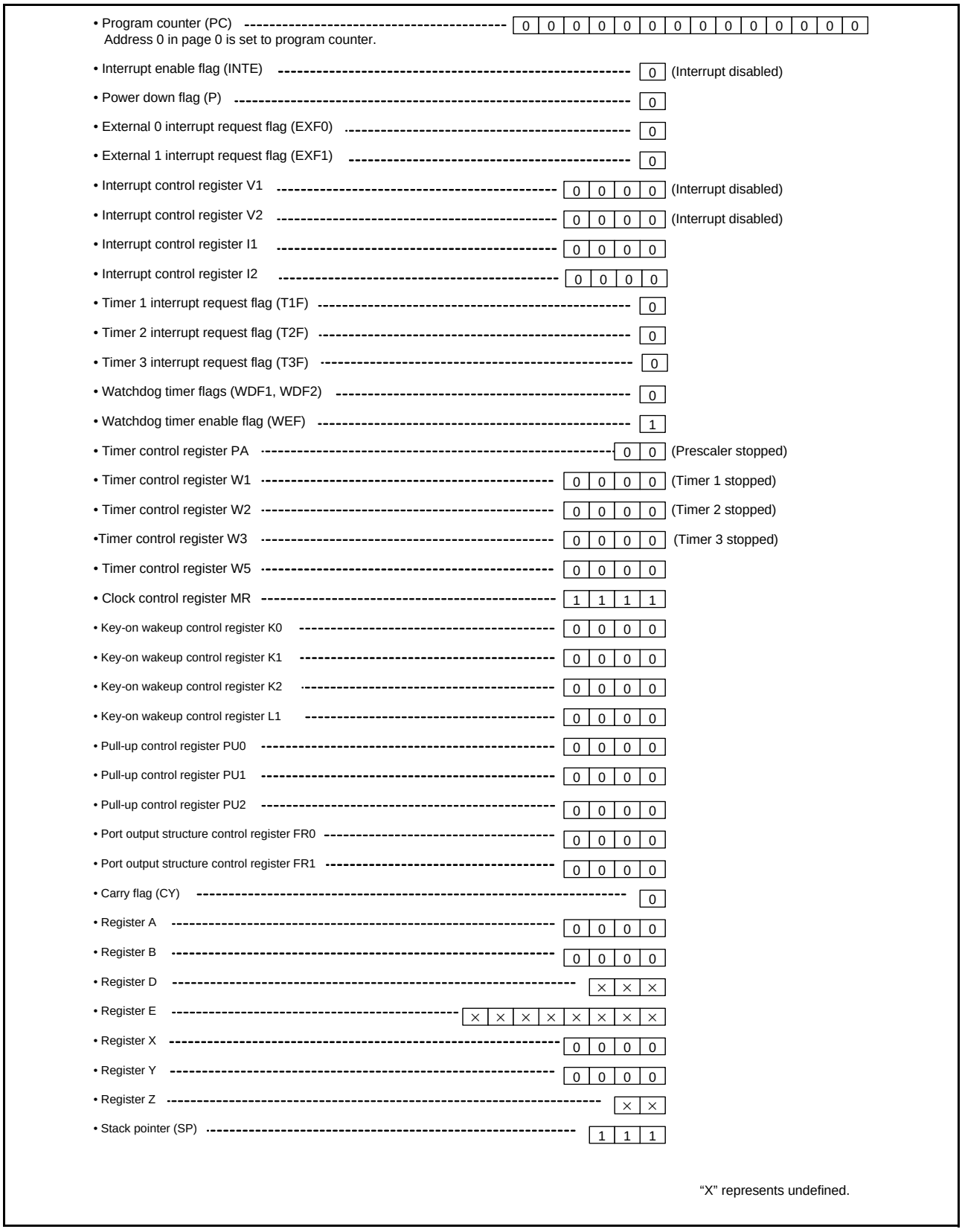
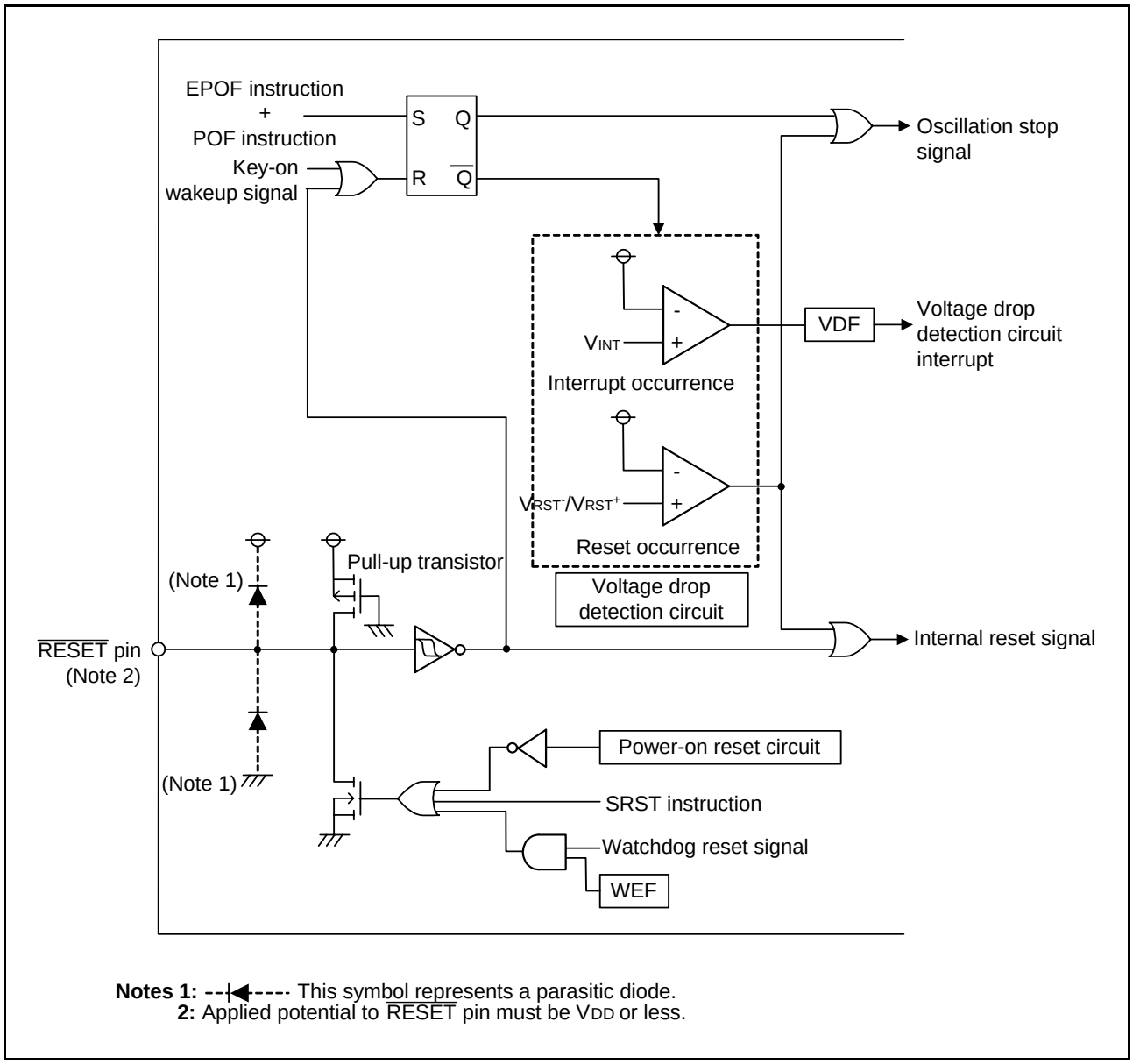


Fig 43. Internal state at reset

VOLTAGE DROP DETECTION CIRCUIT

The built-in voltage drop detection circuit is used to set the voltage drop detection circuit interrupt request flag (VDF) or to perform system reset.
The voltage drop detection circuit stops at RAM back-up mode.



Notes 1: ---|◀---- This symbol represents a parasitic diode.
2: Applied potential to $\overline{\text{RESET}}$ pin must be V_{DD} or less.

Fig 44. Voltage drop detection reset circuit

(1) Voltage drop detection circuit interrupt request flag (VDF)

Voltage drop detection circuit interrupt request flag (VDF) is set to "1" when the supply voltage goes the defined value (V_{INT}) or less. Moreover, voltage drop detection circuit interrupt request flag (VDF) is cleared to "0" when the supply voltage goes the defined value (V_{INT}) or more. The state of the interrupt request flag can be examined with the skip instruction (SNZVD). Use the interrupt control register V2 to select an interrupt or a skip instruction. Unlike other interrupt request flags, even when the interrupt occurs or the skip instruction is executed, the voltage drop detection circuit interrupt request flag is not cleared to "0".

(2) Voltage drop detection circuit reset

System reset is performed when the supply voltage goes the defined value (V_{RST-}) or less ("L" level is not output to $\overline{RESET}$ pin.). However, unlike the normal system reset, the oscillation circuit is stopped.

When the supply voltage goes the defined value (V_{RST-}) or more, the oscillation circuit goes to be in the operating enabled state and system reset is released.

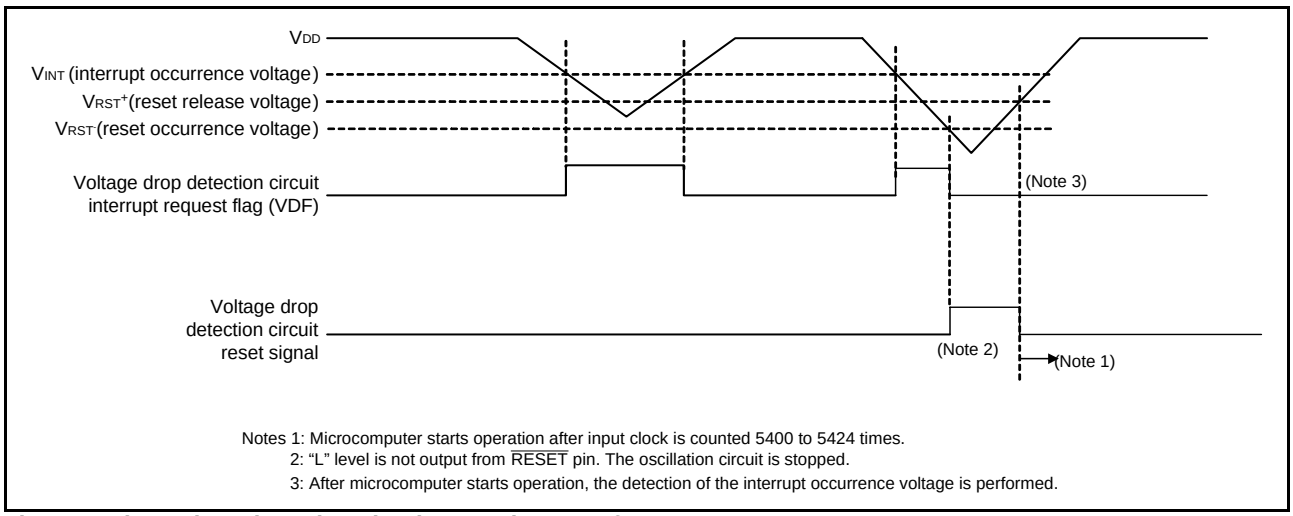


Fig 45. Voltage drop detection circuit operation waveform

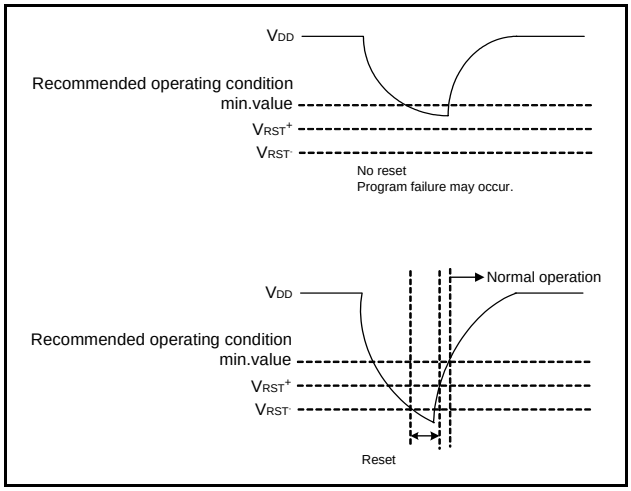


Fig 46. V_{DD} and V_{RST-}

(3) Note on voltage drop detection circuit

The voltage drop detection circuit detection voltage of this product is set up lower than the minimum value of the supply voltage of the recommended operating conditions.

When the supply voltage of a microcomputer falls below to the minimum value of recommended operating conditions and rises again, depending on the capacity value of the bypass capacitor added to the power supply pin, the following case may cause program failure (Figure 46);

supply voltage does not fall below to V_{RST-}, and
its voltage rises again with no reset.

In such a case, please design a system which supply voltage is once reduced below to V_{RST-} and re-goes up after that.

RAM BACK-UP MODE

The 4571 Group has the RAM back-up mode.
When the POF instruction is executed continuously after the EPOF instruction, system enters the RAM back-up state.
The POF instruction is equal to the NOP instruction when the EPOF instruction is not executed before the POF instruction.
As oscillation stops retaining RAM, the function of reset circuit and states at RAM back-up mode, current dissipation can be reduced without losing the contents of RAM.
Table 19 shows the function and states retained at RAM back-up.
Figure 47 shows the state transition.

(1) Identification of the start condition

Warm start (return from the RAM back-up state) or cold start (return from the normal reset state) can be identified by examining the state of the power down flag (P) with the SNZP instruction.

(2) Warm start condition

When the external wakeup signal is input after the system enters the RAM back-up state by executing the EPOF instruction and POF instruction continuously, the CPU starts executing the program from address 0 in page 0. In this case, the P flag is "1."

(3) Cold start condition

The CPU starts executing the program from address 0 in page 0 when;

- "L" level is applied to $\overline{\text{RESET}}$ pin,
- system reset (SRST) is performed,
- reset by watchdog timer is performed,
- reset by the built-in power-on reset circuit is performed, or
- reset by the voltage drop detection circuit is performed.

In this case, the P flag is "0."

Table 19 Functions and states retained at RAM back-up

Function	RAM back-up
Program counter (PC), stack pointer (SP) (Table 2), carry flag (CY), registers A, B	×
Contents of RAM	O
Interrupt control registers V1, V2	×
Interrupt control registers I1, I2	O
Clock control register MR	×
Timer 1, Timer 2, Timer 3 function	(Note 3)
Watchdog timer function	× (Note 4)
Timer control registers PA, W3	×
Timer control registers W1, W2, W5	O
Voltage drop detection circuit	(Note 5)
Port level	(Note 6)
Key-on wakeup control registers K0 to K2, L1	O
Pull-up control registers PU0 to PU2	O
Port output structure control registers FR0, FR1	O
External interrupt request flags (EXF0, EXF1)	×
Timer interrupt request flags (T1F, T2F, T3F)	(Note 3)
Voltage drop detection circuit interrupt request flag (VDF)	×
Interrupt enable flag (INTE)	×
Watchdog timer flags (WDF1, WDF2)	× (Note 4)
Watchdog timer enable flag (WEF)	× (Note 4)

Note 1."O" represents that the function can be retained, and "×" represents that the function is initialized.
Registers and flags other than the above are undefined at RAM back-up, and set an initial value after returning.
Note 2.The stack pointer (SP) points the level of the stack register and is initialized to "7" at RAM back-up.
Note 3.The state of the timer is undefined.
Note 4.Initialize the watchdog timer flag WDF1 with the WRST instruction, and then set the system to be in the RAM back-up mode.
Note 5.The voltage drop detection circuit is invalid.
Note 6.C/CNTR pin outputs "L" level. Other ports retain their output levels.

(4) Return signal

An external wakeup signal is used to return from the RAM back-up mode because the oscillation is stopped. Table 20 shows the return condition for each return source.

(5) Control registers

- Key-on wakeup control register K0
  Register K0 controls the port P0 key-on wakeup function. Set the contents of this register through register A with the TK0A instruction. In addition, the TAK0 instruction can be used to transfer the contents of register K0 to register A.
- Key-on wakeup control register K1
  Register K1 controls the port P1 key-on wakeup function. Set the contents of this register through register A with the TK1A instruction. In addition, the TAK1 instruction can be used to transfer the contents of register K1 to register A.
- Key-on wakeup control register K2
  Register K2 controls the ports K and P2 key-on wakeup function. Set the contents of this register through register A with the TK2A instruction. In addition, the TAK2 instruction can be used to transfer the contents of register K2 to register A.
- Key-on wakeup control register L1
  Register L1 controls the selection of the selection of the INT0 pin return condition and INT0 pin key-on wakeup function and the selection of the INT1 pin return condition and INT1 pin key-on wakeup function. Set the contents of this register through register A with the TL1A instruction. In addition, the TAL1 instruction can be used to transfer the contents of register L1 to register A.
- Pull-up control register PU0
  Register PU0 controls the ON/OFF of the port P0 pull-up transistor. Set the contents of this register through register A with the TPU0A instruction. In addition, the TAPU0 instruction can be used to transfer the contents of register PU0 to register A.
- Pull-up control register PU1
  Register PU1 controls the ON/OFF of the port P1 pull-up transistor. Set the contents of this register through register A with the TPU1A instruction. In addition, the TAPU1 instruction can be used to transfer the contents of register PU1 to register A.
- Pull-up control register PU2
  Register PU2 controls the ON/OFF of the ports P2 pull-up transistor. Set the contents of this register through register A with the TPU2A instruction. In addition, the TAPU2 instruction can be used to transfer the contents of register PU2 to register A.
- Interrupt control register I1
  Register I1 controls the valid waveform/level of the INT0 pin and the input control of INT0 pin. Set the contents of this register through register A with the TI1A instruction. In addition, the TAI1 instruction can be used to transfer the contents of register I1 to register A.

Table 20 Return source and return condition

Return source		Return condition	Remarks
External wakeup signal	Port $P0_0$–$P0_3$ Port $P1_0$–$P1_3$ Port $P2_0$, $P2_1$ Port K	Return by an external falling edge (“H” → “L”) input.	The key-on wakeup function can be selected by one port unit. Set the port using the key-on wakeup function to “H” level before going into the RAM back-up state.
	INT pin	Return by an external “H” level or “L” level input, or falling edge (“H” → “L”) or rising edge (“L” → “H”). When the return level is input, the EXF0 flag is not set.	The key-on wakeup function can be selected by one port unit. Select the return level (“L” level or “H” level) with the register I1 and return condition (level or edge) with the register L1 according to the external state before going into the RAM back-up state.

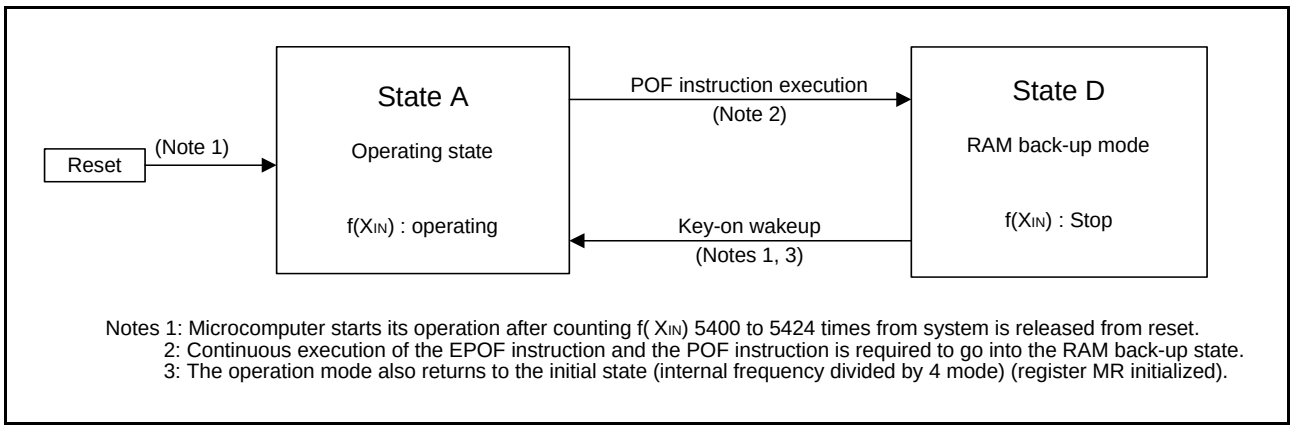


Fig 47. State transition

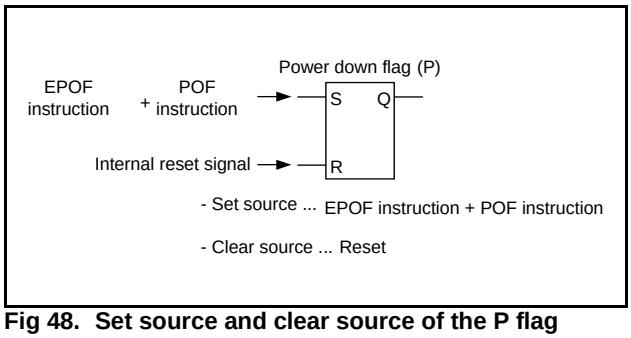


Fig 48. Set source and clear source of the P flag

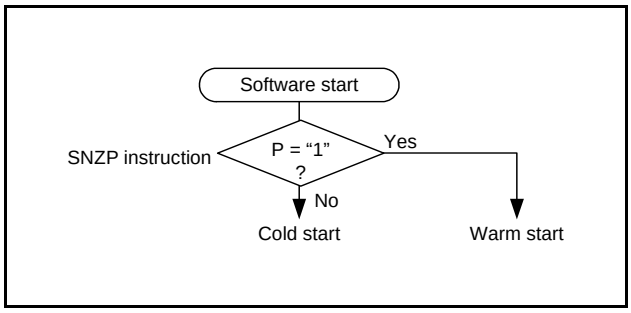


Fig 49. Start condition identified example using the SNZP instruction

Table 21 Key-on wakeup control registers

Key-on wakeup control register K0		at reset : 0000_2	at RAM back-up : state retained	R/W TAK0/TK0A
$K0_3$	Port $P0_3$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$K0_2$	Port $P0_2$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$K0_1$	Port $P0_1$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$K0_0$	Port $P0_0$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	

Key-on wakeup control register K1		at reset : 0000_2	at RAM back-up : state retained	R/W TAK1/TK1A
$K1_3$	Port $P1_3$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$K1_2$	Port $P1_2$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$K1_1$	Port $P1_1$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$K1_0$	Port $P1_0$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	

Key-on wakeup control register K2		at reset : 0000_2	at RAM back-up : state retained	R/W TAK2/TK2A
$K2_3$	Not used	0	This bit has no function, but read/write is enabled.	
		1		
$K2_2$	Port K key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$K2_1$	Port $P2_1$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$K2_0$	Port $P2_0$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	

Key-on wakeup control register L1		at reset : 0000_2	at RAM back-up : state retained	R/W TAL1/TL1A
$L1_3$	INT1 pin return condition selection bit	0	Return by level	
		1	Return by edge	
$L1_2$	INT1 pin valid waveform/ level selection bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$L1_1$	INT0 pin return condition selection bit	0	Return by level	
		1	Return by edge	
$L1_0$	INT0 pin key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	

Note 1.“R” represents read enabled, and “W” represents write enabled.

Table 22 Pull-up control registers

Pull-up control register PU0		at reset : 0000_2	at RAM back-up : state retained	R/W TAPU0/TPU0A
$PU0_3$	Port $P0_3$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	
$PU0_2$	Port $P0_2$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	
$PU0_1$	Port $P0_1$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	
$PU0_0$	Port $P0_0$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	

Pull-up control register PU1		at reset : 0000_2	at RAM back-up : state retained	R/W TAPU1/TPU1A
$PU1_3$	Port $P1_3$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	
$PU1_2$	Port $P1_2$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	
$PU1_1$	Port $P1_1$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	
$PU1_0$	Port $P1_0$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	

Pull-up control register PU2		at reset : 0000_2	at RAM back-up : state retained	R/W TAPU2/TPU2A
$PU2_3$	Not used	0	This bit has no function, but read/write is enabled.	
		1		
$PU2_2$	Not used	0	This bit has no function, but read/write is enabled.	
		1		
$PU2_1$	Port $P2_1$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	
$PU2_0$	Port $P2_0$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	

Note 1.“R” represents read enabled, and “W” represents write enabled.

CLOCK CONTROL

The clock control circuit consists of the following circuits.

- Ceramic oscillation circuit
- Frequency divider
- Internal clock generating circuit

The system clock and the instruction clock are generated as the source clock for operation by these circuits.
Figure 50 shows the structure of the clock control circuit.

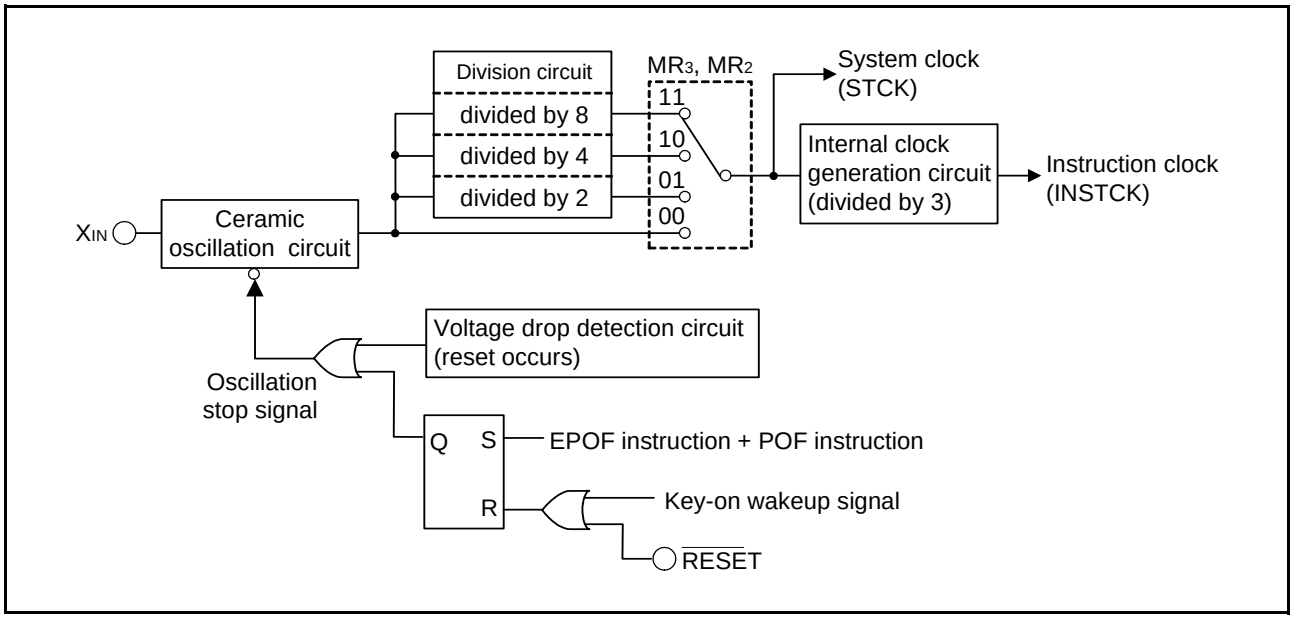


Fig 50. Clock control circuit structure

(1) Ceramic resonator

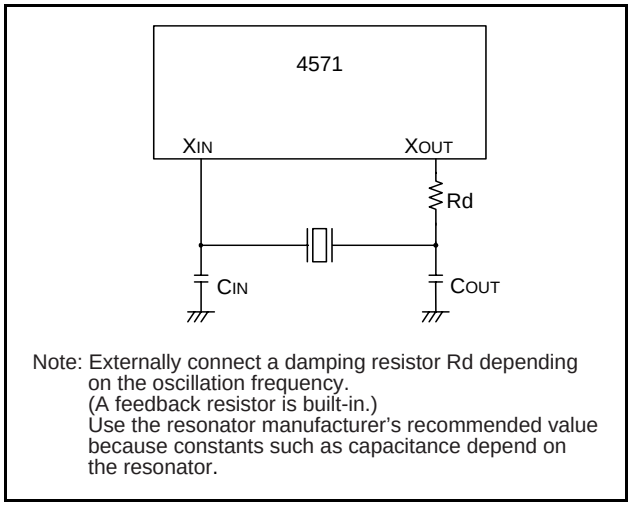
When the ceramic resonator is used as the main clock (f(XIN)), connect the ceramic resonator and the external circuit to pins XIN and XOUT at the shortest distance. A feedback resistor is built in between pins XIN and XOUT (Figure 51).

(2) External clock

When the external signal clock is used for the main clock (f(XIN)), connect the XIN pin to the clock source and leave XOUT pin open (Figure 52).
Be careful that the maximum value of the oscillation frequency when using the external clock differs from the value when using the ceramic resonator (refer to the recommended operating condition).
Also, note that the RAM back-up mode (POF instruction) cannot be used when using the external clock.

(3) Clock control register MR

Register MR controls the selection of operation mode. Set the contents of this register through register A with the TMRA instruction. In addition, the TAMR instruction can be used to transfer the contents of register MR to register A.



Note: Externally connect a damping resistor Rd depending on the oscillation frequency.
(A feedback resistor is built-in.)
Use the resonator manufacturer's recommended value because constants such as capacitance depend on the resonator.

Fig 51. Ceramic resonator external circuit

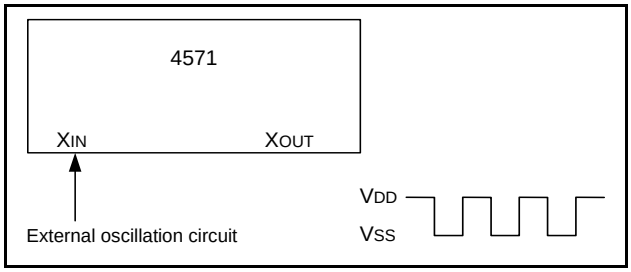


Fig 52. External clock input circuit

Table 23 Return source and return condition

Clock control register MR		at reset : 1111_2		at RAM back-up : 1111_2	R/W TAMR/TMRA
MR_3	Operation mode selection bits	MR_3	MR_2	Operation mode	
		0	0	Through mode (frequency not divided)	
MR_2		0	1	Frequency divided by 2 mode	
		1	0	Frequency divided by 4 mode	
		1	1	Frequency divided by 8 mode	
MR_1	Not used	0	This bit has no function, but read/write is enabled.		
		1			
MR_0	Not used	0	This bit has no function, but read/write is enabled.		
		1			

Note 1."R" represents read enabled, and "W" represents write enabled.

QzROM Writing Mode

In the QzROM writing mode, the user ROM area can be rewritten while the microcomputer is mounted on-board by using a serial pro-grammer which is applicable for this microcomputer. Table 24 lists the pin description (QzROM writing mode) and Figure 53 shows the pin connections.
Refer to Figure 54 for examples of a connection with a serial pro-grammer.
Contact the manufacturer of your serial programmer for serial pro-grammer. Refer to the user's manual of your serial programmer for details on how to use it.

Table 24 Pin description (QzROM writing mode)

Pin	Name	I/O	Function
V_{DD}	Power source	–	• Power supply voltage pin.
V_{SS}	GND	–	• GND pin.
K	V_{PP} input	–	• QzROM programmable power source pin.
$P0_1$	SDA input/output	I/O	• QzROM serial data I/O pin.
$P0_0$	SCLK input	Input	• QzROM serial clock input pin.
$P1_0$	$\overline{PGM}$ input	Input	• QzROM read/program pulse input pin.
$\overline{RESET}$	Reset input	Input	• Reset input pin. • Input "L" level signal.
X_{IN}	Clock input	–	• Either connect an oscillation circuit or connect X_{IN} pin to V_{SS} and leave the X_{OUT} pin open.
X_{OUT}	Clock output	–	
$P0_2$, $P0_3$, $P1_1$–$P1_3$, $P2_0$/INT0, $P2_1$/INT1, $P3_0$, $P3_1$, D_0–D_3, D4/CNTR0, C/CNTR1	I/O port	I/O	• Input "H" or "L" level signal or leave the pin open.

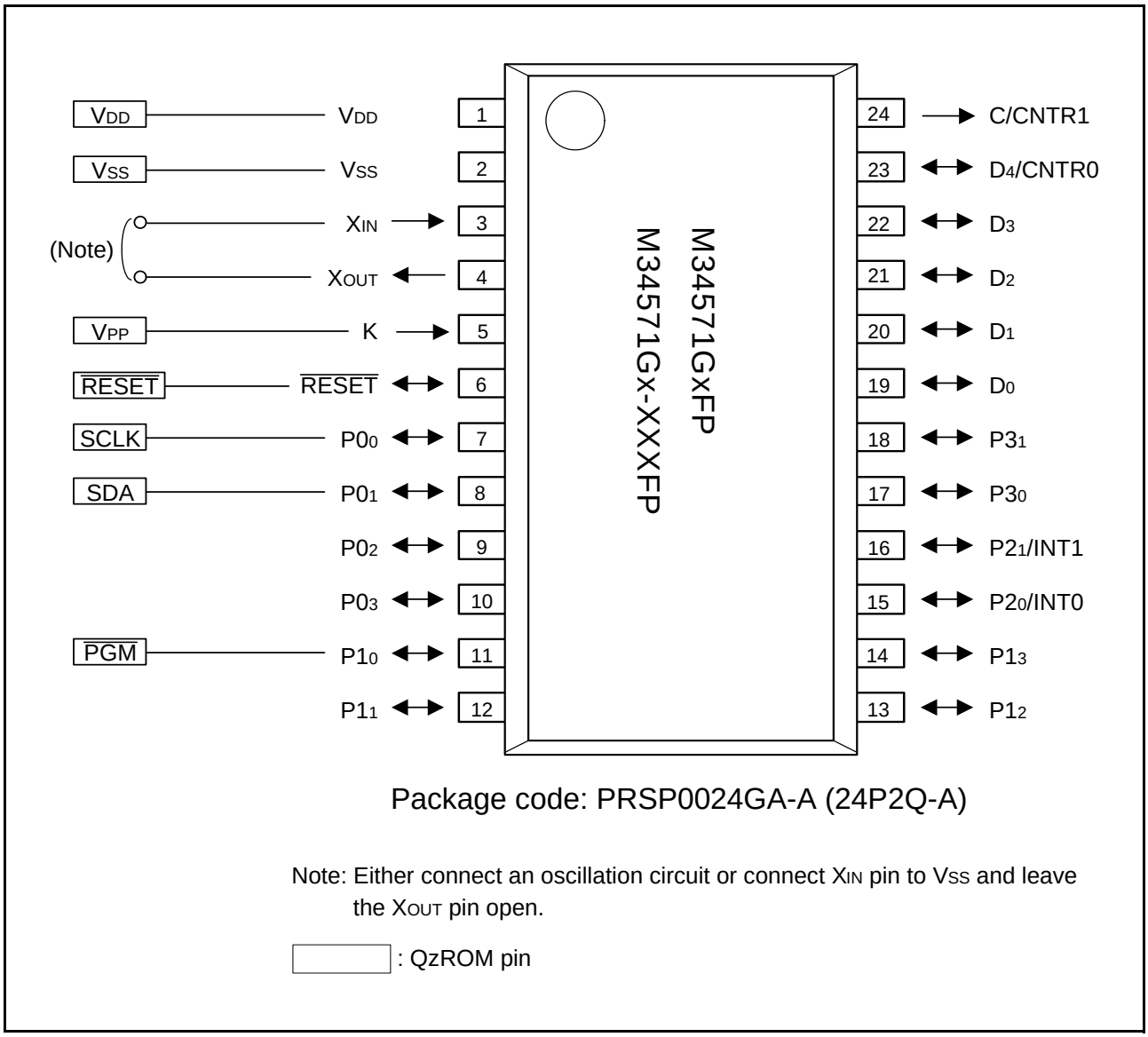


Fig 53. Pin connection diagram

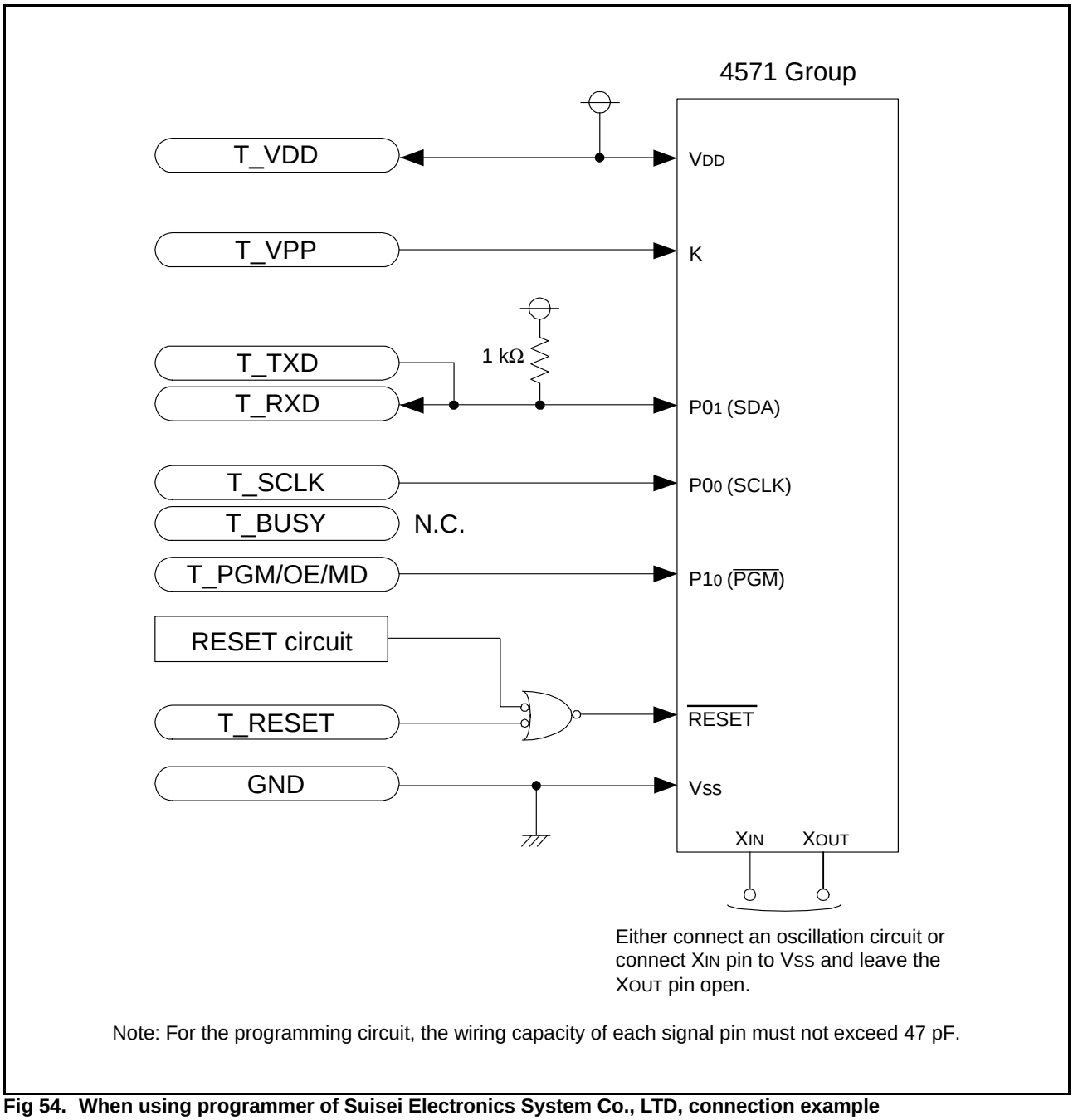


Fig 54. When using programmer of Suisei Electronics System Co., LTD, connection example

DATA REQUIRED FOR QzROM WRITING ORDERS

The following are necessary when ordering a QzROM product shipped after writing:
1. QzROM Writing Confirmation Form*
2. Mark Specification Form*
3. ROM data.....Mask file
* For the QzROM writing confirmation form and the mark specification form, refer to the “Renesas Technology Corp.” Homepage (http://www.renesas.com/homepage.jsp).
Note that we cannot deal with special font marking (customer's trademark etc.) in QzROM microcomputer.

LIST OF PRECAUTIONS

(1) Noise and latch-up prevention

Connect a capacitor on the following condition to prevent noise and latch-up;

- connect a bypass capacitor (approx. 0.1 μF) between pins V_{DD} and V_{SS} at the shortest distance,
- equalize its wiring in width and length, and
- use relatively thick wire.

Port K is also used as V_{PP} pin. Accordingly, when using this pin, connect this pin to V_{SS} or V_{DD}. Do not leave this pin open. When port is used for key matrix, connect it to V_{DD} through a pull-up resistor.

(2) Note on Power Source Voltage

When the power source voltage value of a microcomputer is less than the value which is indicated as the recommended operating conditions, the microcomputer does not operate normally and may perform unstable operation.
In a system where the power source voltage drops slowly when the power source voltage drops or the power supply is turned off, reset a microcomputer when the supply voltage is less than the recommended operating conditions and design a system not to cause errors to the system by this unstable operation.

(3) Register initial values 1

The initial value of the following registers are undefined after system is released from reset. After system is released from reset, set initial values.

- Register Z (2 bits)
- Register D (3 bits)
- Register E (8 bits)

(4) Register initial values 2

The initial value of the following registers are undefined at RAM back-up. After system is returned from RAM back-up, set initial values.

- Register Z (2 bits)
- Register X (4 bits)
- Register Y (4 bits)
- Register D (3 bits)
- Register E (8 bits)

(5) Program counter

Make sure that the PC_H does not specify after the last page of the built-in ROM.

(6) Stack registers (SKS)

Stack registers (SKs) are eight identical registers, so that subroutines can be nested up to 8 levels. However, one of stack registers is used respectively when using an interrupt service routine and when executing a table reference instruction. Accordingly, be careful not to over the stack when performing these operations together.

(7) Multifunction

- The input of D_4 can be used even when CNTR0 (output) is selected. The input/output of D_4 can be used even when CNTR0 (input) is selected. Be careful when using inputs of both CNTR0 and D_4 since the input threshold value of CNTR0 pin is different from that of port D_4.
- "H" output function of port C can be used even when the CNTR1 (output) is used.
- The input/output of $P2_0$ can be used even when INT0 is used. Be careful when using inputs of both INT0 and $P2_0$ since the input threshold value of INT0 pin is different from that of port $P2_0$.
- The input/output of $P2_1$ can be used even when INT1 is used. Be careful when using inputs of both INT1 and $P2_1$ since the input threshold value of INT1 pin is different from that of port $P2_1$.

(8) Power-on reset

When the built-in power-on reset circuit is used, set the time for the supply voltage to rise from 0 V to the minimum voltage of recommended operating conditions to 100 μs or less.
If the rising time exceeds 100 μs, connect a capacitor between the $\overline{\text{RESET}}$ pin and Vss at the shortest distance, and input "L" level to $\overline{\text{RESET}}$ pin until the value of supply voltage reaches the minimum operating voltage.

(9) POF instruction

When the POF instruction is executed continuously after the EPOF instruction, system enters the RAM back-up state.
Note that system cannot enter the RAM back-up state when executing only the POF instruction.
Be sure to disable interrupts by executing the DI instruction before executing the EPOF instruction and the POF instruction continuously.

(10)P2$_0$/INT0 pin

(1) Bit 3 of register I1
When the input of the P2$_0$/INT0 pin is controlled with the bit 3 of register I1 in software, be careful about the following notes.

- Depending on the input state of the P2$_0$/INT0 pin, the external 0 interrupt request flag (EXF0) may be set when the bit 3 of register I1 is changed. In order to avoid the occurrence of an unexpected interrupt, clear the bit 0 of register V1 to "0" (refer to (1) in Figure 55) and then, change the bit 3 of register I1.
In addition, execute the SNZ0 instruction to clear the EXF0 flag to "0" after executing at least one instruction (refer to (2) in Figure 55).
Also, set the NOP instruction for the case when a skip is performed with the SNZ0 instruction (refer to (3) in Figure 55).

```
    ⋮
LA   4          ; (×××0₂)
TV1A            ; The SNZ0 instruction is valid ..... (1)
LA   8          ; (1×××₂)
TI1A            ; Control of INT0 pin input is changed
NOP             ..... (2)
SNZ0            ; The SNZ0 instruction is executed
                  (EXF0 flag cleared)
NOP             ..... (3)
    ⋮

×: these bits are not used here.
```

Fig 55. External 0 interrupt program example-1

(2) Bit 3 of register I1
When the bit 3 of register I1 is cleared to "0", the RAM back-up mode is selected and the input of INT0 pin is disabled, be careful about the following notes.

- When the INT0 pin input is disabled (register I1$_3$ = "0"), set the key-on wakeup of INT0 pin to be invalid (register L1$_0$ = "0") before system enters to the RAM back-up mode. (refer to (1) in Figure 56).

```
    ⋮
LA   0          ; (×××0₂)
TL1A            ; INT0 key-on wakeup disabled .....(1)
DI
EPOF
POF             ; RAM back-up
    ⋮

×: these bits are not used here.
```

Fig 56. External 0 interrupt program example-2

(3) Bit 2 of register I1
When the interrupt valid waveform of the P2$_0$/INT0 pin is changed with the bit 2 of register I1 in software, be careful about the following notes.

- Depending on the input state of the P2$_0$/INT0 pin, the external 1 interrupt request flag (EXF0) may be set when the bit 2 of register I1 is changed. In order to avoid the occurrence of an unexpected interrupt, clear the bit 0 of register V1 to "0" (refer to (1) in Figure 57) and then, change the bit 2 of register I1 is changed.
In addition, execute the SNZ0 instruction to clear the EXF0 flag to "0" after executing at least one instruction (refer to (2) in Figure 57).
Also, set the NOP instruction for the case when a skip is performed with the SNZ0 instruction (refer to (3) in Figure 57).

```
    ⋮
LA   4          ; (×××0₂)
TV1A            ; The SNZ0 instruction is valid .....(1)
LA   12         ; (1×××₂)
TI1A            ; Interrupt valid waveform is changed
NOP             .....(2)
SNZ0            ; The SNZ0 instruction is executed
                  (EXF0 flag cleared)
NOP             .....(3)
    ⋮

×: these bits are not used here.
```

Fig 57. External 0 interrupt program example-3

(11)P21/INT1 pin

(1) Bit 3 of register I2
When the input of the P21/INT1 pin is controlled with the bit 3 of register I2 in software, be careful about the following notes.

- Depending on the input state of the P21/INT1 pin, the external 1 interrupt request flag (EXF1) may be set when the bit 3 of register I2 is changed. In order to avoid the occurrence of an unexpected interrupt, clear the bit 1 of register V1 to "0" (refer to (1) in Figure 58) and then, change the bit 3 of register I2.
In addition, execute the SNZ1 instruction to clear the EXF1 flag to "0" after executing at least one instruction (refer to (2) in Figure 58).
Also, set the NOP instruction for the case when a skip is performed with the SNZ1 instruction (refer to (3) in Figure 58).

```
    ⋮
LA   4        ; (××0×2)
TV1A          ; The SNZ1 instruction is valid ..... (1)
LA   8        ; (1×××2)
TI1A          ; Control of INT1 pin input is changed
NOP           ..... (2)
SNZ0          ; The SNZ1 instruction is executed
                (EXF1 flag cleared)
NOP           ..... (3)
    ⋮

×: these bits are not used here.
```

Fig 58. External 1 interrupt program example-1

(2) Bit 3 of register I2
When the bit 3 of register I2 is cleared to "0", the RAM back-up mode is selected and the input of INT1 pin is disabled, be careful about the following notes.

- When the INT1 pin input is disabled (register I23 = "0"), set the key-on wakeup of INT1 pin to be invalid (register L20 = "0") before system enters to the RAM back-up mode. (refer to (1) in Figure 59)

.

```
    ⋮
LA   0        ; (×0××2)
TL1A          ; INT1 key-on wakeup disabled .....(1)
DI
EPOF
POF           ; RAM back-up
    ⋮

×: these bits are not used here.
```

Fig 59. External 1 interrupt program example-2

(3) Bit 2 of register I2
When the interrupt valid waveform of the P21/INT1 pin is changed with the bit 2 of register I2 in software, be careful about the following notes.

- Depending on the input state of the P21/INT1 pin, the external 1 interrupt request flag (EXF1) may be set when the bit 2 of register I2 is changed. In order to avoid the occurrence of an unexpected interrupt, clear the bit 1 of register V1 to "0" (refer to (1) in Figure 60) and then, change the bit 2 of register I2 is changed.
In addition, execute the SNZ1 instruction to clear the EXF1 flag to "0" after executing at least one instruction (refer to (2) in Figure 60).
Also, set the NOP instruction for the case when a skip is performed with the SNZ1 instruction (refer to (3) in Figure 60).

```
    ⋮
LA   4        ; (××0×2)
TV1A          ; The SNZ1 instruction is valid .....(1)
LA   12       ; (1×××2)
TI1A          ; Interrupt valid waveform is changed
NOP           .....(2)
SNZ0          ; The SNZ1 instruction is executed
                (EXF1 flag cleared)
NOP           .....(3)
    ⋮

×: these bits are not used here.
```

Fig 60. External 1 interrupt program example-3

(12)Prescaler

Stop prescaler counting and then execute the TABPS instruction to read its data.
Stop prescaler counting and then execute the TPSAB instruction to write data to prescaler.

(13)Timer count source

Stop timer 1, 2 or 3 counting to change its count source.

(14)Reading the count value

Stop timer 1, 2 or 3 counting and then execute the TAB1, TAB2 or TAB3 instruction to read its data.

(15)Writing to the timer

Stop timer 1, 2 or 3 counting and then execute the T1AB, T2AB, T3AB or T3R3L instruction to write data to timer.

(16)Writing to reload register

In order to write a data to the reload register R1 while the timer 1 is operating, execute the TR1AB instruction except a timing of the timer 1 underflow.
In order to write a data to the reload register R3H while the timer 3 is operating, execute the T3HAB instruction except a timing of the timer 3 underflow.

(17)PWM signal

If the timer 3 count stop timing and the timer 3 underflow timing overlap during output of the PWM signal, a hazard may occur in the PWM output waveform.
When "H" interval expansion function of the PWM signal is used, set "1" or more to reload register R3H.
Set the port C output latch to "0" to output the PWM signal from C/CNTR1 pin.

(18)Prescaler, timer 1, timer 2 and timer 3 count start timing and count time when operation starts

Count starts from the first rising edge of the count source (2) in Figure 61 after prescaler and timer operations start (1) in Figure 61.
Time to first underflow (3) in Figure 61 is shorter (for up to 1 period of the count source) than time among next underflow (4) in Figure 61 by the timing to start the timer and count source operations after count starts.
When selecting CNTR0 input as the count source of timer 1, timer 1 operates synchronizing with the count edge (falling edge or rising edge) of CNTR0 input selected by software.

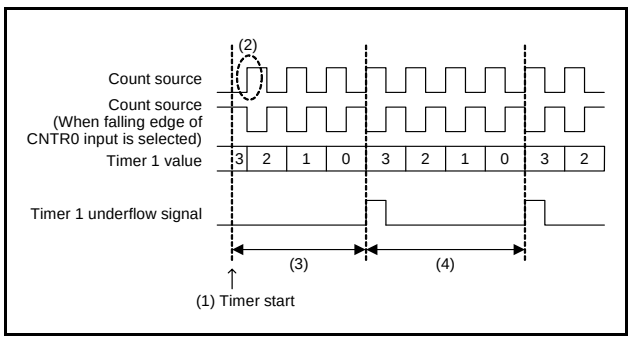


Fig 61. Timer count start timing and count time when operation starts

(19)Watchdog timer

- The watchdog timer function is valid after system is released from reset. When not using the watchdog timer function, execute the DWDT instruction and the WRST instruction continuously, and clear the WEF flag to "0" to stop the watchdog timer function.
- The contents of WDF1 flag and timer WDT are initialized at the RAM back-up mode.
- When using the watchdog timer and the RAM back-up mode, initialize the WDF1 flag with the WRST instruction just before the microcomputer enters the RAM back-up state.
  Also, set the NOP instruction after the WRST instruction, for the case when a skip is performed with the WRST instruction.

(20)External clock

Be careful that the maximum value of the oscillation frequency when using the external clock differs from the value when using the ceramic resonator (refer to the recommended operating condition).
Also, note that the RAM back-up mode (POF instruction) cannot be used when using the external clock.

(21)QzROM

(1) Be careful not to apply overvoltage to MCU. The contents of QzROM may be overwritten because of overvoltage. Take care especially at turning on the power.
(2) As for the product shipped in blank, Renesas does not perform the writing test to user ROM area after the assembly process though the QzROM writing test is performed enough before the assembly process. Therefore, a writing error of approx.0.1 % may occur. Moreover, please note the contact of cables and foreign bodies on a socket, etc. because a writing environment may cause some writing errors.

(22)Notes On ROM Code Protect (QzROM product shipped after writing)

As for the QzROM product shipped after writing, the ROM code protect is specified according to the ROM option setup data in the mask file which is submitted at ordering.
The ROM option setup data in the mask file is "00_{16}" for protect enabled or "FF_{16}" for protect disabled.
Note that the mask file which has nothing at the ROM option data or has the data other than "00_{16}" and "FF_{16}" can not be accepted.

NOTES ON NOISE

Countermeasures against noise are described below.
The following countermeasures are effective against noise in theory, however, it is necessary not only to take measures as follows but to evaluate before actual use.

1. Shortest wiring length

(1) Wiring for $\overline{\text{RESET}}$ pin

Make the length of wiring which is connected to the $\overline{\text{RESET}}$ pin as short as possible. Especially, connect a capacitor across the $\overline{\text{RESET}}$ pin and the VSS pin with the shortest possible wiring.

<Reason>
In order to reset a microcomputer correctly, 1 machine cycle or more of the width of a pulse input into the $\overline{\text{RESET}}$ pin is required.
If noise having a shorter pulse width than this is input to the $\overline{\text{RESET}}$ input pin, the reset is released before the internal state of the microcomputer is completely initialized.
This may cause a program runaway.

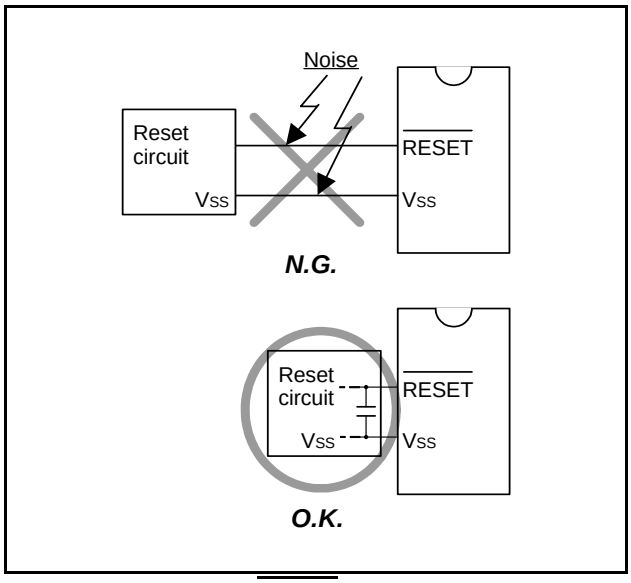


Fig 62. Wiring for the $\overline{\text{RESET}}$ pin

(2) Wiring for clock input/output pins

- Make the length of wiring which is connected to clock I/O pins as short as possible.
- Make the length of wiring across the grounding lead of a capacitor which is connected to an oscillator and the VSS pin of a microcomputer as short as possible.
- Separate the VSS pattern only for oscillation from other VSS patterns.

<Reason>
If noise enters clock I/O pins, clock waveforms may be deformed. This may cause a program failure or program runaway. Also, if a potential difference is caused by the noise between the VSS level of a microcomputer and the VSS level of an oscillator, the correct clock will not be input in the microcomputer.

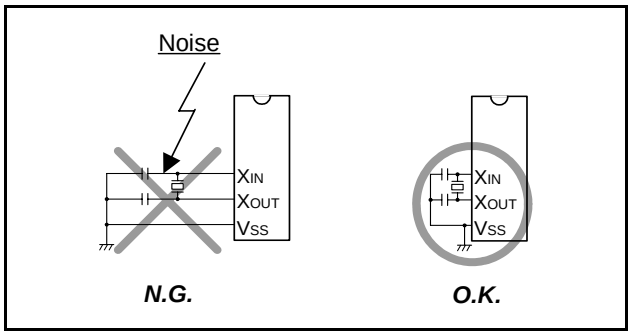


Fig 63. Wiring for clock I/O pins

(3) Port K Wiring

Do not leave port K open. Always connect it to the VDD pin or VSS pin using the thickest wire at the shortest distance.
When port K is used for key matrix, connect it to the VDD pin through a pull-up resistor.
In that case too, place a pull-up resistor close to port K and connect it to port K or the VDD pin using the thickest wire at the shortest distance as above.

<Reason>
Port K is also used as the power source input pin (VPP pin) for the built-in QzROM.
When programming to the QzROM, the impedance of port K is low so that the electric writing current will flow into the QzROM. This allows noise to enter easily. If noise enters from port K, abnormal instruction codes or data are read from the QzROM, which may cause a program runaway.

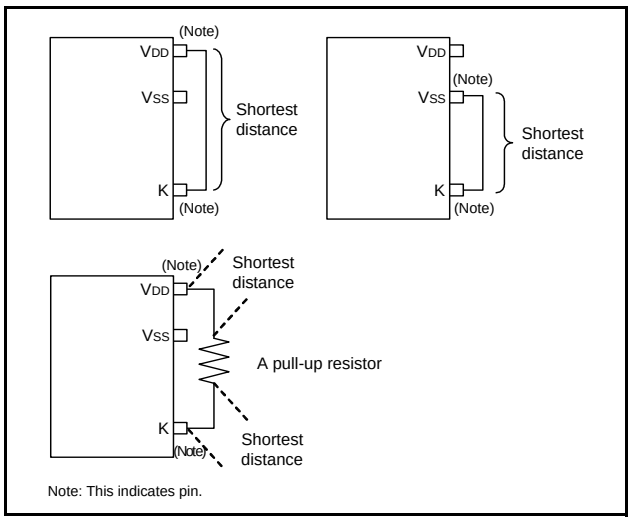


Fig 64. Wiring for port K

2. Connection of bypass capacitor across Vss line and VDD line

Connect an approximately 0.1 µF bypass capacitor across the Vss line and the VDD line as follows:

- Connect a bypass capacitor across the Vss pin and the VDD pin at equal length.
- Connect a bypass capacitor across the Vss pin and the VDD pin with the shortest possible wiring.
- Use lines with a larger diameter than other signal lines for Vss line and VDD line.
- Connect the power source wiring via a bypass capacitor to the Vss pin and the VDD pin.

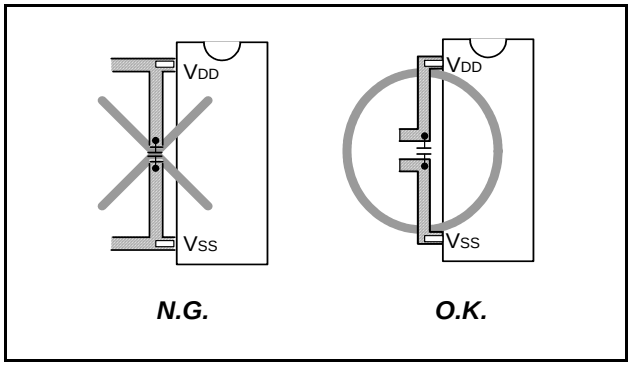


Fig 65. Bypass capacitor across the Vss line and the VDD line

3. Wiring to analog input pins

- Connect an approximately 100 Ω to 1 kΩ resistor to an analog signal line which is connected to an analog input pin in series. Besides, connect the resistor to the microcomputer as close as possible.
- Connect an approximately 1000 pF capacitor across the Vss pin and the analog input pin. Besides, connect the capacitor to the Vss pin as close as possible. Also, connect the capacitor across the analog input pin and the Vss pin at equal length.

<Reason>
Signals which is input in an analog input pin (such as an A/D converter/comparator input pin) are usually output signals from sensor. The sensor which detects a change of event is installed far from the printed circuit board with a microcomputer, the wiring to an analog input pin is longer necessarily. This long wiring functions as an antenna which feeds noise into the microcomputer, which causes noise to an analog input pin.

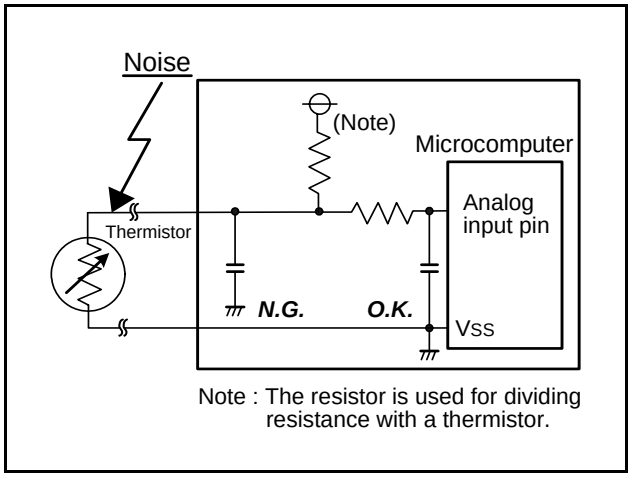


Fig 66. Analog signal line and a resistor and a capacitor

4. Oscillator concerns

Take care to prevent an oscillator that generates clocks for a microcomputer operation from being affected by other signals.

(1) Keeping oscillator away from large current signal lines

Install a microcomputer (and especially an oscillator) as far as possible from signal lines where a current larger than the tolerance of current value flows.

<Reason>

In the system using a microcomputer, there are signal lines for controlling motors, LEDs, and thermal heads or others. When a large current flows through those signal lines, strong noise occurs because of mutual inductance.

(2) Installing oscillator away from signal lines where potential levels change frequently

Install an oscillator and a connecting pattern of an oscillator away from signal lines where potential levels change frequently. Also, do not cross such signal lines over the clock lines or the signal lines which are sensitive to noise.

<Reason>

Signal lines where potential levels change frequently (such as the CNTR pin signal line) may affect other lines at signal rising edge or falling edge. If such lines cross over a clock line, clock waveforms may be deformed, which causes a microcomputer failure or a program runaway.

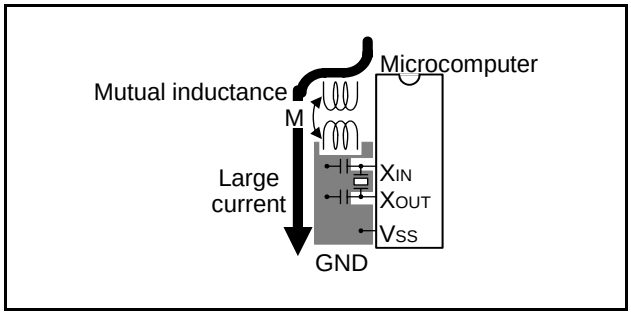


Fig 67. Wiring for a large current signal line

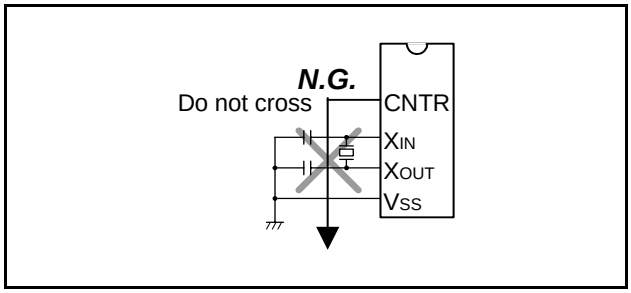


Fig 68. Wiring to a signal line where potential levels change frequently

(3) Oscillator protection using Vss pattern

As for a two-sided printed circuit board, print a Vss pattern on the underside (soldering side) of the position (on the component side) where an oscillator is mounted.

Connect the Vss pattern to the microcomputer Vss pin with the shortest possible wiring. Besides, separate this Vss pattern from other Vss patterns.

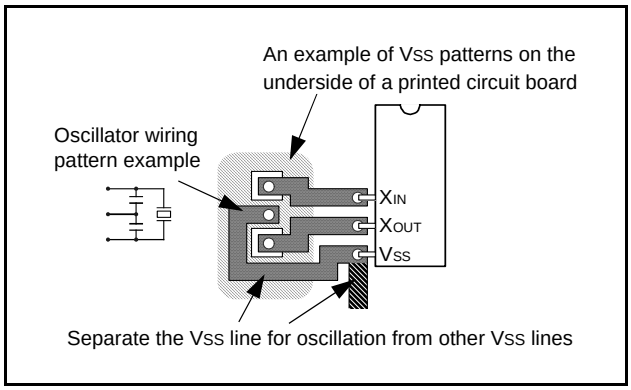


Fig 69. Vss pattern on the underside of an oscillator

5. Setup for I/O ports

Setup I/O ports using hardware and software as follows:

<Hardware>

- Connect a resistor of 100 Ω or more to an I/O port in series.

<Software>

- As for an input port, read data several times by a program for checking whether input levels are equal or not.
- As for an output port or an I/O port, since the output data may reverse because of noise, rewrite data to its port latch at fixed periods.
- Rewrite data to pull-up control registers at fixed periods.

6. Providing of watchdog timer function by software

If a microcomputer runs away because of noise or others, it can be detected by a software watchdog timer and the microcomputer can be reset to normal operation. This is equal to or more effective than program runaway detection by a hardware watchdog timer. The following shows an example of a watchdog timer provided by software. In the following example, to reset a microcomputer to normal operation, the main routine detects errors of the interrupt processing routine and the interrupt processing routine detects errors of the main routine.

This example assumes that interrupt processing is repeated multiple times in a single main routine processing.

<The main routine>

- Assigns a single word of RAM to a software watchdog timer (SWDT) and writes the initial value N in the SWDT once at each execution of the main routine. The initial value N should satisfy the following condition:
  N+1 ≥ (Counts of interrupt processing executed in each main routine)
  As the main routine execution cycle may change because of an interrupt processing or others, the initial value N should have a margin.
- Watches the operation of the interrupt processing routine by comparing the SWDT contents with counts of interrupt processing after the initial value N has been set.
- Detects that the interrupt processing routine has failed and deter-mines to branch to the program initialization routine for recovery processing in the following case:
  If the SWDT contents do not change after interrupt processing.

<The interrupt processing routine>

- Decrements the SWDT contents by 1 at each interrupt processing.
- Determines that the main routine operates normally when the SWDT contents are reset to the initial value N at almost fixed cycles (at the fixed interrupt processing count).
- Detects that the main routine has failed and determines to branch to the program initialization routine for recovery processing in the following case:
  If the SWDT contents are not initialized to the initial value N but continued to decrement and if they reach 0 or less.

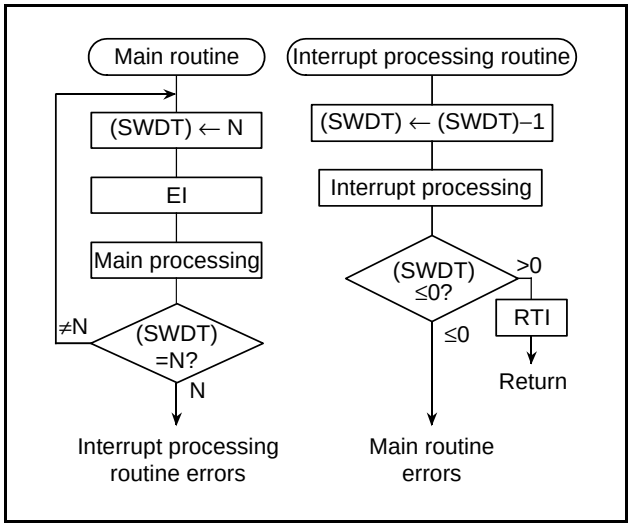


Fig 70. Watchdog timer by software

CONTROL REGISTERS

Interrupt control register V1		at reset : 0000_2	at RAM back-up : 0000_2	R/W TAV1/TV1A
$V1_3$	Timer 2 interrupt enable bit	0	Interrupt disabled (SNZT2 instruction is valid)	
		1	Interrupt enabled (SNZT2 instruction is invalid)	
$V1_2$	Timer 1 interrupt enable bit	0	Interrupt disabled (SNZT1 instruction is valid)	
		1	Interrupt enabled (SNZT1 instruction is invalid)	
$V1_1$	External 1 interrupt enable bit	0	Interrupt disabled (SNZ1 instruction is valid)	
		1	Interrupt enabled (SNZ1 instruction is invalid)	
$V1_0$	External 0 interrupt enable bit	0	Interrupt disabled (SNZ0 instruction is valid)	
		1	Interrupt enabled (SNZ0 instruction is invalid)	

Interrupt control register V2		at reset : 0000_2	at RAM back-up : 0000_2	R/W TAV2/TV2A
$V2_3$	Voltage drop detector interrupt enable bit	0	Interrupt disabled (SNZVD instruction is valid)	
		1	Interrupt enabled (SNZVD instruction is invalid)	
$V2_2$	Not used	0	This bit has no function, but read/write is enabled.	
		1		
$V2_1$	Not used	0	This bit has no function, but read/write is enabled.	
		1		
$V2_0$	Timer 3 interrupt enable bit	0	Interrupt disabled (SNZT3 instruction is valid)	
		1	Interrupt enabled (SNZT3 instruction is invalid)	

Interrupt control register I1		at reset : 0000_2	at RAM back-up : state retained	R/W TAI1/TI1A
$I1_3$	INT0 pin input control bit (Note 2)	0	INT0 pin input disabled	
		1	INT0 pin input enabled	
$I1_2$	Interrupt valid waveform for INT0 pin/ return level selection bit (Note 2)	0	Falling waveform (“L” level of INT0 pin is recognized with the SNZI0 instruction)/“L” level	
		1	Rising waveform (“H” level of INT0 pin is recognized with the SNZI0 instruction)/“H” level	
$I1_1$	INT0 pin edge detection circuit control bit	0	One-sided edge detected	
		1	Both edges detected	
$I1_0$	INT0 pin timer 1 control enable bit	0	Timer 1 disabled	
		1	Timer 1 enabled	

Interrupt control register I2		at reset : 0000_2	at RAM back-up : state retained	R/W TAI2/TI2A
$I2_3$	INT1 pin input control bit (Note 3)	0	INT0 pin input disabled	
		1	INT0 pin input enabled	
$I2_2$	Interrupt valid waveform for INT1 pin/ return level selection bit (Note 3)	0	Falling waveform (“L” level of INT0 pin is recognized with the SNZI1 instruction)/“L” level	
		1	Rising waveform (“H” level of INT0 pin is recognized with the SNZI1 instruction)/“H” level	
$I2_1$	INT1 pin edge detection circuit control bit	0	One-sided edge detected	
		1	Both edges detected	
$I2_0$	Not used	0	This bit has no function, but read/write is enabled.	
		1		

Note 1.”R” represents read enabled, and “W” represents write enabled.
Note 2.When the contents of $I1_2$ and $I1_3$ are changed, the external interrupt request flag EXF0 may be set.
Note 3.When the contents of $I2_2$ and $I2_3$ are changed, the external interrupt request flag EXF1 may be set.

Timer control register PA		at reset : 00_2	at RAM back-up : 00_2	W TPAA
PA_1	Prescaler count source selection bit	0	Instruction clock (INSTCK)	
		1	Instruction clock divided by 4 (INSTCK)/4	
PA_0	Prescaler control bit	0	Stop (state initialized)	
		1	Operating	

Timer control register W1		at reset : 0000_2		at RAM back-up : state retained	R/W TAW1/TW1A
$W1_3$	Timer 1 count auto-stop circuit selection bit (Note 2)	0	Timer 1 count auto-stop circuit not selected		
		1	Timer 1 count auto-stop circuit selected		
$W1_2$	Timer 1 control bit	0	Stop (state retained)		
		1	Operating		
$W1_1$ $W1_0$	Timer 1 count source selection bits	$W1_1$	$W1_0$	Count source	
		0	0	PWM output (PWMOUT)	
		0	1	Prescaler output (ORCLK)	
		1	0	System clock (STCK)	
		1	1	CNTR0 input	

Timer control register W2		at reset : 0000_2		at RAM back-up : state retained	R/W TAW2/TW2A
$W2_3$	CNTR0 pin function selection bit	0	Timer 1 underflow signal divided by 2 output		
		1	Timer 2 underflow signal divided by 2 output		
$W2_2$	Timer 2 control bit	0	Stop (state retained)		
		1	Operating		
$W2_1$ $W2_0$	Timer 2 count source selection bits	$W2_1$	$W2_0$	Count source	
		0	0	PWM output (PWMOUT)	
		0	1	Prescaler output (ORCLK)	
		1	0	System clock (STCK)	
		1	1	Timer 1 underflow signal (T1UDF)	

Timer control register W3		at reset : 0000_2	at RAM back-up : 0000_2	R/W TAW3/TW3A
$W3_3$	CNTR1 pin output control bit	0	CNTR1 pin output invalid	
		1	CNTR1 pin output valid	
$W3_2$	PWM signal “H” interval expansion function control bit	0	PWM signal “H” interval expansion function invalid	
		1	PWM signal “H” interval expansion function valid	
$W3_1$	Timer 3 control bit	0	Stop (state retained)	
		1	Operating	
$W3_0$	Timer 3 count source selection bit	0	X_{IN} input	
		1	Prescaler output/2	

Timer control register W5		at reset : 0000_2	at RAM back-up : state retained	R/W TAW5/TW5A
$W5_3$	Timer 1 count start synchronous circuit selection bit (Note 3)	0	Count start synchronous circuit not selected	
		1	Count start synchronous circuit selected	
$W5_2$	CNTR0 pin input count edge selection bit	0	Falling edge	
		1	Rising edge	
$W5_1$	CNTR 1 pin output auto-control circuit selection bit	0	Output auto-control circuit not selected	
		1	Output auto-control circuit selected	
$W5_0$	D_4/CNTR0 pin function selection bit	0	D_4 (I/O) / CNTR0 (input)	
		1	D_4 (input) /CNTR0 (I/O)	

Note 1.“R” represents read enabled, and “W” represents write enabled.
Note 2.This function is valid only when the INT0 pin/timer 1 control is enabled ($I1_0$ =“1”) and the timer 1 count start synchronous circuit is selected ($W5_3$ =“1”).
Note 3.This function is valid only when the INT0 pin/timer 1 control is enabled ($I1_0$ =“1”).

Clock control register MR		at reset : 1111_2		at RAM back-up : 1111_2	R/W TAMR/TMRA
MR_3	Operation mode selection bits	MR_3	MR_2	Operation mode	
		0	0	Through mode (frequency not divided)	
MR_2		0	1	Frequency divided by 2 mode	
		1	0	Frequency divided by 4 mode	
		1	1	Frequency divided by 8 mode	
MR_1	Not used	0	This bit has no function, but read/write is enabled.		
		1			
MR_0	Not used	0	This bit has no function, but read/write is enabled.		
		1			

Key-on wakeup control register K0		at reset : 0000_2	at RAM back-up : state retained	R/W TAK0/TK0A
$K0_3$	Port $P0_3$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$K0_2$	Port $P0_2$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$K0_1$	Port $P0_1$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$K0_0$	Port $P0_0$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	

Key-on wakeup control register K1		at reset : 0000_2	at RAM back-up : state retained	R/W TAK1/TK1A
$K1_3$	Port $P1_3$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$K1_2$	Port $P1_2$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$K1_1$	Port $P1_1$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$K1_0$	Port $P1_0$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	

Key-on wakeup control register K2		at reset : 0000_2	at RAM back-up : state retained	R/W TAK2/TK2A
$K2_3$	Not used	0	This bit has no function, but read/write is enabled.	
		1		
$K2_2$	Port K key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$K2_1$	Port $P2_1$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	
$K2_0$	Port $P2_0$ key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	

Key-on wakeup control register L1		at reset : 0000_2	at RAM back-up : state retained	R/W TAL1/TL1A
$L1_3$	INT1 pin return condition selection bit	0	Return by level	
		1	Return by edge	
$L1_2$	INT1 pin valid waveform/ level selection bit	0	Falling waveform/“L” level	
		1	Rising waveform/“H” level	
$L1_1$	INT0 pin return condition selection bit	0	Return by level	
		1	Return by edge	
$L1_0$	INT0 pin key-on wakeup control bit	0	Key-on wakeup not used	
		1	Key-on wakeup used	

Note 1.“R” represents read enabled, and “W” represents write enabled.

Pull-up control register PU0		at reset : 0000_2	at RAM back-up : state retained	R/W TAPU0/TPU0A
$PU0_3$	Port $P0_3$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	
$PU0_2$	Port $P0_2$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	
$PU0_1$	Port $P0_1$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	
$PU0_0$	Port $P0_0$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	

Pull-up control register PU1		at reset : 0000_2	at RAM back-up : state retained	R/W TAPU1/TPU1A
$PU1_3$	Port $P1_3$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	
$PU1_2$	Port $P1_2$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	
$PU1_1$	Port $P1_1$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	
$PU1_0$	Port $P1_0$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	

Pull-up control register PU2		at reset : 0000_2	at RAM back-up : state retained	R/W TAPU2/TPU2A
$PU2_3$	Not used	0	This bit has no function, but read/write is enabled.	
		1		
$PU2_2$	Not used	0	This bit has no function, but read/write is enabled.	
		1		
$PU2_1$	Port $P2_1$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	
$PU2_0$	Port $P2_0$ pull-up transistor control bit	0	Pull-up transistor OFF	
		1	Pull-up transistor ON	

Port output structure control register FR0		at reset : 0000_2	at RAM back-up : state retained	W TFR0A
$FR0_3$	Not used	0	This bit has no function, but read/write is enabled.	
		1		
$FR0_2$	Not used	0	This bit has no function, but read/write is enabled.	
		1		
$FR0_1$	Port $P3_1$ output structure selection bit	0	N-channel open-drain output	
		1	CMOS output	
$FR0_0$	Port $P3_0$ output structure selection bit	0	N-channel open-drain output	
		1	CMOS output	

Port output structure control register FR1		at reset : 0000_2	at RAM back-up : state retained	W TFR1A
$FR1_3$	Port D_3 output structure selection bit	0	N-channel open-drain output	
		1	CMOS output	
$FR1_2$	Port D_2 output structure selection bit	0	N-channel open-drain output	
		1	CMOS output	
$FR1_1$	Port D_1 output structure selection bit	0	N-channel open-drain output	
		1	CMOS output	
$FR1_0$	Port D_0 output structure selection bit	0	N-channel open-drain output	
		1	CMOS output	

Note 1. “R” represents read enabled, and “W” represents write enabled.

INSTRUCTIONS

Each instruction is described as follows;
1. Index list of instruction function
2. Machine instructions (index by alphabet)
3. Machine instructions (index by function)
4. Instruction code table

The symbols shown below are used in the following list of instruction function and the machine instructions.

SYMBOL

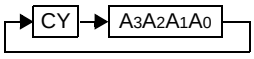
Symbol	Contents	Symbol	Contents
A	Register A (4 bits)	T1F	Timer 1 interrupt request flag
B	Register B (4 bits)	T2F	Timer 2 interrupt request flag
DR	Register DR (3 bits)	T3F	Timer 3 interrupt request flag
E	Register E (8 bits)	WDF1	Watchdog timer flag
V1	Interrupt control register V1 (4 bits)	WEF	Watchdog timer enable flag
V2	Interrupt control register V2 (4 bits)	INTE	Interrupt enable flag
I1	Interrupt control register I1 (4 bits)	EXF0	External 0 interrupt request flag
I2	Interrupt control register I2 (4 bits)	EXF1	External 1 interrupt request flag
PA	Timer control register PA (2 bits)	VDF	Voltage drop detection circuit interrupt request flag
W1	Timer control register W1 (4 bits)	P	Power down flag
W2	Timer control register W2 (4 bits)	D	Port D (5 bits)
W3	Timer control register W3 (4 bits)	P0	Port P0 (4 bits)
W5	Timer control register W5 (4 bits)	P1	Port P1 (4 bits)
MR	Clock control register MR (4 bits)	P2	Port P2 (2 bits)
K0	Key-on wakeup control register K0 (4 bits)	P3	Port P3 (2 bits)
K1	Key-on wakeup control register K1 (4 bits)		
K2	Key-on wakeup control register K2 (4 bits)	x	Hexadecimal variable
L1	Key-on wakeup control register L1 (4 bits)	y	Hexadecimal variable
PU0	Pull-up control register PU0 (4 bits)	z	Hexadecimal variable
PU1	Pull-up control register PU1 (4 bits)	p	Hexadecimal variable
PU2	Pull-up control register PU2 (4 bits)	n	Hexadecimal constant
FR0	Port output structure control register FR0 (4 bits)	i	Hexadecimal constant
FR1	Port output structure control register FR1 (4 bits)	j	Hexadecimal constant
X	Register X (4 bits)	A_3 A_2 A_1 A_0	Binary notation of hexadecimal variable A (same for others)
Y	Register Y (4 bits)		
Z	Register Z (2 bits)		
DP	Data pointer (10 bits) (It consists of registers X, Y, and Z)	←	Direction of data movement
		( )	Contents of registers and memories
PC	Program counter (14 bits)	–	Negate, Flag unchanged after executing instruction
PC_H	High-order 7 bits of program counter	M (DP)	RAM address pointed by the data pointer
PC_L	Low-order 7 bits of program counter	a	Label indicating address a_6 a_5 a_4 a_3 a_2 a_1 a_0
SK	Stack register (14 bits × 8)	p, a	Label indicating address a_6 a_5 a_4 a_3 a_2 a_1 a_0 in page p_6 p_5 p_4 p_3 p_2 p_1 p_0
SP	Stack pointer (3 bits)		
CY	Carry flag		
RPS	Prescaler reload register (8 bits)	C	Hex. C + Hex. number x (also same for others)
R1L	Timer 1 reload register (8 bits)	+	
R2	Timer 2 reload register (8 bits)	x	
R3L	Timer 3 reload register (8 bits)	?	Decision of state shown before “?”
R3H	Timer 3 reload register (8 bits)	← →	Data exchange between a register and memory
PS	Prescaler		
T1	Timer 1	AND	Logical multiplication
T2	Timer 2	OR	Logical addition
T3	Timer 3		

Note 1. The 4571 Group just invalidates the next instruction when a skip is performed. The contents of program counter is not increased by 2. Accordingly, the number of cycles does not change even if skip is not performed. However, the cycle count becomes “1” if the TABP p, RT, or RTS instruction is skipped.

INDEX LIST OF INSTRUCTION FUNCTION

Grouping	Mnemonic	Function	Page
Register to register transfer	TAB	(A) ← (B)	88, 103
	TBA	(B) ← (A)	95, 103
	TAY	(A) ← (Y)	95, 103
	TYA	(Y) ← (A)	101, 103
	TEAB	(E_7–E_4) ← (B) (E_3–E_0) ← (A)	96, 103
	TABE	(B) ← (E_7–E_4) (A) ← (E_3–E_0)	89, 103
	TDA	(DR_2–DR_0) ← (A_2–A_0)	95, 103
	TAD	(A_2–A_0) ← (DR_2–DR_0) (A_3) ← 0	90, 103
	TAZ	(A_1, A_0) ← (Z_1, Z_0) (A_3, A_2) ← 0	95, 103
	TAX	(A) ← (X)	94, 103
	TASP	(A_2–A_0) ← (SP_2–SP_0) (A_3) ← 0	93, 103
RAM addresses	LXY x, y	(X) ← x, x = 0 to 15 (Y) ← y, y = 0 to 15	77, 103
	LZ z	(Z) ← z, z = 0 to 3	77, 103
	INY	(Y) ← (Y) + 1	76, 103
	DEY	(Y) ← (Y) – 1	74, 103
RAM to register transfer	TAM j	(A) ← (M(DP)) (X) ← (X)EXOR(j) j = 0 to 15	91, 103
	XAM j	(A) ←→ (M(DP)) (X) ← (X)EXOR(j) j = 0 to 15	102, 103
	XAMD j	(A) ←→ (M(DP)) (X) ← (X)EXOR(j) j = 0 to 15 (Y) ← (Y) – 1	102, 103
	XAMI j	(A) ←→ (M(DP)) (X) ← (X)EXOR(j) j = 0 to 15 (Y) ← (Y) + 1	102, 103
	TMA j	(M(DP)) ← (A) (X) ← (X)EXOR(j) j = 0 to 15	98, 103

M34571G4: p=0 to 31
M34571G6: p=0 to 47
M34571GD: p=0 to 127

Grouping	Mnemonic	Function	Page
Arithmetic operation	LA n	(A) ← n n = 0 to 15	76, 105
	TABP p	(SP) ← (SP) + 1 (SK(SP)) ← (PC) (PC_H) ← p (PC_L) ← (DR_2–DR_0, A_3–A_0) (UPTF) = 1, (DR_2) ← 0 (DR_1, DR_0) ← (ROM(PC))$_{9, 8}$ (B) ← (ROM(PC))$_{7–4}$ (A) ← (ROM(PC))$_{3–0}$ (PC) ← (SK(SP)) (SP) ← (SP) – 1	89, 105
	AM	(A) ← (A) + (M(DP))	71, 105
	AMC	(A) ← (A) + (M(DP)) + (CY) (CY) ← Carry	71, 105
	A n	(A) ← (A) + n n = 0 to 15	71, 105
	AND	(A) ← (A)AND(M(DP))	71, 105
	OR	(A) ← (A)OR(M(DP))	78, 105
	SC	(CY) ← 1	82, 105
	RC	(CY) ← 0	80, 105
	SZC	(CY) = 0 ?	86, 105
	CMA	(A) ← $\overline{(A)}$	73, 105
	RAR		79, 105
Bit operation	SB j	(Mj(DP)) ← 1 j = 0 to 3	81, 105
	RB j	(Mj(DP)) ← 0 j = 0 to 3	79, 105
	SZB j	(Mj(DP)) = 0 ? j = 0 to 3	86, 105
Comparison operation	SEAM	(A) = (M(DP)) ?	83, 107
	SEA n	(A) = n n = 0 to 15	83, 107
Branch operation	B a	(PC_L) ← a_6–a_0	72, 107
	BL p, a	(PC_H) ← p (PC_L) ← a_6–a_0	72, 107
	BLA p	(PC_H) ← p (PC_L) ← (DR_2–DR_0, A_3–A_0)	72, 107

INDEX LIST OF INSTRUCTION FUNCTION (continued)

Grouping	Mnemonic	Function	Page
Subroutine operation	BM a	(SP) ← (SP) + 1 (SK(SP)) ← (PC) (PC_H) ← 2 (PC_L) ← a_6–a_0	72, 107
	BML p, a	(SP) ← (SP) + 1 (SK(SP)) ← (PC) (PC_H) ← p (PC_L) ← a_6–a_0	73, 107
	BMLA p	(SP) ← (SP) + 1 (SK(SP)) ← (PC) (PC_H) ← p (PC_L) ← (DR_2–DR_0, A_3–A_0)	73, 107
Return operation	RTI	(PC) ← (SK(SP)) (SP) ← (SP) – 1	81, 107
	RT	(PC) ← (SK(SP)) (SP) ← (SP) – 1	80, 107
	RTS	(PC) ← (SK(SP)) (SP) ← (SP) – 1	81, 107
Interrupt operation	DI	(INTE) ← 0	74, 109
	EI	(INTE) ← 1	74, 109
	SNZ0	$V1_0$ = 0 : (EXF0) = 1 ? (EXF0) ← 0 $V1_0$ = 1 : NOP	83, 109
	SNZI0	$I1_2$ = 0 : (INT0) = “L” ? $I1_2$ = 1 : (INT0) = “H” ?	84, 109
	SNZ1	$V1_1$ = 0 : (EXF1) = 1 ? (EXF1) ← 0 $V1_1$ = 1 : NOP	83, 109
	SNZI1	$I2_2$ = 0 : (INT1) = “L” ? $I2_2$ = 1 : (INT1) = “H” ?	84, 109
	TAV1	(A) ← (V1)	93, 109
	TV1A	(V1) ← (A)	100, 109
	TAV2	(A) ← (V2)	93, 109
	TV2A	(V2) ← (A)	100, 109
	TAI1	(A) ← (I1)	90, 109
	TI1A	(I1) ← (A)	96, 109
	TAI2	(A) ← (I2)	90, 109
	TI2A	(I2) ← (A)	97, 109

Grouping	Mnemonic	Function	Page
Timer operation	TPAA	(PA_0) ← (A_0)	98, 109
	TAW1	(A) ← (W1)	93, 109
	TW1A	(W1) ← (A)	100, 109
	TAW2	(A) ← (W2)	94, 109
	TW2A	(W2) ← (A)	101, 109
	TAW3	(A) ← (W3)	94, 109
	TW3A	(W3) ← (A)	101, 109
	TAW5	(A) ← (W5)	94, 109
	TW5A	(W5) ← (A)	101, 109
	TABPS	(B) ← (TPS_7–TPS_4) (A) ← (TPS_3–TPS_0)	89, 111
	TPSAB	(RPS_7–RPS_4) ← (B) (TPS_7–TPS_4) ← (B) (RPS_3–RPS_0) ← (A) (TPS_3–TPS_0) ← (A)	99, 111
	TAB1	(B) ← ($T1_7$–$T1_4$) (A) ← ($T1_3$–$T1_0$)	88, 111
	T1AB	($R1_7$–$R1_4$) ← (B) ($T1_7$–$T1_4$) ← (B) ($R1_3$–$R1_0$) ← (A) ($T1_3$–$T1_0$) ← (A)	87, 111
	TR1AB	($R1_7$–$R1_4$) ← (B) ($R1_3$–$R1_0$) ← (A)	100, 111
	TAB2	(B) ← ($T2_7$–$T2_4$) (A) ← ($T2_3$–$T2_0$)	88, 111
	T2AB	($R2_7$–$R2_4$) ← (B) ($T2_7$–$T2_4$) ← (B) ($R2_3$–$R2_0$) ← (A) ($T2_3$–$T2_0$) ← (A)	87, 111
	TAB3	(B) ← ($T3_7$–$T3_4$) (A) ← ($T3_3$–$T3_0$)	89, 111
	T3AB	($R3L_7$–$R3L_4$) ← (B) ($T3_7$–$T3_4$) ← (B) ($R3L_3$–$R3L_0$) ← (A) ($T3_3$–$T3_0$) ← (A)	87, 111
	T3R3L	($T3_7$–$T3_0$) ← ($R3L_7$–$R3L_0$)	88, 111
	T3HAB	($R3H_7$–$R3H_4$) ← (B) ($R3H_3$–$R3H_0$) ← (A)	87, 111

M34571G4: p=0 to 31
M34571G6: p=0 to 47
M34571GD: p=0 to 127

INDEX LIST OF INSTRUCTION FUNCTION (continued)

Grouping	Mnemonic	Function	Page
Timer operation	SNZT1	$V1_2$ = 0 : (T1F) = 1 ? (T1F) ← 0 $V1_2$ = 1 : SNZT1=NOP	84, 111
	SNZT2	$V1_3$ = 0 : (T2F) = 1 ? (T2F) ← 0 $V1_3$ = 1 : SNZT2=NOP	85, 111
	SNZT3	$V2_0$ = 0 : (T3F) = 1 ? (T3F) ← 0 $V2_0$ = 1 : SNZT3=NOP	85, 111
Input/Output operation	IAP0	(A) ← (P0)	75, 113
	OP0A	(P0) ← (A)	77, 113
	IAP1	(A) ←(P1)	75, 113
	OP1A	(P1) ← (A)	78, 113
	IAP2	(A_1, A_0) ← ($P2_1$, $P2_0$) (A_3, A_2) ← 0	76, 113
	OP2A	($P2_1$, $P2_0$) ← (A_1, A_0)	78, 113
	IAP3	(A_1, A_0) ← ($P3_1$, $P3_0$) (A_3, A_2) ← 0	76, 113
	OP3A	($P3_1$, $P3_0$) ← (A_1, A_0)	78, 113
	CLD	(D) ← 1	73, 113
	RD	(D(Y)) ← 0 (Y) = 0 to 4	80, 113
	SD	(D(Y)) ← 1 (Y) = 0 to 4	82, 113
	SZD	(D(Y)) = 0 ? (Y) = 0 to 4	86, 113
	RCP	(C) ← (0)	80, 113
	SCP	(C) ← (1)	82, 113
	IAK	(A_0) ← (K) (A_3–A_1) ← 0	75, 113
	TFR0A	(FR0) ← (A)	96, 113
	TFR1A	(FR1) ← (A)	96, 113
	TAPU0	(A) ← (PU0)	92, 113
	TPU0A	(PU0) ← (A)	99, 113
	TAPU1	(A) ← (PU1)	92, 113
	TPU1A	(PU1) ← (A)	99, 113
	TAPU2	(A) ← (PU2)	92, 113
	TPU2A	(PU2) ← (A)	99, 113

Grouping	Mnemonic	Function	Page
Input/Output operation	TAK0	(A) ← (K0)	90, 115
	TK0A	(K0) ← (A)	97, 115
	TAK1	(A) ← (K1)	91, 115
	TK1A	(K1) ← (A)	97, 115
	TAK2	(A) ← (K2)	91, 115
	TK2A	(K2) ← (A)	97, 115
	TAL1	(A) ← (L1)	91, 115
	TL1A	(L1) ← (A)	98, 115
Other operation	TAMR	(A) ← (MR)	92, 115
	TMRA	(MR) ← (A)	98, 115
	NOP	(PC) ← (PC)+1	77, 115
	POF	RAM back-up	79, 115
	EPOF	POF instruction valid	75, 115
	SNZP	(P) = 1 ?	84, 115
	SNZVD	$V2_3$ = 0 : (VDF) = 1? $V2_3$ = 0 : NOP	85, 115
	WRST	(WDF1) = 1 ? (WDF_1) ← 0	102, 115
	DWDT	Stop of watchdog timer function enabled	74, 115
	SRST	System reset	85, 115
	RUPT	(UPTF) ←0	81, 115
	SUPT	(UPTF) ←1	86, 115
	RBK	p_6 ← 0 when TABP p instruction is executed	79, 115
	SBK	p_6 ← 1 when TABP p instruction is executed	82, 115

MACHINE INSTRUCTIONS (INDEX BY ALPHABET)

A n (Add n and accumulator)

Instruction code: D9 … D0

0	0	0	1	1	0	n	n	n	n	2	0	6	n	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	Overflow = 0

Operation: (A) ← (A) + n
n = 0 to 15

Grouping: Arithmetic operation

Description: Adds the value n in the immediate field to register A, and stores a result in register A.
The contents of carry flag CY remains unchanged.
Skips the next instruction when there is no overflow as the result of operation.
Executes the next instruction when there is overflow as the result of operation.

AM (Add accumulator and Memory)

Instruction code: D9 … D0

0	0	0	0	0	0	1	0	1	0	2	0	0	A	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (A) ← (A)Å{(M(DP))

Grouping: Arithmetic operation

Description: Adds the contents of M(DP) to register A.
Stores the result in register A. The contents of carry flag CY remains unchanged.

AMC (Add accumulator, Memory and Carry)

Instruction code: D9 … D0

0	0	0	0	0	0	1	0	1	1	2	0	0	B	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	0/1	-

Operation: (A) ← (A) + (M(DP)) + (CY)
(CY) ← Carry

Grouping: Arithmetic operation

Description: Adds the contents of M(DP) and carry flag CY to register A. Stores the result in register A and carry flag CY.

AND (logical AND between accumulator and memory)

Instruction code: D9 … D0

0	0	0	0	0	1	1	0	0	0	2	0	1	8	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (A) ← (A) AND (M(DP))

Grouping: Arithmetic operation

Description: Takes the AND operation between the contents of register A and the contents of M(DP), and stores the result in register A.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

B a (Branch to address a)

Instruction code

D_9									D_0				
0	1	1	a_6	a_5	a_4	a_3	a_2	a_1	a_0	2	1	8+a	a	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: $(PC_L) \leftarrow a_6$ to a_0

Grouping: Branch operation

Description: Branch within a page : Branches to address a in the identical page.

Note: Specify the branch address within the page including this instruction.

BL p,a (Branch Long to address a in page p)

Instruction code

D_9									D_0				
0	0	1	1	1	p_4	p_3	p_2	p_1	p_0	2	0	E+p	p	16
1	p_6	p_5	a_6	a_5	a_4	a_3	a_2	a_1	a_0	2	2	a	a	16

Number of words	Number of cycles	Flag CY	Skip condition
2	2	-	-

Operation: $(PC_H) \leftarrow p$
$(PC_L) \leftarrow a_6$ to a_0

Grouping: Branch operation

Description: Branch out of a page : Branches to address a in page p.

Note: M34571G4 : p = 0 to 31
M34571G6 : p = 0 to 47
M34571GD : p = 0 to 127

BLA p (Branch Long to address (D)+(A) in page p)

Instruction code

D_9									D_0				
0	0	0	0	0	1	0	0	0	0	2	0	1	0	16
1	p_6	p_5	p_4	0	0	p_3	p_2	p_1	p_0	2	2	p	p	16

Number of words	Number of cycles	Flag CY	Skip condition
2	2	-	-

Operation: $(PC_H) \leftarrow p$
$(PC_L) \leftarrow (DR_2$–R_0, A_3–$A_0)$

Grouping: Branch operation

Description: Branch out of a page : Branches to address (DR_2 DR_1 DR_0 A_3 A_2 A_1 A_0)$_2$ specified by registers D and A in page p.

Note: M34571G4 : p = 0 to 31
M34571G6 : p = 0 to 47
M34571GD : p = 0 to 127

BM a (Branch and Mark to address a in page 2)

Instruction code

D_9									D_0				
0	1	0	a_6	a_5	a_4	a_3	a_2	a_1	a_0	2	1	a	a	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: $(SP) \leftarrow (SP) + 1$
$(SK(SP)) \leftarrow (PC)$
$(PC_H) \leftarrow 2$
$(PC_L) \leftarrow a_6$–a_0

Grouping: Subroutine call operation

Description: Call the subroutine in page 2 : Calls the subroutine at address a in page 2.

Note: Subroutine extending from page 2 to another page can also be called with the BM instruction when it starts on page 2.
Be careful not to over the stack because the maximum level of subroutine nesting is 8.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

BML p,a (Branch and Mark Long to address a in page p)

Instruction code

D_9									D_0				
0	0	1	1	0	p_4	p_3	p_2	p_1	p_0	2	0	c+p	p	16
1	p_6	p_5	a_6	a_5	a_4	a_3	a_2	a_1	a_0	2	2	a	a	16

Number of words	Number of cycles	Flag CY	Skip condition
2	2	-	-

Operation: (SP) ← (SP) + 1
(SK(SP)) ← (PC)
(PC_H) ← p
(PC_L) ← a_6–a_0

Grouping: Subroutine call operation

Description: Call the subroutine : Calls the subroutine at address a in page p.

Note: M34571G4 : p = 0 to 31
M34571G6 : p = 0 to 47
M34571GD : p = 0 to 127
Be careful not to over the stack because the maximum level of subroutine nesting is 8.

BMLA p (Branch and Mark Long to address (D)+(A) in page p)

Instruction code

D_9									D_0				
0	0	0	0	1	1	0	0	0	0	2	0	3	0	16
1	p_6	p_5	p_4	0	0	p_3	p_2	p_1	p_0	2	2	p	p	16

Number of words	Number of cycles	Flag CY	Skip condition
2	2	-	-

Operation: (SP) ← (SP) + 1
(SK(SP)) ← (PC)
(PC_H) ← p
(PC_L) ← (DR_2–DR_0, A_3–A_0)

Grouping: Subroutine call operation

Description: Call the subroutine : Calls the subroutine at address (DR_2 DR_1 DR_0 A_3 A_2 A_1 A_0)$_2$ specified by registers D and A in page p.

Note: M34571G4 : p = 0 to 31
M34571G6 : p = 0 to 47
M34571GD : p = 0 to 127
Be careful not to over the stack because the maximum level of subroutine nesting is 8.

CLD (CLear port D)

Instruction code

D_9									D_0				
0	0	0	0	0	1	0	0	0	1	2	0	1	1	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (D) ← (1)

Grouping: Input/Output operation

Description: Sets (1) to port D.

CMA (CoMplement of Accumulator)

Instruction code

D_9									D_0				
0	0	0	0	0	1	1	1	0	0	2	0	1	C	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (A) ←$\overline{(A)}$

Grouping: Arithmetic operation

Description: Stores the one's complement for register A's contents in register A.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

DEY (DEcrement register Y)

Instruction code: D9 … D0: 0 0 0 0 0 1 0 1 1 1 (2) 0 1 7 (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	(Y) = 15

Operation: (Y) ← (Y) –1

Grouping: RAM addresses

Description: Subtracts 1 from the contents of register Y.
As a result of subtraction, when the contents of register Y is 15, the next instruction is skipped. When the contents of register Y is not 15, the next instruction is executed.

DI (Disable Interrupt)

Instruction code: D9 … D0: 0 0 0 0 0 0 0 1 0 0 (2) 0 0 4 (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (INTE) ← 0

Grouping: Interrupt control operation

Description: Clears (0) to interrupt enable flag INTE, and disables the interrupt.

Note: Interrupt is disabled by executing the DI instruction after executing 1 machine cycle.

DWDT (Disable WatchDog Timer)

Instruction code: D9 … D0: 1 0 1 0 0 1 1 1 0 0 (2) 2 9 C (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: Stop of watchdog timer function enabled

Grouping: Other operation

Description: Stops the watchdog timer function by the WRST instruction after executing the DWDT instruction.

EI (Enable Interrupt)

Instruction code: D9 … D0: 0 0 0 0 0 0 0 1 0 1 (2) 0 0 5 (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (INTE) ← 1

Grouping: Interrupt control operation

Description: Sets (1) to interrupt enable flag INTE, and enables the interrupt.

Note: Interrupt is enabled by executing the EI instruction after executing 1 machine cycle.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

EPOF (Enable POF instruction)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 0 0 0 1 0 1 1 0 1 1 $_2$ = 0 5 B $_{16}$	1	1	-	-

Operation: POF instruction valid

Grouping: Other operation

Description: Makes the immediate after POF instruction valid by executing the EPOF instruction.

IAK (Input Accumulator from port K)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 1 0 0 1 1 0 1 1 1 1 $_2$ = 2 6 F $_{16}$	1	1	-	-

Operation: (A_0) ← (K)
(A_3–A_1) ← 0

Grouping: Input/Output operation

Description: Transfers the input of port K to the least significant bit (A_0) of register A. “0” is stored to the high-order 3 bits (A_3–A_1) of register A.

IAP0 (Input Accumulator from port P0)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 1 0 0 1 1 0 0 0 0 0 $_2$ = 2 6 0 $_{16}$	1	1	-	-

Operation: (A) ← (P0)

Grouping: Input/Output operation

Description: Transfers the input of port P0 to register A.

IAP1 (Input Accumulator from port P1)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 1 0 0 1 1 0 0 0 0 1 $_2$ = 2 6 1 $_{16}$	1	1	-	-

Operation: (A) ← (P1)

Grouping: Input/Output operation

Description: Transfers the input of port P1 to register A.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

IAP2 (Input Accumulator from port P2)

Instruction code: D9 … D0: 1 0 0 1 1 0 0 0 1 0 (2) 2 6 2 (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: $(A_1, A_0) \leftarrow (P2_1, P2_0)$
$(A_3, A_2) \leftarrow 0$

Grouping: Input/Output operation

Description: Transfers the input of port P2 to the low-order 2 bits (A_1, A_0) of register A.
"0" is stored to the high-order 2 bits (A_3, A_2) of register A.

IAP3 (Input Accumulator from port P3)

Instruction code: D9 … D0: 1 0 0 1 1 0 0 0 1 1 (2) 2 6 3 (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: $(A_1, A_0) \leftarrow (P3_1, P3_0)$
$(A_3, A_2) \leftarrow 0$

Grouping: Input/Output operation

Description: Transfers the input of port P3 to the low-order 2 bits (A_1, A_0) of register A.
"0" is stored to the high-order 2 bits (A_3, A_2) of register A.

INY (INcrement register Y)

Instruction code: D9 … D0: 0 0 0 0 0 1 0 0 1 1 (2) 0 1 3 (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	(Y) = 0

Operation: (Y) ← (Y) + 1

Grouping: RAM addresses

Description: Adds 1 to the contents of register Y. As a result of addition, when the contents of register Y is 0, the next instruction is skipped. When the contents of register Y is not 0, the next instruction is executed.

LA n (Load n in Accumulator)

Instruction code: D9 … D0: 0 0 0 1 1 1 n n n n (2) 0 7 n (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	Continuous description

Operation: (A) ← n
n = 0 to 15

Grouping: Arithmetic operation

Description: Loads the value n in the immediate field to register A.
When the LA instructions are continuously coded and executed, only the first LA instruction is executed and other LA instructions coded continuously are skipped.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

LXY x,y (Load register X and Y with x and y)

Instruction code

D_9									D_0				
1	1	x_3	x_2	x_1	x_0	y_3	y_2	y_1	y_0	2	3	x	y	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	Continuous description

Operation: (X) ← x x = 0 to 15
(Y) ← y y = 0 to 15

Grouping: RAM addresses

Description: Loads the value x in the immediate field to register X, and the value y in the immediate field to register Y. When the LXY instructions are continuously coded and executed, only the first LXY instruction is executed and other LXY instructions coded continuously are skipped.

LZ z (Load register Z with z)

Instruction code

D_9									D_0				
0	0	0	1	0	0	1	0	Z_1	Z_0	2	0	4	8 +z	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (Z) ← z z = 0 to 3

Grouping: RAM addresses

Description: Loads the value z in the immediate field to register Z.

NOP (No OPeration)

Instruction code

D_9									D_0				
0	0	0	0	0	0	0	0	0	0	2	0	0	0	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (PC) ← (PC) + 1

Grouping: Other operation

Description: No operation; Adds 1 to program counter value, and others remain unchanged.

OP0A (Output port P0 from Accumulator)

Instruction code

D_9									D_0				
1	0	0	0	1	0	0	0	0	0	2	2	2	0	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (P0) ← (A)

Grouping: Input/Output operation

Description: Outputs the contents of register A to port P0.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

OP1A (Output port P1 from Accumulator)

Instruction code: D9 … D0

1	0	0	0	1	0	0	0	0	1	2	2	2	1	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (P1) ← (A)

Grouping: Input/Output operation

Description: Outputs the contents of register A to port P1.

OP2A (Output port P2 from Accumulator)

Instruction code: D9 … D0

1	0	0	0	1	0	0	0	1	0	2	2	2	2	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: ($P2_1$, $P2_0$) ← (A_1, A_0)

Grouping: Input/Output operation

Description: Outputs the contents of the low-order 2 bits (A_1, A_0) of register A to port P2.

OP3A (Output port P3 from Accumulator)

Instruction code: D9 … D0

1	0	0	0	1	0	0	0	1	1	2	2	2	3	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: ($P3_1$, $P3_0$) ← (A_1, A_0)

Grouping: Input/Output operation

Description: Outputs the contents of the low-order 2 bits (A_1, A_0) of register A to port P3.

OR (logical OR between accumulator and memory)

Instruction code: D9 … D0

0	0	0	0	0	1	1	0	0	1	2	0	1	9	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (A) ← (A) OR (M(DP))

Grouping: Arithmetic operation

Description: Takes the OR operation between the contents of register A and the contents of M(DP), and stores the result in register A.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

POF (Power OFf)

Instruction code: D_9 … D_0: 0 0 0 0 0 0 0 1 0 (2) 0 0 2 (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: RAM back-up

Grouping: Other operation

Description: Puts the system in RAM back-up state by executing the POF instruction after executing the EPOF instruction.

Note: If the EPOF instruction is not executed just before this instruction, this instruction is equivalent to the NOP instruction.

RAR (Rotate Accumulator Right)

Instruction code: D_9 … D_0: 0 0 0 0 0 1 1 1 0 1 (2) 0 1 D (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	0/1	-

Operation: CY → $A_3A_2A_1A_0$ → CY

Grouping: Arithmetic operation

Description: Rotates 1 bit of the contents of register A including the contents of carry flag CY to the right.

RB j (Reset Bit)

Instruction code: D_9 … D_0: 0 0 0 1 0 0 1 1 j j (2) 0 4 C+j (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (Mj(DP)) ← 0
j = 0 to 3

Grouping: Bit operation

Description: Clears (0) the contents of bit j (bit specified by the value j in the immediate field) of M(DP).

RBK (Reset BanK flag))

Instruction code: D_9 … D_0: 0 0 0 1 0 0 1 1 0 0 (2) 0 4 0 (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: p_6 ← 0 when TABP p instruction is executed.

Grouping: Other operation

Description: Sets referring data area to pages 0 to 63 when the TABP p instruction is executed. This instruction is valid only for the TABP p instruction.

Note: This instruction cannot be used for the M34571G4/G6.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

RC (Reset Carry flag)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 D0: 0 0 0 0 0 0 0 1 1 0 (2) 0 0 6 (16)	1	1	0	-

Operation: (CY) ← 0

Grouping: Arithmetic operation

Description: Clears (0) to carry flag CY.

RCP (Reset Port C)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 D0: 1 0 1 0 0 0 1 1 0 0 (2) 2 8 C (16)	1	1	-	-

Operation: (C) ← 0

Grouping: Input/Output operation

Description: Clears (0) to port C.

RD (Reset port D specified by register Y)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 D0: 0 0 0 0 0 1 0 1 0 0 (2) 0 1 4 (16)	1	1	-	-

Operation: (D(Y)) ← 0
However,
(Y) = 0 to 4

Grouping: Input/Output operation

Description: Clears (0) to a bit of port D specified by register Y.

Note: (Y) = 0 to 4.
Do not execute this instruction if values except above are set to register Y.

RT (ReTurn from subroutine)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 D0: 0 0 0 1 0 0 0 1 0 0 (2) 0 4 4 (16)	1	2	-	-

Operation: (PC) ← (SK(SP))
(SP) ← (SP) –1

Grouping: Return operation

Description: Returns from subroutine to the routine called the subroutine.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

RTI (ReTurn from Interrupt)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 … D0: 0 0 0 1 0 0 0 1 1 0 (2) 0 4 6 (16)	1	2	-	-

Operation: (PC) ← (SK(SP))
(SP) ← (SP) – 1

Grouping: Return operation

Description: Returns from interrupt service routine to main routine. Returns each value of data pointer (X, Y, Z), carry flag, skip status, NOP mode status by the continuous description of the LA/LXY instruction, register A and register B to the states just before interrupt.

RTS (ReTurn from subroutine and Skip)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 … D0: 0 0 0 1 0 0 0 1 0 1 (2) 0 4 5 (16)	1	2	-	Skip at uncondition

Operation: (PC) ← (SK(SP))
(SP) ← (SP) – 1

Grouping: Return operation

Description: Returns from subroutine to the routine called the subroutine, and skips the next instruction at uncondition.

RUPT (Reset UPT flag)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 … D0: 0 0 0 1 0 1 1 0 0 0 (2) 0 5 8 (16)	1	1	-	-

Operation: (UPTF) ←0

Grouping: Other operation

Description: Clears (0) to the high-order bit reference enable flag UPTF.

Note: Even when the table reference instruction (TABP p) is executed, the high-order 2 bits of ROM reference data is not transferred to register D.

SB j (Set Bit)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 … D0: 0 0 0 1 0 1 1 1 j j (2) 0 5 C+j (16)	1	1	-	-

Operation: (Mj(DP)) ← 1
j = 0 to 3

Grouping: Bit operation

Description: Sets (1) the contents of bit j (bit specified by the value j in the immediate field) of M(DP).

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

SBK (Set BanK flag)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 0 0 0 1 0 0 0 0 0 1 $_2$ = 0 4 1 $_{16}$	1	1	-	-

Operation: $p_6 \leftarrow 1$ when TABP p instruction is executed.

Grouping: Other operation

Description: Sets referring data area to pages 64 to 127 when the TABP p instruction is executed. This instruction is valid only for the TABP p instruction.

Note: This instruction cannot be used for the M34571G4/G6.

SC (Set Carry flag)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 0 0 0 0 0 0 0 1 1 1 $_2$ = 0 0 7 $_{16}$	1	1	1	-

Operation: (CY) ← 1

Grouping: Arithmetic operation

Description: Sets (1) to carry flag CY.

SCP (Set Port C)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 1 0 1 0 0 0 1 1 0 1 $_2$ = 2 8 D $_{16}$	1	1	-	-

Operation: (C) ← 1

Grouping: Input/Output operation

Description: Sets (1) to port C.

SD (Set port D specified by register Y)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 0 0 0 0 0 1 0 1 0 1 $_2$ = 0 1 5 $_{16}$	1	1	-	-

Operation: (D(Y)) ← 1
(Y) = 0 to 4

Grouping: Input/Output operation

Description: Sets (1) to a bit of port D specified by register Y.

Note: (Y) = 0 to 4.
Do not execute this instruction if values except above are set to register Y.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

SEA n (Skip Equal, Accumulator with immediate data n)

Instruction code

D_9									D_0					
0	0	0	0	1	0	0	1	0	1	2	0	2	5	16
0	0	0	1	1	1	n	n	n	n	2	0	7	n	16

Number of words	Number of cycles	Flag CY	Skip condition
2	2	-	(A) = n n = 0 to 15

Operation: (A) = n ?
n = 0 to 15

Grouping: Comparison operation

Description: Skips the next instruction when the contents of register A is equal to the value n in the immediate field.
Executes the next instruction when the contents of register A is not equal to the value n in the immediate field.

SEAM (Skip Equal, Accumulator with Memory)

Instruction code

D_9									D_0					
0	0	0	0	1	0	0	1	1	0	2	0	2	6	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	(A) = (M(DP))

Operation: (A) = (M(DP)) ?

Grouping: Comparison operation

Description: Skips the next instruction when the contents of register A is equal to the contents of M(DP).
Executes the next instruction when the contents of register A is not equal to the contents of M(DP).

SNZ0 (Skip if Non Zero condition of external interrupt 0 request flag)

Instruction code

D_9									D_0					
0	0	0	0	1	1	1	0	0	0	2	0	3	8	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	$V1_0$ = 0 : (EXF0) = 1

Operation: $V1_0$ = 0 : (EXF0) = 1 ?
(EXF0) ← 0
$V1_0$ = 1 : SNZ0 = NOP
($V1_0$: bit 0 of the interrupt control register V1)

Grouping: Interrupt operation

Description: When $V1_0$ = 0 : Clears (0) to the EXF0 flag and skips the next instruction when external 0 interrupt request flag EXF0 is “1”. When the EXF0 flag is “0”, executes the next instruction.
When $V1_0$ = 1 : This instruction is equivalent to the NOP instruction.

SNZ1 (Skip if Non Zero condition of external interrupt 1 request flag)

Instruction code

D_9									D_0					
0	0	0	0	1	1	1	0	0	1	2	0	3	9	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	$V1_1$ = 0 : (EXF1) = 1

Operation: $V1_1$ = 0 : (EXF1) = 1 ?
(EXF1) ← 0
$V1_1$ = 1 : SNZ1 = NOP
($V1_1$: bit 1 of the interrupt control register V1)

Grouping: Interrupt operation

Description: When $V1_1$ = 0 : Clears (0) to the EXF1 flag and skips the next instruction when external 1 interrupt request flag EXF1 is “1”. When the EXF1 flag is “0”, executes the next instruction.
When $V1_1$ = 1 : This instruction is equivalent to the NOP instruction.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

SNZI0 (Skip if Non Zero condition of external Interrupt 0 input pin)

Instruction code: D_9 … D_0: 0 0 0 0 1 1 1 0 1 0 $_2$ 0 3 A $_{16}$

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	$I1_2$ = 0 : (INT0) = "L" $I1_2$ = 1 : (INT0) = "H"

Operation: $I1_2$ = 0 : (INT0) = "L" ?
$I1_2$ = 1 : (INT0) = "H" ?
($I1_2$: bit 2 of the interrupt control register I1)

Grouping: Interrupt operation

Description: When $I1_2$ = 0 : Skips the next instruction when the level of INT0 pin is "L". Executes the next instruction when the level of INT0 pin is "H".
When $I1_2$ = 1 : Skips the next instruction when the level of INT0 pin is "H." Executes the next instruction when the level of INT0 pin is "L".

SNZI1 (Skip if Non Zero condition of external Interrupt 1 input pin)

Instruction code: D_9 … D_0: 0 0 0 0 1 1 1 0 1 1 $_2$ 0 3 B $_{16}$

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	$I2_2$ = 0 : (INT1) = "L" $I2_2$ = 1 : (INT1) = "H"

Operation: $I2_2$ = 0 : (INT1) = "L" ?
$I2_2$ = 1 : (INT1) = "H" ?

Grouping: Interrupt operation

Description: When $I2_2$ = 0 : Skips the next instruction when the level of INT1 pin is "L". Executes the next instruction when the level of INT1 pin is "H".
When $I2_2$ = 1 : Skips the next instruction when the level of INT1 pin is "H". Executes the next instruction when the level of INT1 pin is "L".

SNZP (Skip if Non Zero condition of Power down flag)

Instruction code: D_9 … D_0: 0 0 0 0 0 0 0 0 1 1 $_2$ 0 0 3 $_{16}$

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	(P) = 1

Operation: (P) = 1 ?

Grouping: Other operation

Description: Skips the next instruction when the P flag is "1".
After skipping, the P flag remains unchanged.
Executes the next instruction when the P flag is "0".

SNZT1 (Skip if Non Zero condition of Timer 1 interrupt request flag)

Instruction code: D_9 … D_0: 1 0 1 0 0 0 0 0 0 0 $_2$ 2 8 0 $_{16}$

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	$V1_2$ = 0 : (T1F) = 1

Operation: $V1_2$ = 0 : (T1F) = 1 ?
(T1F) ← 0
$V1_2$ = 1 : SNZT1 = NOP
($V1_2$ = bit 2 of interrupt control register V1)

Grouping: Timer operation

Description: When $V1_2$ = 0 : Clears (0) to the T1F flag and skips the next instruction when timer 1 interrupt request flag T1F is "1". When the T1F flag is "0," executes the next instruction.
When $V1_2$ = 1 : This instruction is equivalent to the NOP instruction.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

SNZT2 (Skip if Non Zero condition of Timer 2 interrupt request flag)

Instruction code: D9 … D0

1	0	1	0	0	0	0	0	0	1	2	2	8	1	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	$V1_3$ = 0 : (T2F) = 1

Operation: $V1_3$ = 0 : (T2F) = 1 ?
(T2F) ← 0
$V1_3$ = 1 : SNZT2 = NOP
($V1_3$ = bit 3 of interrupt control register V1)

Grouping: Timer operation

Description: When $V1_3$ = 0 : Clears (0) to the T2F flag and skips the next instruction when timer 2 interrupt request flag T2F is "1". When the T2F flag is "0", executes the next instruction.
When $V1_3$ = 1 : This instruction is equivalent to the NOP instruction.

SNZT3 (Skip if Non Zero condition of Timer 3 interrupt request flag)

Instruction code: D9 … D0

1	0	1	0	0	0	0	0	1	0	2	2	8	2	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	$V2_0$ = 0 : (T3F) = 1

Operation: $V2_0$ = 0 : (T3F) = 1 ?
(T3F) ← 0
$V2_0$ = 1 : SNZT3 = NOP

Grouping: Timer operation

Description: When $V2_0$ = 0 : Clears (0) to the T3F flag and skips the next instruction when timer 3 interrupt request flag T3F is "1". When the T3F flag is "0", executes the next instruction.
When $V2_0$ = 1 : This instruction is equivalent to the NOP instruction.

SNZVD (Skip if Non Zero condition of Voltage Detector interrupt request flag)

Instruction code: D9 … D0

1	0	1	0	0	0	1	0	1	0	2	2	8	A	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	$V2_3$ = 0 : (VDF) = 1

Operation: $V2_3$ = 0 : (VDF) = 1?
$V2_3$ = 1 : SNZVD = NOP

Grouping: Other operation

Description: When $V2_3$ = 0 : Skips the next instruction when voltage detector interrupt request flag VDF is "1". After skipping, clears (0) to the VDF flag. The VDF flag is not cleared to "0".
When $V2_3$ = 1 : This instruction is equivalent to the NOP instruction.

SRST (System ReSet)

Instruction code: D9 … D0

0	0	0	0	0	0	0	0	0	1	2	0	0	1	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: System reset

Grouping: Other operation

Description: System reset occurs.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

SUPT (Set UPT flag)

Instruction code: D9 … D0: 0 0 0 1 0 1 1 0 0 1 (2) 0 5 9 (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (UPTF) ←1

Grouping: Other operation

Description: Sets (1) to the high-order bit reference enable flag UPTF.

Note: When the table reference instruction (TABP p) is executed, the high-order 2 bits of ROM reference data is transferred to the low-order 2 bits of register D.

SZB j (Skip if Zero, Bit)

Instruction code: D9 … D0: 0 0 0 0 1 0 0 0 j j (2) 0 2 j (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	(Mj(DP)) = 0 j = 0 to 3

Operation: (Mj(DP)) = 0 ?
j = 0 to 3

Grouping: Bit operation

Description: Skips the next instruction when the contents of bit j (bit specified by the value j in the immediate field) of M(DP) is “0”.
Executes the next instruction when the contents of bit j of M(DP) is “1”.

SZC (Skip if Zero, Carry flag)

Instruction code: D9 … D0: 0 0 0 0 1 0 1 1 1 1 (2) 0 2 F (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	(CY) = 0

Operation: (CY) = 0 ?

Grouping: Arithmetic operation

Description: Skips the next instruction when the contents of carry flag CY is “0”.
After skipping, the CY flag remains unchanged.
Executes the next instruction when the contents of the CY flag is “1”.

SZD (Skip if Zero, port D specified by register Y)

Instruction code: D9 … D0: 0 0 0 0 1 0 0 1 0 0 (2) 0 2 4 (16)

0 0 0 0 1 0 1 0 1 1 (2) 0 2 B (16)

Number of words	Number of cycles	Flag CY	Skip condition
2	2	-	(D(Y)) = 0

Operation: (D(Y)) = 0 ?
(Y) = 0 to 4

Grouping: Input/Output operation

Description: Skips the next instruction when a bit of port D specified by register Y is “0”. Executes the next instruction when the bit is “1”.

Note: (Y) = 0 to 4.
Do not execute this instruction if values except above are set to register Y.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

T1AB (Transfer data to timer 1 and register R1 from Accumulator and register B)

Instruction code: D9 … D0

1	0	0	0	1	1	0	0	0	0	2	2	3	0	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation:
$(T1_7–T1_4) \leftarrow (B)$
$(R1_7–R1_4) \leftarrow (B)$
$(T1_3–T1_0) \leftarrow (A)$
$(R1_3–R1_0) \leftarrow (A)$

Grouping: Timer operation

Description: Transfers the contents of register B to the high-order 4 bits of timer 1 and timer 1 reload register R1. Transfers the contents of register A to the low-order 4 bits of timer 1 and timer 1 reload register R1.

T2AB (Transfer data to timer 2 and register R2 from Accumulator and register B)

Instruction code: D9 … D0

1	0	0	0	1	1	0	0	0	1	2	2	3	1	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation:
$(T2_7–T2_4) \leftarrow (B)$
$(R2_7–R2_4) \leftarrow (B)$
$(T2_3–T2_0) \leftarrow (A)$
$(R2_3–R2_0) \leftarrow (A)$

Grouping: Timer operation

Description: Transfers the contents of register B to the high-order 4 bits of timer 2 and timer 2 reload register R2. Transfers the contents of register A to the low-order 4 bits of timer 2 and timer 2 reload register R2.

T3AB (Transfer data to timer 3 and register R3L from Accumulator and register B)

Instruction code: D9 … D0

1	0	0	0	1	1	0	0	1	0	2	2	3	2	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation:
$(T3_7–T3_4) \leftarrow (B)$
$(R3L_7–R3L_4) \leftarrow (B)$
$(T3_3–T3_0) \leftarrow (A)$
$(R3L_3–R3L_0) \leftarrow (A)$

Grouping: Timer operation

Description: Transfers the contents of register B to the high-order 4 bits of timer 3 and timer 3 reload register R3L. Transfers the contents of register A to the low-order 4 bits of timer 3 and timer 3 reload register R3L.

T3HAB (Transfer data to register R3H from Accumulator and register B)

Instruction code: D9 … D0

1	0	0	0	1	1	1	1	0	1	2	2	3	D	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation:
$(R3H_7–R3H_4) \leftarrow (B)$
$(R3H_3–R3H_0) \leftarrow (A)$

Grouping: Timer operation

Description: Transfers the contents of register B to the high-order 4 bits of timer 3 and timer 3 reload register R3H. Transfers the contents of register A to the low-order 4 bits of timer 3 and timer 3 reload register R3H.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

T3R3L (Transfer data to timer 3 from register R3L)

Instruction code: D9 … D0: 1 0 0 0 1 1 0 1 0 0 (2) 2 3 4 (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: $(T3_7–T3_0) \leftarrow (R3L_7–R3L_0)$

Grouping: Timer operation

Description: Transfers the contents of reload register R3L to timer 3.

TAB (Transfer data to Accumulator from register B)

Instruction code: D9 … D0: 0 0 0 0 0 1 1 1 1 0 (2) 0 1 E (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (A) ← (B)

Grouping: Register to register transfer

Description: Transfers the contents of register B to register A.

TAB1 (Transfer data to Accumulator and register B from timer 1)

Instruction code: D9 … D0: 1 0 0 1 1 1 0 0 0 0 (2) 2 7 0 (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (B) ← $(T1_7–T1_4)$
(A) ← $(T1_3–T1_0)$

Grouping: Timer operation

Description: Transfers the high-order 4 bits $(T1_7–T1_4)$ of timer 1 to register B.
Transfers the low-order 4 bits $(T1_3–T1_0)$ of timer 1 to register A.

TAB2 (Transfer data to Accumulator and register B from timer 2)

Instruction code: D9 … D0: 1 0 0 1 1 1 0 0 0 1 (2) 2 7 1 (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (B) ← $(T2_7–T2_4)$
(A) ← $(T2_3–T2_0)$

Grouping: Timer operation

Description: Transfers the high-order 4 bits $(T2_7–T2_4)$ of timer 2 to register B.
Transfers the low-order 4 bits $(T2_3–T2_0)$ of timer 2 to register A.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

TAB3 (Transfer data to Accumulator and register B from timer 3)

Instruction code

D_9									D_0					
1	0	0	1	1	1	0	0	1	0	2	2	7	2	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation:
(B) ← ($T3_7$–$T3_4$)
(A) ← ($T3_3$–$T3_0$)

Grouping: Timer operation

Description: Transfers the high-order 4 bits ($T3_7$–$T3_4$) of timer 3 to register B.
Transfers the low-order 4 bits ($T3_3$–$T3_0$) of timer 3 to register A.

TABE (Transfer data to Accumulator and register B from register E)

Instruction code

D_9									D_0					
0	0	0	0	1	0	1	0	1	0	2	0	2	A	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation:
(B) ← (E_7–E_4)
(A) ← (E_3–E_0)

Grouping: Register to register transfer

Description: Transfers the high-order 4 bits (E_7–E_4) of register E to register B, and low-order 4 bits of register E to register A.

TABP p (Transfer data to Accumulator and register B from Program memory in page p)

Instruction code

D_9									D_0				
0	0	1	0	p_5	p_4	p_3	p_2	p_1	p_0	2	0	8 +p	p	16

Number of words	Number of cycles	Flag CY	Skip condition
1	3	-	-

Operation:
(SP) ← (SP) + 1
(SK(SP)) ← (PC)
(PC_H) ← p
(PC_L) ← (DR_2–DR_0, A_3–A_0)
(B) ← (ROM(PC))$_{7–4}$
(A) ← (ROM(PC))$_{3–0}$
(UPTF) ← 1
(DR_1, DR_0) ← (ROM(PC))$_{9, 8}$
(DR_2) ← 0
(PC) ← (SK(SP))
(SP) ← (SP) – 1

Grouping: Arithmetic operation

Description: Transfers bits 7 to 4 to register B and bits 3 to 0 to register A. These bits 7 to 0 are the ROM pattern in address (DR_2 DR_1 DR_0 A_3 A_2 A_1 A_0)$_2$ specified by registers A and D in page p. When UPTF is 1, Transfers bits 9, 8 to the low-order 2 bits (DR_1, DR_0) of register D, and "0" is stored to the least significant bit (DR_2) of register D.
When this instruction is executed, 1 stage of stack register (SK) is used.

Note: M34571G4 : p = 0 to 31
M34571G6 : p = 0 to 47
M34571GD : p = 0 to 127
When this instruction is executed, be careful not to over the stack because 1 stage of stack register is used.

TABPS (Transfer data to Accumulator and register B from Prescaler)

Instruction code

D_9									D_0					
1	0	0	1	1	1	0	1	0	1	2	2	7	5	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation:
(B) ← (TPS_7–TPS_4)
(A) ← (TPS_3–TPS_0)

Grouping: Timer operation

Description: Transfers the high-order 4 bits of prescaler to register B.
Transfers the low-order 4 bits of prescaler to register A.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

TAD (Transfer data to Accumulator from register D)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 0 0 0 1 0 1 0 0 0 1 $_2$ 0 5 1 $_{16}$	1	1	-	-

Operation: $(A_2–A_0) \leftarrow (DR_2–DR_0)$, $(A_3) \leftarrow 0$

Grouping: Register to register transfer

Description: Transfers the contents of register D to the low-order 3 bits ($A_2–A_0$) of register A.
"0" is stored to the bit 3 (A_3) of register A.

TAI1 (Transfer data to Accumulator from register I1)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 1 0 0 1 0 1 0 0 1 1 $_2$ 2 5 3 $_{16}$	1	1	-	-

Operation: (A) ← (I1)

Grouping: Interrupt operation

Description: Transfers the contents of interrupt control register I1 to register A.

TAI2 (Transfer data to Accumulator from register I2)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 1 0 0 1 0 1 0 1 0 0 $_2$ 2 5 4 $_{16}$	1	1	-	-

Operation: (A) ← (I2)

Grouping: Interrupt operation

Description: Transfers the contents of interrupt control register I2 to register A.

TAK0 (Transfer data to Accumulator from register K0)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 1 0 0 1 0 1 0 1 1 0 $_2$ 2 5 6 $_{16}$	1	1	-	-

Operation: (A) ← (K0)

Grouping: Input/Output operation

Description: Transfers the contents of key-on wakeup control register K0 to register A.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

TAK1 (Transfer data to Accumulator from register K1)

Instruction code: D9 … D0

1	0	0	1	0	1	1	0	0	1	2	2	5	9	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (A) ← (K1)

Grouping: Input/Output operation

Description: Transfers the contents of key-on wakeup control register K1 to register A.

TAK2 (Transfer data to Accumulator from register K2)

Instruction code: D9 … D0

1	0	0	1	0	1	1	0	1	0	2	2	5	A	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (A) ← (K2)

Grouping: Input/Output operation

Description: Transfers the contents of key-on wakeup control register K2 to register A.

TAL1 (Transfer data to Accumulator from register L1)

Instruction code: D9 … D0

1	0	0	1	0	0	1	0	1	0	2	2	4	A	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (A) ← (L1)

Grouping: Input/Output operation

Description: Transfers the contents of key-on wakeup control register L1 to register A.

TAM j (Transfer data to Accumulator from Memory)

Instruction code: D9 … D0

1	0	1	1	0	0	j	j	j	j	2	2	C	j	16

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (A) ← (M(DP))
(X) ← (X)EXOR(j)
j = 0 to 15

Grouping: RAM to register transfer

Description: After transferring the contents of M(DP) to register A, an exclusive OR operation is performed between register X and the value j in the immediate field, and stores the result in register X.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

TAMR (Transfer data to Accumulator from register MR)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 D0: 1 0 0 1 0 1 0 0 1 0 (2) 2 5 2 (16)	1	1	-	-

Operation: (A) ← (MR)

Grouping: Clock operation

Description: Transfers the contents of clock control register MR to register A.

TAPU0 (Transfer data to Accumulator from register PU0)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 D0: 1 0 0 1 0 1 0 1 1 1 (2) 2 5 7 (16)	1	1	-	-

Operation: (A) ← (PU0)

Grouping: Input/Output operation

Description: Transfers the contents of pull-up control register PU0 to register A.

TAPU1 (Transfer data to Accumulator from register PU1)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 D0: 1 0 0 1 0 1 1 1 1 0 (2) 2 5 E (16)	1	1	-	-

Operation: (A) ← (PU1)

Grouping: Input/Output operation

Description: Transfers the contents of pull-up control register PU1 to register A.

TAPU2 (Transfer data to Accumulator from register PU2)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 D0: 1 0 0 1 0 1 1 1 1 1 (2) 2 5 F (16)	1	1	-	-

Operation: (A) ← (PU2)

Grouping: Input/Output operation

Description: Transfers the contents of pull-up control register PU2 to register A.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

TASP (Transfer data to Accumulator from Stack Pointer)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 0 0 0 1 0 1 0 0 0 0 $_2$ 0 5 0 $_{16}$	1	1	-	-

Operation: $(A_2–A_0) \leftarrow (SP_2–SP_0)$, $(A_3) \leftarrow 0$

Grouping: Register to register transfer

Description: Transfers the contents of stack pointer (SP) to the low-order 3 bits ($A_2–A_0$) of register A.
"0" is stored to the bit 3 (A_3) of register A.

TAV1 (Transfer data to Accumulator from register V1)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 0 0 0 1 0 1 0 1 0 0 $_2$ 0 5 4 $_{16}$	1	1	-	-

Operation: (A) ← (V1)

Grouping: Interrupt operation

Description: Transfers the contents of interrupt control register V1 to register A.

TAV2 (Transfer data to Accumulator from register V2)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 0 0 0 1 0 1 0 1 0 1 $_2$ 0 5 5 $_{16}$	1	1	-	-

Operation: (A) ← (V2)

Grouping: Interrupt operation

Description: Transfers the contents of interrupt control register V2 to register A.

TAW1 (Transfer data to Accumulator from register W1)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 1 0 0 1 0 0 1 0 1 1 $_2$ 2 4 B $_{16}$	1	1	-	-

Operation: (A) ← (W1)

Grouping: Timer operation

Description: Transfers the contents of timer control register W1 to register A.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

TAW2 (Transfer data to Accumulator from register W2)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 D0: 1 0 0 1 0 0 1 1 0 0 (2) 2 4 C (16)	1	1	-	-

Operation: (A) ← (W2)

Grouping: Timer operation

Description: Transfers the contents of timer control register W2 to register A.

TAW3 (Transfer data to Accumulator from register W3)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 D0: 1 0 0 1 0 0 1 1 0 1 (2) 2 4 D (16)	1	1	-	-

Operation: (A) ← (W3)

Grouping: Timer operation

Description: Transfers the contents of timer control register W3 to register A.

TAW5 (Transfer data to Accumulator from register W5)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 D0: 1 0 0 1 0 0 1 1 1 1 (2) 2 4 F (16)	1	1	-	-

Operation: (A) ← (W5)

Grouping: Timer operation

Description: Transfers the contents of timer control register W5 to register A.

TAX (Transfer data to Accumulator from register X)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 D0: 0 0 0 1 0 1 0 0 1 0 (2) 0 5 2 (16)	1	1	-	-

Operation: (A) ← (X)

Grouping: Register to register transfer

Description: Transfers the contents of register X to register A.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

TAY (Transfer data to Accumulator from register Y)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 0 0 0 0 0 1 1 1 1 1 $_2$ 0 1 F $_{16}$	1	1	-	-

Operation: (A) ← (Y)

Grouping: Register to register transfer

Description: Transfers the contents of register Y to register A.

TAZ (Transfer data to Accumulator from register Z)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 0 0 0 1 0 1 0 0 1 1 $_2$ 0 5 3 $_{16}$	1	1	-	-

Operation: (A_1, A_0) ← (Z_1, Z_0)
(A_3, A_2) ← 0

Grouping: Register to register transfer

Description: Transfers the contents of register Z to the low-order 2 bits (A_1, A_0) of register A. “0” is stored to the high-order 2 bits (A_3, A_2) of register A.

TBA (Transfer data to register B from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 0 0 0 0 0 0 1 1 1 0 $_2$ 0 0 E $_{16}$	1	1	-	-

Operation: (B) ← (A)

Grouping: Register to register transfer

Description: Transfers the contents of register A to register B.

TDA (Transfer data to register D from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 0 0 0 0 1 0 1 0 0 1 $_2$ 0 2 9 $_{16}$	1	1	-	-

Operation: (DR_2–DR_0) ← (A_2–A_0)

Grouping: Register to register transfer

Description: Transfers the contents of the low-order 3 bits (A_2–A_0) of register A to register D.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

TEAB (Transfer data to register E from Accumulator and register B)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 D0: 0 0 0 0 0 1 1 0 1 0 (2) 0 1 A (16)	1	1	-	-

Operation: $(E_7–E_4) \leftarrow (B)$, $(E_3–E_0) \leftarrow (A)$

Grouping: Register to register transfer

Description: Transfers the contents of register B to the high-order 4 bits ($E_3–E_0$) of register E, and the contents of register A to the low-order 4 bits ($E_3–E_0$) of register E.

TFR0A (Transfer data to register FR0 from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 D0: 1 0 0 0 1 0 1 0 0 0 (2) 2 2 8 (16)	1	1	-	-

Operation: (FR0) ← (A)

Grouping: Input/Output operation

Description: Transfers the contents of register A to port output structure control register FR0.

TFR1A (Transfer data to register FR1 from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 D0: 1 0 0 0 1 0 1 0 0 1 (2) 2 2 9 (16)	1	1	-	-

Operation: (FR1) ← (A)

Grouping: Input/Output operation

Description: Transfers the contents of register A to port output structure control register FR1.

TI1A (Transfer data to register I1 from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 D0: 1 0 0 0 0 1 0 1 1 1 (2) 2 1 7 (16)	1	1	-	-

Operation: (I1) ← (A)

Grouping: Interrupt operation

Description: Transfers the contents of register A to interrupt control register I1.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

TI2A (Transfer data to register I2 from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 … D0: 1 0 0 0 0 1 1 0 0 0 (2) 2 1 8 (16)	1	1	-	-

Operation: (I2) ← (A)

Grouping: Interrupt operation

Description: Transfers the contents of register A to interrupt control register I2.

TK0A (Transfer data to register K0 from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 … D0: 1 0 0 0 0 1 1 0 1 1 (2) 2 1 B (16)	1	1	-	-

Operation: (K0) ← (A)

Grouping: Input/Output operation

Description: Transfers the contents of register A to key-on wakeup control register K0.

TK1A (Transfer data to register K1 from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 … D0: 1 0 0 0 0 1 0 1 0 0 (2) 2 1 4 (16)	1	1	-	-

Operation: (K1) ← (A)

Grouping: Input/Output operation

Description: Transfers the contents of register A to key-on wakeup control register K1.

TK2A (Transfer data to register K2 from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 … D0: 1 0 0 0 0 1 0 1 0 1 (2) 2 1 5 (16)	1	1	-	-

Operation: (K2) ← (A)

Grouping: Input/Output operation

Description: Transfers the contents of register A to key-on wakeup control register K2.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

TL1A (Transfer data to register L1 from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 … D0: 1 0 0 0 0 0 1 0 1 0 (2) 2 0 A (16)	1	1	-	-

Operation: (L1) ← (A)

Grouping: Input/Output operation

Description: Transfers the contents of register A to key-on wakeup control register L1.

TMA j (Transfer data to Memory from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 … D0: 1 0 1 0 1 1 j j j j (2) 2 B j (16)	1	1	-	-

Operation: (M(DP)) ← (A)
(X) ← (X)EXOR(j)
j = 0 to 15

Grouping: RAM to register transfer

Description: After transferring the contents of register A to M(DP), an exclusive OR operation is performed between register X and the value j in the immediate field, and stores the result in register X.

TMRA (Transfer data to register MR from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 … D0: 1 0 0 0 0 1 0 1 1 0 (2) 2 1 6 (16)	1	1	-	-

Operation: (MR) ← (A)

Grouping: Clock operation

Description: Transfers the contents of register A to clock control register MR.

TPAA (Transfer data to register PA from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D9 … D0: 1 0 1 0 1 0 1 0 1 0 (2) 2 A A (16)	1	1	-	-

Operation: (PA_0) ← (A_0)

Grouping: Timer operation

Description: Transfers the least significant bit of register A (A_0) to timer control register PA.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

TPSAB (Transfer data to Prescaler and register RPS from Accumulator and register B)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 1 0 0 0 1 1 0 1 0 1 $_2$ = 2 3 5 $_{16}$	1	1	-	-

Operation: $(RPS_7–RPS_4) \leftarrow (B)$
$(TPS_7–TPS_4) \leftarrow (B)$
$(RPS_3–RPS_0) \leftarrow (A)$
$(TPS_3–TPS_0) \leftarrow (A)$

Grouping: Timer operation

Description: Transfers the contents of register B to the high-order 4 bits of prescaler and prescaler reload register RPS. Transfers the contents of register A to the low-order 4 bits of prescaler and prescaler reload register RPS.

TPU0A (Transfer data to register PU0 from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 1 0 0 0 1 0 1 1 0 1 $_2$ = 2 2 D $_{16}$	1	1	-	-

Operation: $(PU0) \leftarrow (A)$

Grouping: Input/Output operation

Description: Transfers the contents of register A to pull-up control register PU0.

TPU1A (Transfer data to register PU1 from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 1 0 0 0 1 0 1 1 1 0 $_2$ = 2 2 E $_{16}$	1	1	-	-

Operation: $(PU1) \leftarrow (A)$

Grouping: Input/Output operation

Description: Transfers the contents of register A to pull-up control register PU1.

TPU2A (Transfer data to register PU2 from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 1 0 0 0 1 0 1 1 1 1 $_2$ = 2 2 F $_{16}$	1	1	-	-

Operation: $(PU2) \leftarrow (A)$

Grouping: Input/Output operation

Description: Transfers the contents of register A to pull-up control register PU2.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

TR1AB (Transfer data to register R1 from Accumulator and register B)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 1 0 0 0 1 1 1 1 1 1 $_2$ = 2 3 F $_{16}$	1	1	-	-

Operation: $(R1_7–R1_4) \leftarrow (B)$, $(R1_3–R1_0) \leftarrow (A)$

Grouping: Input/Output operation

Description: Transfers the contents of register B to the high-order 4 bits ($R1_7–R1_4$) of reload register R1, and the contents of register A to the low-order 4 bits ($R1_3–R1_0$) of reload register R1.

TV1A (Transfer data to register V1 from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 0 0 0 0 1 1 1 1 1 1 $_2$ = 0 3 F $_{16}$	1	1	-	-

Operation: $(V1) \leftarrow (A)$

Grouping: Interrupt operation

Description: Transfers the contents of register A to interrupt control register V1.

TV2A (Transfer data to register V2 from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 0 0 0 0 1 1 1 1 1 0 $_2$ = 0 3 E $_{16}$	1	1	-	-

Operation: $(V2) \leftarrow (A)$

Grouping: Interrupt operation

Description: Transfers the contents of register A to interrupt control register V2.

TW1A (Transfer data to register W1 from Accumulator)

Instruction code	Number of words	Number of cycles	Flag CY	Skip condition
D_9 … D_0: 1 0 0 0 0 0 1 1 1 0 $_2$ = 2 0 E $_{16}$	1	1	-	-

Operation: $(W1) \leftarrow (A)$

Grouping: Timer operation

Description: Transfers the contents of register A to timer control register W1.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

TW2A (Transfer data to register W2 from Accumulator)

Instruction code: D9 … D0: 1 0 0 0 0 0 1 1 1 1 (2) 2 0 F (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (W2) ← (A)

Grouping: Timer operation

Description: Transfers the contents of register A to timer control register W2.

TW3A (Transfer data to register W3 from Accumulator)

Instruction code: D9 … D0: 1 0 0 0 0 1 0 0 0 0 (2) 2 1 0 (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (W3) ← (A)

Grouping: Timer operation

Description: Transfers the contents of register A to timer control register W3.

TW5A (Transfer data to register W5 from Accumulator)

Instruction code: D9 … D0: 1 0 0 0 0 1 0 0 1 0 (2) 2 1 2 (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (W5) ← (A)

Grouping: Timer operation

Description: Transfers the contents of register A to timer control register W5.

TYA (Transfer data to register Y from Accumulator)

Instruction code: D9 … D0: 0 0 0 0 0 0 1 1 0 0 (2) 0 0 C (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (Y) ← (A)

Grouping: Register to register transfer

Description: Transfers the contents of register A to register Y.

MACHINE INSTRUCTIONS (INDEX BY ALPHABET) (continued)

WRST (Watchdog timer ReSeT)

Instruction code: D9 … D0: 1 0 1 0 1 0 0 0 0 0 (2) 2 A 0 (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	(WDF1) = 1

Operation: (WDF1) = 1 ?
(WDF1) ← 0

Grouping: Other operation

Description: Clears (0) to the WDF1 flag and skips the next instruction when watchdog timer flag WDF1 is “1”. When the WDF1 flag is “0”, executes the next instruction. Also, stops the watchdog timer function when executing the WRST instruction immediately after the DWDT instruction.

XAM j (eXchange Accumulator and Memory data)

Instruction code: D9 … D0: 1 0 1 1 0 1 j j j j (2) 2 D j (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	-

Operation: (A) ← → (M(DP))
(X) ← (X)EXOR(j)
j = 0 to 15

Grouping: RAM to register transfer

Description: After exchanging the contents of M(DP) with the contents of register A, an exclusive OR operation is performed between register X and the value j in the immediate field, and stores the result in register X.

XAMD j (eXchange Accumulator and Memory data and Decrement register Y and skip)

Instruction code: D9 … D0: 1 0 1 1 1 1 j j j j (2) 2 F j (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	(Y) = 15

Operation: (A) ← → (M(DP))
(X) ← (X)EXOR(j)
j = 0 to 15
(Y) ← (Y) −1

Grouping: RAM to register transfer

Description: After exchanging the contents of M(DP) with the contents of register A, an exclusive OR operation is performed between register X and the value j in the immediate field, and stores the result in register X.
Subtracts 1 from the contents of register Y.
As a result of subtraction, when the contents of register Y is 15, the next instruction is skipped. When the contents of register Y is not 15, the next instruction is executed.

XAMI j (eXchange Accumulator and Memory data and Increment register Y and skip)

Instruction code: D9 … D0: 1 0 1 1 1 0 j j j j (2) 2 E j (16)

Number of words	Number of cycles	Flag CY	Skip condition
1	1	-	(Y) = 0

Operation: (A) ← → (M(DP))
(X) ← (X)EXOR(j)
j = 0 to 15
(Y) ← (Y) + 1

Grouping: RAM to register transfer

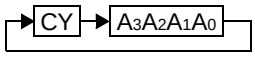
Description: After exchanging the contents of M(DP) with the contents of register A, an exclusive OR operation is performed between register X and the value j in the immediate field, and stores the result in register X.
Adds 1 to the contents of register Y. As a result of addition, when the contents of register Y is 0, the next instruction is skipped. When the contents of register Y is not 0, the next instruction is executed.

MACHINE INSTRUCTIONS (INDEX BY TYPES)

Parameter / Type of instructions	Mnemonic	Instruction code D_9	D_8	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Hexadecimal notation			Number of words	Number of cycles	Function
Register to register transfer	TAB	0	0	0	0	0	1	1	1	1	0	0	1	E	1	1	(A) ← (B)
	TBA	0	0	0	0	0	0	1	1	1	0	0	0	E	1	1	(B) ← (A)
	TAY	0	0	0	0	0	1	1	1	1	1	0	1	F	1	1	(A) ← (Y)
	TYA	0	0	0	0	0	0	1	1	0	0	0	0	C	1	1	(Y) ← (A)
	TEAB	0	0	0	0	0	1	1	0	1	0	0	1	A	1	1	(E_7–E_4) ← (B) (E_3–E_0) ← (A)
	TABE	0	0	0	0	1	0	1	0	1	0	0	2	A	1	1	(B) ← (E_7–E_4) (A) ← (E_3–E_0)
	TDA	0	0	0	0	1	0	1	0	0	1	0	2	9	1	1	(DR_2–DR_0) ← (A_2–A_0)
	TAD	0	0	0	1	0	1	0	0	0	1	0	5	1	1	1	(A_2–A_0) ← (DR_2–DR_0) (A_3) ← 0
	TAZ	0	0	0	1	0	1	0	0	1	1	0	5	3	1	1	(A_1, A_0) ← (Z_1, Z_0) (A_3, A_2) ← 0
	TAX	0	0	0	1	0	1	0	0	1	0	0	5	2	1	1	(A) ← (X)
	TASP	0	0	0	1	0	1	0	0	0	0	0	5	0	1	1	(A_2–A_0) ← (SP_2–SP_0) (A_3) ← 0
RAM addresses	LXY x, y	1	1	x_3	x_2	x_1	x_0	y_3	y_2	y_1	y_0	3	x	y	1	1	(X) ← x x = 0 to 15 (Y) ← y y = 0 to 15
	LZ z	0	0	0	1	0	0	1	0	z_1	z_0	0	4	8+z	1	1	(Z) ← z z = 0 to 3
	INY	0	0	0	0	0	1	0	0	1	1	0	1	3	1	1	(Y) ← (Y) + 1
	DEY	0	0	0	0	0	1	0	1	1	1	0	1	7	1	1	(Y) ← (Y) – 1
RAM to register transfer	TAM j	1	0	1	1	0	0	j	j	j	j	2	C	j	1	1	(A) ← (M(DP)) (X) ← (X)EXOR(j) j = 0 to 15
	XAM j	1	0	1	1	0	1	j	j	j	j	2	D	j	1	1	(A) ←→ (M(DP)) (X) ← (X)EXOR(j) j = 0 to 15
	XAMD j	1	0	1	1	1	1	j	j	j	j	2	F	j	1	1	(A) ←→ (M(DP)) (X) ← (X)EXOR(j) j = 0 to 15 (Y) ← (Y) – 1
	XAMI j	1	0	1	1	1	0	j	j	j	j	2	E	j	1	1	(A) ←→ (M(DP)) (X) ← (X)EXOR(j) j = 0 to 15 (Y) ← (Y) + 1
	TMA j	1	0	1	0	1	1	j	j	j	j	2	B	j	1	1	(M(DP)) ← (A) (X) ← (X)EXOR(j) j = 0 to 15

Skip condition	Carry flag CY	Detailed description
–	–	Transfers the contents of register B to register A.
–	–	Transfers the contents of register A to register B.
–	–	Transfers the contents of register Y to register A.
–	–	Transfers the contents of register A to register Y.
–	–	Transfers the contents of register B to the high-order 4 bits (E_3–E_0) of register E, and the contents of register A to the low-order 4 bits (E_3–E_0) of register E.
–	–	Transfers the high-order 4 bits (E_7–E_4) of register E to register B, and low-order 4 bits of register E to register A.
–	–	Transfers the contents of the low-order 3 bits (A_2–A_0) of register A to register D.
–	–	Transfers the contents of register D to the low-order 3 bits (A_2–A_0) of register A. "0" is stored to the bit 3 (A_3) of register A.
–	–	Transfers the contents of register Z to the low-order 2 bits (A_1, A_0) of register A. "0" is stored to the high-order 2 bits (A_3, A_2) of register A.
–	–	Transfers the contents of register X to register A.
–	–	Transfers the contents of stack pointer (SP) to the low-order 3 bits (A_2–A_0) of register A. "0" is stored to the bit 3 (A_3) of register A.
Continuous description	–	Loads the value x in the immediate field to register X, and the value y in the immediate field to register Y. When the LXY instructions are continuously coded and executed, only the first LXY instruction is executed and other LXY instructions coded continuously are skipped.
–	–	Loads the value z in the immediate field to register Z.
(Y) = 0	–	Adds 1 to the contents of register Y. As a result of addition, when the contents of register Y is 0, the next instruction is skipped. When the contents of register Y is not 0, the next instruction is executed.
(Y) = 15	–	Subtracts 1 from the contents of register Y. As a result of subtraction, when the contents of register Y is 15, the next instruction is skipped. When the contents of register Y is not 15, the next instruction is executed.
–	–	After transferring the contents of M(DP) to register A, an exclusive OR operation is performed between register X and the value j in the immediate field, and stores the result in register X.
–	–	After exchanging the contents of M(DP) with the contents of register A, an exclusive OR operation is performed between register X and the value j in the immediate field, and stores the result in register X.
(Y) = 15	–	After exchanging the contents of M(DP) with the contents of register A, an exclusive OR operation is performed between register X and the value j in the immediate field, and stores the result in register X. Subtracts 1 from the contents of register Y. As a result of subtraction, when the contents of register Y is 15, the next instruction is skipped. When the contents of register Y is not 15, the next instruction is executed.
(Y) = 0	–	After exchanging the contents of M(DP) with the contents of register A, an exclusive OR operation is performed between register X and the value j in the immediate field, and stores the result in register X. Adds 1 to the contents of register Y. As a result of addition, when the contents of register Y is 0, the next instruction is skipped. when the contents of register Y is not 0, the next instruction is executed.
–	–	After transferring the contents of register A to M(DP), an exclusive OR operation is performed between register X and the value j in the immediate field, and stores the result in register X.

MACHINE INSTRUCTIONS (INDEX BY TYPES) (continued)

Parameter / Type of instructions	Mnemonic	Instruction code D_9	D_8	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Hexadecimal notation			Number of words	Number of cycles	Function
Arithmetic operation	LA n	0	0	0	1	1	1	n	n	n	n	0	7	n	1	1	(A) ← n n = 0 to 15
	TABP p	0	0	1	0	p_5	p_4	p_3	p_2	p_1	p_0	0	8 +p	p	1	3	(SP) ← (SP) + 1 (SK(SP)) ← (PC) (PC_H) ← p (Note 1) (PC_L) ← (DR_2–DR_0, A_3–A_0) (B) ← (ROM(PC))$_{7\text{-}4}$ (A) ← (ROM(PC))$_{3\text{-}0}$ (UPTF) = 1 (DR_1, DR_0) ← (ROM(PC))$_{9,\ 8}$ (DR_2) ← 0 (PC) ← (SK(SP)) (SP) ← (SP) – 1
	AM	0	0	0	0	0	0	1	0	1	0	0	0	A	1	1	(A) ← (A) + (M(DP))
	AMC	0	0	0	0	0	0	1	0	1	1	0	0	B	1	1	(A) ← (A) + (M(DP)) + (CY) (CY) ← Carry
	A n	0	0	0	1	1	0	n	n	n	n	0	6	n	1	1	(A) ← (A) + n n = 0 to 15
	AND	0	0	0	0	0	1	1	0	0	0	0	1	8	1	1	(A) ← (A) AND (M(DP))
	OR	0	0	0	0	0	1	1	0	0	1	0	1	9	1	1	(A) ← (A) OR (M(DP))
	SC	0	0	0	0	0	0	0	1	1	1	0	0	7	1	1	(CY) ← 1
	RC	0	0	0	0	0	0	0	1	1	0	0	0	6	1	1	(CY) ← 0
	SZC	0	0	0	0	1	0	1	1	1	1	0	2	F	1	1	(CY) = 0 ?
	CMA	0	0	0	0	0	1	1	1	0	0	0	1	C	1	1	(A) ← $\overline{(A)}$
	RAR	0	0	0	0	0	1	1	1	0	1	0	1	D	1	1	
Bit operation	SB j	0	0	0	1	0	1	1	1	j	j	0	5	C +j	1	1	(Mj(DP)) ← 1 j = 0 to 3
	RB j	0	0	0	1	0	0	1	1	j	j	0	4	C +j	1	1	(Mj(DP)) ← 0 j = 0 to 3
	SZB j	0	0	0	0	1	0	0	0	j	j	0	2	j	1	1	(Mj(DP)) = 0 ? j = 0 to 3

Note 1.M34571G4: p=0 to 31, M34571G6: p=0 to 47 and M34571GD: p=0 to 127.

Skip condition	Carry flag CY	Detailed description
Continuous description	–	Loads the value n in the immediate field to register A. When the LA instructions are continuously coded and executed, only the first LA instruction is executed and other LA instructions coded continuously are skipped.
–	–	Transfers bits 7 to 4 to register B and bits 3 to 0 to register A. These bits 7 to 0 are the ROM pattern in address (DR_2 DR_1 DR_0 A_3 A_2 A_1 A_0)$_2$ specified by registers A and D in page p. When UPTF is 1, Transfers bits 9, 8 to the low-order 2 bits (DR_1, DR_0) of register D, and "0" is stored to the least significant bit (DR_2) of register D. When this instruction is executed, 1 stage of stack register (SK) is used.
–	–	Adds the contents of M(DP) to register A. Stores the result in register A. The contents of carry flag CY remains unchanged.
–	0/1	Adds the contents of M(DP) and carry flag CY to register A. Stores the result in register A and carry flag CY.
Overflow = 0	–	Adds the value n in the immediate field to register A, and stores a result in register A. The contents of carry flag CY remains unchanged. Skips the next instruction when there is no overflow as the result of operation. Executes the next instruction when there is overflow as the result of operation.
–	–	Takes the AND operation between the contents of register A and the contents of M(DP), and stores the result in register A.
–	–	Takes the OR operation between the contents of register A and the contents of M(DP), and stores the result in register A.
–	1	Sets (1) to carry flag CY.
–	0	Clears (0) to carry flag CY.
(CY) = 0	–	Skips the next instruction when the contents of carry flag CY is "0". Executes the next instruction when the contents of carry flag CY is "1". The contents of carry flag CY remains unchanged.
–	–	Stores the one's complement for register A's contents in register A.
–	0/1	Rotates 1 bit of the contents of register A including the contents of carry flag CY to the right.
–	–	Sets (1) the contents of bit j (bit specified by the value j in the immediate field) of M(DP).
–	–	Clears (0) the contents of bit j (bit specified by the value j in the immediate field) of M(DP).
(Mj(DP)) = 0 j = 0 to 3	–	Skips the next instruction when the contents of bit j (bit specified by the value j in the immediate field) of M(DP) is "0". Executes the next instruction when the contents of bit j of M(DP) is "1".

MACHINE INSTRUCTIONS (INDEX BY TYPES) (continued)

Parameter / Type of instructions	Mnemonic	Instruction code D_9	D_8	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0	Hexadecimal notation			Number of words	Number of cycles	Function
Comparison operation	SEAM	0	0	0	0	1	0	0	1	1	0	0	2	6	1	1	(A) = (M(DP)) ?
	SEA n	0	0	0	0	1	0	0	1	0	1	0	2	5	2	2	(A) = n n = 0 to 15
		0	0	0	1	1	1	n	n	n	n	0	7	n			
Branch operation	B a	0	1	1	a_6	a_5	a_4	a_3	a_2	a_1	a_0	1	8 +a	a	1	1	$(PC_L) \leftarrow a_6$–a_0
	BL p, a	0	0	1	1	1	p_4	p_3	p_2	p_1	p_0	0	E +p	p	2	2	$(PC_H) \leftarrow$p (Note 1) $(PC_L) \leftarrow a_6$–a_0
		1	p_6	p_5	a_6	a_5	a_4	a_3	a_2	a_1	a_0	2	a	a			
	BLA p	0	0	0	0	0	1	0	0	0	0	0	1	0	2	2	$(PC_H) \leftarrow$ p (Note 1) $(PC_L) \leftarrow (DR_2$–DR_0, A_3–$A_0)$
		1	p_6	p_5	p_4	0	0	p_3	p_2	p_1	p_0	2	p	p			
Subroutine operation	BM a	0	1	0	a_6	a_5	a_4	a_3	a_2	a_1	a_0	1	a	a	1	1	(SP) ← (SP) + 1 (SK(SP)) ← (PC) $(PC_H) \leftarrow 2$ $(PC_L) \leftarrow a_6$–a_0
	BML p, a	0	0	1	1	0	p_4	p_3	p_2	p_1	p_0	0	C +p	p	2	2	(SP) ← (SP) + 1 (SK(SP)) ← (PC) $(PC_H) \leftarrow$ p (Note 1)
		1	p_6	p_5	a_6	a_5	a_4	a_3	a_2	a_1	a_0	2	a	a			$(PC_L) \leftarrow a_6$–a_0
	BMLA p	0	0	0	0	1	1	0	0	0	0	0	3	0	2	2	(SP) ← (SP) + 1 (SK(SP)) ← (PC) $(PC_H) \leftarrow$ p (Note 1) $(PC_L) \leftarrow (DR_2$–DR_0, A_3–$A_0)$
		1	p_6	p_5	p_4	0	0	p_3	p_2	p_1	p_0	2	p	p			
Return operation	RTI	0	0	0	1	0	0	0	1	1	0	0	4	6	1	1	(PC) ← (SK(SP)) (SP) ← (SP) – 1
	RT	0	0	0	1	0	0	0	1	0	0	0	4	4	1	2	(PC) ← (SK(SP)) (SP) ← (SP) – 1
	RTS	0	0	0	1	0	0	0	1	0	1	0	4	5	1	2	(PC) ← (SK(SP)) (SP) ← (SP) – 1

Note 1.M34571G4: p=0 to 31, M34571G6: p=0 to 47 and M34571GD: p=0 to 127.

Skip condition	Carry flag CY	Detailed description
(A) = (M(DP))	–	Skips the next instruction when the contents of register A is equal to the contents of M(DP). Executes the next instruction when the contents of register A is not equal to the contents of M(DP).
(A) = n n = 0 to 15	–	Skips the next instruction when the contents of register A is equal to the value n in the immediate field. Executes the next instruction when the contents of register A is not equal to the value n in the immediate field.
–	–	Branch within a page : Branches to address a in the identical page.
–	–	Branch out of a page : Branches to address a in page p.
–	–	Branch out of a page : Branches to address $(DR_2\ DR_1\ DR_0\ A_3\ A_2\ A_1\ A_0)_2$ specified by registers D and A in page p.
–	–	Call the subroutine in page 2 : Calls the subroutine at address a in page 2.
–	–	Call the subroutine : Calls the subroutine at address a in page p.
–	–	Call the subroutine : Calls the subroutine at address $(DR_2\ DR_1\ DR_0\ A_3\ A_2\ A_1\ A_0)_2$ specified by registers D and A in page p.
–	–	Returns from interrupt service routine to main routine. Returns each value of data pointer (X, Y, Z), carry flag, skip status, NOP mode status by the continuous description of the LA/LXY instruction, register A and register B to the states just before interrupt.
–	–	Returns from subroutine to the routine called the subroutine.
No conditional skip	–	Returns from subroutine to the routine called the subroutine, and skips the next instruction at with no condition.

MACHINE INSTRUCTIONS (INDEX BY TYPES) (continued)

Parameter / Type of instructions	Mnemonic	Instruction code										Hexadecimal notation			Number of words	Number of cycles	Function
		D_9	D_8	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0						
Interrupt operation	DI	0	0	0	0	0	0	0	1	0	0	0	0	4	1	1	(INTE) ← 0
	EI	0	0	0	0	0	0	0	1	0	1	0	0	5	1	1	(INTE) ← 1
	SNZ0	0	0	0	0	1	1	1	0	0	0	0	3	8	1	1	$V1_0$ = 0 : (EXF0) = 1 ? (EXF0) ← 0 $V1_0$ = 1 : SNZ0 = NOP
	SNZI0	0	0	0	0	1	1	1	0	1	0	0	3	A	1	1	$I1_2$ = 0 : (INT0) = “L”? $I1_2$ = 1 : (INT0) = “H”?
	SNZ1	0	0	0	0	1	1	1	0	0	1	0	3	9	1	1	$V1_1$ = 0 : (EXF1) = 1 ? (EXF1) ← 0 $V1_1$ = 1 : SNZ1 = NOP
	SNZI1	0	0	0	0	1	1	1	0	1	1	0	3	B	1	1	$I2_2$ = 0 : (INT1) = “L” ? $I2_2$ = 1 : (INT1) = “H” ?
	TAV1	0	0	0	1	0	1	0	1	0	0	0	5	4	1	1	(A) ← (V1)
	TV1A	0	0	0	0	1	1	1	1	1	1	0	3	F	1	1	(V1) ← (A)
	TAV2	0	0	0	1	0	1	0	1	0	1	0	5	5	1	1	(A) ← (V2)
	TV2A	0	0	0	0	1	1	1	1	1	0	0	3	E	1	1	(V2) ← (A)
	TAI1	1	0	0	1	0	1	0	0	1	1	2	5	3	1	1	(A) ← (I1)
	TI1A	1	0	0	0	0	1	0	1	1	1	2	1	7	1	1	(I1) ← (A)
	TAI2	1	0	0	1	0	1	0	1	0	0	2	5	4	1	1	(A) ← (I2)
	TI2A	1	0	0	0	0	1	1	0	0	0	2	1	8	1	1	(I2) ← (A)
Timer operation	TPAA	1	0	1	0	1	0	1	0	1	0	2	A	A	1	1	(PA_1, PA_0) ← (A_1, A_0)
	TAW1	1	0	0	1	0	0	1	0	1	1	2	4	B	1	1	(A) ← (W1)
	TW1A	1	0	0	0	0	0	1	1	1	0	2	0	E	1	1	(W1) ← (A)
	TAW2	1	0	0	1	0	0	1	1	0	0	2	4	C	1	1	(A) ← (W2)
	TW2A	1	0	0	0	0	0	1	1	1	1	2	0	F	1	1	(W2) ← (A)
	TAW3	1	0	0	1	0	0	1	1	0	1	2	4	D	1	1	(A) ← (W3)
	TW3A	1	0	0	0	0	1	0	0	0	0	2	1	0	1	1	(W3) ← (A)
	TAW5	1	0	0	1	0	0	1	1	1	1	2	4	F	1	1	(A) ← (W5)
	TW5A	1	0	0	0	0	1	0	0	1	0	2	1	2	1	1	(W5) ← (A)

Skip condition	Carry flag CY	Detailed description
–	–	Clears (0) to interrupt enable flag INTE, and disables the interrupt.
–	–	Sets (1) to interrupt enable flag INTE, and enables the interrupt.
$V1_0$ = 0 : (EXF0) = 1	–	When $V1_0$ = 0 : Clears (0) to the EXF0 flag and skips the next instruction when external 0 interrupt request flag EXF0 is “1”. When the EXF0 flag is “0”, executes the next instruction. When $V1_0$ = 1 : This instruction is equivalent to the NOP instruction. ($V1_0$: bit 0 of interrupt control register V1)
(INT0) = “L” However, $I1_2$ = 0	–	When $I1_2$ = 0 : Skips the next instruction when the level of INT0 pin is “L”. Executes the next instruction when the level of INT0 pin is “H”.
(INT0) = “H” However, $I1_2$ = 1		When $I1_2$ = 1 : Skips the next instruction when the level of INT0 pin is “H”. Executes the next instruction when the level of INT0 pin is “L”. ($I1_2$: bit 2 of interrupt control register I1)
$V1_1$ = 0 : (EXF1) = 1	–	When $V1_1$ = 0 : Clears (0) to the EXF1 flag and skips the next instruction when external 1 interrupt request flag EXF1 is “1”. When the EXF1 flag is “0”, executes the next instruction. When $V1_1$ = 1 : This instruction is equivalent to the NOP instruction. ($V1_1$: bit 1 of interrupt control register V1)
$I2_2$ = 0 : (INT1) = “L”	–	When $I2_2$ = 0 : Skips the next instruction when the level of INT1 pin is “L”. Executes the next instruction when the level of INT1 pin is “H”.
$I2_2$ = 1 : (INT1) = “H”		When $I2_2$ = 1 : Skips the next instruction when the level of INT1 pin is “H”. Executes the next instruction when the level of INT1 pin is “L”. ($I2_2$: bit 2 of interrupt control register I2)
–	–	Transfers the contents of interrupt control register V1 to register A.
–	–	Transfers the contents of register A to interrupt control register V1.
–	–	Transfers the contents of interrupt control register V2 to register A.
–	–	Transfers the contents of register A to interrupt control register V2.
–	–	Transfers the contents of interrupt control register I1 to register A.
–	–	Transfers the contents of register A to interrupt control register I1.
–	–	Transfers the contents of interrupt control register I2 to register A.
–	–	Transfers the contents of register A to interrupt control register I2.
–	–	Transfers the contents of register A (A_1, A_0) to timer control register PA.
–	–	Transfers the contents of timer control register W1 to register A.
–	–	Transfers the contents of register A to timer control register W1.
–	–	Transfers the contents of timer control register W2 to register A.
–	–	Transfers the contents of register A to timer control register W2.
–	–	Transfers the contents of timer control register W3 to register A.
–	–	Transfers the contents of register A to timer control register W3.
–	–	Transfers the contents of timer control register W5 to register A.
–	–	Transfers the contents of register A to timer control register W5.

MACHINE INSTRUCTIONS (INDEX BY TYPES) (continued)

Parameter / Type of instructions	Mnemonic	Instruction code										Hexadecimal notation			Number of words	Number of cycles	Function
		D_9	D_8	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0						
Timer operation	TABPS	1	0	0	1	1	1	0	1	0	1	2	7	5	1	1	(B) ← (TPS_7–TPS_4) (A) ← (TPS_3–TPS_0)
	TPSAB	1	0	0	0	1	1	0	1	0	1	2	3	5	1	1	(RPS_7–RPS_4) ← (B) (TPS_7–TPS_4) ← (B) (RPS_3–RPS_0) ← (A) (TPS_3–TPS_0) ← (A)
	TAB1	1	0	0	1	1	1	0	0	0	0	2	7	0	1	1	(B) ← ($T1_7$–$T1_4$) (A) ← ($T1_3$–$T1_0$)
	T1AB	1	0	0	0	1	1	0	0	0	0	2	3	0	1	1	($R1_7$–$R1_4$) ← (B) ($T1_7$–$T1_4$) ← (B) ($R1_3$–$R1_0$) ← (A) ($T1_3$–$T1_0$) ← (A)
	TR1AB	1	0	0	0	1	1	1	1	1	1	2	3	F	1	1	($R1_7$–$R1_4$) ← (B) ($R1_3$–$R1_0$) ← (A)
	TAB2	1	0	0	1	1	1	0	0	0	1	2	7	1	1	1	(B) ← ($T2_7$–$T2_4$) (A) ← ($T2_3$–$T2_0$)
	T2AB	1	0	0	0	1	1	0	0	0	1	2	3	1	1	1	($R2_7$–$R2_4$) ← (B) ($T2_7$–$T2_4$) ← (B) ($R2_3$–$R2_0$) ← (A) ($T2_3$–$T2_0$) ← (A)
	TAB3	1	0	0	1	1	1	0	0	1	0	2	7	2	1	1	(B) ← ($T3_7$–$T3_4$) (A) ← ($T3_3$–$T3_0$)
	T3AB	1	0	0	0	1	1	0	0	1	0	2	3	2	1	1	($R3L_7$–$R3L_4$) ← (B) ($T3_7$–$T3_4$) ← (B) ($R3L_3$–$R3L_0$) ← (A) ($T3_3$–$T3_0$) ← (A)
	T3HAB	1	0	0	0	1	1	1	1	0	1	2	3	D	1	1	($R3H_7$–$R3H_4$) ← (B) ($R3H_3$–$R3H_0$) ← (A)
	T3R3L	1	0	0	0	1	1	0	1	0	0	2	3	4	1	1	($T3_7$) ← (R3L)
	SNZT1	1	0	1	0	0	0	0	0	0	0	2	8	0	1	1	$V1_2$ = 0 : (T1F) = 1 ? After skipping, (T1F) ← 0 $V1_2$ = 1 : SNZT1=NOP
	SNZT2	1	0	1	0	0	0	0	0	0	1	2	8	1	1	1	$V1_3$ = 0 : (T2F) = 1 ? After skipping, (T2F) ← 0 $V1_3$ = 1 : SNZT2=NOP
	SNZT3	1	0	1	0	0	0	0	0	1	0	2	8	2	1	1	$V2_0$ = 0 : (T3F) = 1 ? After skipping, (T3F) ← 0 $V2_0$ = 1 : SNZT3=NOP

Skip condition	Carry flag CY	Detailed description
–	–	Transfers the high-order 4 bits of prescaler to register B. Transfers the low-order 4 bits of prescaler to register A.
–	–	Transfers the contents of register B to the high-order 4 bits of prescaler and prescaler reload register RPS. Transfers the contents of register A to the low-order 4 bits of prescaler and prescaler reload register RPS.
–	–	Transfers the high-order 4 bits ($T1_7$–$T1_4$) of timer 1 to register B. Transfers the low-order 4 bits ($T1_3$–$T1_0$) of timer 1 to register A.
–	–	Transfers the contents of register B to the high-order 4 bits of timer 1 and timer 1 reload register R1L. Transfers the contents of register A to the low-order 4 bits of timer 1 and timer 1 reload register R1L.
–	–	Transfers the contents of register B to the high-order 4 bits ($R1_7$–$R1_4$) of reload register R1, and the contents of register A to the low-order 4 bits ($R1_3$–$R1_0$) of reload register R1.
–	–	Transfers the high-order 4 bits ($T2_7$–$T2_4$) of timer 2 to register B. Transfers the low-order 4 bits ($T2_3$–$T2_0$) of timer 2 to register A.
–	–	Transfers the contents of register B to the high-order 4 bits of timer 2 and timer 2 reload register R2L. Transfers the contents of register A to the low-order 4 bits of timer 2 and timer 2 reload register R2L.
–	–	Transfers the high-order 4 bits ($T3_7$–$T3_4$) of timer 3 to register B. Transfers the low-order 4 bits ($T3_3$–$T3_0$) of timer 3 to register A.
–	–	Transfers the contents of register B to the high-order 4 bits of timer 3 and timer 3 reload register R3L. Transfers the contents of register A to the low-order 4 bits of timer 3 and timer 3 reload register R3L.
–	–	Transfers the contents of register B to the high-order 4 bits of timer 3 reload register R3H. Transfers the contents of register A to the low-order 4 bits of timer 3 reload register R3H.
–	–	Transfers the contents of timer 3 reload register R3L to timer 3.
$V1_2$ = 0 : (T1F) = 1	–	When $V1_2$ = 0 : Clears (0) to the T1F flag and skips the next instruction when timer 1 interrupt request flag T1F is “1”. When the T1F flag is “0”, executes the next instruction. When $V1_2$ = 1 : This instruction is equivalent to the NOP instruction. ($V1_2$: bit 2 of interrupt control register V1)
$V1_3$ = 0 : (T2F) = 1	–	When $V1_3$ = 0 : Clears (0) to the T2F flag and skips the next instruction when timer 2 interrupt request flag T2F is “1”. When the T2F flag is “0”, executes the next instruction. When $V1_3$ = 1 : This instruction is equivalent to the NOP instruction. ($V1_3$: bit 3 of interrupt control register V1)
$V2_0$ = 0 : (T3F) = 1	–	When $V2_0$ = 0 : Clears (0) to the T3F flag and skips the next instruction when timer 3 interrupt request flag T3F is “1”. When the T3F flag is “0”, executes the next instruction. When $V2_0$ = 1 : This instruction is equivalent to the NOP instruction. ($V2_0$: bit 0 of interrupt control register V2)

MACHINE INSTRUCTIONS (INDEX BY TYPES) (continued)

Parameter / Type of instructions	Mnemonic	Instruction code										Hexadecimal notation			Number of words	Number of cycles	Function
		D_9	D_8	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0						
Input/Output operation	IAP0	1	0	0	1	1	0	0	0	0	0	2	6	0	1	1	(A) ← (P0)
	OP0A	1	0	0	0	1	0	0	0	0	0	2	2	0	1	1	(P0) ← (A)
	IAP1	1	0	0	1	1	0	0	0	0	1	2	6	1	1	1	(A) ← (P1)
	OP1A	1	0	0	0	1	0	0	0	0	1	2	2	1	1	1	(P1) ← (A)
	IAP2	1	0	0	1	1	0	0	0	1	0	2	6	2	1	1	(A_1, A_0) ← ($P2_1$, $P2_0$) (A_3, A_2) ← 0
	OP2A	1	0	0	0	1	0	0	0	1	0	2	2	2	1	1	($P2_1$, $P2_0$) ← (A_1, A_0)
	IAP3	1	0	0	1	1	0	0	0	1	1	2	6	3	1	1	(A_1, A_0) ← ($P3_1$, $P3_0$) (A_3, A_2) ← 0
	OP3A	1	0	0	0	1	0	0	0	1	1	2	2	3	1	1	($P3_1$, $P3_0$) ← (A_1, A_0)
	CLD	0	0	0	0	0	1	0	0	0	1	0	1	1	1	1	(D) ← 1
	RD	0	0	0	0	0	1	0	1	0	0	0	1	4	1	1	(D(Y)) ← 0 (Y) = 0 to 4
	SD	0	0	0	0	0	1	0	1	0	1	0	1	5	1	1	(D(Y)) ← 1 (Y) = 0 to 4
	SZD	0	0	0	0	1	0	0	1	0	0	0	2	4	2	2	(D(Y)) = 0 ? (Y) = 0 to 4
		0	0	0	0	1	0	1	0	1	1	0	2	B			
	RCP	1	0	1	0	0	0	1	1	0	0	2	8	C	1	1	(C) ← (0)
	SCP	1	0	1	0	0	0	1	1	0	1	2	8	D	1	1	(C) ← (1)
	TFR0A	1	0	0	0	1	0	1	0	0	0	2	2	8	1	1	(FR0) ← (A)
	TFR1A	1	0	0	0	1	0	1	0	0	1	2	2	9	1	1	(FR1) ← (A)
	TAPU0	1	0	0	1	0	1	0	1	1	1	2	5	7	1	1	(A) ← (PU0)
	TPU0A	1	0	0	0	1	0	1	1	0	1	2	2	D	1	1	(PU0) ← (A)
	TAPU1	1	0	0	1	0	1	1	1	1	0	2	5	E	1	1	(A) ← (PU1)
	TPU1A	1	0	0	0	1	0	1	1	1	0	2	2	E	1	1	(PU1) ← (A)
	TAPU2	1	0	0	1	0	1	1	1	1	1	2	5	F	1	1	(A) ← (PU2)
	TPU2A	1	0	0	0	1	0	1	1	1	1	2	2	F	1	1	(PU2) ← (A)
	IAK	1	0	0	1	1	0	1	1	1	1	2	6	F	1	1	(A_0) ← (K) (A_3–A_1) ← 0

Skip condition	Carry flag CY	Detailed description
–	–	Transfers the input of port P0 to register A.
–	–	Outputs the contents of register A to port P0.
–	–	Transfers the input of port P1 to register A.
–	–	Outputs the contents of register A to port P1.
–	–	Transfers the input of port P2 to the low-order 2 bits (A_1, A_0) of register A. "0" is stored to the high-order 2 bits (A_3, A_2) of register A.
–	–	Outputs the contents of the low-order 2 bits (A_1, A_0) of register A to port P2.
–	–	Transfers the input of port P3 to the low-order 2 bits (A_1, A_0) of register A. "0" is stored to the high-order 2 bits (A_3, A_2) of register A.
–	–	Outputs the contents of the low-order 2 bits (A_1, A_0) of register A to port P3.
–	–	Sets (1) to port D.
–	–	Clears (0) to a bit of port D specified by register Y.
–	–	Sets (1) to a bit of port D specified by register Y.
(D(Y)) = 0 Y = 0 to 4	–	Skips the next instruction when a bit of port D specified by register Y is "0". Executes the next instruction when a bit of port D specified by register Y is "1".
–	–	Clears (0) to port C.
–	–	Sets (1) to port C.
–	–	Transfers the contents of register A to port output structure control register FR0.
–	–	Transfers the contents of register A to port output structure control register FR1.
–	–	Transfers the contents of pull-up control register PU0 to register A.
–	–	Transfers the contents of register A to pull-up control register PU0.
–	–	Transfers the contents of pull-up control register PU1 to register A.
–	–	Transfers the contents of register A to pull-up control register PU1.
–	–	Transfers the contents of pull-up control register PU2 to register A.
–	–	Transfers the contents of register A to pull-up control register PU2.
–	–	Transfers the input of port K to the least significant bit (A_0) of register A. "0" is stored to the high-order 3 bits (A_3–A_1) of register A.

MACHINE INSTRUCTIONS (INDEX BY TYPES) (continued)

Parameter / Type of instructions	Mnemonic	Instruction code										Hexadecimal notation			Number of words	Number of cycles	Function
		D_9	D_8	D_7	D_6	D_5	D_4	D_3	D_2	D_1	D_0						
Input/Output operation	TAK0	1	0	0	1	0	1	0	1	1	0	2	5	6	1	1	(A) ← (K0)
	TK0A	1	0	0	0	0	1	1	0	1	1	2	1	B	1	1	(K0) ← (A)
	TAK1	1	0	0	1	0	1	1	0	0	1	2	5	9	1	1	(A) ← (K1)
	TK1A	1	0	0	0	0	1	0	1	0	0	2	1	4	1	1	(K1) ← (A)
	TAK2	1	0	0	1	0	1	1	0	1	0	2	5	A	1	1	(A) ← (K2)
	TK2A	1	0	0	0	0	1	0	1	0	1	2	1	5	1	1	(K2) ← (A)
	TAL1	1	0	0	1	0	0	1	0	1	0	2	4	A	1	1	(A) ← (L1)
	TL1A	1	0	0	0	0	0	1	0	1	0	2	0	A	1	1	(L1) ← (A)
Other operation	TAMR	1	0	0	1	0	1	0	0	1	0	2	5	2	1	1	(A) ← (MR)
	TMRA	1	0	0	0	0	1	0	1	1	0	2	1	6	1	1	(MR) ← (A)
	NOP	0	0	0	0	0	0	0	0	0	0	0	0	0	1	1	(PC) ← (PC) + 1
	POF	0	0	0	0	0	0	0	0	1	0	0	0	2	1	1	RAM back-up
	EPOF	0	0	0	1	0	1	1	0	1	1	0	5	B	1	1	POF instruction valid
	SNZP	0	0	0	0	0	0	0	0	1	1	0	0	3	1	1	(P) = 1 ?
	WRST	1	0	1	0	1	0	0	0	0	0	2	A	0	1	1	(WDF1) = 1 ? (WDF1) ← 0
	DWDT	1	0	1	0	0	1	1	1	0	0	2	9	C	1	1	Stop of watchdog timer function enabled
	SRST	0	0	0	0	0	0	0	0	0	1	0	0	1	1	1	System reset
	RUPT	0	0	0	1	0	1	1	0	0	0	0	5	8	1	1	(UPTF) ← 0
	SUPT	0	0	0	1	0	1	1	0	0	1	0	5	9	1	1	(UPTF) ← 1
	SNZVD	1	0	1	0	0	0	1	0	1	0	2	8	A	1	1	$V2_3$ = 0 : (VDF) = 1? $V2_3$ = 1 : SNZVD = NOP
	RBK (Note 1)	0	0	0	1	0	0	0	0	0	0	0	4	0	1	1	p_6 ← 0 when TABP p instruction is executed.
	SBK (Note 1)	0	0	0	1	0	0	0	0	0	1	0	4	1	1	1	p_6 ← 1 when TABP p instruction is executed.

Note 1.This instruction cannot be used for the M34571G4/G6.

Skip condition	Carry flag CY	Detailed description
–	–	Transfers the contents of key-on wakeup control register K0 to register A.
–	–	Transfers the contents of register A to key-on wakeup control register K0.
–	–	Transfers the contents of key-on wakeup control register K1 to register A.
–	–	Transfers the contents of register A to key-on wakeup control register K1.
–	–	Transfers the contents of key-on wakeup control register K2 to register A.
–	–	Transfers the contents of register A to key-on wakeup control register K2.
–	–	Transfers the contents of key-on wakeup control register L1 to register A.
–	–	Transfers the contents of register A to key-on wakeup control register L1.
–	–	Transfers the contents of clock control register MR to register A.
–	–	Transfers the contents of register A to clock control register MR.
–	–	No operation; Adds 1 to program counter value, and others remain unchanged.
–	–	Puts the system in RAM back-up state by executing the POF instruction after executing the EPOF instruction. Operations of all functions are stopped.
–	–	Makes the immediate after POF instruction valid by executing the EPOF instruction.
(P) = 1	–	Skips the next instruction when the P flag is “1”. After skipping, the P flag remains unchanged. Executes the next instruction when the P flag is “0”.
(WDF1) = 1		Clears (0) to the WDF1 flag and skips the next instruction when watchdog timer flag WDF1 is “1”. When the WDF1 flag is “0”, executes the next instruction. Also, stops the watchdog timer function when executing the WRST instruction immediately after the DWDT instruction.
–	–	Stops the watchdog timer function by the WRST instruction after executing the DWDT instruction.
–	–	System reset occurs.
–	–	Clears (0) to the high-order bit reference enable flag UPTF.
–	–	Sets (1) to the high-order bit reference enable flag UPTF.
$V2_3$ = 0 : (VDF) = 1	–	When $V2_3$ = 0 : Skips the next instruction when voltage detector interrupt request flag VDF is “1”. The VDF flag is not cleared to “0“. When the VDF flag is “0”, executes the next instruction. When $V2_3$ = 1 : This instruction is equivalent to the NOP instruction.
–	–	Sets referring data area to pages 0 to 63 when the TABP p instruction is executed. This instruction is valid only for the TABP p instruction.
–	–	Sets referring data area to pages 64 to 127 when the TABP p instruction is executed. This instruction is valid only for the TABP p instruction.

INSTRUCTION CODE TABLE

D_3–D_0 Hex notation \ D_9–D_4	000000	000001	000010	000011	000100	000101	000110	000111	001000	001001	001010	001011	001100	001101	001110	001111	010000 to 010111	011000 to 011111
	00	01	02	03	04	05	06	07	08	09	0A	0B	0C	0D	0E	0F	10–17	18–F
0000	0	NOP	BLA	SZB 0	BMLA	RBK ***	TASP	A 0	LA 0	TABP 0	TABP 16	TABP 32*	TABP 48**	BML	BML	BL	BL	BM	B
0001	1	SRST	CLD	SZB 1	–	SBK ***	TAD	A 1	LA 1	TABP 1	TABP 17	TABP 33*	TABP 49**	BML	BML	BL	BL	BM	B
0010	2	POF	–	SZB 2	–	–	TAX	A 2	LA 2	TABP 2	TABP 18	TABP 34*	TABP 50**	BML	BML	BL	BL	BM	B
0011	3	SNZP	INY	SZB 3	–	–	TAZ	A 3	LA 3	TABP 3	TABP 19	TABP 35*	TABP 51**	BML	BML	BL	BL	BM	B
0100	4	DI	RD	SZD	–	RT	TAV1	A 4	LA 4	TABP 4	TABP 20	TABP 36*	TABP 52**	BML	BML	BL	BL	BM	B
0101	5	EI	SD	SEAn	–	RTS	TAV2	A 5	LA 5	TABP 5	TABP 21	TABP 37*	TABP 53**	BML	BML	BL	BL	BM	B
0110	6	RC	–	SEAM	–	RTI	–	A 6	LA 6	TABP 6	TABP 22	TABP 38*	TABP 54**	BML	BML	BL	BL	BM	B
0111	7	SC	DEY	–	–	–	–	A 7	LA 7	TABP 7	TABP 23	TABP 39*	TABP 55**	BML	BML	BL	BL	BM	B
1000	8	–	AND	–	SNZ0	LZ 0	RUPT	A 8	LA 8	TABP 8	TABP 24	TABP 40*	TABP 56**	BML	BML	BL	BL	BM	B
1001	9	–	OR	TDA	SNZ1	LZ 1	SUPT	A 9	LA 9	TABP 9	TABP 25	TABP 41*	TABP 57**	BML	BML	BL	BL	BM	B
1010	A	AM	TEAB	TABE	SNZI 0	LZ 2	–	A 10	LA 10	TABP 10	TABP 26	TABP 42*	TABP 58**	BML	BML	BL	BL	BM	B
1011	B	AMC	–	–	SNZI 1	LZ 3	EPOF	A 11	LA 11	TABP 11	TABP 27	TABP 43*	TABP 59**	BML	BML	BL	BL	BM	B
1100	C	TYA	CMA	–	–	RB 0	SB 0	A 12	LA 12	TABP 12	TABP 28	TABP 44*	TABP 60**	BML	BML	BL	BL	BM	B
1101	D	–	RAR	–	–	RB 1	SB 1	A 13	LA 13	TABP 13	TABP 29	TABP 45*	TABP 61**	BML	BML	BL	BL	BM	B
1110	E	TBA	TAB	–	TV2A	RB 2	SB 2	A 14	LA 14	TABP 14	TABP 30	TABP 46*	TABP 62**	BML	BML	BL	BL	BM	B
1111	F	–	TAY	SZC	TV1A	RB 3	SB 3	A 15	LA 15	TABP 15	TABP 31	TABP 47*	TABP 63**	BML	BML	BL	BL	BM	B

The above table shows the relationship between machine language codes and machine language instructions. D_3–D_0 show the low-order 4 bits of the machine language code, and D_9–D_4 show the high-order 6 bits of the machine language code. The hexadecimal representation of the code is also provided. There are one-word instructions and two-word instructions, but only the first word of each instruction is shown. Do not use code marked "–."

The codes for the second word of a two-word instruction are described below.

	The second word
BL	10 0aaa aaaa
BML	10 0aaa aaaa
BLA	10 0p00 pppp
BMLA	10 0p00 pppp
SEA	00 0111 nnnn
SZD	00 0010 1011

- *, **, and *** cannot be used in the M34571G4.
- ** and *** cannot be used in the M34571G6.
- A page referred by the TABP instruction can be switched by the SBK and RBK instructions in the M34571GD.
  The pages which can be referred by the TABP instruction after the RBK instruction is executed are 0 to 63.
  The pages which can be referred by the TABP instruction after the SBK instruction is executed are 64 to 127 (Ex. TABP 0 TABP 64).
  When the SBK instruction is not used, the pages which can be referred by the TABP instruction are 0 to 63.

INSTRUCTION CODE TABLE

D_3–D_0 \ D_9–D_4	Hex. notation	100000	100001	100010	100011	100100	100101	100110	100111	101000	101001	101010	101011	101100	101101	101110	101111	110000 to 111111
		20	21	22	23	24	25	26	27	28	29	2A	2B	2C	2D	2E	2F	30–3F
0000	0	–	TW3A	OP0A	T1AB	–	–	IAP0	TAB1	SNZT 1	–	WRST	TMA 0	TAM 0	XAM 0	XAMI 0	XAMD 0	LXY
0001	1	–	–	OP1A	T2AB	–	–	IAP1	TAB2	SNZT 2	–	–	TMA 1	TAM 1	XAM 1	XAMI 1	XAMD 1	LXY
0010	2	–	TW5A	OP2A	T3AB	–	TAMR	IAP2	TAB3	SNZT 3	–	–	TMA 2	TAM 2	XAM 2	XAMI 2	XAMD 2	LXY
0011	3	–	–	OP3A	–	–	TAI1	IAP3	–	–	–	–	TMA 3	TAM 3	XAM 3	XAMI 3	XAMD 3	LXY
0100	4	–	TK1A	–	T3R3L	–	TAI2	–	–	–	–	–	TMA 4	TAM 4	XAM 4	XAMI 4	XAMD 4	LXY
0101	5	–	TK2A	–	TPSAB	–	–	–	TABPS	–	–	–	TMA 5	TAM 5	XAM 5	XAMI 5	XAMD 5	LXY
0110	6	–	TMRA	–	–	–	TAK0	–	–	–	–	–	TMA 6	TAM 6	XAM 6	XAMI 6	XAMD 6	LXY
0111	7	–	TI1A	–	–	–	TAPU0	–	–	–	–	–	TMA 7	TAM 7	XAM 7	XAMI 7	XAMD 7	LXY
1000	8	–	TI2A	TFR0A	–	–	–	–	–	–	–	–	TMA 8	TAM 8	XAM 8	XAMI 8	XAMD 8	LXY
1001	9	–	–	TFR1A	–	–	TAK1	–	–	–	–	–	TMA 9	TAM 9	XAM 9	XAMI 9	XAMD 9	LXY
1010	A	TL1A	–	–	–	TAL1	TAK2	–	–	SNZVD	–	TPAA	TMA 10	TAM 10	XAM 10	XAMI 10	XAMD 10	LXY
1011	B	–	TK0A	–	–	TAW1	–	–	–	–	–	–	TMA 11	TAM 11	XAM 11	XAMI 11	XAMD 11	LXY
1100	C	–	–	–	–	TAW2	–	–	–	RCP	DWDT	–	TMA 12	TAM 12	XAM 12	XAMI 12	XAMD 12	LXY
1101	D	–	–	TPU0A	T3HAB	TAW3	–	–	–	SCP	–	–	TMA 13	TAM 13	XAM 13	XAMI 13	XAMD 13	LXY
1110	E	TW1A	–	TPU1A	–	–	TAPU1	–	–	–	–	–	TMA 14	TAM 14	XAM 14	XAMI 14	XAMD 14	LXY
1111	F	TW2A	–	TPU2A	TR1AB	TAW5	TAPU2	IAK	–	–	–	–	TMA 15	TAM 15	XAM 15	XAMI 15	XAMD 15	LXY

The above table shows the relationship between machine language codes and machine language instructions. D_3–D_0 show the low-order 4 bits of the machine language code, and D_9–D_4 show the high-order 6 bits of the machine language code. The hexadecimal representation of the code is also provided. There are one-word instructions and two-word instructions, but only the first word of each instruction is shown. Do not use code marked "–."

The codes for the second word of a two-word instruction are described below.

	The second word
BL	10 0aaa aaaa
BML	10 0aaa aaaa
BLA	10 0p00 pppp
BMLA	10 0p00 pppp
SEA	00 0111 nnnn
SZD	00 0010 1011

Electrical characteristics

Absolute maximum ratings

Table 25 Absolute maximum ratings

Symbol	Parameter	Conditions	Ratings	Unit
V_{DD}	Supply voltage	-	–0.3 to 6.5	V
V_I	Input voltage P0, P1, $P2_0$/INT0, $P2_1$/INT1, P3, D_0–D_3, D_4/CNTR0, K, $\overline{RESET}$, X_{IN}	-	–0.3 to V_{DD}+0.3	V
V_O	Output voltage P0, P1, P2, P3, D_0–D_3, D_4/CNTR0, $\overline{RESET}$	Output transistors in cut-off state	–0.3 to V_{DD}+0.3	V
V_O	Output voltage C, X_{OUT}	-	–0.3 to V_{DD}+0.3	V
P_d	Power dissipation	Ta = 25 °C	300	mW
T_{opr}	Operating temperature range	-	–20 to 85	°C
T_{stg}	Storage temperature range	-	–40 to 125	°C

Recommended operating conditions

Table 26 Recommended operating conditions 1 (Ta = –20 °C to 85 °C, V_{DD} = 1.8 to 5.5 V, unless otherwise noted)

Symbol	Parameter	Conditions		Limits			Unit
				Min.	Typ.	Max.	
V_{DD}	Supply voltage (with a ceramic resonator)	f(STCK) ≤ 6MHz		4		5.5	V
		f(STCK) ≤ 4.4MHz		2.7		5.5	
		f(STCK) ≤ 2.2MHz		2		5.5	
		f(STCK) ≤ 1.1MHz		1.8		5.5	
V_{DD}	Supply voltage (when an external clock is used)	f(STCK) ≤ 4.8MHz		4		5.5	V
		f(STCK) ≤ 3.2MHz		2.7		5.5	
		f(STCK) ≤ 1.6MHz		2		5.5	
		f(STCK) ≤ 0.8MHz		1.8		5.5	
V_{RAM}	RAM back-up voltage	(at RAM back-up)		1.6		5.5	V
V_{SS}	Supply voltage				0		V
V_{IH}	“H” level input voltage	P0, P1, P2, P3, D_0–D_4, K		0.8V_{DD}		V_{DD}	V
		X_{IN}		0.7V_{DD}		V_{DD}	
		$\overline{RESET}$, INT0, INT1		0.85V_{DD}		V_{DD}	
		CNTR0		0.85V_{DD}		V_{DD}	
V_{IL}	“L” level input voltage	P0, P1, P2, P3, D_0–D_4, K		0		0.3V_{DD}	mA
		X_{IN}		0		0.3V_{DD}	
		$\overline{RESET}$, INT0, INT1		0		0.3V_{DD}	
		CNTR0		0		0.15V_{DD}	
$I_{OH(peak)}$	“H” level peak output current	P3, D_0–D_3	V_{DD} = 5V			–20	mA
			V_{DD} = 3V			–10	
		C, CNTR1	V_{DD} = 5V			–30	
			V_{DD} = 3V			–15	
$I_{OH(avg)}$	“H” level average output current (Note 1)	P3, D_0–D_3	V_{DD} = 5V			–10	mA
			V_{DD} = 3V			–5	
		C, CNTR1	V_{DD} = 5V			–15	
			V_{DD} = 3V			–7	
$I_{OL(peak)}$	“L” level peak output current	$\overline{P0}$, $\overline{P1}$, P2, P3, D_0–D_4, C, $\overline{RESET}$, CNTR0, CNTR1,	V_{DD} = 5V			24	mA
			V_{DD} = 3V			12	
$I_{OL(avg)}$	“L” level average output current (Note 1)	$\overline{P0}$, $\overline{P1}$, P2, P3, D_0–D_4, C, $\overline{RESET}$, CNTR0, CNTR1,	V_{DD} = 5V			12	mA
			V_{DD} = 3V			6	
$\Sigma I_{OH(avg)}$	“H” level total average current	P3, D_0–D_3, C, CNTR1				–30	mA
$\Sigma I_{OL(avg)}$	“L” level total average current	P0, $P1_0$, $P1_1$, $\overline{RESET}$				30	mA
		$P1_0$, $P1_1$, P2, P3, D_0–D_4, C, CNTR0, CNTR1				30	

Note 1.The average output current is the average value during 100ms.

Table 27 Recommended operating conditions 2 (Ta = –20 °C to 85 °C, V_{DD} = 1.8 to 5.5 V, unless otherwise noted)

Symbol	Parameter	Conditions		Limits			Unit
				Min.	Typ.	Max.	
f(X_{IN})	Oscillation frequency (with a ceramic resonator)	Through mode	V_{DD} = 4.0 V to 5.5 V			6	MHz
			V_{DD} = 2.7 V to 5.5 V			4.4	
			V_{DD} = 2.0 V to 5.5 V			2.2	
			V_{DD} = 1.8 V to 5.5 V			1.1	
		Internal frequency divided by 2	V_{DD} = 2.7 V to 5.5 V			6	
			V_{DD} = 2.0 V to 5.5 V			4.4	
			V_{DD} = 1.8 V to 5.5 V			2.2	
		Internal frequency divided by 4, 8	V_{DD} = 2.0 V to 5.5 V			6	
			V_{DD} = 1.8 V to 5.5 V			4.4	
f(X_{IN})	Oscillation frequency (with an external clock input)	Through mode	V_{DD} = 4.0 V to 5.5 V			4.8	MHz
			V_{DD} = 2.7 V to 5.5 V			3.2	
			V_{DD} = 2.0 V to 5.5 V			1.6	
			V_{DD} = 1.8 V to 5.5 V			0.8	
		Internal frequency divided by 2	V_{DD} = 2.7 V to 5.5 V			4.8	
			V_{DD} = 2.0 V to 5.5 V			3.2	
			V_{DD} = 1.8 V to 5.5 V			1.6	
		Internal frequency divided by 4, 8	V_{DD} = 2.0 V to 5.5 V			4.8	
			V_{DD} = 1.8 V to 5.5 V			3.2	
f(CNTR)	Timer external input frequency	CNTR0, CNTR1				f(STCK)/6	Hz
tw(CNTR)	Timer external input period (“H” and “L” pulse width)	CNTR0, CNTR1		3/f(STCK)			s
T_{PON}	Power-on reset circuit valid supply voltage rising time (Note 1)	V_{DD} = 0 → 1.8V				100	μs

Note 1. If the rising time exceeds the maximum rating value, connect a capacitor between the $\overline{\text{RESET}}$ pin and Vss at the shortest distance, and input “L” level to $\overline{\text{RESET}}$ pin until the value of supply voltage reaches the minimum operating voltage.

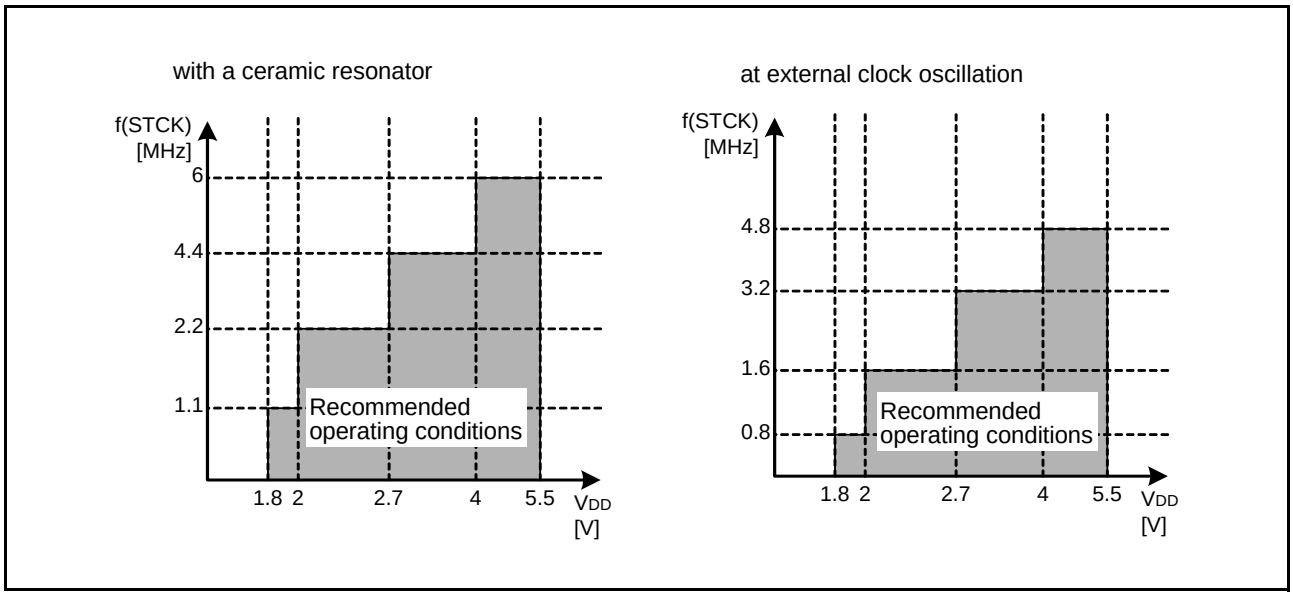


Fig 71. System clock (STCK) operating condition map

Electrical characteristics

Table 28 Electrical characteristics 1 (Ta = –20 °C to 85 °C, V_{DD} = 1.8 to 5.5 V, unless otherwise noted)

Symbol	Parameter		Test conditions		Limits			Unit
					Min.	Typ.	Max.	
V_{OH}	"H" level output voltage	P3, D_0–D_4 CNTR0	V_{DD} = 5V	I_{OH} = –10mA	3			V
				I_{OH} = –3mA	4.1			
			V_{DD} = 3V	I_{OH} = –5mA	2.1			
				I_{OH} = –1mA	2.4			
V_{OH}	"H" level output voltage	C CNTR1	V_{DD} = 5V	I_{OL} = –20mA	3			V
				I_{OL} = –6mA	4.1			
			V_{DD} = 3V	I_{OL} =–10mA	2.1			
				I_{OL} = –3mA	2.4			
V_{OL}	"L" level output voltage	P0, P1, P2, P3, D_0–D_4 $\overline{\text{RESET}}$, C, CNTR0, CNTR1	V_{DD} = 5V	I_{OL} = 15mA			2	V
				I_{OL} = 5mA			0.9	
			V_{DD} = 3V	I_{OL} = 9mA			1.4	
				I_{OL} = 3mA			0.9	
I_{IH}	"H" level input current	P0, P1, P2, P3, D_0–D_4, K $\overline{\text{RESET}}$, INT0, INT1 CNTR0	V_I = V_{DD}				2	μA
I_{IL}	"L" level input current	P0, P1, P2, P3, D_0–D_4, K $\overline{\text{RESET}}$, INT0, INT1 CNTR0	V_I = 0V P0, P1, P2 No pull-up				–2	μA
R_{PU}	Pull-up resistor value	P0, P1, P2 $\overline{\text{RESET}}$	V_I = 0V	V_{DD} = 5V	30	60	125	kΩ
				V_{DD} = 3V	50	120	250	
V_{T+} –$V_{T–}$	Hysteresis	$\overline{\text{RESET}}$, INT0, INT1	V_{DD} = 5V			1		V
			V_{DD} = 3V			0.4		
V_{T+} –$V_{T–}$	Hysteresis	CNTR0	V_{DD} = 5V			0.2		V
			V_{DD} = 3V			0.2		
I_{DD}	Supply current	at active mode (with a ceramic resonator) (Note 1)	V_{DD} = 5V f(X_{IN}) = 6MHz f(RING) = stop	f(STCK) = f(X_{IN})/8		1.2	2.4	mA
				f(STCK) = f(X_{IN})/4		1.3	2.6	
				f(STCK) = f(X_{IN})/2		1.6	3.2	
				f(STCK) = f(X_{IN})		2.2	4.4	
			V_{DD} = 3V f(X_{IN}) = 4MHz f(RING) = stop	f(STCK) = f(X_{IN})/8		0.3	0.6	mA
				f(STCK) = f(X_{IN})/4		0.4	0.8	
				f(STCK) = f(X_{IN})/2		0.6	1.2	
				f(STCK) = f(X_{IN})		0.8	1.6	
		at RAM back-up mode (POF instruction execution)	Ta = 25°C			0.1	3	μA
			V_{DD} = 5V				10	
			V_{DD} = 3V				6	

Note 1.The voltage drop detection circuit operation current (I_{RST}) is added.

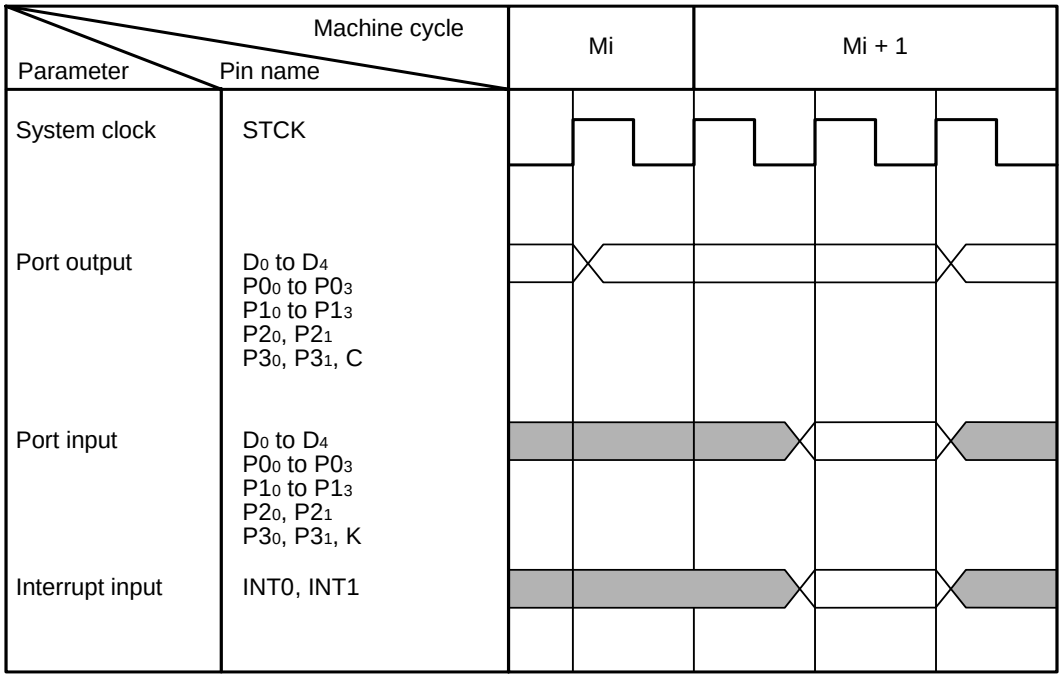
Voltage drop detection circuit characteristics

Table 29 Voltage drop detection circuit characteristics (Ta = –20 °C to 85 °C, unless otherwise noted)

Symbol	Parameter	Test conditions	Limits			Unit
			Min.	Typ.	Max.	
V_{RST-}	Detection voltage (reset occurs) (Note 1)	Ta = 25°C		1.65		V
		–20°C≤ Ta < 0°C	1.6		2.2	
		0°C≤ Ta < 50°C	1.3		2.1	
		50°C≤ Ta ≤ 85°C	1.1		1.8	
V_{RST+}	Detection voltage (reset release) (Note 2)	Ta = 25°C		1.75		V
		–20°C≤ Ta < 0°C	1.7		2.3	
		0°C≤ Ta < 50°C	1.4		2.2	
		50°C≤ Ta ≤ 85°C	1.2		1.9	
V_{INT}	Detection voltage (Interrupt occurs) (Note 3)	Ta = 25°C		1.85		V
		–20°C≤ Ta < 0°C	1.8		2.4	
		0°C≤ Ta < 50°C	1.5		2.3	
		50°C≤ Ta ≤ 85°C	1.3		2.2	
$V_{RST+} – V_{RST-}$	Detection voltage hysteresis			0.1		V
I_{RST}	Voltage drop detection circuit operation current (Note 4)	V_{DD} = 5V		40	80	μA
		V_{DD} = 3V		20	40	
		V_{DD} = 1.65V		7	15	
T_{RST}	Detection time (Note 5)	$V_{DD} \rightarrow (V_{RST-} – 0.1V)$		0.2	1.2	ms

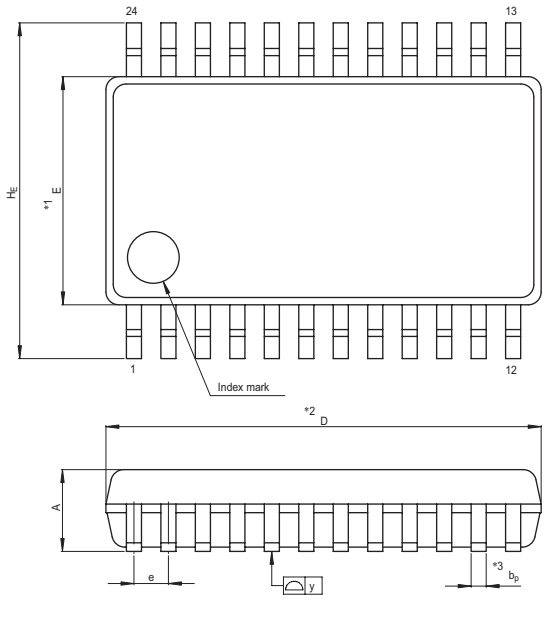
Note 1.The detection voltage ($V_{RST–}$) is defined as the voltage when reset occurs when the supply voltage (V_{DD}) is falling.
Note 2.The detection voltage (V_{RST+}) is defined as the voltage when reset is released when the supply voltage (V_{DD}) is rising from reset occurs.
Note 3.When the supply voltage goes lower than the detection voltage (V_{INT}), the voltage drop detection circuit interrupt request flag (VDF) is set to "1".
Note 4.I_{RST} is added to I_{DD} (power current).
Note 5.The detection time (T_{RST}) is defined as the time until reset occurs when the supply voltage (V_{DD}) is falling to [V_{RST-} –0.1V].

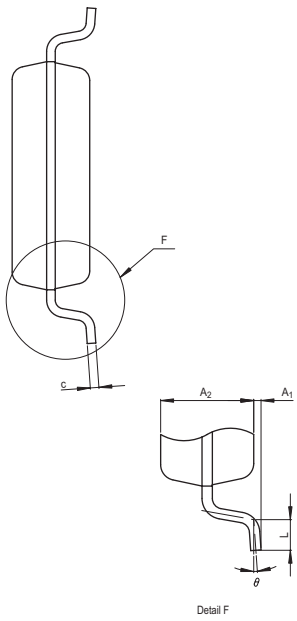
Basic timing diagram



PACKAGE OUTLINE

JEITA Package Code	RENESAS Code	Previous Code	MASS[Typ.]
P-SSOP24-5.3x10.1-0.80	PRSP0024GA-A	24P2Q-A	0.2g





NOTE)
1. DIMENSIONS "*1" AND "*2" DO NOT INCLUDE MOLD FLASH.
2. DIMENSION "*3" DOES NOT INCLUDE TRIM OFFSET.

Reference Symbol	Dimension in Millimeters		
	Min	Nom	Max
D	10.0	10.1	10.2
E	5.2	5.3	5.4
A_2	—	1.8	—
A	—	—	2.1
A_1	0	0.1	0.2
b_p	0.3	0.35	0.45
c	0.18	0.2	0.25
θ	0°	—	8°
H_E	7.5	7.8	8.1
e	0.65	0.8	0.95
y	—	—	0.10
L	0.4	0.6	0.8

REVISION HISTORY	4571 Group Datasheet

Rev.	Date	Description	
		Page	Summary
1.00	Feb. 20, 2006	–	First edition issued
1.01	Apr. 18, 2007	1	FEATURES: Description revised
		4	Table 2: Subroutine nesting added
		6	Table 5: Port P2; $P2_0$ → $P2_0$/INT0, $P2_1$ → $P2_1$/INT1
		30	Table 17: Timer control register W1; CNTR1 input → CNTR0 input
		31	• Timer control register PA: Description revised • Timer control register W3: Description revised (2) Prescaler: PRS → RPS
		32	(5) Timer 3: Description revised
		37	WATCHDOG TIMER: Description revised
		47	Table 21: Title revised
		48	Table 22: Title revised
		49	Fig 50: Ceramic resonator circuit → Ceramic oscillation circuit
		51	QzROM Writing Mode added
		59	NOTES ON NOISE added
		64	Timer control register W1: CNTR1 input → CNTR0 input
		69	SNZ1: $V1_{\underline{0}}$ → $V1_{\underline{1}}$
		86	SUPT: Description revised
		112	T3HAB: Description revised
		122	Table 28: I_{DD}; (with a ceramic oscillator) → (with a ceramic resonator)
1.02	May. 25, 2007	All pages	“PRELIMINARY” deleted

Renesas Technology Corp. Sales Strategic Planning Div. Nippon Bldg., 2-6-2, Ohte-machi, Chiyoda-ku, Tokyo 100-0004, Japan

RENESAS SALES OFFICES

http://www.renesas.com

Refer to "**http://www.renesas.com/en/network**" for the latest and detailed information.

Renesas Technology America, Inc.
450 Holger Way, San Jose, CA 95134-1368, U.S.A
Tel: <1> (408) 382-7500, Fax: <1> (408) 382-7501

Renesas Technology Europe Limited
Dukes Meadow, Millboard Road, Bourne End, Buckinghamshire, SL8 5FH, U.K.
Tel: <44> (1628) 585-100, Fax: <44> (1628) 585-900

Renesas Technology (Shanghai) Co., Ltd.
Unit 204, 205, AZIACenter, No.1233 Lujiazui Ring Rd, Pudong District, Shanghai, China 200120
Tel: <86> (21) 5877-1818, Fax: <86> (21) 6887-7898

Renesas Technology Hong Kong Ltd.
7th Floor, North Tower, World Finance Centre, Harbour City, 1 Canton Road, Tsimshatsui, Kowloon, Hong Kong
Tel: <852> 2265-6688, Fax: <852> 2730-6071

Renesas Technology Taiwan Co., Ltd.
10th Floor, No.99, Fushing North Road, Taipei, Taiwan
Tel: <886> (2) 2715-2888, Fax: <886> (2) 2713-2999

Renesas Technology Singapore Pte. Ltd.
1 Harbour Front Avenue, #06-10, Keppel Bay Tower, Singapore 098632
Tel: <65> 6213-0200, Fax: <65> 6278-8001

Renesas Technology Korea Co., Ltd.
Kukje Center Bldg. 18th Fl., 191, 2-ka, Hangang-ro, Yongsan-ku, Seoul 140-702, Korea
Tel: <82> (2) 796-3115, Fax: <82> (2) 796-2145

Renesas Technology Malaysia Sdn. Bhd
Unit 906, Block B, Menara Amcorp, Amcorp Trade Centre, No.18, Jalan Persiaran Barat, 46050 Petaling Jaya, Selangor Darul Ehsan, Malaysia
Tel: <603> 7955-9390, Fax: <603> 7955-9510

Colophon .7.0