PRELIMINARY PRODUCT INFORMATION

NEC

MOS INTEGRATED CIRCUIT

μPD780921, 780922, 780923, 780924

8-BIT SINGLE-CHIP MICROCONTROLLER

The μPD780921, 780922, 780923, and 780924, which are members of the μPD780924 subseries of the 78K/0 series, are suited to control general-purpose inverters.

Compared to the conventional μPD78014 subseries or μPD78018F subseries products, EMI (Electro Magnetic Interface) noise is reduced for the μPD780924 subseries products.

A flash memory version, the μPD78F0924, that can operate within the same power supply voltage range as the mask ROM version, and various development tools are also provided.

The details of functions are described in the following user's manuals. Be sure to read them before designing.

μPD780924 Subseries User's Manual : Planned

78K/0 Series User's Manual Instructions : IEU-1372

FEATURES

- Internal ROM and RAM

Item / Part Number	Program memory (ROM)	Data memory (internal high-capacity RAM)	Package
μPD780921	8 Kbytes	512 bytes	• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 x 14 mm)
μPD780922	16 Kbytes		
μPD780923	24 Kbytes	1024 bytes	
μPD780924	32 Kbytes		

- External memory expansion space: 48 Kbytes
- Minimum instruction execution time: 0.24 μs (@ fx = 8.38-MHz operation)
- I/O ports: 47
- 8-bit resolution A/D converter: eight channels
- Serial interface: two channels
- Timer: Five channels
- Power supply voltage: V_{DD} = 2.7 to 5.5 V

APPLICATIONS

Motor control for inverter-type air conditioners, washing machines, etc.

Document No. U11804EJ1V0PM00 (1st edition)
Date Published November 1996 N
Printed in Japan

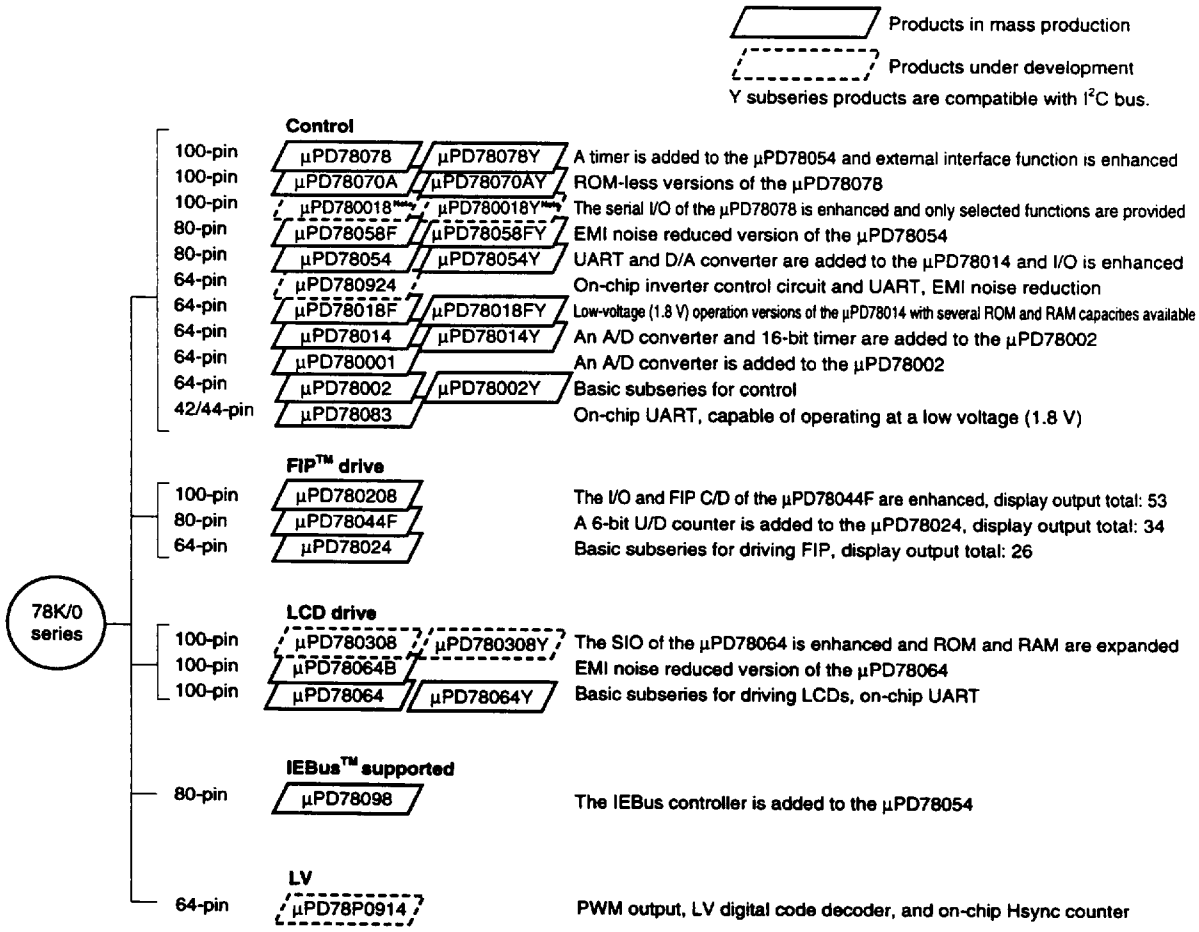
ORDERING INFORMATION

Part Number	Package
μPD780921CW-xxx	64-pin plastic shrink DIP (750 mil)
μPD780922CW-xxx	64-pin plastic shrink DIP (750 mil)
μPD780923CW-xxx	64-pin plastic shrink DIP (750 mil)
μPD780924CW-xxx	64-pin plastic shrink DIP (750 mil)
μPD780921GC-xxx-AB8	64-pin plastic QFP (14 x 14 mm)
μPD780922GC-xxx-AB8	64-pin plastic QFP (14 x 14 mm)
μPD780923GC-xxx-AB8	64-pin plastic QFP (14 x 14 mm)
μPD780924GC-xxx-AB8	64-pin plastic QFP (14 x 14 mm)

Remark xxx indicates the ROM code suffix.

78K/0 SERIES DEVELOPMENT

The 78K/0 series product line-up is shown below. Subseries names are shown inside frames.



Note Under planning

The table below shows the main differences between subseries.

Subseries \ Functions		ROM Capacity	Timer				8-bit A/D	8-bit D/A	Serial Interface	I/O	V_{DD} MIN. Value	External Expansion
			8-bit	16-bit	Watch	WDT						
For Control	μPD78078	32 K to 60 K	4ch	1ch	1ch	1ch	8ch	2ch	3ch (UART: 1ch)	88	1.8 V	Available
	μPD78070A	—								61	2.7 V	
	μPD780018	48 K to 60 K						—	2ch	88		
	μPD78058F		2ch					2ch	3ch (UART : 1ch)	69		
	μPD78054	16 K to 60 K									2.0 V	
	μPD780924	8 K to 32K	3ch	Note	—			—	2ch (UART : 2ch)	47	2.7ch	
	μPD78018F	8 K to 60 K	2ch	1ch	1ch				2ch	53	1.8 V	
	μPD78014	8 K to 32 K									2.7 V	
	μPD780001	8 K		—	—				1ch	39		—
	μPD78002	8 K to 16 K			1ch		—			53		Available
	μPD78083	8 K to 16 K			—		8ch		1ch (UART : 1ch)	33	1.8 V	—
For FIP drive	μPD780208	32 K to 60 K	2ch	1ch	1ch	1ch	8ch	—	2ch	74	2.7 V	—
	μPD78044F	16 K to 40 K								68		
	μPD78024	24 K to 32 K								54		
For LCD drive	μPD780308	48 K to 60 K	2ch	1ch	1ch	1ch	8ch	—	3ch (UART: 1ch)	57	1.8 V	—
	μPD78064B	32 K							2ch (UART : 1ch)		2.0 V	
	μPD78064	16 K to 32 K										
IEBus Support	μPD78098	32 K to 60 K	2ch	1ch	1ch	1ch	8ch	2ch	3ch (UART: 1ch)	69	2.7 V	Available
For LV	μPD78P0914	32K	6ch	—	—	1ch	8ch	—	2ch	54	4.5 V	Available

Note 10-bit timer: one channel

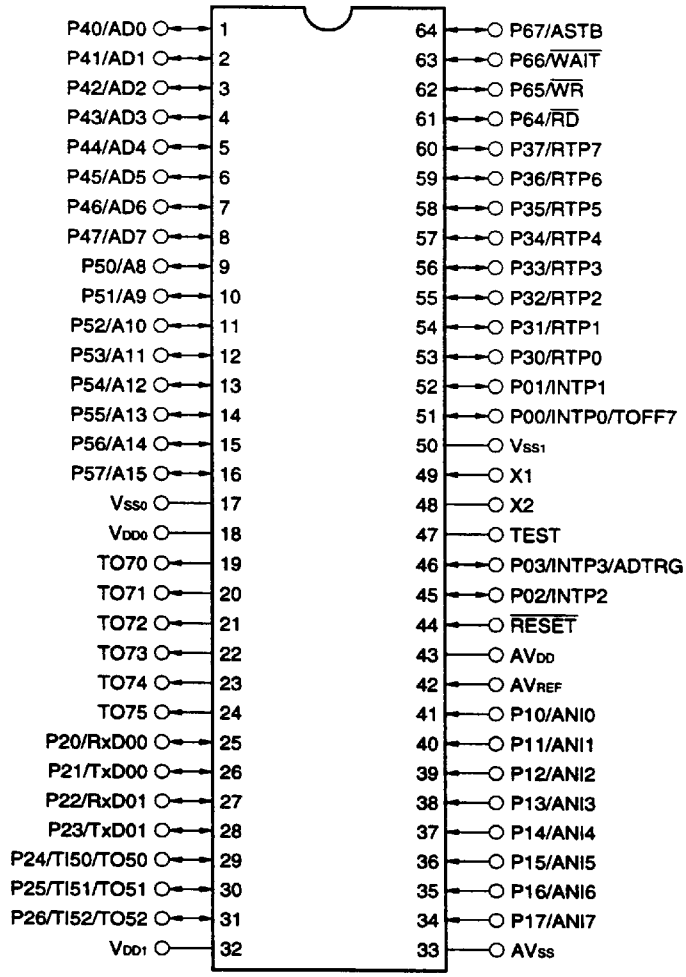
FUNCTION OVERVIEW

Item \ Product Name		μPD780921	μPD780922	μPD780923	μPD780924
Internal memory	ROM	8 Kbytes	16 Kbytes	24 Kbytes	32 Kbytes
	Internal high-speed RAM	512 bytes		1024 bytes	
Memory space		64 Kbytes			
General-purpose register		8 bits x 32 registers (8 bits x 8 registers x 4 banks)			
Instruction cycle		On-chip instruction execution time variable function 0.24 μs/0.48 μs/0.96 μs/1.9 μs/3.8 μs (@ 8.38-MHz operation with system clock)			
Instruction set		• 16-bit operation • Multiply/divide (8 bits x 8 bits, 16 bits ÷ 8 bits) • Bit manipulate (set, reset, test, Boolean operation) • BCD adjust, etc.			
I/O ports		Total : 47 • CMOS inputs : 8 • CMOS I/Os : 39			
Real-time output ports		8 bits x 1 or 4 bits x 2			
A/D converter		• 8-bit resolution x 8 channels • Power supply voltage: AV_{DD} = 2.7 to 5.5 V			
Serial interface		UART x 2 channels			
Timer		• 8-bit timer/event counter : 3 channels • 10-bit timer : 1 channel • Watchdog timer : 1 channel			
Timer output		9 (8-bit PWM output x 3, and inverter control output x 6)			
Vectored interrupt	Maskable interrupts	Internal: 12, external: 4			
	Non-maskable interrupt	Internal: 1			
	Software interrupt	1			
Power supply voltage		V_{DD} = 2.7 to 5.5 V			
Operating ambient temperature		T_A = −40 to +85 °C			
Package		• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 x 14 mm)			

CONTENTS

1. PIN CONFIGURATION (Top View)

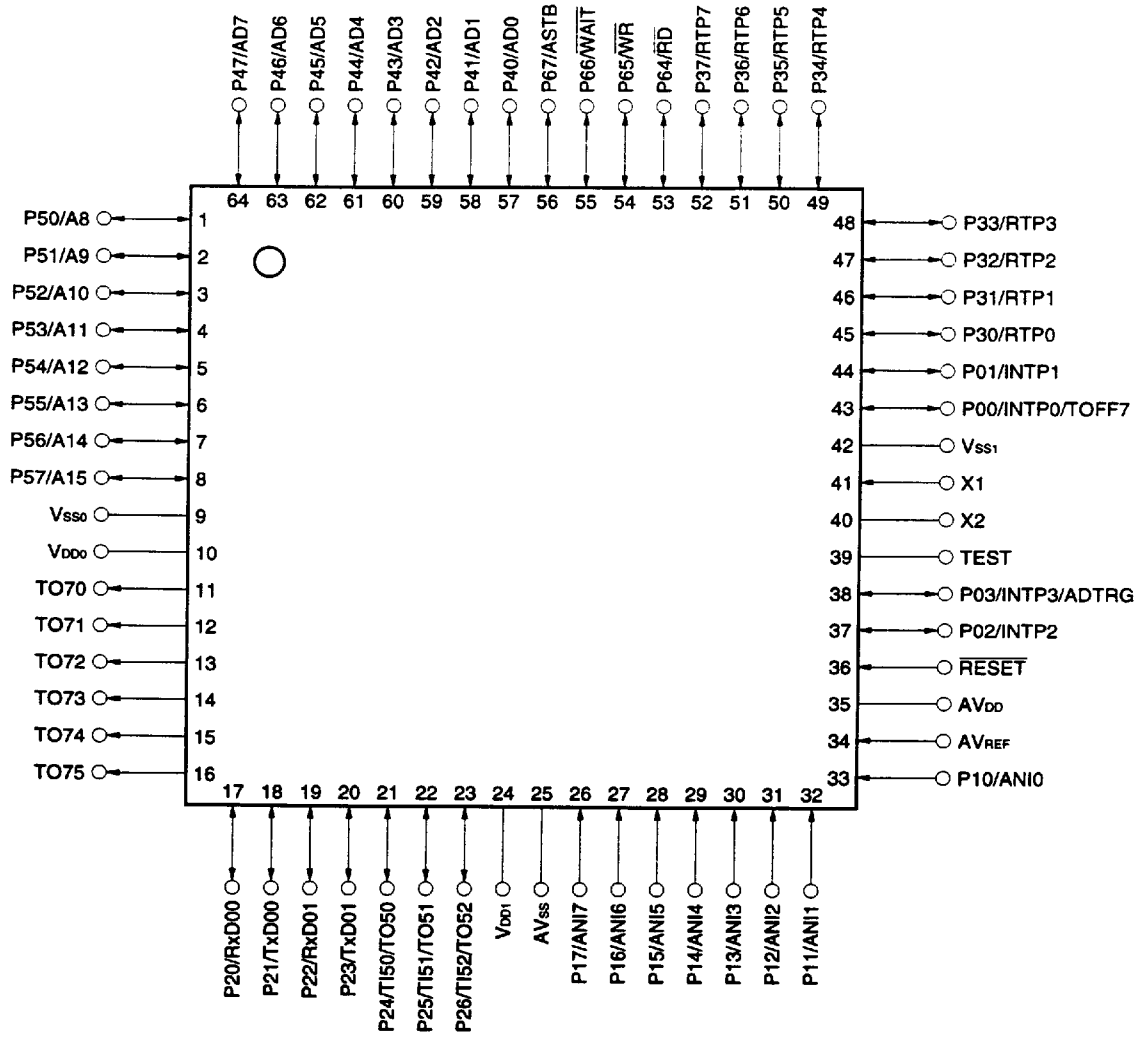
- **64-Pin Plastic Shrink DIP (750 mil)**
  μPD780921CW-xxx, 780922CW-xxx, 780923CW-xxx, 780924CW-xxx



Cautions
1. **Connect the TEST pin to V_{SS0} directly.**
2. **Connect the AV_{DD} pin to V_{DD0}.**
3. **Connect the AV_{SS} pin to V_{SS0}.**

Remark When the μPD780921, 780922, 780923, or 780924 is used in application fields that require reduction of the noise generated from inside the microcontroller, the implementation of noise reduction measures, such as supplying voltage to V_{DD0} and V_{SS1} individually and connecting V_{SS0} and V_{SS1} to different ground lines, is recommended.

- **64-Pin Plastic QFP (14 x 14 mm)**
  μPD780921GC-xxx-AB8, 780922GC-xxx-AB8, 780923GC-xxx-AB8, 780924GC-xxx-AB8

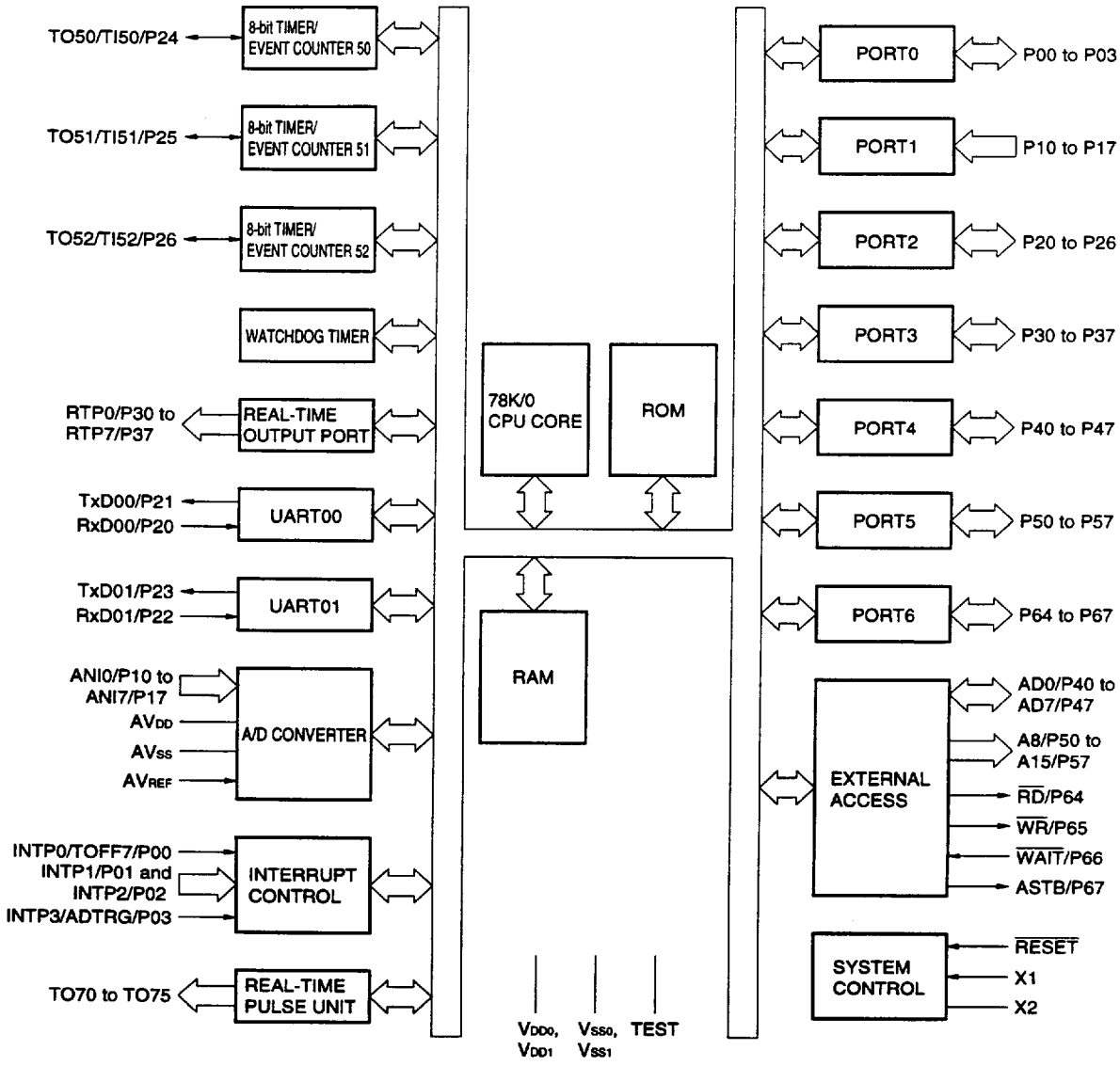


Cautions 1. Connect the TEST pin to V_{SS0} directly.
2. Connect the AV_{DD} pin to V_{DD0}.
3. Connect the AV_{SS} pin to V_{SS0}.

Remark When the μPD780921, 780922, 780923, or 780924 is used in application fields that require reduction of the noise generated from inside the microcontroller, the implementation of noise reduction measures, such as supplying voltage to V_{DD0} and V_{SS1} individually and connecting V_{SS0} and V_{SS1} to different ground lines, is recommended.

A8 to A15	:	Address Bus
AD0 to AD7	:	Address/Data Bus
ADTRG	:	AD Trigger Input
ANI0 to ANI7	:	Analog Input
ASTB	:	Address Strobe
AV_{DD}	:	Analog Power Supply
AV_{REF}	:	Analog Reference Voltage
AV_{SS}	:	Analog Ground
INTP0 to INTP3	:	Interrupt From Peripherals
P00 to P03	:	Port 0
P10 to P17	:	Port 1
P20 to P26	:	Port 2
P30 to P37	:	Port 3
P40 to P47	:	Port 4
P50 to P57	:	Port 5
P64 to P67	:	Port 6
$\overline{RD}$	:	Read Strobe
$\overline{RESET}$	:	Reset
RTP0 to RTP7	:	Real-time Port
RxD00, RxD01	:	Receive Data
TEST	:	Test
TI50 to TI52	:	Timer Input
TO50 to TO52, TO70 to TO75	:	Timer Output
TOFF7	:	Timer Output Off
TxD00, TxD01	:	Transmit Data
V_{DD0}, V_{DD1}	:	Power Supply
V_{SS0}, V_{SS1}	:	Ground
$\overline{WAIT}$	:	Wait
$\overline{WR}$	:	Write Strobe
X1, X2	:	Crystal

2. BLOCK DIAGRAM



Remark Internal ROM and RAM capacity varies depending on the product.

3. PIN FUNCTION LIST

3.1 PORT PINS

Pin Name	I/O	Function	After Reset	Alternate Function
P00	I/O	Port 0. 4-bit I/O port. Input/output can be specified bit-wise. When used as an input port, an on-chip pull-up resistor can be connected by software.	Input	INTP0/TOFF7
P01				INTP1
P02				INTP2
P03				INTP3/ADTRG
P10 to P17	Input	Port 1. 8-bit input only port.	Input	ANI0 to ANI7
P20	I/O	Port 2. 7-bit I/O port. Input/output can be specified bit-wise. When used as an input port, an on-chip pull-up resistor can be connected by software.	Input	RxD00
P21				TxD00
P22				RxD01
P23				TxD01
P24				TI50/TO50
P25				TI51/TO51
P26				TI52/TO52
P30 to P37	I/O	Port 3. 8-bit I/O port. Input/output can be specified bit-wise. When used as an input port, an on-chip pull-up resistor can be connected by software.	Input	RTP0 to RTP7
P40 to P47	I/O	Port 4. 8-bit I/O port. Input/output can be specified bit-wise. When used as an input port, an on-chip pull-up resistor can be connected by software.	Input	AD0 to AD7
P50 to P57	I/O	Port 5. 8-bit I/O port. Input/output can be specified bit-wise. LEDs can be driven directly. When used as an input port, an on-chip pull-up resistor can be connected by software..	Input	A8 to A15
P64	I/O	Port 6. 4-bit I/O port. Input/output can be specified bit-wise. When used as an input port, an on-chip pull-up resistor can be connected by software.	Input	$\overline{RD}$
P65				$\overline{WR}$
P66				$\overline{WAIT}$
P67				ASTB

3.2 NON PORT PINS

Pin Name	I/O	Function	After Reset	Alternate Function
INTP0	Input	External interrupt input that can specify the effective edge (rising edge, falling edge, or both rising and falling edges) can be specified.	Input	P00/TOFF7
INTP1			Input	P01
INTP2			Input	P02
INTP3			Input	P03/ADTRG
TI50	Input	External count clock input to timer (TM50).	Input	P24/TO50
TI51		External count clock input to timer (TM51).	Input	P25/TO51
TI52		External count clock input to timer (TM52).	Input	P26/TO52
TO50	Output	Timer (TM50) output.	Input	P24/TI50
TO51		Timer (TM51) output.	Input	P25/TI51
TO52		Timer (TM52) output.	Input	P26/TI52
RTP0 to RTP7	Output	Real-time output port that outputs pulses in synchronization with trigger signals outputs from the real-time pulse unit.	Input	P30 to P37
TxD00	Output	Asynchronous serial interface serial data output.	Input	P21
TxD01			Input	P23
RxD00	Input	Asynchronous serial interface serial data input.	Input	P20
RxD01			Input	P22
ANI0 to ANI7	Input	A/D converter analog input.	Input	P10 to P17
ADTRG	Input	External trigger signal input to the A/D converter.	Input	P03/INTP3
TO70 to TO75	Output	Timer output for the 3-phase PWM inverter control.	Hi-Z	–
TOFF7	Input	Timer output (TO70 to TO75) stop interrupt input.	Input	P00/INTP0
AD0 to AD7	I/O	Lower address/data bus when memory is expanded externally.	Input	P40 to P47
A8 to A15	Output	Upper address bus when memory is expanded externally.	Input	P50 to P57
$\overline{RD}$	Output	Strobe signal output for external memory read operation.	Input	P64
$\overline{WR}$		Strobe signal output for external memory write operation.	Input	P65
$\overline{WAIT}$	Input	Wait insertion when accessing external memory.	Input	P66
ASTB	Output	Strobe output that externally latches address information output to ports 4 and 5 to access external memory.	Input	P67
AV_{REF}	Input	A/D converter reference voltage input.	–	–
AV_{DD}	–	A/D converter analog power supply. Connect to V_{DD0}.	–	–
AV_{SS}	–	A/D converter ground potential. Connect to V_{SS0}.	–	–
$\overline{RESET}$	Input	System reset input.	–	–
X1	Input	Crystal connection for system clock oscillation.	–	–
X2	–		–	–
V_{DD0}	–	Positive power supply for ports.	–	–
V_{SS0}	–	Ground potential for ports.	–	–
V_{DD1}	–	Positive power supply except for ports.	–	–
V_{SS1}	–	Ground potential except for ports.	–	–
TEST	–	Test mode set pin. Connect to V_{SS0} directly.	–	–

3.3 RECOMMENDED CONNECTION OF UNUSED PINS

The recommended connections of unused pins are shown in Table 3-1.

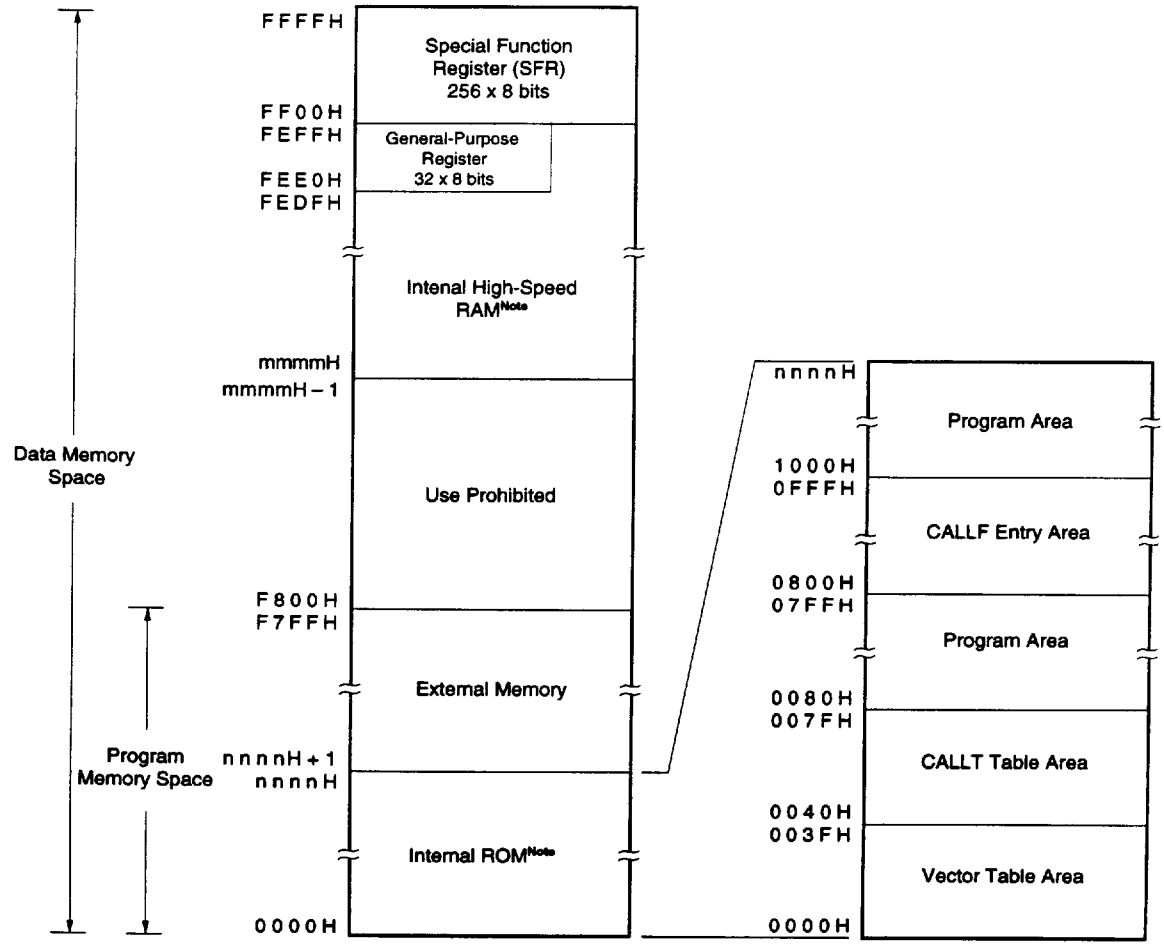
Table 3-1. Recommended Connection of Unused Pins

Pin Name	I/O	Recommended Connection of Unused Pins
P00/INTP0/TOFF7	I/O	Individually connect to V_{SS0} via a resistor.
P01/INTP1		
P02/INTP2		
P03/INTP3/ADTRG		
P10/ANI0 to P17/ANI7	Input	Individually connect to V_{DD0} or V_{SS0} via a resistor.
P20/RxD00	I/O	
P21/TxD00		
P22/RxD01		
P23/TxD01		
P24/TI50/TO50		
P25/TI51/TO51		
P26/TI52/TO52		
P30/RTP0 to P37/RTP7		
P40/AD0 to P47/AD7		
P50/A8 to P57/A15		
P64/$\overline{RD}$		
P65/$\overline{WR}$		
P66/$\overline{WAIT}$		
P67/ASTB		
TO70 to TO75	Output	Open
AV_{DD}	–	Connect to V_{DD0}.
AV_{REF}		Connect to V_{SS0}.
AV_{SS}		
TEST		Connect to V_{SS0} directly.

4. MEMORY SPACE

The memory map of the μPD780921, 780922, 780923, and 780924 is shown in Figure 4-1.

Figure 4-1. Memory Map



Note The internal ROM and internal high-speed RAM capacity differ depending on the product (see the table below).

Product Name	Intenal ROM End Address nnnnH	Internal High-Speed RAM Start Address mmmmH
μPD780921	1FFFH	FD00H
μPD780922	3FFFH	
μPD780923	5FFFH	FB00H
μPD780924	7FFFH	

5. PERIPHERAL HARDWARE FUNCTION FEATURES

5.1 PORTS

Two kinds of I/O ports are provided.

- CMOS input (Port 1) : 8
- CMOS input/output (Port 0, ports 2 to 6) : 39

Total : 47

Table 5-1. Functions of Ports

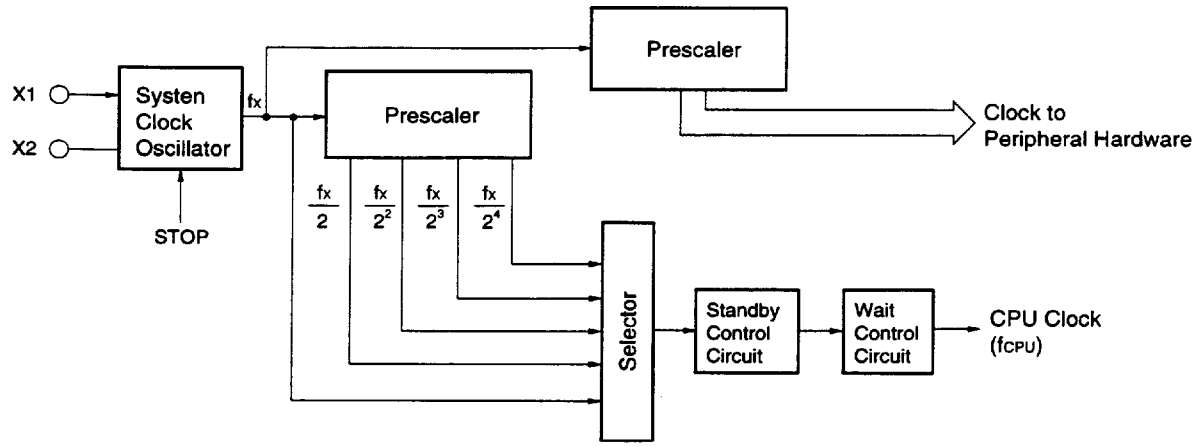
Port Name	Pin Name	Function
Port 0	P01 to P03	I/O port. Input/output can be specified bit-wise. When used as an input port, an on-chip pull-up resistor can be connected by software.
Port 1	P10 to P17	Input only port.
Port 2	P20 to P26	I/O port. Input/output can be specified bit-wise. When used as an input port, an on-chip pull-up resistor can be connected by software.
Port 3	P30 to P37	I/O port. Input/output can be specified bit-wise. When used as an input port, an on-chip pull-up resistor can be connected by software.
Port 4	P40 to P47	I/O port. Input/output can be specified bit-wise. When used as an input port, an on-chip pull-up resistor can be connected by software.
Port 5	P50 to P57	I/O port. Input/output can be specified bit-wise. When used as an input port, an on-chip pull-up resistor can be connected by software. LEDs can be driven directly.
Port 6	P64 to P67	I/O port. Input/output can be specified bit-wise. When used as an input port, an on-chip pull-up resistor can be connected by software.

5.2 CLOCK GENERATOR

A system clock generator is incorporated.
The instruction exection time can be changed.

- 0.24 μs/0.48 μs/0.96 μs/1.9 μs/3.8 μs (@ 8.38-MHz operation with system clock)

Figure 5-1. Clock Generator Block Diagram



5.3 TIMER/EVENT COUNTERS

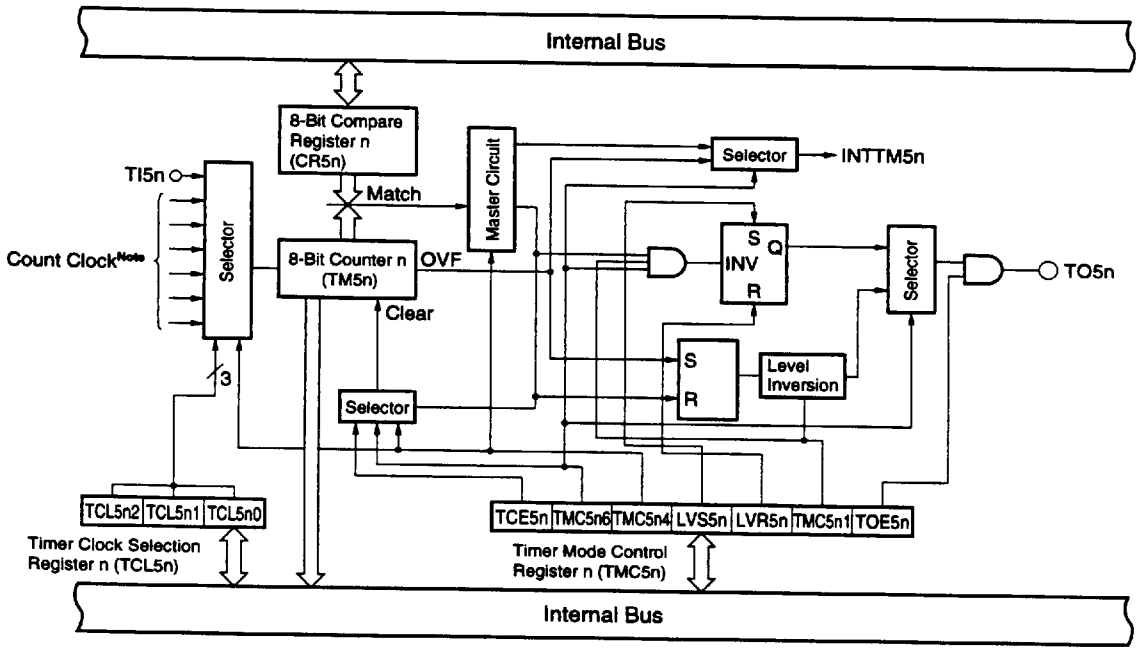
Five timer/event counter channels are incorporated.

- 8-bit timer/event counter : 3 channels
- 10-bit timer : 1 channel
- Watchdog timer : 1 channel

Table 5-2. Types and Functions of Timer/Event Counters

		8-bit Timer/Event Counter	10-bit Timer	Watchdog Timer
Type	Interval timer	3 channels	1 channel	1 channel
	External event counter	3 channels	–	–
Functions	Timer output	3 outputs	6 outputs	–
	PWM output	3 outputs	–	–
	Square wave output	3 outputs	–	–
	Interrupt request	3	1	1

Figure 5-2. 8-Bit Timer/Event Counter Block Diagram



Note Count clock differs depending on the timer
TM50 : fx/2, fx2³, fx2⁵, fx2⁷, fx2⁹, fx¹¹
TM51 : fx, fx/2, fx2², fx2³, fx2⁴, fx2⁵
TM52 : fx2⁴, fx2⁵, fx2⁶, fx2⁷, fx2⁸, fx2⁹

Remark n = 0 to 2

Figure 5-3. 10-bit Timer Block Diagram

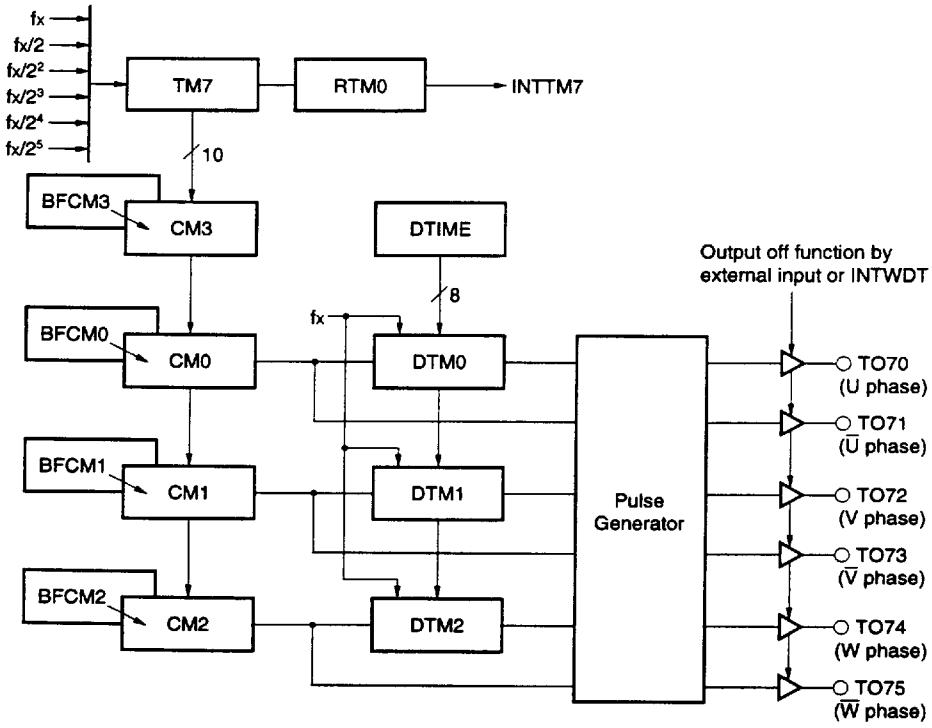
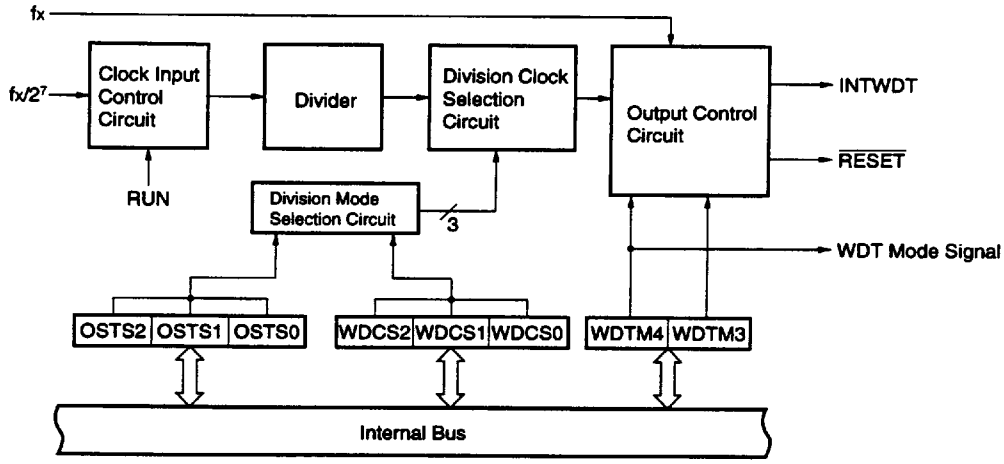


Figure 5-4. Watchdog Timer Block Diagram

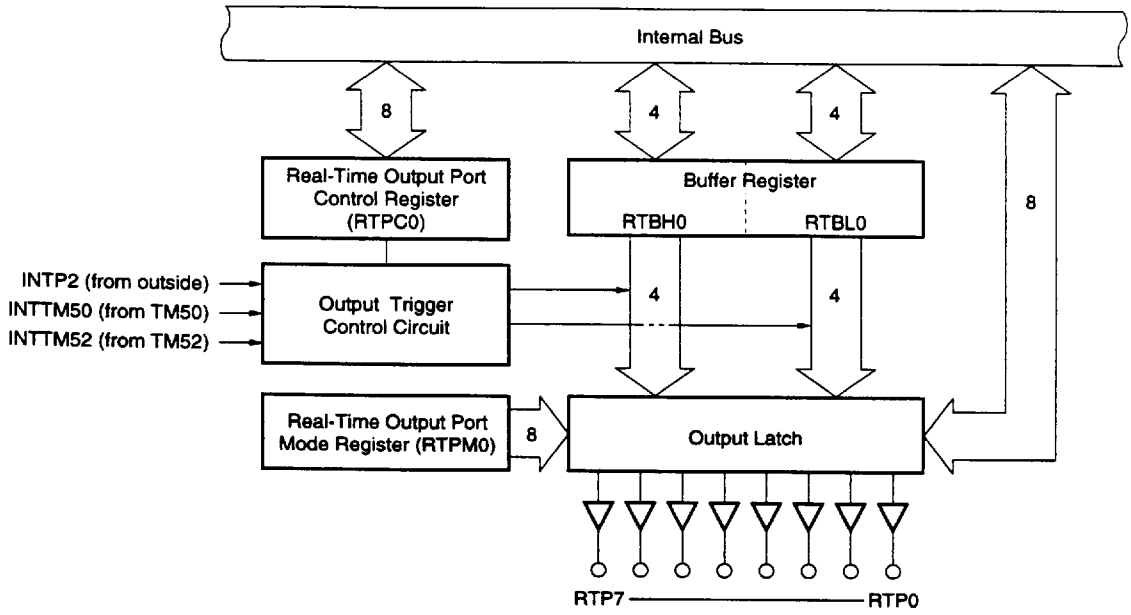


5.4 REAL-TIME OUTPUT PORT

The real-time output port outputs data stored in buffers in synchronization with match interrupts of 8-bit timer/event counters (TM50 and TM52) or external interrupts, enabling to output pulses without jitter.

Therefore, the real-time output method is suited for applications which output given patterns at set intervals.

Figure 5-5. Real-Time Output Port Block Diagram

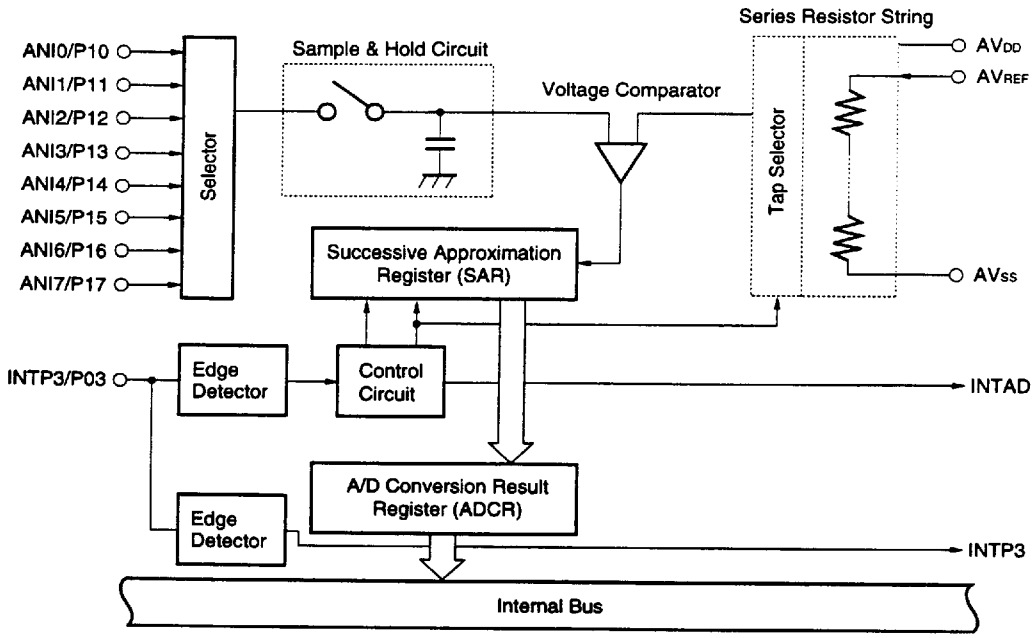


5.5 A/D CONVERTER

Eight 8-bit resolution A/D converter channels are incorporated.
A/D conversion by the following two methods.

- Hardware start
- Software start

Figure 5-6. A/D Converter Block Diagram



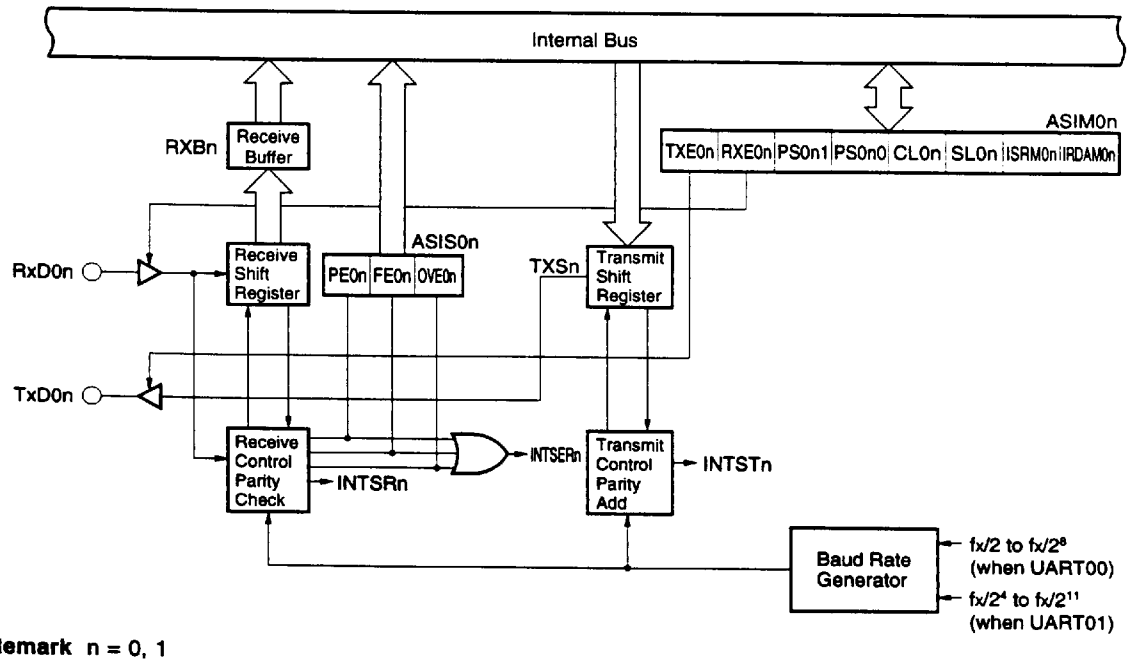
5.6 SERIAL INTERFACE

Two independent serial interface channels (UART00 and UART01) are incorporated.

Each serial interface incorporates a baud-rate generator. Therefore, it is possible to set a serial transfer rate that is independent of the operating clock frequency.

The serial transfer rate can be set from 75 to 76800 bps (@ fx = 8.38-MHz operation) by setting the mode register.

Table 5-7. Serial Interface Block Diagram



Remark n = 0, 1

6. INTERRUPT FUNCTIONS

There are 18 interrupt functions of three different kinds as shown below.

- Non-maskable interrupt : 1
- Maskable interrupts : 16
- Software interrupt : 1

Table 6-1. Interrupt Source List

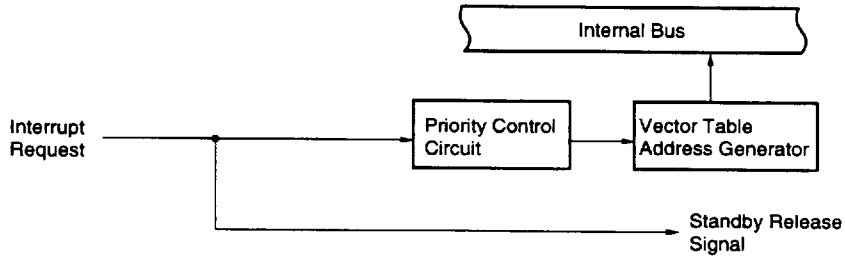
Interrupt Type	Default Priority Note 1	Interrupt Source		Internal/ External	Vector Table Address	Basic Configuratin Type Note 2
		Name	Trigger			
Non-maskable	—	INTWDT	Watchdog timer overflow (when non-maskable interrupt is selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Watchdog timer overflow (when interval timer is selected)			(B)
	1	INTP0	Pin input edge detection	External	0006H	(C)
	2	INTP1			0008H	
	3	INTP2			000AH	
	4	INTP3			000CH	
	5	INTTM7	TM7 under flow	Internal	000EH	(B)
	6	INTSER0	UART00 receive error generation		0010H	
	7	INTSR0	UART00 receive termination		0012H	
	8	INTST0	UART00 transmit termination		0014H	
	9	INTSER1	UART01 receive error generation		0016H	
	10	INTSR1	UART01 receive termination		0018H	
	11	INTST1	UART01 transmit termination		001AH	
	12	INTTM50	TM50 and CR50 match signal generation		001CH	
	13	INTTM51	TM51 and CR51 match signal generation		001EH	
	14	INTTM52	TM52 and CR52 match signal generation		0020H	
	15	INTAD0	A/D conversion termination		0022H	
Software	—	BRK	BRK instruction execution	—	003EH	(D)

Notes 1. The default priority is the priority applicable when more than one maskable interrupt is generated. 0 is the highest priority and 15, the lowest.

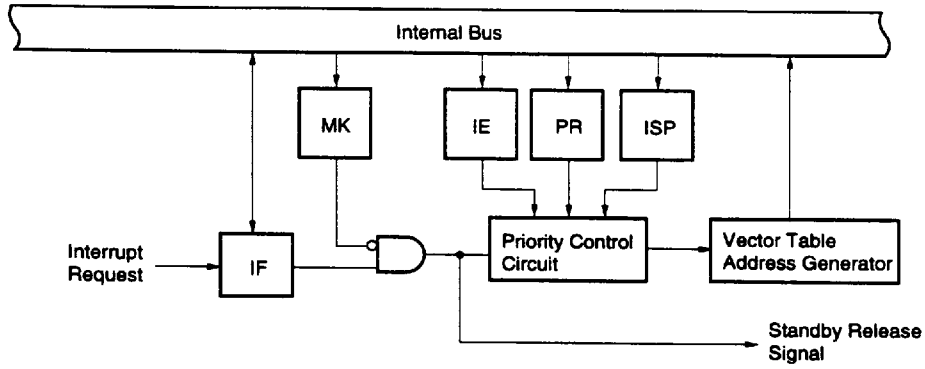
2. Basic configuration types (A) to (D) correspond to (A) to (D) on the next page.

Figure 6-1. Basic Interrupt Function Configuration (1/2)

(A) Internal non-maskable interrupt



(B) Internal maskable interrupt



(C) External Maskable Interrupt

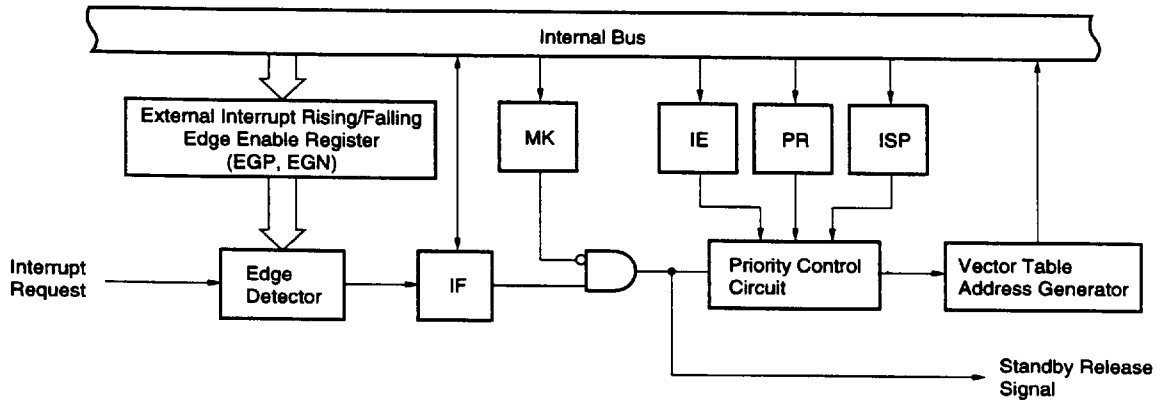
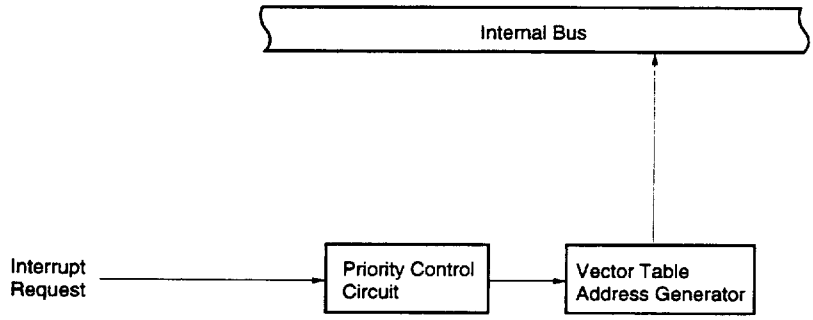


Figure 6-1. Basic Interrupt Function Configuration (2/2)

(D) Software Interrupt



IF : Interrupt request flag
IE : Interrupt enable flag
ISP : In-service priority flag
MK : Interrupt mask flag
PR : Priority specification flag

7. EXTERNAL DEVICE EXPANSION FUNCTION

The external device expansion function is used to connect external devices to areas other than the internal ROM, RAM and SFR.

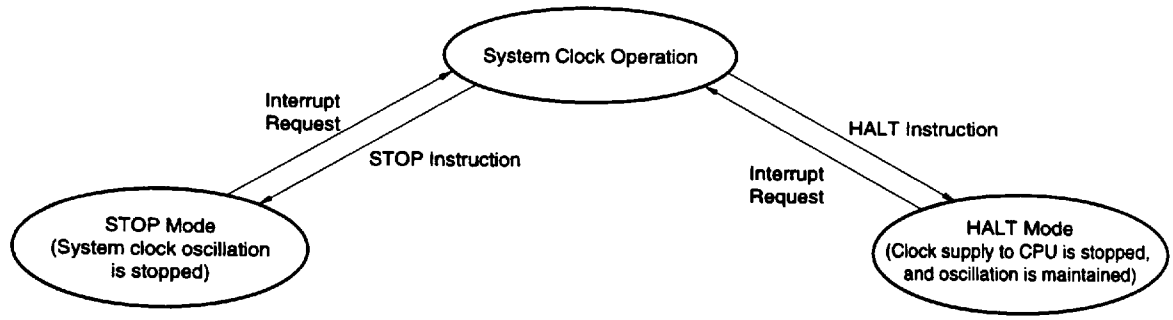
Ports 4 to 6 are used for connection with external devices.

8. STANDBY FUNCTIONS

There are the following two standby functions to reduce the current consumption.

- HALT mode : The CPU operating clock is stopped. The average current consumption can be reduced by intermittent operation in combination with the normal operating mode.
- STOP mode : The system clock oscillation is stopped. The whole operation by the system clock is stopped, so that the system operates with ultra-low power dissipation.

Figure 8-1. Standby Functions



9. RESET FUNCTIONS

There are the following two reset methods.

- External reset by $\overline{\text{RESET}}$ input.
- Internal reset by watchdog timer runaway time detection

10. INSTRUCTION SET

(1) 8-bit instructions

MOV, XCH, ADD, ADDC, SUB, SUBC, AND, OR, XOR, CMP, MULU, DIVUW, INC, DEC, ROR, ROL, RORC, ROLC, ROR4, ROL4, PUSH, POP, DBNZ

1st Operand \ 2nd Operand	#byte	A	r Note	sfr	saddr	!addr16	PSW	[DE]	[HL]	[HL+byte] [HL+B] [HL+C]	$addr16	1	None
A	ADD ADDC SUB SUBC AND OR XOR CMP		MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP		ROR ROL RORC ROLC	
r	MOV	MOV ADD ADDC SUB SUBC AND OR XOR CMP											INC DEC
B, C											DBNZ		
sfr	MOV	MOV											
saddr	MOV ADD ADDC SUB SUBC AND OR XOR CMP	MOV									DBNZ		INC DEC
!addr16		MOV											
PSW	MOV	MOV											PUSH POP
[DE]		MOV											
[HL]		MOV											ROR4 ROL4
[HL+byte] [HL+B] [HL+C]		MOV											
X													MULU
C													DIVUW

Note Except r=A

(2) 16-bit instructions

MOVW, XCHW, ADDW, SUBW, CMPW, PUSH, POP, INCW, DECW

2nd Operand / 1st Operand	#byte	AX	rp Note	sfrp	saddrp	!addr16	SP	None
AX	ADDW SUBW CMPW		MOVW XCHW	MOVW	MOVW	MOVW	MOVW	
rp	MOVW	MOVW Note						INCW, DECW PUSH, POP
sfrp	MOVW	MOVW						
saddrp	MOVW	MOVW						
!addr16		MOVW						
SP	MOVW	MOVW						

Note Only when rp = BC, DE, HL.

(3) Bit manipulation instructions

MOV1, AND1, OR1, XOR1, SET1, CLR1, NOT1, BT, BF, BTCLR

2nd Operand / 1st Operand	A.bit	sfr.bit	saddr.bit	PWS.bit	[HL].bit	CY	$addr16	None
A.bit						MOV1	BT BF BTCLR	SET1 CLR1
sfr.bit						MOV1	BT BF BTCLR	SET1 CLR1
saddr.bit						MOV1	BT BF BTCLR	SET1 CLR1
PSW.bit						MOV1	BT BF BTCLR	SET1 CLR1
[HL].bit						MOV1	BT BF BTCLR	SET1 CLR1
CY	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1			SET1 CLR1 NOT1

(4) Call instruction/branch instructions

CALL, CALLF, CALLT, BR, BC, BNC, BZ, BNZ, BT, BF, BTCLR, DBNZ

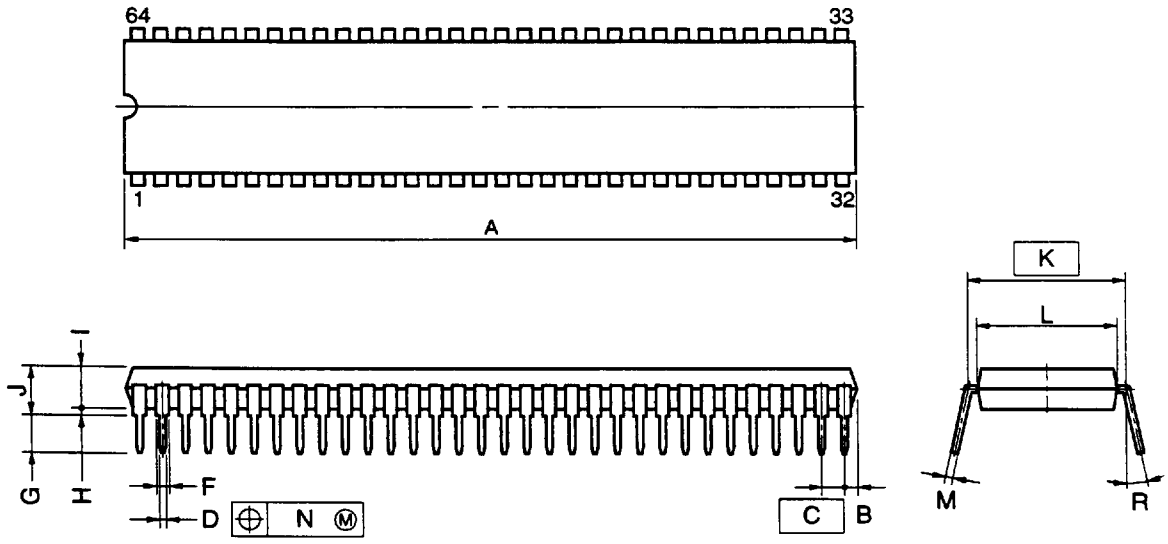
2nd Operand / 1st Operand	AX	!addr16	!addr11	[addr5]	$addr16
Basic instruction	BR	CALL, BR	CALLF	CALLT	BR, BC, BNC, BZ, BNZ
Compound instruction					BT,BF,BTCLR, DBNZ

(5) Other Instructions

ADJBA, ADJBS, BRK, RET, RETI, RETB, SEL, NOP, EI, DI, HALT, STOP

11. PACKAGE DRAWINGS

64 PIN PLASTIC SHRINK DIP (750 mil)



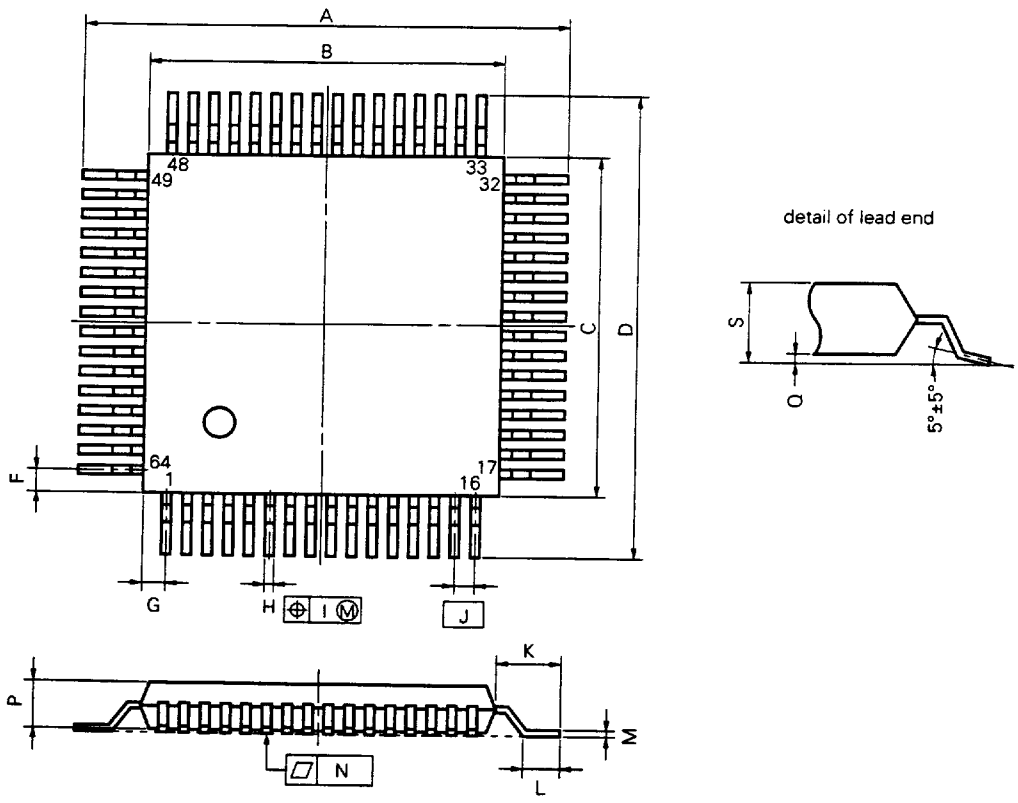
NOTE

1) Each lead centerline is located within 0.17 mm (0.007 inch) of its true position (T.P.) at maximum material condition.
2) Item "K" to center of leads when formed parallel.

ITEM	MILLIMETERS	INCHES
A	58.68 MAX.	2.311 MAX.
B	1.78 MAX.	0.070 MAX.
C	1.778 (T.P.)	0.070 (T.P.)
D	0.50 ± 0.10	$0.020^{+0.004}_{-0.005}$
F	0.9 MIN.	0.035 MIN.
G	3.2 ± 0.3	0.126 ± 0.012
H	0.51 MIN.	0.020 MIN.
I	4.31 MAX.	0.170 MAX.
J	5.08 MAX.	0.200 MAX.
K	19.05 (T.P.)	0.750 (T.P.)
L	17.0	0.669
M	$0.25^{+0.10}_{-0.05}$	$0.010^{+0.004}_{-0.003}$
N	0.17	0.007
R	0~15°	0~15°

P64C-70-750A,C-1

64 PIN PLASTIC QFP (□14)



P64GC-80-AB8-3

NOTE

Each lead centerline is located within 0.15 mm (0.006 inch) of its true position (T.P.) at maximum material condition.

ITEM	MILLIMETERS	INCHES
A	17.6±0.4	0.693±0.016
B	14.0±0.2	$0.551^{+0.009}_{-0.008}$
C	14.0±0.2	$0.551^{+0.009}_{-0.008}$
D	17.6±0.4	0.693±0.016
F	1.0	0.039
G	1.0	0.039
H	0.35±0.10	$0.014^{+0.004}_{-0.005}$
I	0.15	0.006
J	0.8 (T.P.)	0.031 (T.P.)
K	1.8±0.2	0.071±0.008
L	0.8±0.2	$0.031^{+0.009}_{-0.008}$
M	$0.15^{+0.10}_{-0.05}$	$0.006^{+0.004}_{-0.003}$
N	0.10	0.004
P	2.55	0.100
Q	0.1±0.1	0.004±0.004
S	2.85 MAX.	0.112 MAX.

APPENDIX A. DEVELOPMENT TOOLS

The following development tools are available for system development using the μPD780924 subseries.

Language Processing Software

RA78K/0 Notes 1, 2, 3, 4	78K/0 series common assembler package
CC78K/0 Notes 1, 2, 3, 4	78K/0 series common C compiler package
DF780964 Notes 1, 2, 3, 4, 8	μPD780924 subseries common device file
CC78K/0-L Notes 1, 2, 3, 4	78K/0 series common C compiler library source file

Flash Memory Writing Tools

Flashpro	Dedicated flash memory writer. The Flashpro is a product of Naitou Densei Machida Seisakusho Co., Ltd.
PA-FLASH64CW (temporary name) Note 8 PA-FLASH64GC (temporary name) Note 8	Adapter to write data to the flash memory

Debugging Tools

IE-780000-SL (temporary name) Note 8	75XL, 78K/0S, 78K/0, and 78K/IV series common in-circuit emulator
IE-78K0-SL-EM (temporary name) Note 8	78K/0 series common CPU core board
IE-780964-SL-EM1 (temporary name) Note 8	Probe board to emulate μPD780924 subseries products
EP-64CW-SL (temporary name) Note 8	Emulation probe for 64-pin plastic shrink DIP (CW type)
EP-64GC-SL (temporary name) Note 8	Emulation probe for 64-pin plastic QFP (GC-AB8 type)
EV-9200GC-64	Socket to be mounted on target system board created for the 64-pin plastic QFP (GC-AB8 type)
SM78K0 Notes 5, 6, 7	78K/0 series common system simulator
ID78K0 Notes 4, 5, 6, 7	IE-780000-SL integrated debugger
DF780964 Notes 4, 5, 6, 7, 8	Device file common to μPD780924 subseries

Real-Time OSs

RX78K/0 Notes 1, 2, 3, 4	78K/0 series real-time OS
MX78K0 Notes 1, 2, 3, 4	78K/0 series OS

Fuzzy Inference Development Support Systems

FE9000 Note 1/FE9200 Note 6	Fuzzy knowledge data creation tool
FT9080 Note 1/FT9085 Note 2	Translator
FI78K0 Notes 1, 2	Fuzzy inference module
FD78K0 Notes 1, 2	Fuzzy inference debugger

Notes 1. PC-9800 series (MS-DOS™) based
2. IBM PC/AT™ and compatibles (PC DOS™/IBM DOS™/MS-DOS) based
3. HP9000 series 300™ (HP-UX™) based
4. HP9000 series 700™ (HP-UX) based, SPARCstation™ (SunOS™) based, EWS4800 series (EWS-UX/V) based
5. PC-9800 series (MS-DOS + Windows™) based
6. IBM PC/AT and compatibles (PC DOS/IBM DOS/MS-DOS + Windows) based
7. NEWS™ (NEWS-OS™) based
8. Under development

Remark RA78K/0, CC78K/0, SM78K0, ID78K0, and RX78K/0 are used in combination with DF780964.

APPENDIX B. RELATED DOCUMENTS

Device Related Documents

Document Name	Document No.	
	English	Japanese
μPD780924 Subseries User's Manual	Planned	Planned
μPD780921, 780922, 780923, 780924 Preliminary Product Information	This manual	U11804J
μPD78F0924 Preliminary Product Information	Planned	Planned
μPD780924 Subseries Special Function Register Table	—	Planned
78K/0 Series User's Manual Instructions	IEU-1372	IEU-849
78K/0 Series Instruction Table	—	U10903J
78K/0 Series Instruction Set	—	U10904J

Development Tool Documents (User's Manuals)

Document Name		Document No.	
		English	Japanese
RA78K Series Assembler Package	Operation	EEU-1399	EEU-809
	Language	EEU-1404	EEU-815
RA78K Series Structured Assembler Preprocessor		EEU-1402	EEU-817
CC78K Series C Compiler	Operation	EEU-1280	EEU-656
	Language	EEU-1284	EEU-655
CC 78K0 C Compiler	Operation	U11517E	U11517J
	Language	U11518E	U11518J
CC78K/0 C Compiler Application Note	Programming Know-how	EEA-1208	EEA-618
CC78K Series Library Source File		—	EEU-777
IE-780000-SL		Planned	Planned
IE-78K0-SL-EM		Planned	Planned
IE-780924-SL-EM1		Planned	Planned
EP-64CW-SL		Planned	Planned
EP-64GC-SL		Planned	Planned
SM78K0 System Simulator Windows-based	Reference	U10181E	U10181J
SM78K Series System Simulator	External parts user open interface specification	U10092E	U10092J
ID78K0 Integrated Debugger EWS-based	Reference	U11151E	U11151J
ID78K0 Integrated Debugger PC-based	Reference	U11539E	U11539J
ID78K0 Integrated Debugger Windows-based	Guide	U11649E	U11649J

Caution The above related documents are subject to change without notice. Be sure to use the latest documents when starting design.

Embedded Software Documents (User's Manuals)

Document Name		Document No.	
		English	Japanese
78K/0 Series Real-Time OS	Basics	—	U11537J
	Installation	—	U11536J
	Technical	—	U11538J
78K/0 Series OS MX78K0	Basics	EEU-1532	EEU-5010
Fuzzy Knowledge Data Creation Tool		EEU-1438	EEU-829
78K/0, 78K/II, 87AD Series Fuzzy Inference Development Support System Translator		EEU-1444	EEU-862
78K/0 Series Fuzzy Inference Development Suport System Fuzzy Inference Module		EEU-1441	EEU-858
78K/0 Series Fuzzy Inference Development Support System Fuzzy Inference Debugger		EEU-1458	EEU-921

Other Documents

Document Name	Document No.	
	English	Japanese
IC Package Manual	C10943X	
Semiconductor Device Mounting Technology Manual	C10535E	C10535J
Quality Grades on NEC Semiconductor Devices	C11531E	C11531J
NEC Semiconductor Device Reliability/Quality Control System	C10983E	C10983J
Electrostatic Discharge (ESD) Test	IEI-1201	MEM-539
Guide to Quality Assurance for Semiconductor Devices	MEI-1202	MEI-603
Microcomputer Product Series Guide	—	MEI-604

Caution The above related documents are subject to change without notice. Be sure to use the latest documents when starting design.

[MEMO]

PAGE(S) INTENTIONALLY BLANK

[MEMO]

PAGE(S) INTENTIONALLY BLANK

[MEMO]

The following supersedes any statement which may be found elsewhere in this document purporting to address the subjects of quality, reliability or suitability of any devices listed in this document for applications other than as noted.

NOTES FOR CMOS DEVICES

① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note: Strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and quickly dissipate it once, when it has occurred. Environmental control must be adequate. When it is dry, humidifier should be used. It is recommended to avoid using insulators that easily build static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work bench and floor should be grounded. The operator should be grounded using wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with semiconductor devices on it.

② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note: No connection for CMOS device inputs can be cause of malfunction. If no connection is provided to the input pins, it is possible that an internal input level may be generated due to noise, etc., hence causing malfunction. CMOS device behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using a pull-up or pull-down circuitry. Each unused pin should be connected to V_{DD} or GND with a resistor, if it is considered to have a possibility of being an output pin. All handling related to the unused pins must be judged device by device and related specifications governing the devices.

③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note: Power-on does not necessarily define initial status of MOS device. Production process of MOS does not define the initial operation status of the device. Immediately after the power source is turned ON, the devices with reset function have not yet been initialized. Hence, power-on does not guarantee out-pin levels, I/O settings or contents of registers. Device is not initialized until the reset signal is received. Reset operation must be executed immediately after power-on for devices having reset function.

Regional Information

Some information contained in this document may vary from country to country. Before using any NEC product in your application, please contact the NEC office in your country to obtain a list of authorized representatives and distributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and other legal issues may also vary from country to country.

NEC Electronics Inc. (U.S.)
Mountain View, California
Tel: 800-366-9782
Fax: 800-729-9288

NEC Electronics (Germany) GmbH
Duesseldorf, Germany
Tel: 0211-65 03 02
Fax: 0211-65 03 490

NEC Electronics (UK) Ltd.
Milton Keynes, UK
Tel: 01908-691-133
Fax: 01908-670-290

NEC Electronics Italiana s.r.1.
Milano, Italy
Tel: 02-66 75 41
Fax: 02-66 75 42 99

NEC Electronics (Germany) GmbH
Benelux Office
Eindhoven, The Netherlands
Tel: 040-2445845
Fax: 040-2444580

NEC Electronics (France) S.A.
France
Tel: 01-30-67 58 00
Fax: 01-30-67 58 99

NEC Electronics (France) S.A.
Spain Office
Madrid, Spain
Tel: 01-504-2787
Fax: 01-504-2860

NEC Electronics (Germany) GmbH
Scandinavia Office
Taeby Sweden
Tel: 8-63 80 820
Fax: 8-63 80 388

NEC Electronics Hong Kong Ltd.
Hong Kong
Tel: 2886-9318
Fax: 2886-9022/9044

NEC Electronics Hong Kong Ltd.
Seoul Branch
Seoul, Korea
Tel: 02-528-0303
Fax: 02-528-4411

NEC Electronics Singapore Pte. Ltd.
United Square, Singapore 1130
Tel: 253-8311
Fax: 250-3583

NEC Electronics Taiwan Ltd.
Taipei, Taiwan
Tel: 02-719-2377
Fax: 02-719-5951

NEC do Brasil S.A.
Sao Paulo-SP, Brasil
Tel: 011-889-1680
Fax: 011-889-1689

J96. 3

The related documents in this publication may include preliminary versions. However, preliminary versions are not marked as such.

The export of this product from Japan is regulated by the Japanese government. To export this product may be prohibited without governmental license, the need for which must be judged by the customer. The export or re-export of this product from a country other than Japan may also be prohibited without a license from that country. Please call an NEC sales representative.

M4 96.5

USER'S MANUAL

NEC

μPD780924, 780964 SUBSERIES

8-BIT SINGLE-CHIP MICROCONTROLLER

(PRELIMINARY)

μPD780921
μPD780922
μPD780923
μPD780924
μPD78F0924

μPD780961
μPD780962
μPD780963
μPD780964
μPD78F0964

A

Document No. U12071EJ1V0UM00 (1st edition)
Date Published January 1997 N
Printed in Japan

NOTES FOR CMOS DEVICES

① PRECAUTION AGAINST ESD FOR SEMICONDUCTORS

Note:

Strong electric field, when exposed to a MOS device, can cause destruction of the gate oxide and ultimately degrade the device operation. Steps must be taken to stop generation of static electricity as much as possible, and quickly dissipate it once, when it has occurred. Environmental control must be adequate. When it is dry, humidifier should be used. It is recommended to avoid using insulators that easily build static electricity. Semiconductor devices must be stored and transported in an anti-static container, static shielding bag or conductive material. All test and measurement tools including work bench and floor should be grounded. The operator should be grounded using wrist strap. Semiconductor devices must not be touched with bare hands. Similar precautions need to be taken for PW boards with semiconductor devices on it.

② HANDLING OF UNUSED INPUT PINS FOR CMOS

Note:

No connection for CMOS device inputs can be cause of malfunction. If no connection is provided to the input pins, it is possible that an internal input level may be generated due to noise, etc., hence causing malfunction. CMOS devices behave differently than Bipolar or NMOS devices. Input levels of CMOS devices must be fixed high or low by using a pull-up or pull-down circuitry. Each unused pin should be connected to V_{DD} or GND with a resistor, if it is considered to have a possibility of being an output pin. All handling related to the unused pins must be judged device by device and related specifications governing the devices.

③ STATUS BEFORE INITIALIZATION OF MOS DEVICES

Note:

Power-on does not necessarily define initial status of MOS device. Production process of MOS does not define the initial operation status of the device. Immediately after the power source is turned ON, the devices with reset function have not yet been initialized. Hence, power-on does not guarantee out-pin levels, I/O settings or contents of registers. Device is not initialized until the reset signal is received. Reset operation must be executed immediately after power-on for devices having reset function.

For literature, call toll-free 7 a.m. to 6 p.m. Pacific time: **1-800-366-9782**

or FAX your request to: **1-800-729-9288**

6427525 0099442 73T

The export of these products from Japan is regulated by the Japanese government. The export of some or all of these products may be prohibited without governmental license. To export or re-export some or all of these products from a country other than Japan may also be prohibited without a license from that country. Please call an NEC sales representative.

License not needed	: μPD78F0924, 78F0964
The customer must judge the need for license	: μPD780921, 780922, 780923, 780924 μPD780961, 780962, 780963, 780964

The application circuits and their parameters are for reference only and are not intended for use in actual design-ins.

The information contained in this document is being issued in advance of the production cycle for the device. The parameters for the device may change before final production or NEC Corporation, at its own discretion, may withdraw the device prior to its production.

No part of this document may be copied or reproduced in any form or by any means without the prior written consent of NEC Corporation. NEC Corporation assumes no responsibility for any errors which may appear in this document.

NEC Corporation does not assume any liability for infringement of patents, copyrights or other intellectual property rights of third parties by or arising from use of a device described herein or any other liability arising from use of such device. No license, either express, implied or otherwise, is granted under any patents, copyrights or other intellectual property rights of NEC Corporation or others.

While NEC Corporation has been making continuous effort to enhance the reliability of its semiconductor devices, the possibility of defects cannot be eliminated entirely. To minimize risks of damage or injury to persons or property arising from a defect in an NEC semiconductor device, customers must incorporate sufficient safety measures in its design, such as redundancy, fire-containment, and anti-failure features.

NEC devices are classified into the following three quality grades:

"Standard", "Special", and "Specific". The Specific quality grade applies only to devices developed based on a customer designated "quality assurance program" for a specific application. The recommended applications of a device depend on its quality grade, as indicated below. Customers must check the quality grade of each device before using it in a particular application.

- 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
- Special: 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: Aircrafts, aerospace equipment, submersible repeaters, nuclear reactor control systems, life support systems or medical equipment for life support, etc.

The quality grade of NEC devices is "Standard" unless otherwise specified in NEC's Data Sheets or Data Books. If customers intend to use NEC devices for applications other than those specified for Standard quality grade, they should contact an NEC sales representative in advance.

Anti-radioactive design is not implemented in this product.

Regional Information

Some information contained in this document may vary from country to country. Before using any NEC product in your application, please contact the NEC office in your country to obtain a list of authorized representatives and distributors. They will verify:

- Device availability
- Ordering information
- Product release schedule
- Availability of related technical literature
- Development environment specifications (for example, specifications for third-party tools and components, host computers, power plugs, AC supply voltages, and so forth)
- Network requirements

In addition, trademarks, registered trademarks, export restrictions, and other legal issues may also vary from country to country.

NEC Electronics Inc. (U.S.)
Santa Clara, California
Tel: 800-366-9782
Fax: 800-729-9288

NEC Electronics (Germany) GmbH
Duesseldorf, Germany
Tel: 0211-65 03 02
Fax: 0211-65 03 490

NEC Electronics (UK) Ltd.
Milton Keynes, UK
Tel: 01908-691-133
Fax: 01908-670-290

NEC Electronics Italiana s.r.1.
Milano, Italy
Tel: 02-66 75 41
Fax: 02-66 75 42 99

NEC Electronics (Germany) GmbH
Benelux Office
Eindhoven, The Netherlands
Tel: 040-2445845
Fax: 040-2444580

NEC Electronics (France) S.A.
Velizy-Villacoublay, France
Tel: 01-30-67 58 00
Fax: 01-30-67 58 99

NEC Electronics (France) S.A.
Spain Office
Madrid, Spain
Tel: 01-504-2787
Fax: 01-504-2860

NEC Electronics (Germany) GmbH
Scandinavia Office
Taeby, Sweden
Tel: 08-63 80 820
Fax: 08-63 80 388

NEC Electronics Hong Kong Ltd.
Hong Kong
Tel: 2886-9318
Fax: 2886-9022/9044

NEC Electronics Hong Kong Ltd.
Seoul Branch
Seoul, Korea
Tel: 02-528-0303
Fax: 02-528-4411

NEC Electronics Singapore Pte. Ltd.
United Square, Singapore 1130
Tel: 253-8311
Fax: 250-3583

NEC Electronics Taiwan Ltd.
Taipei, Taiwan
Tel: 02-719-2377
Fax: 02-719-5951

NEC do Brasil S.A.
Sao Paulo-SP, Brasil
Tel: 011-889-1680
Fax: 011-889-1689

E

J96. 8

INTRODUCTION

Readers

This manual is intended for user engineers who understand the functions of the μPD780924 and μPD780964 subseries and wish to design and develop its application systems and programs.
Target products are as follows:

- μPD780924 subseries: μPD780921, 780922, 780923, 780924, 78F0924
- μPD780964 subseries: μPD780961, 780962, 780963, 780964, 78F0964

Purpose

This manual is designed to deepen your understanding of the following functions using the following organization.

Organization

Two manuals are available for the μPD780924 and μPD780964 subseries: this manual and Instruction Manual (common to the 78K/0 series).

μPD780924, 780964 subseries User's Manual	78K/0 series User's Manual Instruction
• Pin functions • Internal block functions • Interrupt • Other internal peripheral functions	• CPU function • Instruction set • Instruction description

How to Read This Manual

It is assumed that the readers of this manual have general knowledge on electric engineering, logic circuits, and microcontrollers.

☐ To understand the overall functions of the μPD780924 and μPD780964 subseries
→ Read this manual in the order of the **TABLE OF CONTENTS.**

☐ How to read register formats
→ The name of a bit whose number is encircled is reserved for the RA78K/0 and is defined for the CC78K/0 by the header file sfrbit.h.

☐ To learn the detailed functions of a register whose register name is known
→ Refer to **APPENDIX C REGISTER INDEX.**

☐ To learn the details of the instruction functions of the μPD780924 and μPD780964 subseries
→ Refer to **78K/0 Series User's Manual - Instruction (IEU-1372)** separately available.

F

Legend

Data significance	:	Left: higher digit, right: lower digit
Active low	:	$\overline{XXX}$ (top bar over pin or signal name)
Note	:	Footnote
Caution	:	Important information
Remark	:	Supplement
Numerical representation	:	Binary ... XXXX or XXXXB
		Decimal ... XXXX
		Hexadecimal ... XXXXH

Related Documents

The related documents indicated in this publication may include preliminary versions. However, preliminary versions are not marked as such.

- **Device-related documents**

Document Name	Document Number	
	Japanese	English
μPD780921, 780922, 780923, 780924 Preliminary Product Information	U11804J	Planned to be published
μPD780961, 780962, 780963, 780964 Preliminary Product Information	U11879J	Planned to be published
μPD78F0924 Preliminary Product Information	U11930J	Planned to be published
μPD78F0964 Preliminary Product Information	U11956J	Planned to be published
μPD780924, 780964 Subseries User's Manual	U12071J	This manual
78K/0 Series User's Manual - Instruction	IEU-849	IEU-1372
78K/0 Series Instruction List	U10903J	—
78K/0 Series Instruction Set	U10904J	—
μPD780924, 780964 Subseries Special Function Register List	Planned to be published	—

Caution The contents of the above related documents are subject to change without notice. Be sure to use the latest version of a document for designing.

• **Development tool-related documents (user's manual)**

Document Name		Document Number	
		Japanese	English
RA78K Series Assembler Package	Operation	EEU-809	EEU-1399
	Language	EEU-815	EEU-1404
RA78K Series Structured Assembler Preprocessor		EEU-817	EEU-1402
CC78K Series C Compiler	Operation	EEU-656	EEU-1280
	Language	EEU-655	EEU-1284
CC78K0 C Compiler	Operation	U11517J	—
	Language	U11518J	—
CC78K/0 C Compiler Application Note	Programming Know-how	EEA-618	EEA-1208
CC78K Series Library Source File		EEU-777	—
IE-780000-SL		Planned to be published	Planned to be published
IE-78K0-SL-EM		Planned to be published	Planned to be published
IE-780964-SL-EM1		Planned to be published	Planned to be published
EP-64CW-SL		Planned to be published	Planned to be published
EP-64GC-SL		Planned to be published	Planned to be published
SM78K0 System Simulator Windows Base	Reference	U10181J	U10181E
SM78K Series System Simulator	External Parts User Open Interface Specifications	U10092J	U10092E
ID78K0 Integrated Debugger EWS Base	Reference	U11151J	—
ID78K0 Integrated Debugger PC Base	Reference	U11539J	—
ID78K0 Integrated Debugger Windows Base	Guide	U11649J	—

• **Embedded software-related documents (user's manual)**

Document Name		Document Number	
		Japanese	English
78K/0 Series Real-time OS	Fundamental	U11537J	—
	Installation	U11536J	—
	Technical	U11538J	—
78K/0 Series OS MX78K0	Fundamental	EEU-5010	—
Fuzzy Knowledge Data Creation Tool		EEU-829	EEU-1438
78K/0, 78K/II, 87AD Series Fuzzy Inference Development Support System - Translator		EEU-862	EEU-1444
78K/0 Series Fuzzy Inference Development Support System - Fuzzy Inference Module		EEU-858	EEU-1441
78K/0 Series Fuzzy Inference Development Support System - Fuzzy Inference Debugger		EEU-921	EEU-1458

Caution The contents of the above related documents are subject to change without notice. Be sure to use the latest version of a document for designing.

H

- **Other related documents**

Document Name	Document Number	
	Japanese	English
IC Package Manual	C10943X	
Semiconductor Device Mounting Technology Manual	C10535J	C10535E
Quality Grades on NEC Semiconductor Devices	C11531J	C11531E
NEC Semiconductor Device Reliability/Quality Control System	C10983J	C10983E
Electrostatic Discharge (ESD) Test	MEM-539	—
Guide to Quality Assurance for Semiconductor Devices	C11893J	MEI-1202
Microcontroller Related Product Guide - Other Manufactures	U11416J	—

Caution The contents of the above related documents are subject to change without notice. Be sure to use the latest version of a document for designing.

CONTENTS

LIST OF FIGURES (1/4)

LIST OF FIGURES (2/4)

LIST OF FIGURES (3/4)

LIST OF FIGURES (4/4)

LIST OF TABLES (1/2)

LIST OF TABLES (2/2)

[MEMO]

PAGE(S) INTENTIONALLY BLANK

CHAPTER 1 GENERAL

1.1 Features

- On-chip ROM and RAM

Item / Part Number	Program Memory (ROM)	Data Memory
		Internal High-Speed RAM
μPD780921, 780961	8 Kbytes	512 bytes
μPD780922, 780962	16 Kbytes	
μPD780923, 780963	24 Kbytes	1024 bytes
μPD780924, 780964	32 Kbytes	
μPD78F0924, 78F0964	32 Kbytes[Note 1] (flash memory)	1024 bytes[Note 2]

Notes 1. 8, 16, 24, or 32 Kbytes is selectable by using memory size select register (IMS).
2. 512 or 1024 bytes is selectable by using IMS.

- Lower EMI (Electro Magnetic Interfarence) noise than existing μPD78014 and 78018F subseries
- External memory extension space: 48 Kbytes
- Minimum instruction execution time: 0.24 μs (fx = 8.38-MHz operation)
- Instruction set suitable for system control
  - Bit processing in entire address space
  - Multiplication/division instructions
- I/O port: 47 lines
- A/D converter
  - 8-bit resolution × 8 channels (μPD780924 subseries)
  - 10-bit resolution × 8 channels (μPD780964 subseries)
- Serial interface: UART × 2 channels
- Timer: 5 channels
  - 10-bit inverter control timer : 1 channel
  - 8-bit timer/event counter : 3 channels
  - Watchdog timer : 1 channel
- Vectored interrupt: 18 lines
- Supply voltage: V_{DD} = 2.7 to 5.5 V

1.2 Application Field

Inverter air conditioners, washing machine, etc.

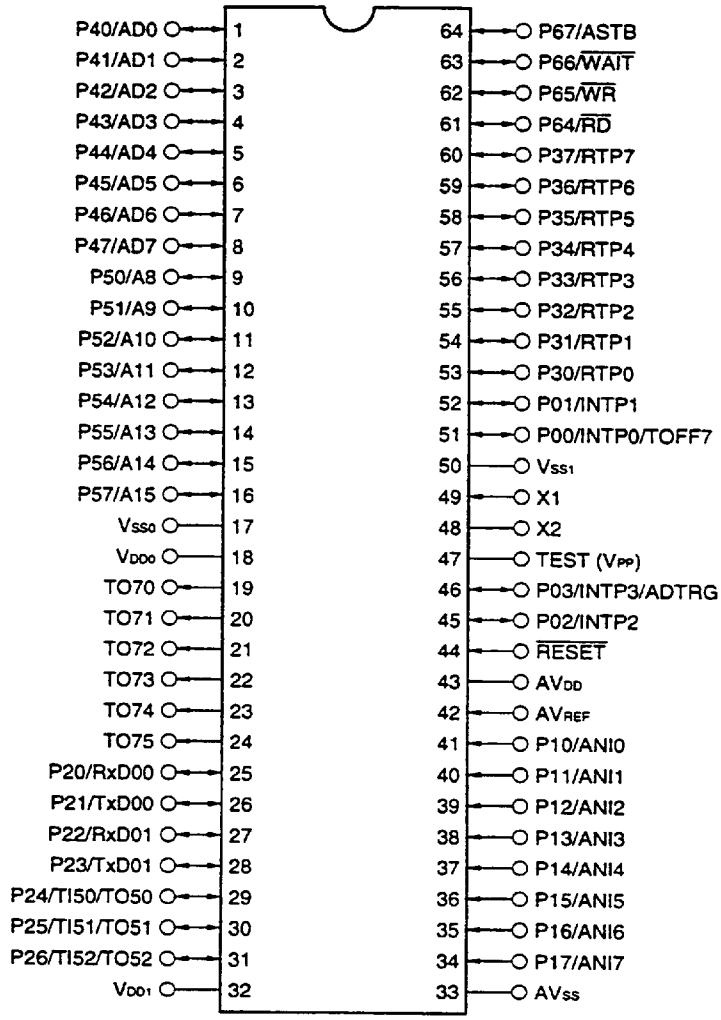
1.3 Ordering Information

Part Number	Package	Internal ROM
μPD780921CW	64-pin plastic shrink DIP (750 mil)	Mask ROM
μPD780921GC-xxx-AB8	64-pin plastic QFP (14 × 14 mm)	Mask ROM
μPD780922CW	64-pin plastic shrink DIP (750 mil)	Mask ROM
μPD780922GC-xxx-AB8	64-pin plastic QFP (14 × 14 mm)	Mask ROM
μPD780923CW	64-pin plastic shrink DIP (750 mil)	Mask ROM
μPD780923GC-xxx-AB8	64-pin plastic QFP (14 × 14 mm)	Mask ROM
μPD780924CW	64-pin plastic shrink DIP (750 mil)	Mask ROM
μPD780924GC-xxx-AB8	64-pin plastic QFP (14 × 14 mm)	Mask ROM
μPD780961CW	64-pin plastic shrink DIP (750 mil)	Mask ROM
μPD780961GC-xxx-AB8	64-pin plastic QFP (14 × 14 mm)	Mask ROM
μPD780962CW	64-pin plastic shrink DIP (750 mil)	Mask ROM
μPD780962GC-xxx-AB8	64-pin plastic QFP (14 × 14 mm)	Mask ROM
μPD780963CW	64-pin plastic shrink DIP (750 mil)	Mask ROM
μPD780963GC-xxx-AB8	64-pin plastic QFP (14 × 14 mm)	Mask ROM
μPD780964CW	64-pin plastic shrink DIP (750 mil)	Mask ROM
μPD780964GC-xxx-AB8	64-pin plastic QFP (14 × 14 mm)	Mask ROM
μPD78F0924CW	64-pin plastic shrink DIP (750 mil)	Flash memory
μPD78F0924GC-AB8	64-pin plastic QFP (14 × 14 mm)	Flash memory
μPD78F0964CW	64-pin plastic shrink DIP (750 mil)	Flash memory
μPD78F0964GC-AB8	64-pin plastic QFP (14 × 14 mm)	Flash memory

Remark xxx indicates a ROM code suffix.

1.4 Pin Configuration (Top View)

- **64-pin plastic shrink DIP (750 mil)**
  μPD780921CW-xxx, 780922CW-xxx, 780923CW-xxx, 780924CW-xxx, 78F0924CW
  μPD780961CW-xxx, 780962CW-xxx, 780963CW-xxx, 780964CW-xxx, 78F0964CW



Cautions 1. Connect the TEST pin to V_{SS0} directly.

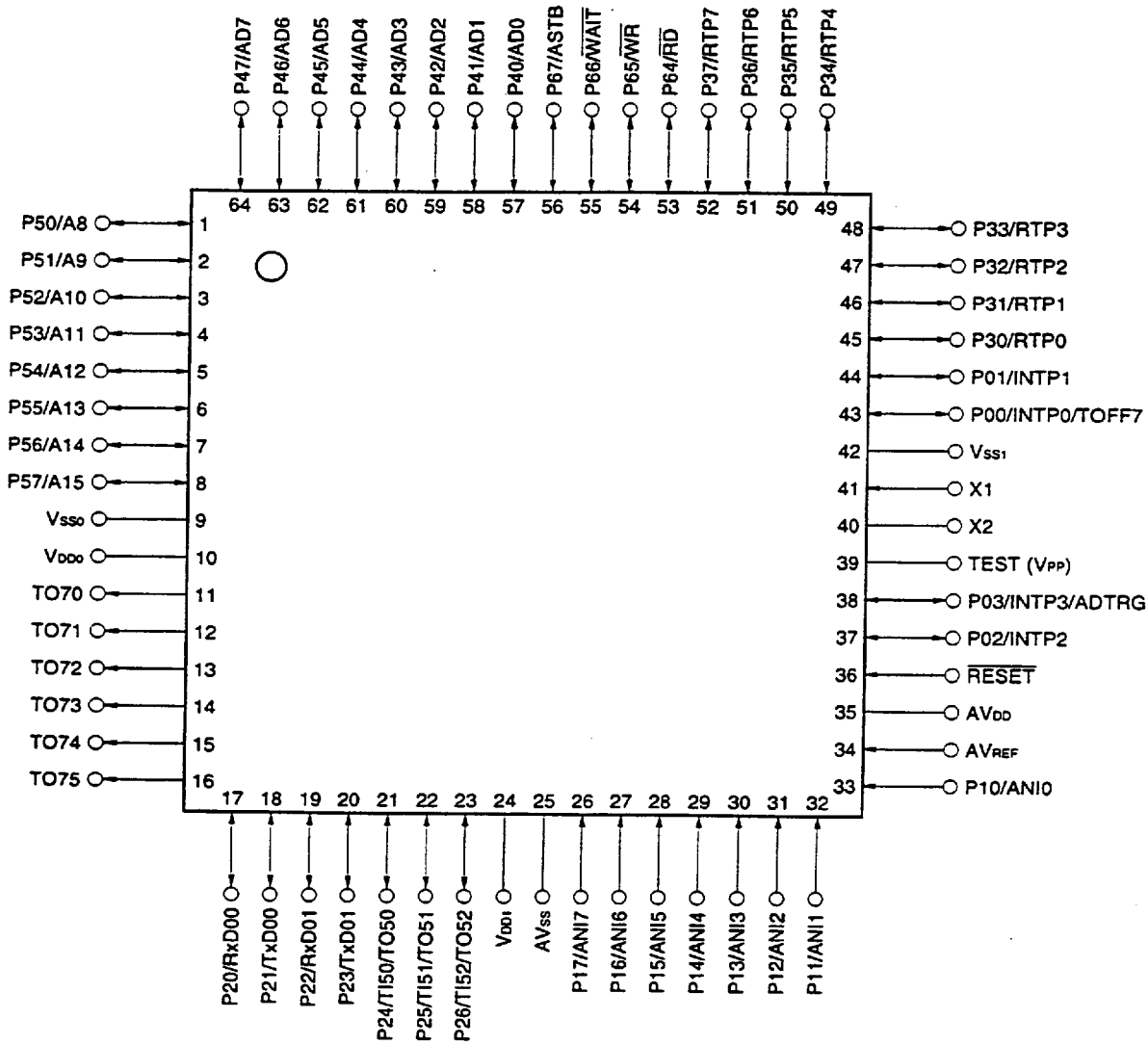
2. Connect the AV_{DD} pin to V_{DD0}.

3. Connect the AVss pin to V_{SS0}.

Remarks 1. ( ): μPD78F0924, 78F0964

2. When the μPD780924 and 780964 subseries are used in application fields that require reduction of the noise generated from inside the microcontroller, the implementation of noise reduction measures, such as supplying voltage to V_{DD0} and V_{DD1} individually and connecting V_{SS0} and V_{SS1} to different ground lines, is recommended.

- **64-pin plastic QFP (14 × 14 mm)**
  μPD780921GC-×××-AB8, 780922GC-×××-AB8, 780923GC-×××-AB8, 780924GC-×××-AB8
  μPD78F0924GC-AB8
  μPD780961GC-×××-AB8, 780962GC-×××-AB8, 780963GC-×××-AB8, 780964GC-×××-AB8
  μPD78F0964GC-AB8



Cautions 1. Connect the TEST pin to V_{SS0} directly.

2. Connect the AV_{DD} pin to V_{DD0}.

3. Connect the AVss pin to V_{SS0}.

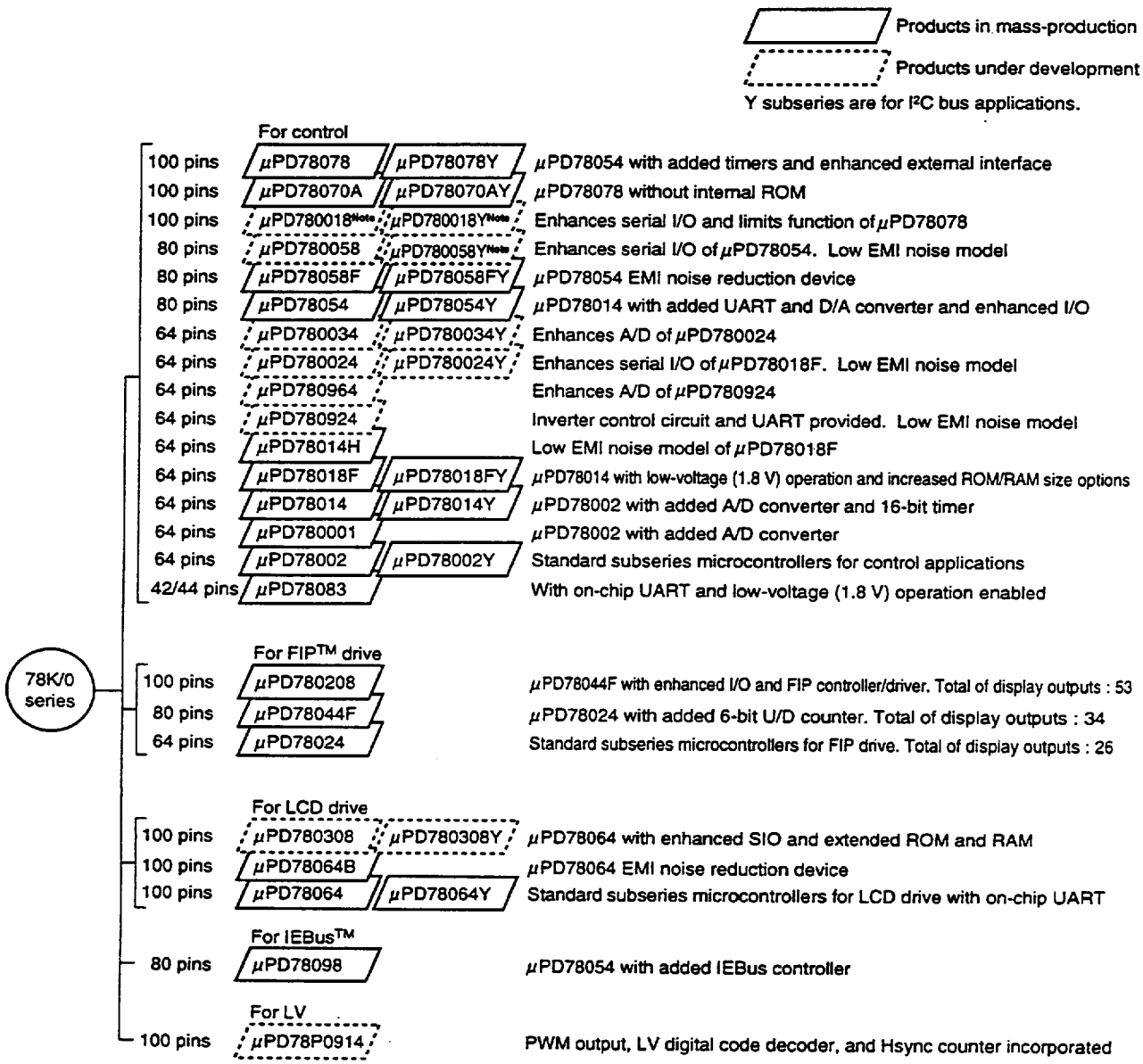
Remarks 1. ( ): μPD78F0924, 78F0964

2. When the μPD780924 and 780964 subseries are used in application fields that require reduction of the noise generated from inside the microcontroller, the implementation of noise reduction measures, such as supplying voltage to V_{DD0} and V_{DD1} individually and connecting V_{SS0} and V_{SS1} to different ground lines, is recommended.

A8 to A15	:	Address Bus
AD0 to AD7	:	Address/Data Bus
ADTRG	:	AD Trigger Input
ANI0 to ANI7	:	Analog Input
ASTB	:	Address Strobe
AV_{DD}	:	Analog Power Supply
AV_{REF}	:	Analog Reference Voltage
AVss	:	Analog Ground
INTP0 to INTP3	:	Interrupt From Peripherals
P00 to P03	:	Port 0
P10 to P17	:	Port 1
P20 to P26	:	Port 2
P30 to P37	:	Port 3
P40 to P47	:	Port 4
P50 to P57	:	Port 5
P64 to P67	:	Port 6
$\overline{RD}$	:	Read Strobe
$\overline{RESET}$	:	Reset
RTP0 to RTP7	:	Real-time Port
RxD00, RxD01	:	Receive Data
TEST	:	Test
TI50 to TI52	:	Timer Input
TO50 to TO52, TO70 to TO75	:	Timer Output
TOFF7	:	Timer Output Off
TxD00, TxD01	:	Transmit Data
V_{DD0}, V_{DD1}	:	Power Supply
V_{PP}	:	Programming Power Supply
V_{SS0}, V_{SS1}	:	Ground
$\overline{WAIT}$	:	Wait
$\overline{WR}$	:	Write Strobe
X1, X2	:	Crystal

1.5 Product Development of 78K/0 Series

The products in the 78K/0 series are listed below. The names enclosed in boxes are subseries names.



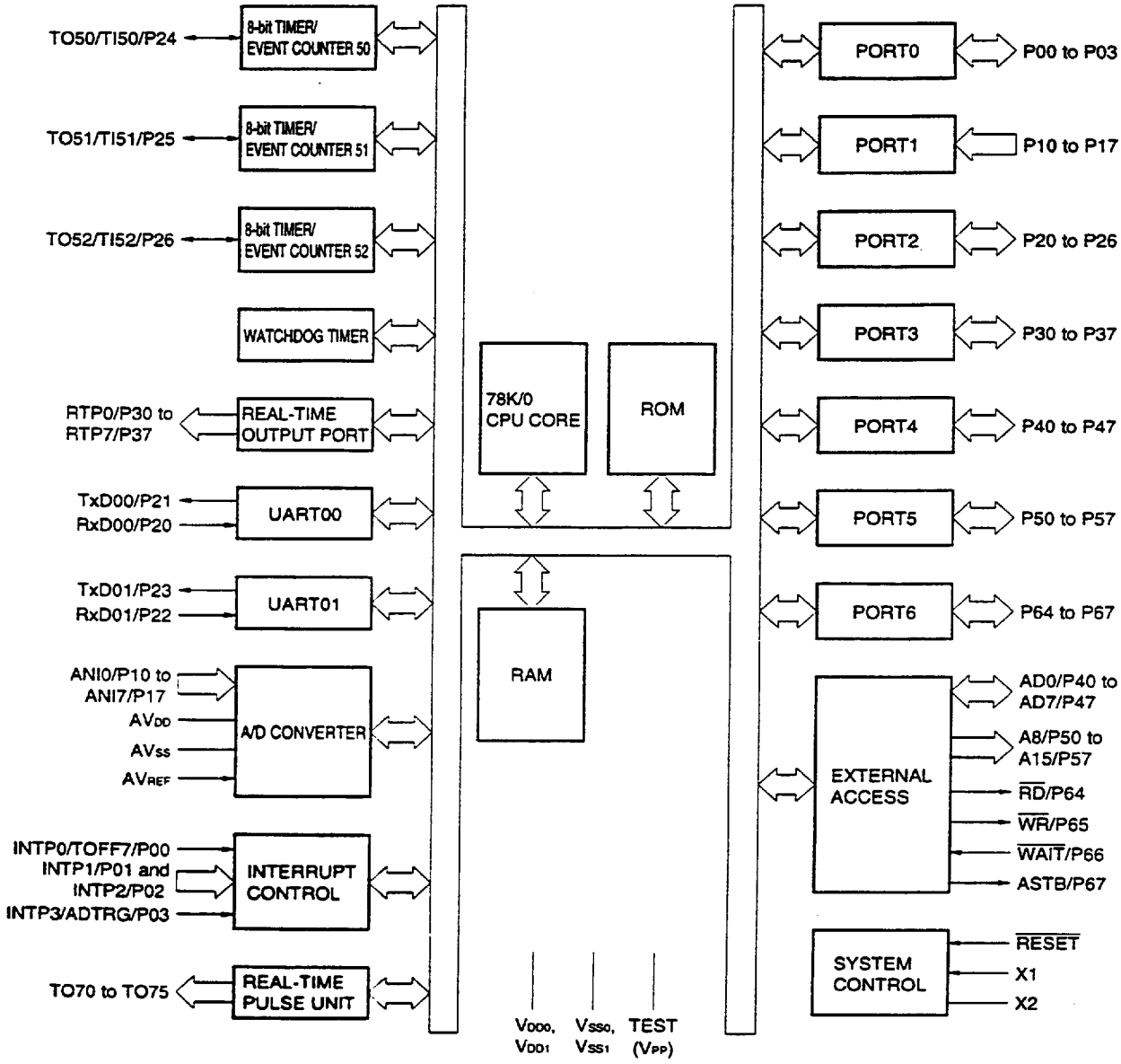
Note Under planning

The following shows the major differences among subseries products.

Subseries Name	Function	ROM Capacity	Timer 8-bit	Timer 16-bit	Timer Watch	Timer WDT	8-bit A/D	10-bit A/D	8-bit D/A	Serial Interface	I/O	Vdd MIN. Value	External Expansion
Control	μPD78078	32 K to 60 K	4ch	1ch	1ch	1ch	8ch	–	2ch	3ch (UART: 1ch)	88	1.8 V	√
	μPD78070A	–									61	2.7 V	
	μPD780018	48 K to 60 K							–	2ch (time-division 3-wire: 1ch)	88		
	μPD780058	24 K to 60 K	2ch						2ch	3ch (time-division UART: 1ch)	68	1.8 V	
	μPD78058F	48 K to 60 K								3ch (UART: 1ch)	69	2.7 V	
	μPD78054	16 K to 60 K										2.0 V	
	μPD780034	8 K to 32 K					–	8ch	–	3ch (UART: 1ch, time-division 3-wire: 1ch)	51	1.8 V	
	μPD780024						8ch	–					
	μPD780964		3ch	Note	–		–	8ch		2ch (UART: 2ch)	47	2.7 V	
	μPD780924						8ch	–					
	μPD78014H		2ch	1ch	1ch					2ch	53	1.8 V	
	μPD78018F	8 K to 60 K										2.7 V	
	μPD78014	8 K to 32 K											
	μPD780001	8 K		–	–					1ch	39		–
	μPD78002	8 K to 16 K			1ch		–				53		√
	μPD78083				–		8ch			1ch (UART: 1ch)	33	1.8 V	–
FIP drive	μPD780208	32 K to 60 K	2ch	1ch	1ch	1ch	8ch	–	–	2ch	74	2.7 V	–
	μPD78044F	16 K to 40 K									68		
	μPD78024	24 K to 32 K									54		
LCD drive	μPD780308	48 K to 60 K	2ch	1ch	1ch	1ch	8ch	–	–	3ch (time-division UART: 1ch)	57	1.8 V	–
	μPD78064B	32 K								2ch (UART: 1ch)		2.0 V	
	μPD78064	16 K to 32 K											
IEBus	μPD78098	32 K to 60 K	2ch	1ch	1ch	1ch	8ch	–	2ch	3ch (UART: 1ch)	69	2.7 V	√
LV	μPD78P0914	32 K	6ch	–	–	1ch	8ch	–	–	2ch	54	4.5 V	√

Note 10-bit timer: 1 channel

1.6 Block Diagram



Remarks 1. The internal ROM and RAM capacities differ depending on the product.

2. ( ): μPD78F0924, 78F0964

1.7 Functional Outline

Item \ Product Name		μPD780921 μPD780961	μPD780922 μPD780962	μPD780923 μPD780963	μPD780924 μPD780964	μPD78F0924 μPD78F0964
Internal memory	ROM capacity	8 Kbytes	16 Kbytes	24 Kbytes	32 Kbytes	32 Kbytes
	ROM structure	Mask ROM				Flash memory
	High-speed RAM capacity	512 bytes		1024 bytes		
Memory space		64 Kbytes				
General-purpose register		8 bits x 32 registers (8 bits x 8 registers x 4 banks)				
Instruction cycle		On-chip instruction execution time variable function 0.24 μs/0.48 μs/0.96 μs/1.9 μs/3.8 μs (@ 8.38-MHz operation with system clock)				
Instruction set		• 16-bit operation • Multiply/divide (8 bits x 8 bits, 16 bits ÷ 8 bits) • Bit manipulate (set, reset, test, Boolean operation) • BCD adjust, etc.				
I/O ports		Total : 47 • CMOS inputs : 8 • CMOS I/Os : 39				
Real-time output ports		8 bits x 1 or 4 bits x 2				
A/D converter		• 8-bit resolution x 8 channels (μPD780921, 780922, 780923, 780924, 78F0924) • 10-bit resolution x 8 channels (μPD780961, 780962, 780963, 780964, 78F0964) • Power supply voltage: AV_{DD} = 2.7 to 5.5 V				
Serial interface		UART x 2 channels				
Timer		• 8-bit timer/event counter : 3 channels • 10-bit inverter control timer : 1 channel • Watchdog timer : 1 channel				
Timer output		9 (8-bit PWM output x 3, and inverter control output x 6)				
Vectored interrupt source	Maskable	Internal: 12, external: 4				
	Non-maskable	Internal: 1				
	Software	1				
Power supply voltage		V_{DD} = 2.7 to 5.5 V				
Operating ambient temperature		T_A = −40 to +85 °C				
Package		• 64-pin plastic shrink DIP (750 mil) • 64-pin plastic QFP (14 x 14 mm)				

[MEMO]

PAGE(S) INTENTIONALLY BLANK

2.1 List of Pin Functions

(1) Port pins

Pin Name	I/O	Function	After Reset	Alternate Function
P00	I/O	Port 0 4-bit I/O port Input/output can be specified bit-wise. When used as an input port, an on-chip pull-up resistor can be connected by software.	Input	INTP0/TOFF7
P01				INTP1
P02				INTP2
P03				INTP3/ADTRG
P10 to P17	Input	Port 1 8-bit input only port	Input	ANI0 to ANI7
P20	I/O	Port 2 7-bit I/O port Input/output can be specified bit-wise. When used as an input port, an on-chip pull-up resistor can be connected by software.	Input	RxD00
P21				TxD00
P22				RxD01
P23				TxD01
P24				TI50/TO50
P25				TI51/TO51
P26				TI52/TO52
P30 to P37	I/O	Port 3 8-bit I/O port Input/output can be specified bit-wise. When used as an input port, an on-chip pull-up resistor can be connected by software.	Input	RTP0 to RTP7
P40 to P47	I/O	Port 4 8-bit I/O port Input/output can be specified bit-wise. When used as an input port, an on-chip pull-up resistor can be connected by software.	Input	AD0 to AD7
P50 to P57	I/O	Port 5 8-bit I/O port Input/output can be specified bit-wise. LEDs can be driven directly. When used as an input port, an on-chip pull-up resistor can be connected by software..	Input	A8 to A15
P64	I/O	Port 6 4-bit I/O port Input/output can be specified bit-wise. When used as an input port, an on-chip pull-up resistor can be connected by software.	Input	$\overline{RD}$
P65				$\overline{WR}$
P66				$\overline{WAIT}$
P67				ASTB

(2) Pins other than port pins

Pin Name	I/O	Function	After Reset	Alternate Function
INTP0	Input	External interrupt input that can specify the effective edge (rising edge, falling edge, or both rising and falling edges) can be specified.	Input	P00/TOFF7
INTP1			Input	P01
INTP2			Input	P02
INTP3			Input	P03/ADTRG
TI50	Input	External count clock input to timer (TM50)	Input	P24/TO50
TI51		External count clock input to timer (TM51)	Input	P25/TO51
TI52		External count clock input to timer (TM52)	Input	P26/TO52
TO50	Output	Timer (TM50) output	Input	P24/TI50
TO51		Timer (TM51) output	Input	P25/TI51
TO52		Timer (TM52) output	Input	P26/TI52
RTP0 to RTP7	Output	Real-time output port that outputs pulses in synchronization with trigger signals outputs from the real-time pulse unit.	Input	P30 to P37
TxD00	Output	Asynchronous serial interface serial data output	Input	P21
TxD01			Input	P23
RxD00	Input	Asynchronous serial interface serial data input	Input	P20
RxD01			Input	P22
ANI0 to ANI7	Input	A/D converter analog input	Input	P10 to P17
ADTRG	Input	External trigger signal input to the A/D converter	Input	P03/INTP3
TO70 to TO75	Output	Timer output for the 3-phase PWM inverter control	Hi-Z	–
TOFF7	Input	Timer output (TO70 to TO75) stop interrupt input	Input	P00/INTP0
AD0 to AD7	I/O	Lower address/data bus when memory is expanded externally	Input	P40 to P47
A8 to A15	Output	Upper address bus when memory is expanded externally	Input	P50 to P57
$\overline{RD}$	Output	Strobe signal output for external memory read operation	Input	P64
$\overline{WR}$		Strobe signal output for external memory write operation	Input	P65
$\overline{WAIT}$	Input	Wait insertion when accessing external memory	Input	P66
ASTB	Output	Strobe output that externally latches address information output to ports 4 and 5 to access external memory	Input	P67
AV_{REF}	Input	A/D converter reference voltage input	–	–
AV_{DD}	–	A/D converter analog power supply. Connect to V_{DD0}	–	–
AV_{SS}	–	A/D converter ground potential. Connect to V_{SS0}	–	–
$\overline{RESET}$	Input	System reset input	–	–
X1	Input	Crystal connection for system clock oscillation	–	–
X2	–		–	–
V_{DD0}	–	Positive power supply for ports	–	–
V_{SS0}	–	Ground potential for ports	–	–
V_{DD1}	–	Positive power supply except for ports	–	–
V_{SS1}	–	Ground potential except for ports	–	–
TEST	–	Test mode set pin. Connect to V_{SS0} directly	–	–
V_{PP}[Note]	–	High-voltage application for program write/verify. Directly connect this pin to V_{SS0} in normal mode.	–	–

Note μPD78F0924, 78F0964 only

2.2 Description of Pin Functions

2.2.1 P00 to P03 (Port0)

These pins constitute a 4-bit I/O port, port 0. In addition, these pins are also used to input external interrupt signals, a timer output stop interrupt signal and an external trigger signal of the A/D converter.

Port 0 can be specified in the following operation modes in 1-bit units.

(1) Port mode

In this mode, P00 to P03 function as 4-bit input/output port pins.

Port 0 can be set in the input or output port mode in 1-bit units by using port mode register 0. When these pins are used as an input port, an internal pull-up resistor can be used if so specified by the pull-up resistor option register.

(2) Control mode

In this mode, P00 to P03 are used to input external interrupts, a timer output stop interrupt, and an external trigger signal of the A/D converter.

(a) INTP0 to INTP3

These pins are external interrupt input pins for which valid edge can be specified (rising edge, falling edge, and both rising and falling edges). INTP2 also functions as an external trigger signal input pin of the real-time output port when a valid edge is input.

(b) TOFF7

Timer output (TO70 to TO75) stop interrupt input pin

(c) ADTRG

External trigger signal input pin of the A/D converter

2.2.2 P10 to P17 (Port1)

These pins constitute an 8-bit input port, port1. In addition to the general-purpose port function, these pins also serve as the analog input pins of the A/D converter.

(1) Port mode

In this mode, P10 to P17 function as 8-bit input port pins.

(2) Control mode

In this mode, P10 to P17 function as the analog input pins (ANI0 through ANI7) of the A/D converter.

2.2.3 P20 to P26 (Port2)

These pins constitute an 7-bit I/O port, port 2. In addition, these pins are also used to input/output the data of the serial interface, input/output timer.

Port 2 can be specified in the following operation modes in 1-bit units.

(1) Port mode

In the port mode, P20 and P27 function as an 7-bit I/O port. Port 2 can be set in the input or output mode in 1-bit units by using the port mode register 2. When the port is used as an input port, an internal pull-up resistor can be used if so specified by the pull-up resistor option register.

(2) Control mode

In this mode, P20 to P27 input/output the data of the serial interface, input/output timer.

(a) RxD00, RxD01, TxD00, TxD01

These are the serial data I/O pins of the serial interface.

(b) TI50 to TI52

External count clock input pins of the 8-bit timer/event counter.

(c) TO50 to TO52

Timer output pins.

2.2.4 P30 to P37 (Port3)

These pins constitute an 8-bit I/O port, port 3. In addition, they also functions as a real-time output port. Port 3 can be set in the following operation modes in 1-bit units.

(1) Port mode

In this mode, port 3 functions as an 8-bit I/O port which can be set in the input or output mode in 1-bit units by using the port mode register 3. When used as an input port, an internal pull-up resistor can be used if so specified by the pull-up resistor option register.

(2) Control mode

In this mode, the pins of port 3 can be used as a real-time output port (RTP0 through RTP7) that outputs data in synchronization with a trigger.

2.2.5 P40 to P47 (Port4)

These pins form an 8-bit I/O port, port 4. In addition, they also form an address/data bus.
This port can be set in the following operation modes in 1-bit units.

(1) Port mode

In this mode, P40 to P47 function as an 8-bit I/O port which can be set in the input or output mode in 1-bit units by using the port mode register 4. When used as an input port, an internal pull-up resistor can be used if so specified by the pull-up resistor option register.

(2) Control mode

In this mode, P40 to P47 function as the lower address/data bus pins (AD0 through AD7) in the external memory extension mode. The pins used as address/data bus pins are automatically disconnected from the internal pull-up resistor.

2.2.6 P50 to P57 (Port5)

These pins form an 8-bit I/O port, port 5, which also serves as an address bus.
These pins can directly drive LEDs.
Port 5 can be set in the following operation modes in 1-bit units.

(1) Port mode

In this mode, P50 to P57 constitute an 8-bit I/O port which can be set in the input or output mode in 1-bit units by using the port mode register 5. When used as an input port, an internal pull-up resistor can be used if so specified by the pull-up resistor option register.

(2) Control mode

In this mode, P50 to P57 function as the higher address bus pins (A8 to A15) in the external memory extension mode. The pins used as address bus pins are automatically disconnected from the internal pull-up resistor.

2.2.7 P64 to P67 (Port6)

These pins constitute an 4-bit I/O port, port 6, which can be also used to output control signals in the external memory extension mode.
Port 6 can be set in the following operation modes in 1-bit units.

(1) Port mode

In this mode, P60 to P67 constitute an 4-bit I/O port, which can be set in the input or output mode in 1-bit units by using the port mode register 6.
When using P64 to P67 as input port pins, an internal pull-up resistor can be used if so specified by the pull-up resistor option register.

(2) Control mode

In this mode, P60 to P67 functions as control signal output pins ($\overline{RD}$, $\overline{WR}$, $\overline{WAIT}$, and ASTB) in the external memory extension mode. The pins used as control signal output pins are automatically disconnected from the internal pull-up resistor.

Caution If the external wait state is not used in the external memory extension mode, P66 can be used as an I/O port pin.

2.2.8 TO70 to TO75

These are the timer output pins for the 3-phase PWM inverter control.

2.2.9 AV_{REF}

This pin inputs a reference voltage to the A/D converter.
Connect this pin to V_{SS0} when the A/D converter is not used.

2.2.10 AV_{DD}

This is the analog power supply pin of the A/D converter.
Keep this pin at the same voltage as the V_{DD0} pin even when the A/D converter is not used.

2.2.11 AV_{SS}

This is the ground pin of the A/D converter.
Keep this pin at the same voltage as the V_{SS0} pin even when the A/D converter is not used.

2.2.12 $\overline{\text{RESET}}$

This pin inputs an active-low system reset signal.

2.2.13 X1 and X2

These pins are used to connect a crystal resonator for system clock oscillation.

To supply an external clock, input the clock to X1 and input the inverted signal to X2.

2.2.14 V_{DD0}, V_{DD1}

V_{DD0} is the positive power supply pin for ports. V_{DD1} is the positive power supply pin for blocks other than ports.

2.2.15 V_{SS0}, V_{SS1}

V_{SS0} is the ground pin for ports. V_{SS1} is the ground pin for blocks other than ports.

2.2.16 V_{PP} (μPD78F0924, 78F0964 only)

A high voltage should be applied to this pin when the program is written or verified.

Directly connect this pin to V_{SS0} in the normal operation mode.

2.2.17 TEST (mask ROM model only)

Pin used or IC testing. Must be directly connected to V_{SS0}.

2.3 Recommended Connection of Unused Pins

The recommended connections of unused pins are shown in Table 2-1.

Table 2-1. Recommended Connection of Unused Pins

Pin Name	I/O	Recommended Connection of Unused Pins
P00/INTP0/TOFF7	I/O	Individually connect to V_{SS0} via a resistor.
P01/INTP1		
P02/INTP2		
P03/INTP3/ADTRG		
P10/ANI0 to P17/ANI7	Input	Individually connect to V_{DD0} or V_{SS0} via a resistor.
P20/RxD00	I/O	
P21/TxD00		
P22/RxD01		
P23/TxD01		
P24/TI50/TO50		
P25/TI51/TO51		
P26/TI52/TO52		
P30/RTP0 to P37/RTP7		
P40/AD0 to P47/AD7		
P50/A8 to P57/A15		
P64/$\overline{RD}$		
P65/$\overline{WR}$		
P66/$\overline{WAIT}$		
P67/ASTB		
TO70 to TO75	Output	Open
AV_{DD}	–	Connect to V_{DD0}.
AV_{REF}		Connect to V_{SS0}.
AV_{SS}		
TEST (Mask ROM Version)		Connect to V_{SS0} directly.
V_{PP} (Flash memory version)		

3.1 Memory Space

Each model in the μPD780924, 780964 subseries can access a memory space of 64 Kbytes. Figures 3-1 to 3-5 show memory maps of the respective models.

Caution The initial value of the memory size select register (IMS) is fixed (to CFH) for all the models in the μPD780924 and 780964 subseries, regardless of the capacity of the internal memory. Therefore, set the value specified for each model to this register.

μPD780921, 780961 : 42H
μPD780922, 780962 : 44H
μPD780923, 780963 : C6H
μPD780924, 780964 : C8H
μPD78F0924, 78F0964 : Value corresponding to those of mask ROM models

Figure 3-1. Memory Map (μPD780921, 780961)

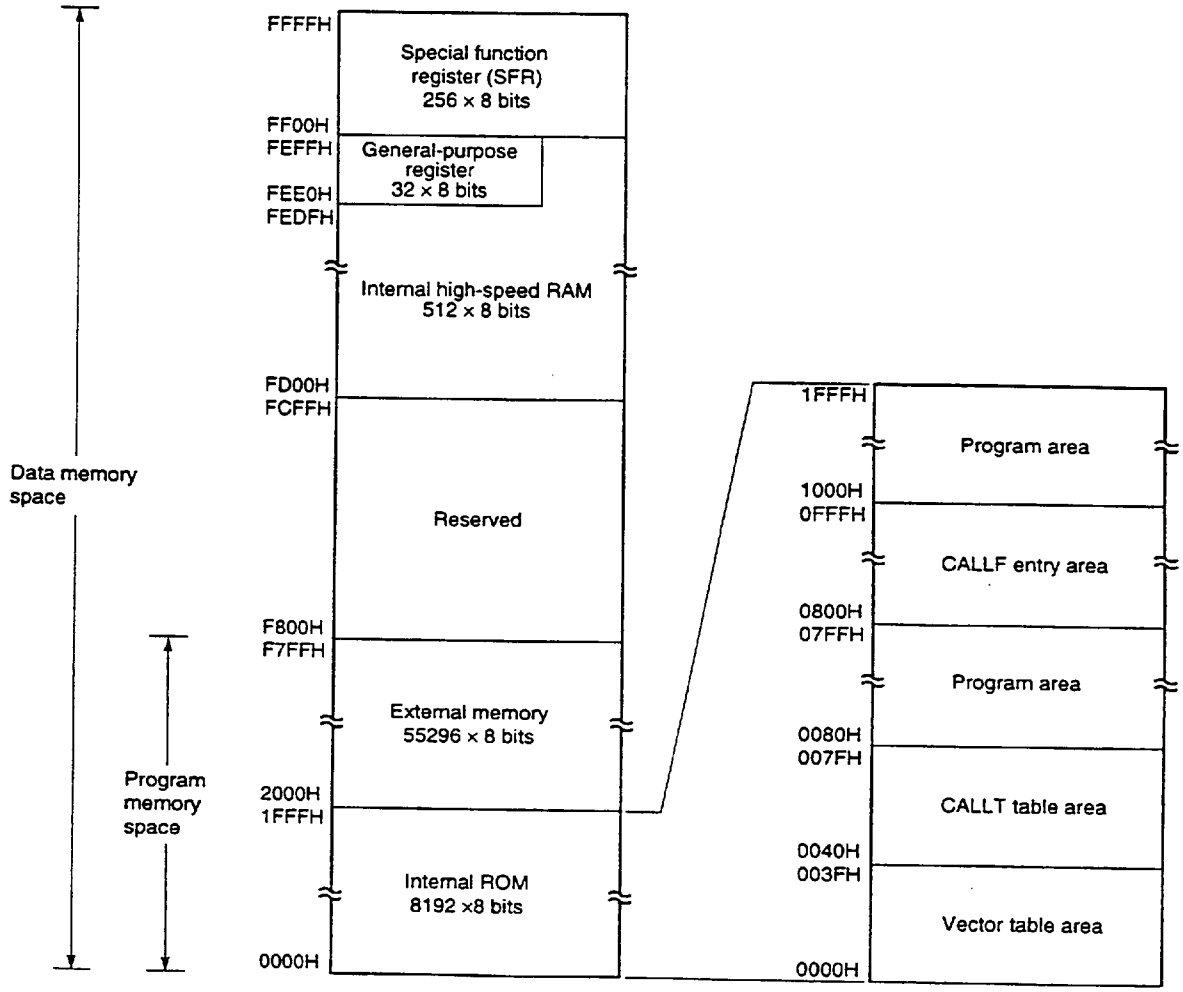


Figure 3-2. Memory Map (μPD780922, 780962)

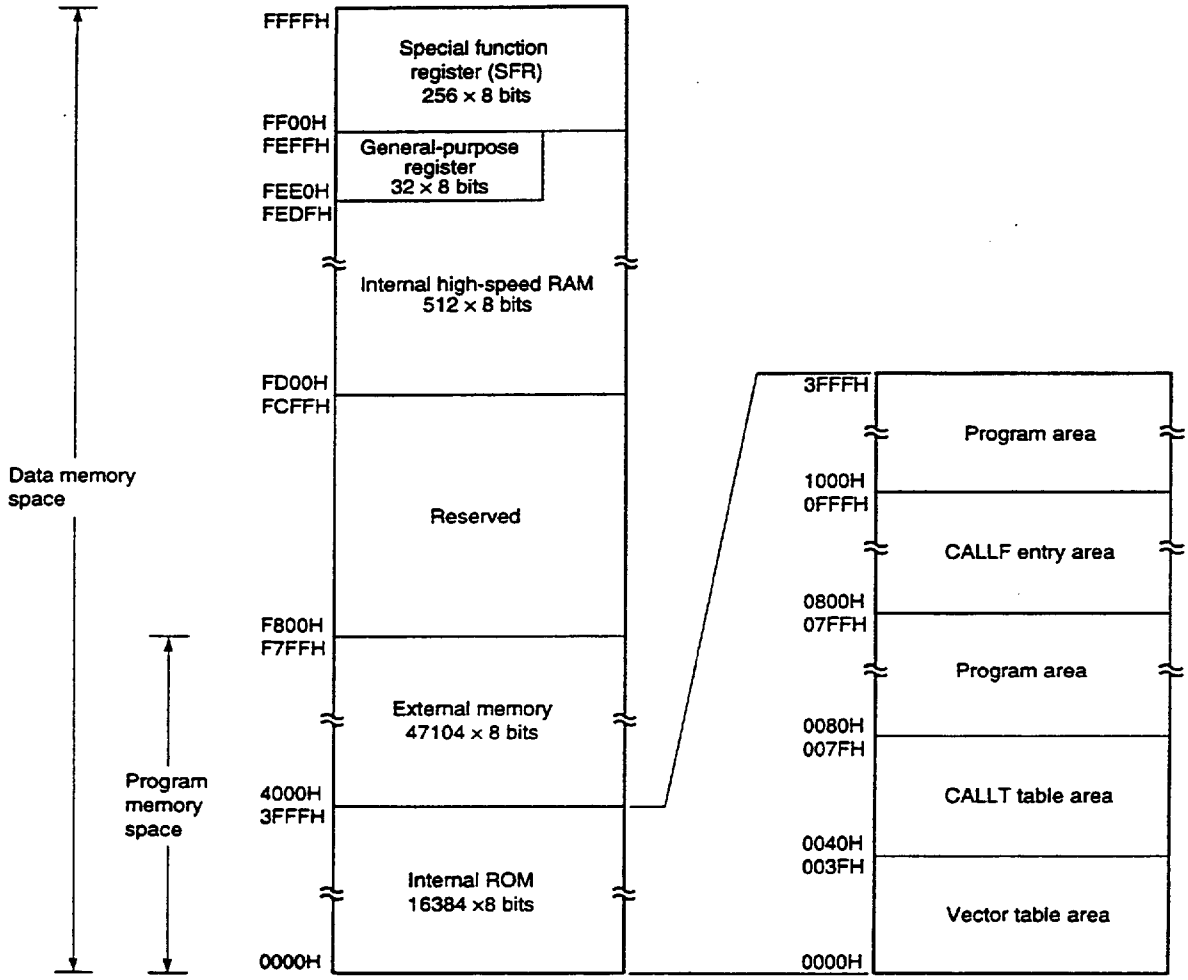


Figure 3-3. Memory Map (μPD780923, 780963)

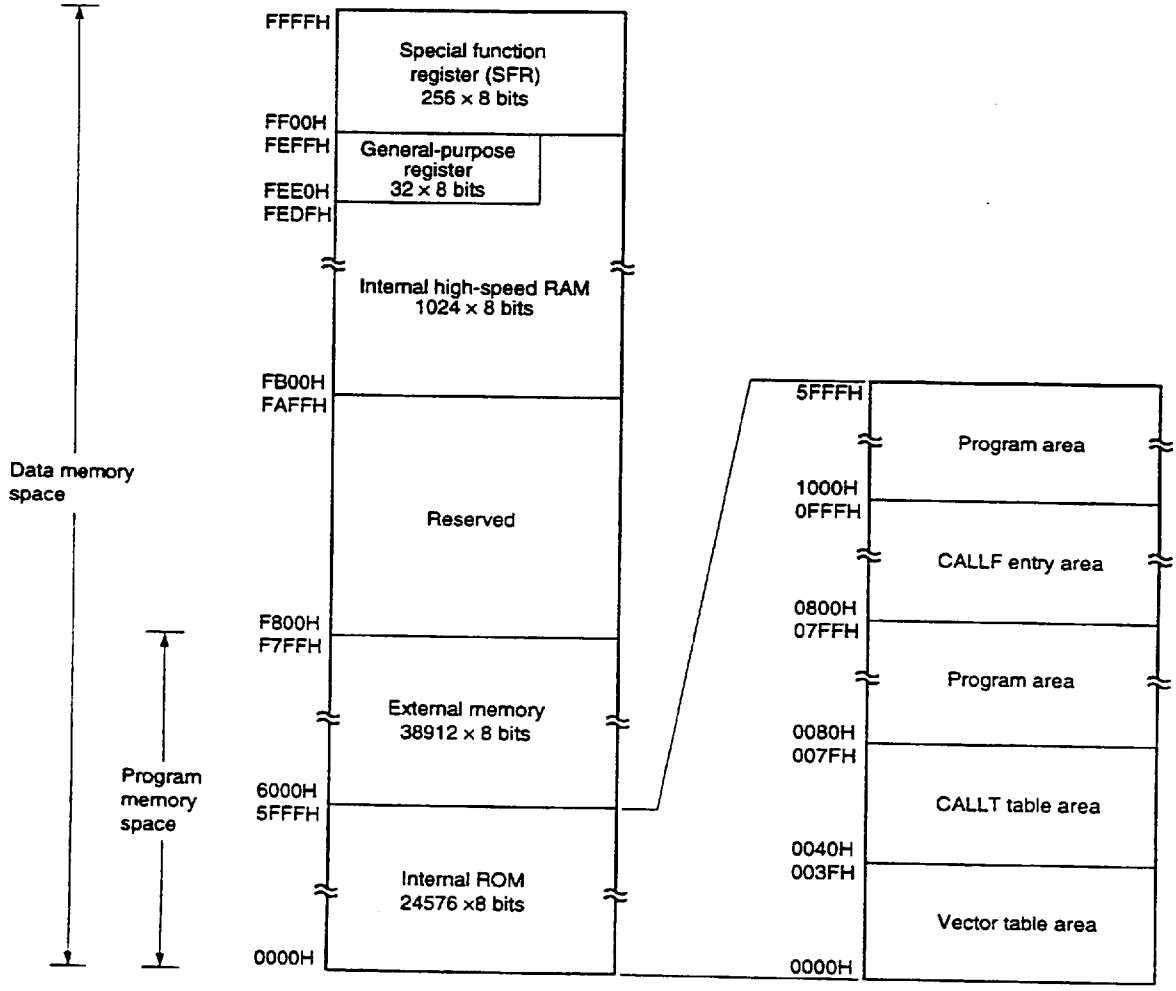


Figure 3-4. Memory Map (μPD780924, 780964)

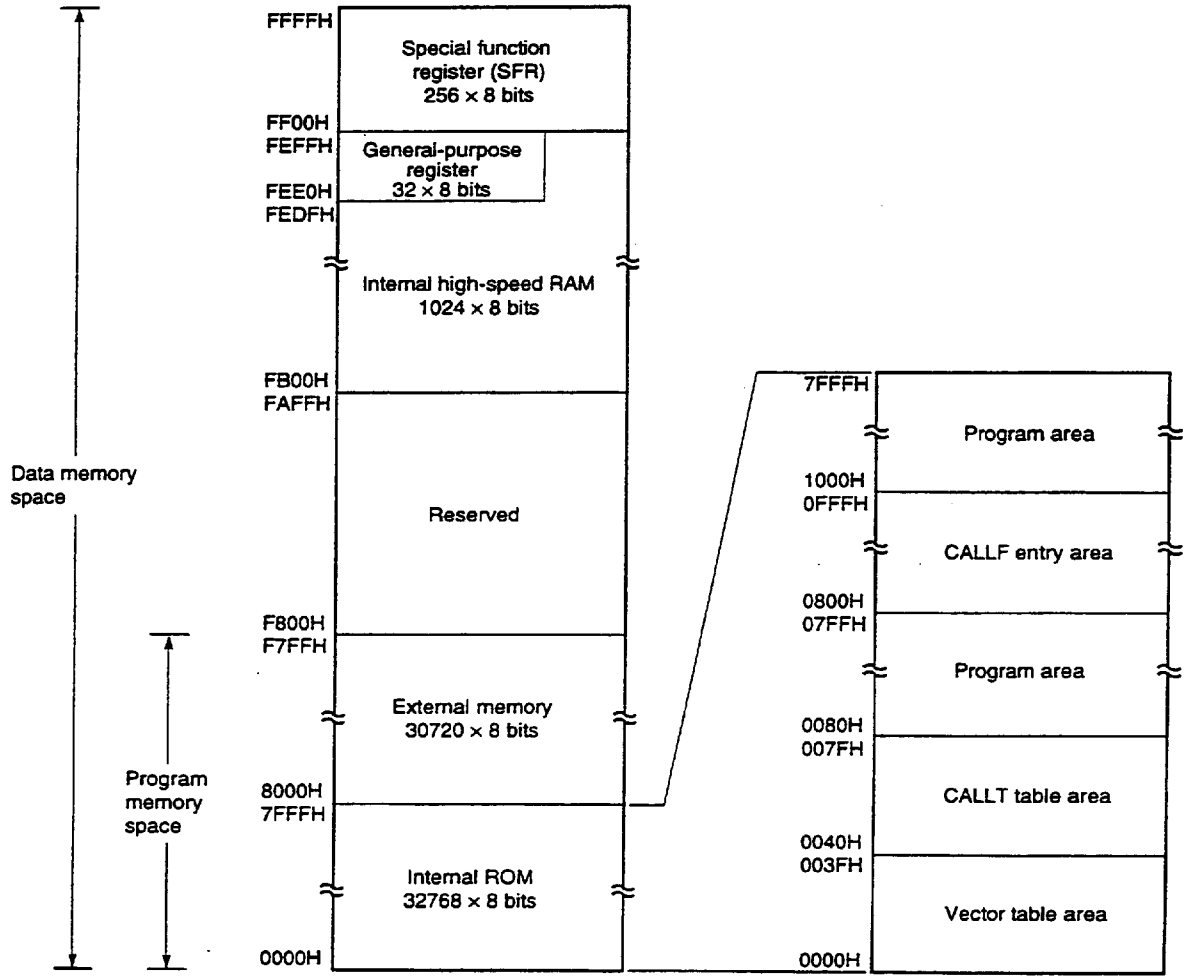
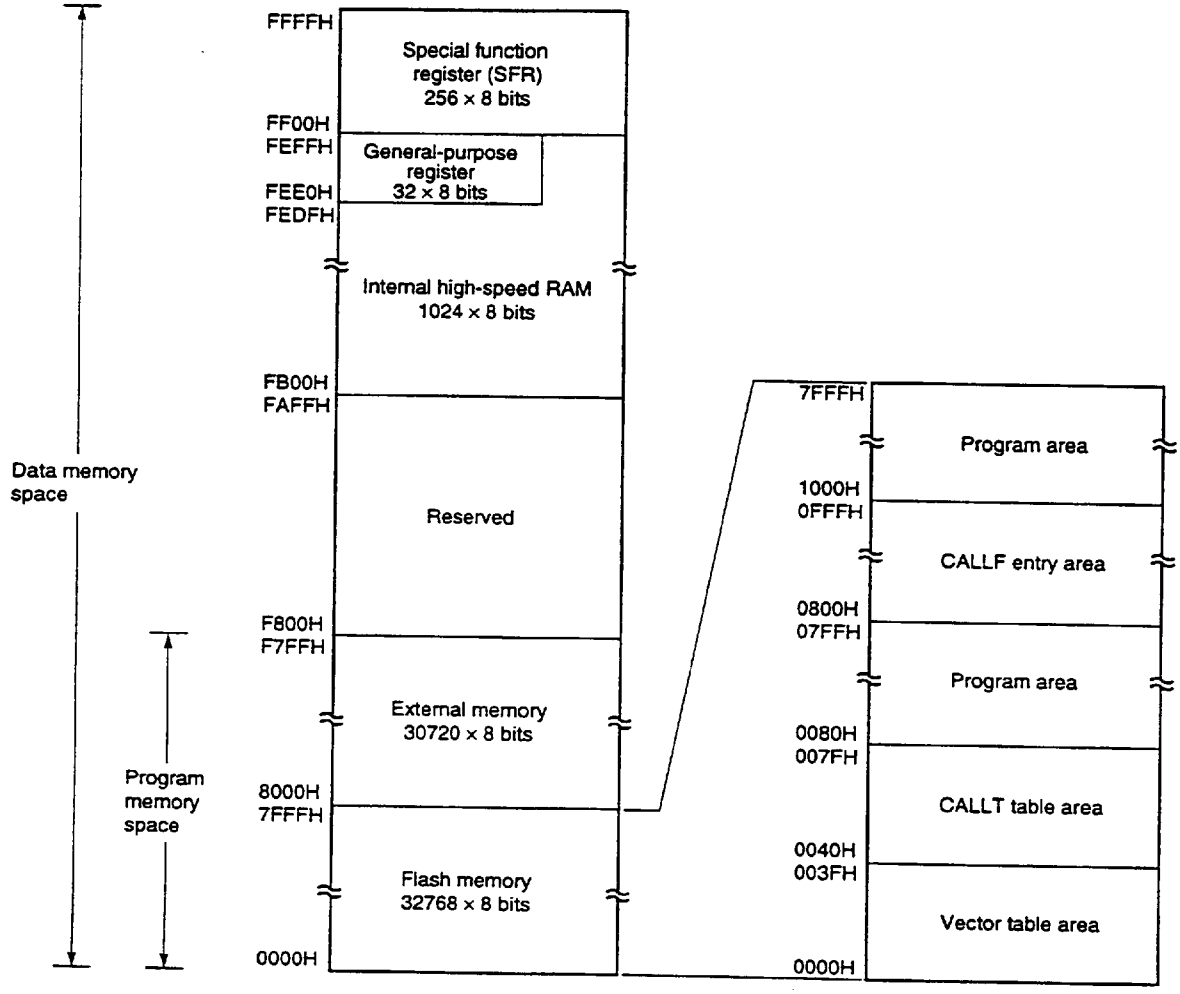


Figure 3-5. Memory Map (μPD78F0924, 78F0964)



3.1.1 Internal program memory space

The internal program memory space stores programs and table data. This space is usually addressed by the program counter (PC).

Each model in the μPD780924, 780964 subseries is provided with the following internal ROM (or flash memory):

Table 3-1. Internal ROM Capacity

Part Number	Structure	Capacity
μPD780921, 780961	Mask ROM	8192 × 8 bits (0000H to 1FFFH)
μPD780922, 780962		16384 × 8 bits (0000H to 3FFFH)
μPD780923, 780963		24576 × 8 bits (0000H to 5FFFH)
μPD780924, 780964		32768 × 8 bits (0000H to 7FFFH)
μPD78F0924, 78F0964	flash memory	32768 × 8 bits (0000H to 7FFFH)

The following areas are allocated to the internal program memory space:

(1) Vector table area

A 64-byte area of addresses 0000H to 003FH is reserved as a vector table area. This area stores program start addresses to which execution branches when the $\overline{\text{RESET}}$ signal is input or when an interrupt request is generated. Of a 16-bit program start address, the lower 8 bits are stored in an even address, and the higher 8 bits are stored in an odd address.

Table 3-2. Vector Table

Vector Table Address	Interrupt Request	Vector Table Address	Interrupt Request
0000H	$\overline{\text{RESET}}$ input	0014H	INTST0
0004H	INTWDT	0016H	INTSER1
0006H	INTP0	0018H	INTSR1
0008H	INTP1	001AH	INTST1
000AH	INTP2	001CH	INTTM50
000CH	INTP3	001EH	INTTM51
000EH	INTTM7	0020H	INTTM52
0010H	INTSER0	0022H	INTAD0
0012H	INTSR0	003EH	BRK instruction

(2) CALLT instruction table area

In a 64-byte area of addresses 0040H to 007FH, the subroutine entry address of a 1-byte call instruction (CALLT) can be stored.

(3) CALLF instruction entry area

From an area of addresses 0800H to 0FFFH, a subroutine can be directly called by using a 2-byte call instruction (CALLF).

3.1.2 Internal data memory space

The μPD780924 and 780964 subseries are provided with the following internal high-speed RAM.

Table 3-3. Internal High-Speed RAM Capacity

Part Number	Internal High-Speed RAM
μPD780921, 780961	512 x 8 bits (FD00H to FEFFH)
μPD780922, 780962	
μPD780923, 780963	1024 x 8 bits (FB00H to FEFFH)
μPD780924, 780964	
μPD78F0924, 78F0964	

A 32-byte area of addresses FEE0H to FEFFH is assigned four banks of general registers. Each bank consists of eight 8-bit registers.

The internal high-speed RAM can also be used as a stack memory.

3.1.3 Special function register (SFR) area

Special function registers (SFRs) of on-chip peripheral hardware are allocated to an area of FF00H to FFFFH (refer to **Table 3-6**).

Caution Do not access an address to which no SFR is allocated.

3.1.4 External memory space

This is an external memory space that can be accessed by using the memory extension mode register (MM). This space can store programs and table data, and can be assigned peripheral devices.

3.2 Processor Registers

The μPD780924, 780964 subseries are provided with the following processor registers:

3.2.1 Control registers

Each of these registers has a dedicated function such as to control the program sequence, status, and stack memory. The control registers include the program counter, program status word, and stack pointer.

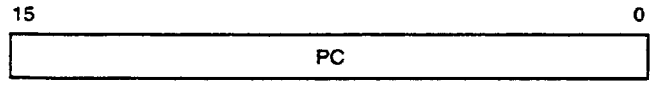
(1) Program counter (PC)

The program counter is a 16-bit register that holds an address of the program to be executed next.

The contents of this register are automatically incremented according to the number of bytes of an instruction to be fetched when a normal operation is performed. When a branch instruction is executed, immediate data or the contents of a register is set to the program counter.

When the $\overline{\text{RESET}}$ signal is input, the value of the reset vector table at addresses 0000H and 0001H is set to the program counter.

Figure 3-6. Program Counter Configuration



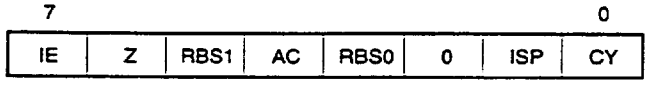
(2) Program status word (PSW)

The program status word is an 8-bit register consisting of flags that are set or reset as a result of instruction execution.

The contents of the program status word are automatically pushed to the stack when an interrupt request is generated or when the PUSH PSW instruction is executed, and are automatically popped from the stack when the RETB, RETI, or POP PSW instruction is executed.

The contents of the program status word are set to 02H when the $\overline{\text{RESET}}$ signal is input.

Figure 3-7. Program Status Word Configuration



(a) Interrupt enable flag (IE)

This flag controls acknowledgement of an interrupt request by the CPU.

When this flag is reset to 0 (DI status), only the non-maskable interrupt can be acknowledged, and all the other interrupts are disabled.

When this flag is set to 1 (EI status), interrupts are enabled, and acknowledging interrupt requests is controlled by the in-service priority flag (ISP), interrupt mask flag corresponding to each interrupt source, and priority flag.

The IE flag is reset to 0 when the DI instruction is executed or when an interrupt acknowledged, and is set to 1 when the EI instruction is executed.

(b) Zero flag (Z)

This flag is set to 1 when the result of an operation performed is 0; otherwise, it is reset to 0.

(c) Register bank select flags (RBS0 and RBS1)

These 2-bit flags select one of the four register banks.

Information of 2 bits that indicate the register bank selected by execution of the "SEL RBn" instruction is stored in these flags.

(d) Auxiliary carry flag (AC)

This flag is set to 1 when a carry occurs from bit 3 or a borrow to bit 3 occurs as a result of an operation performed; otherwise, it is reset to 0.

(e) In-service priority flag (ISP)

This flag controls the priority of maskable vectored interrupts that can be acknowledged.

When this flag is 0, acknowledgement of the vectored interrupt with the lower priority specified by the priority specification flag register (PR0L, PR0H) is disabled.

When it is 1, any interrupt is acknowledged regardless of its priority. Actually, acknowledging interrupts is controlled by the interrupt enable flag (IE).

(f) Carry flag (CY)

This flag records an overflow or underflow that occurs as the result of executing an add or subtract instruction. It also records the value shifted out when a rotate instruction is executed. In addition, it also functions as a bit accumulator when a bit operation instruction is executed.

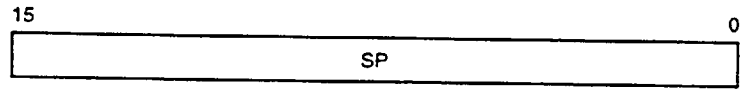
(3) Stack pointer (SP)

This is a 16-bit register that holds the first address of the stack area in the memory. As the stack area, only the internal high-speed RAM area can be specified. The internal high-speed RAM area of each product is as follows:

Table 3-4. Internal High-Speed RAM Area

Part Number	Internal High-Speed RAM Area
μPD780921, 780961, 780922, 780962	FD00H to FEFFH
μPD780923, 780963, 780924, 780964 μPD78F0924, 78F0964	FB00H to FEFFH

Figure 3-8. Stack Pointer Configuration



The contents of the stack pointer are decremented when data is written (saved) to the stack memory, and incremented when data are read (restored) from the stack memory.

The data saved/restored as a result of each stack operation are as shown in Figures 3-9 and 3-10.

Caution The contents of the SP become undefined when the $\overline{\text{RESET}}$ signal is input. Be sure to initialize the SP before executing an instruction.

Figure 3-9. Data Saved to Stack Memory

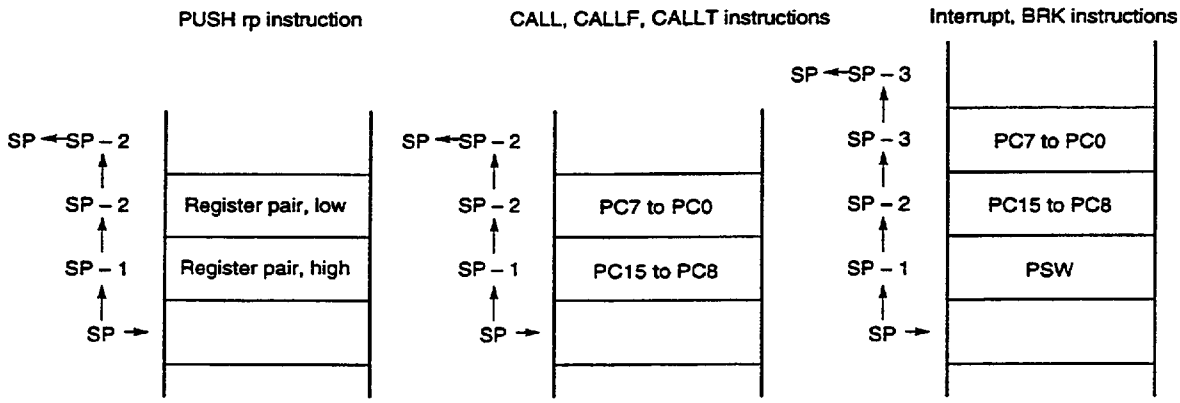
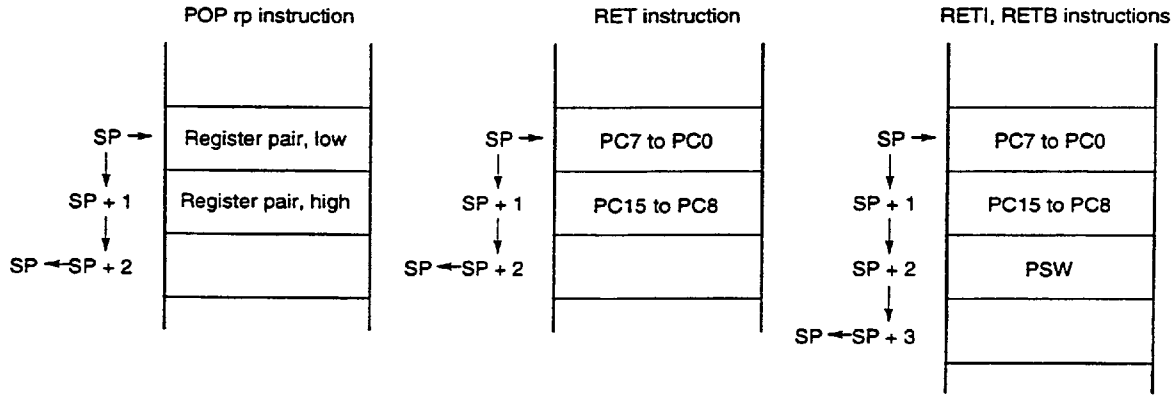


Figure 3-10. Data Restored from Stack Memory



3.2.2 General-purpose registers

General-purpose registers are mapped to the specific addresses of the data memory (FEE0H to FEFFH). Four banks of general-purpose registers, each consisting of eight 8-bit registers (X, A, C, B, E, D, L, and H) are available.

Each register can be used as an 8-bit register. Moreover, two 8-bit registers can be used as a register pair, which are 16-bit registers (AX, BC, DE, and HL).

Each register can be described not only in function name (X, A, C, B, E, D, L, H, AX, BC, DE, or HL) but also in absolute name (R0 to R7, RP0 to RP3).

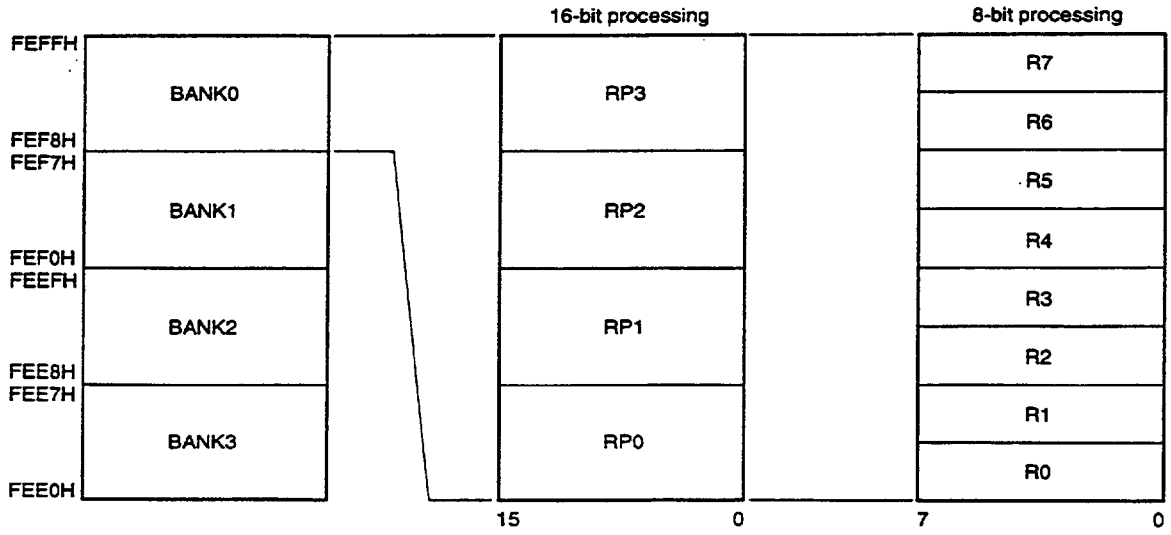
The register bank used for instruction execution is set by the CPU control instruction (SEL RBn). Because four register banks are provided, an efficient program can be developed by using one register bank for ordinary processing and another bank for interrupt processing.

Table 3-5. Absolute Addresses of General-Purpose Registers

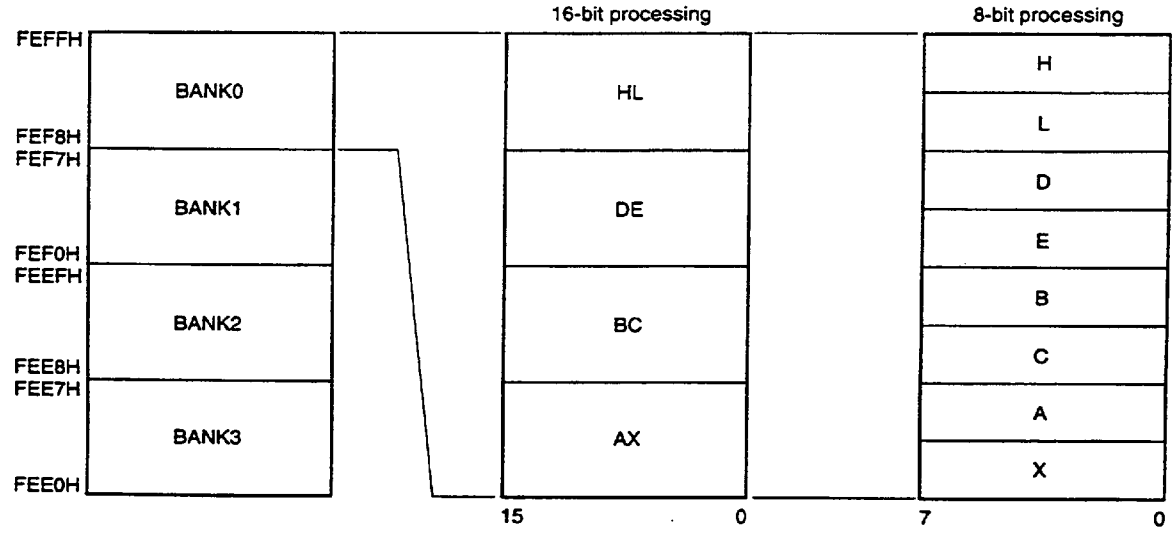
Bank Name	Register		Absolute Address	Bank Name	Register		Absolute Address
	Function Name	Absolute Name			Function Name	Absolute Name	
BANK0	H	R7	FEFFH	BANK2	H	R7	FEEFH
	L	R6	FEFEH		L	R6	FEEEH
	D	R5	FEFDH		D	R5	FEEDH
	E	R4	FEFCH		E	R4	FEECH
	B	R3	FEFBH		B	R3	FEEBH
	C	R2	FEFAH		C	R2	FEEAH
	A	R1	FEF9H		A	R1	FEE9H
	X	R0	FEF8H		X	R0	FEE8H
BANK1	H	R7	FEF7H	BANK3	H	R7	FEE7H
	L	R6	FEF6H		L	R6	FEE6H
	D	R5	FEF5H		D	R5	FEE5H
	E	R4	FEF4H		E	R4	FEE4H
	B	R3	FEF3H		B	R3	FEE3H
	C	R2	FEF2H		C	R2	FEE2H
	A	R1	FEF1H		A	R1	FEE1H
	X	R0	FEF0H		X	R0	FEE0H

Figure 3-11. General-Purpose Register Configuration

(a) Absolute name



(b) Function name



3.2.3 Special function registers (SFRs)

Unlike the general-purpose registers, special function registers have their own functions and are allocated to an area of addresses FF00H to FFFFH.

The special function registers can also be manipulated in the same manner as the general-purpose registers by using operation, transfer, and bit manipulation instructions. The bit units in which one register is to be manipulated (1, 8, or 16 bits) differ from that of another register.

The bit unit for manipulation is specified as follows:

- **1-bit manipulation**
  A symbol reserved by the assembler is described as the operand (sfr.bit) of a 1-bit manipulation instruction. An address can also be specified.
- **8-bit manipulation**
  A symbol reserved by the assembler is described as the operand (sfr) of an 8-bit manipulation instruction. An address can also be specified.
- **16-bit manipulation**
  A symbol reserved by the assembler is described as the operand (sfrp) of a 16-bit manipulation instruction. To specify address, describe an even address.

Table 3-6 lists the special function register. The meanings of the symbols in this table are as follows:

- **Symbol**
  These symbols indicate the addresses of the internal special function registers.
  They are reserved words for the RA78K/0 and defined by header file sfrbit.h for the CC78K/0. These symbols can be described as the operands of instructions when the RA78K/0 or ID78K0 is used.
- **R/W**
  Indicates whether the special function register in question can be read or written.
  R/W : Read/write
  R : Read only
  W : Write only
- **Bit units for manipulation**
  Indicates the bit units (1, 8, 16) in which the special function register in question can be manipulated.
- **At reset**
  Indicates the status of the special function register when the $\overline{\text{RESET}}$ signal is input.

Table 3-6. Special Function Register List (1/3)

Address	Special Function Register (SFR) Name	Symbol		R/W		Address Unit			On Reset
						1 bit	8 bits	16 bits	
FF00H	Port 0	P0		R/W		○	○	–	00H
FF01H	Port 1	P1		R		○	○	–	
FF02H	Port 2	P2		R/W		○	○	–	
FF03H	Port 3	P3				○	○	–	
FF04H	Port 4	P4				○	○	–	Undefined
FF05H	Port 5	P5				○	○	–	
FF06H	Port 6	P6				○	○	–	
FF08H	10-bit buffer register 0	BFCM0	BFCM0L	R/W	R/W	–	○	○	000H
FF09H			–		–		–		
FF0AH	10-bit buffer register 1	BFCM1	BFCM1L	R/W	R/W	–	○	○	
FF0BH			–		–		–		
FF0CH	10-bit buffer register 2	BFCM2	BFCM2L	R/W	R/W	–	○	○	
FF0DH			–		–		–		
FF0EH	10-bit buffer register 3	BFCM3	BFCM3L	R/W	R/W	–	○	○	0FFH
FF0FH			–		–		–		
FF10H	8-bit compare register 50	CR50		R/W		–	○	–	Undefined
FF11H	8-bit compare register 51	CR51				–	○	–	
FF12H	8-bit timer counter 50	TM5	TM50	R		–	○	○	00H
FF13H	8-bit timer counter 51		TM51			–	○		
FF14H	8-bit compare register 52	CR52		R/W		–	○	–	Undefined
FF15H	8-bit timer counter 52	TM52		R		–	○	–	00H
FF16H	A/D conversion result register	ADCR0	ADCR0L			–	–	○Note	Undefined
FF17H			ADCR0H			–	○Note		
FF18H	Transmit shift register 0	TXS00		W		–	○	–	FFH
	Receive buffer register 0	RXB00		R		–	○	–	
FF1AH	Transmit shift register 1	TXS01		W		–	○	–	
	Receive buffer register 1	RXB01		R		–	○	–	
FF20H	Port mode register 0	PM0		R/W		○	○	–	
FF22H	Port mode register 2	PM2				○	○	–	
FF23H	Port mode register 3	PM3				○	○	–	
FF24H	Port mode register 4	PM4				○	○	–	
FF25H	Port mode register 5	PM5				○	○	–	
FF26H	Port mode register 6	PM6				○	○	–	

Note Use ADCR0 when the A/D conversion result register is accessed.
These registers of the μPD780924 subseries can be accessed only in 8-bit units. When ADCR0 is read, the value of FF17H can be read.
These registers of the μPD780964 subseries can be accessed only in 16-bit units and cannot be accessed in 8-bit units.

Table 3-6. Special Function Register List (2/3)

Address	Special Function Register (SFR) Name	Symbol	R/W	Address Unit			On Reset
				1 bit	8 bits	16 bits	
FF30H	Pull-up resistor option register 0	PU0	R/W	○	○	–	00H
FF32H	Pull-up resistor option register 2	PU2		○	○	–	
FF33H	Pull-up resistor option register 3	PU3		○	○	–	
FF34H	Pull-up resistor option register 4	PU4		○	○	–	
FF35H	Pull-up resistor option register 5	PU5		○	○	–	
FF36H	Pull-up resistor option register 6	PU6		○	○	–	
FF42H	Watchdog timer clock select register	WDCS		–	○	–	
FF47H	Memory extension mode register	MEM		○	○	–	
FF48H	External interrupt rising edge enable register	EGP		○	○	–	
FF49H	External interrupt falling edge enable register	EGN		○	○	–	
FF68H	8-bit timer mode control register 50	TMC50		○	○	–	04H
FF69H	Timer clock select register 50	TCL50		–	○	–	00H
FF70H	8-bit timer mode control register 51	TMC51		○	○	–	04H
FF71H	Timer clock select register 51	TCL51		–	○	–	00H
FF78H	8-bit timer mode control register 52	TMC52		○	○	–	04H
FF79H	Timer clock select register 52	TCL52		–	○	–	00H
FF80H	A/D converter mode register 0	ADM0		○	○	–	
FF81H	Analog input channel specification register 0	ADS0		–	○	–	
FF84H	Real-time output buffer register L	RTBL0		○	○	–	
FF85H	Real-time output buffer register H	RTBH0		○	○	–	
FF86H	Real-time output port mode register	RTPM0		○	○	–	
FF87H	Real-time output port control register	RTPC0		○	○	–	
FF90H	Inverter timer control register 7	TMC7		○	○	–	
FF91H	Inverter timer mode register 7	TMM7		○	○	–	
FF92H FF93H	10-bit compare register 0	CM0		–	–	○	
FF94H FF95H	10-bit compare register 1	CM1		–	–	○	
FF96H FF97H	10-bit compare register 2	CM2		–	–	○	
FF98H FF99H	10-bit compare register 3	CM3		–	–	○	FFH
FF9AH	Dead time reload register	DTIME		–	○	–	
FFA0H	Asynchronous serial interface mode register 0	ASIM00		○	○	–	00H
FFA1H	Asynchronous serial interface status register 0	ASIS00	R	–	○	–	
FFA2H	Baud rate generator control register 0	BRGC00	R/W	–	○	–	
FFA4H	Asynchronous serial interface mode register 1	ASIM01		○	○	–	
FFA5H	Asynchronous serial interface status register 1	ASIS01	R	–	○	–	
FFA6H	Baud rate generator control register 1	BRGC01	R/W	–	○	–	

Table 3-6. Special Function Register List (3/3)

Address	Special Function Register (SFR) Name	Symbol		R/W	Address Unit			On Reset
					1 bit	8 bits	16 bits	
FFD0H to FFDFH	External access area[Note 1]			R/W	○	○	–	Undefined
FFE0H	Interrupt request flag register 0L	IF0L	IF0		○	○	○	00H
FFE1H	Interrupt request flag register 0H	IF0H			○	○		
FFE4H	Interrupt mask flag register 0L	MK0L	MK0		○	○	○	FFH
FFE5H	Interrupt mask flag register 0H	MK0H			○	○		
FFE8H	Priority specification flag register 0L	PR0L	PR0		○	○	○	
FFE9H	Priority specification flag register 0H	PR0H			○	○		
FFF0H	Memory size select register	IMS			–	○	–	CFH[Note2]
FFF8H	Memory extension wait setting register	MM			○	○	–	10H
FFF9H	Watchdog timer mode register	WDTM			○	○	–	00H
FFFAH	Oscillation stabilization time select register	OSTS			–	○	–	04H
FFFBH	Processor clock control register	PCC			○	○	–	

Notes
1. The external access area cannot be addressed in the SFR addressing mode. Access this area with an instruction that can specify an address with 16 bits.
2. The initial value of this register is CFH. Set the following value to this register of each model.
   μPD780921, 780961: 42H
   μPD780922, 780962: 44H
   μPD780923, 780963: C6H
   μPD780924, 780964: C8H
   μPD78F0924, 78F0964: Value corresponding to those of mask ROM models

3.3 Addressing Instruction Address

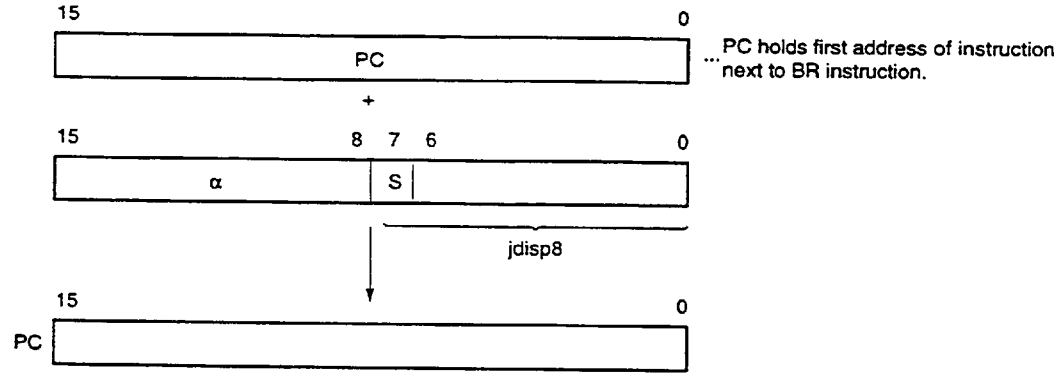
An instruction address is determined by the contents of the program counter (PC). The contents of the PC is usually automatically incremented by the number of bytes of an instruction to be fetched (by 1 per byte) every time an instruction is excuted. When an instruction that causes program execution to branch is performed, the address information of the branch destination is set to the PC by means of the following addressing (for details of each instruction, refer to **78K/0 Series User's Manual - Instruction (IEU-1372)**).

3.3.1 Relative addressing

[Function]

The 8-bit immediate data (displacement value: jdisp8) of the instruction code is added to the first address of the next instruction, the resultant sum is transferred to the program counter (PC), and the program branches. The displacement value is treated as signed 2's complement data (-128 to +127), and bit 7 serves as a sign bit. This addressing is used when "BR $addr16" instruction or conditional branch instruction is executed.

[Operation]



When S = 0, all bits of α are 0.
When S = 1, all bits of α are 1.

3.3.2 Immediate addressing

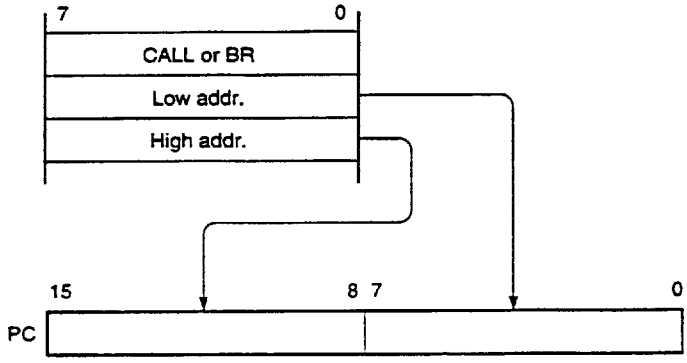
[Function]

The immediate data in an instruction word is transferred to the program counter (PC), and execution branches.

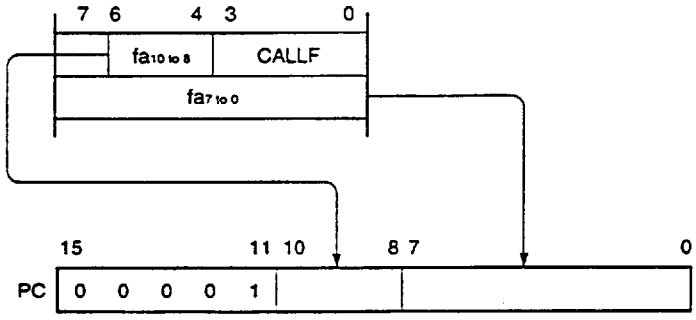
This addressing is used when the "CALL !addr16", "BR !addr16", or "CALLF !addr11" instruction is executed.

[Operation]

When "CALL !addr16" or "BR !addr16" instruction is executed



When "CALLF !addr11" instruction is executed

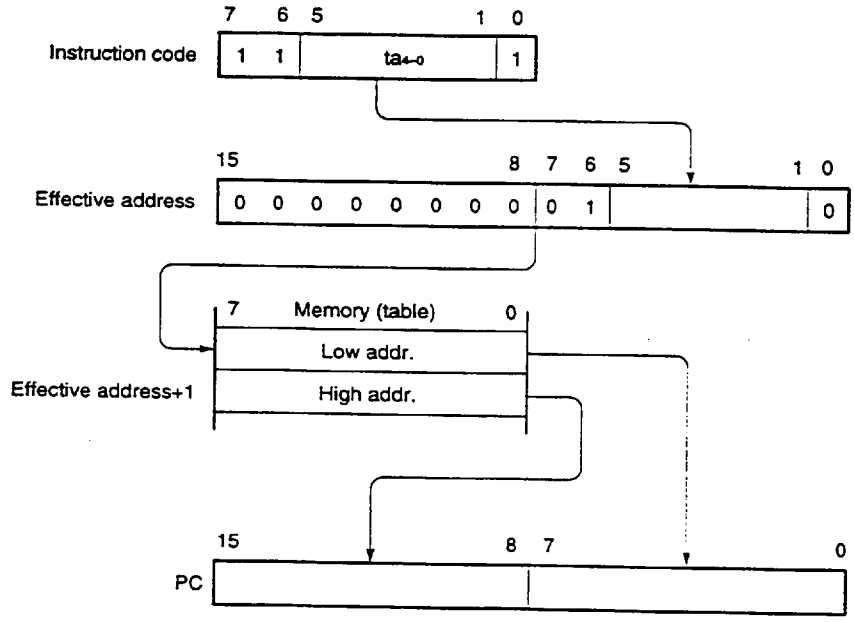


3.3.3 Table indirect addressing

[Function]

The contents of a specific location table (branch destination address) addressed by the immediate data of bits 1 to 5 of an instruction code are transferred to the program counter (PC), and program execution branches. This addressing is used when the "CALLT [addr5]" instruction is executed.

[Operation]



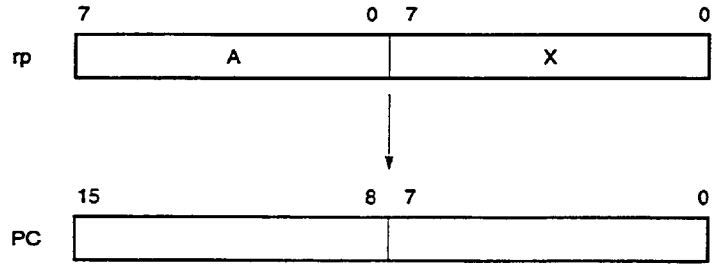
3.3.4 Register addressing

[Function]

The contents of the register pair (AX) specified by an instruction word are transferred to the program counter (PC), and program execution branches.

This addressing is used when the "BR AX" instruction is executed.

[Operation]



3.4 Addressing of Operand Address

3.4.1 Data memory addressing

The μPD780924, 780964 subseries are provided with many addressing modes to facilitate manipulation of the memory. By using these addressing modes, special function registers (SFRs) and general-purpose registers can be addressed according to their own functions. Figures 3-12 to 3-16 illustrate the addressing of the data memory.

Figure 3-12. Data Memory Addressing (μPD780921, 780961)

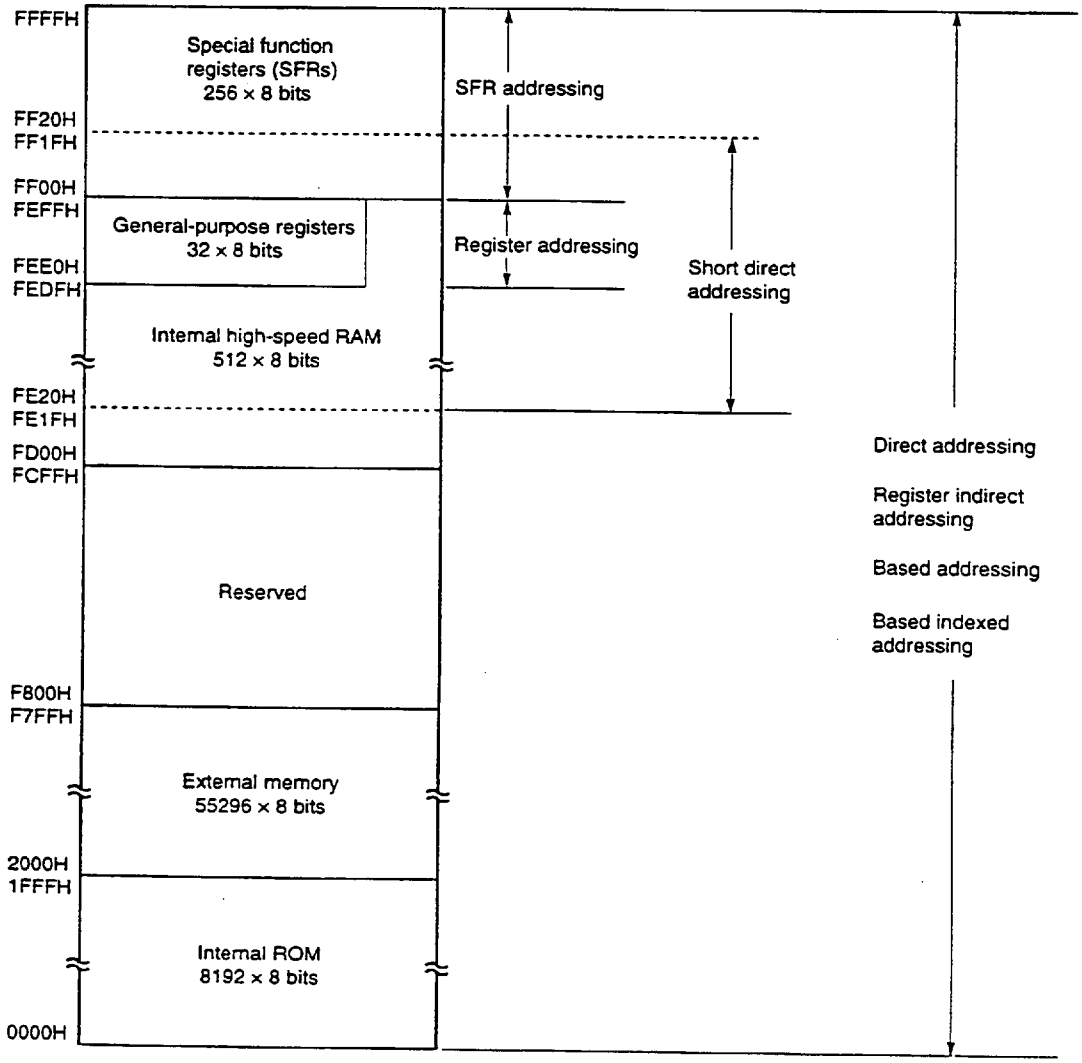


Figure 3-13. Data Memory Addressing (μPD780922, 780962)

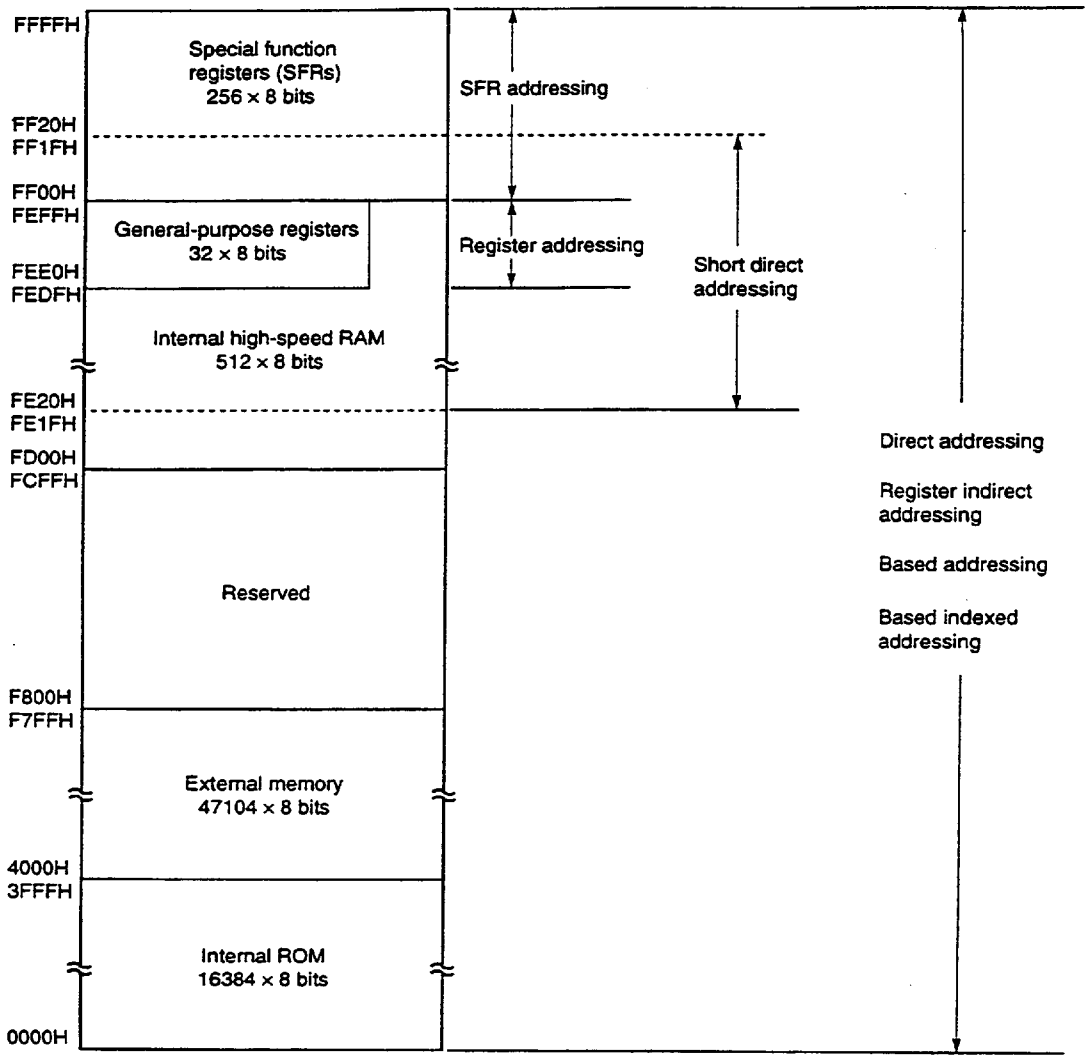


Figure 3-14. Data Memory Addressing (μPD780923, 780963)

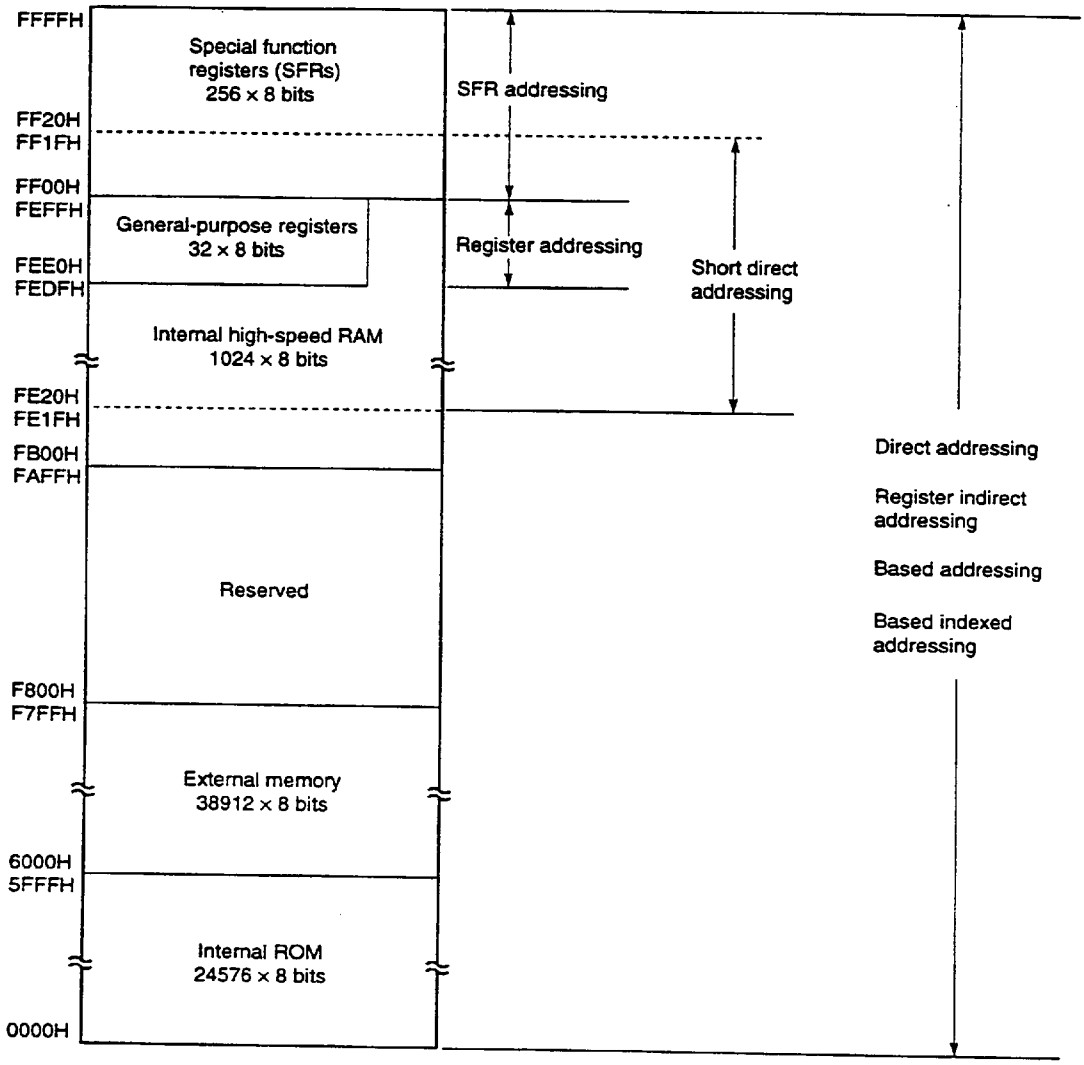


Figure 3-15. Data Memory Addressing (μPD780924, 780964)

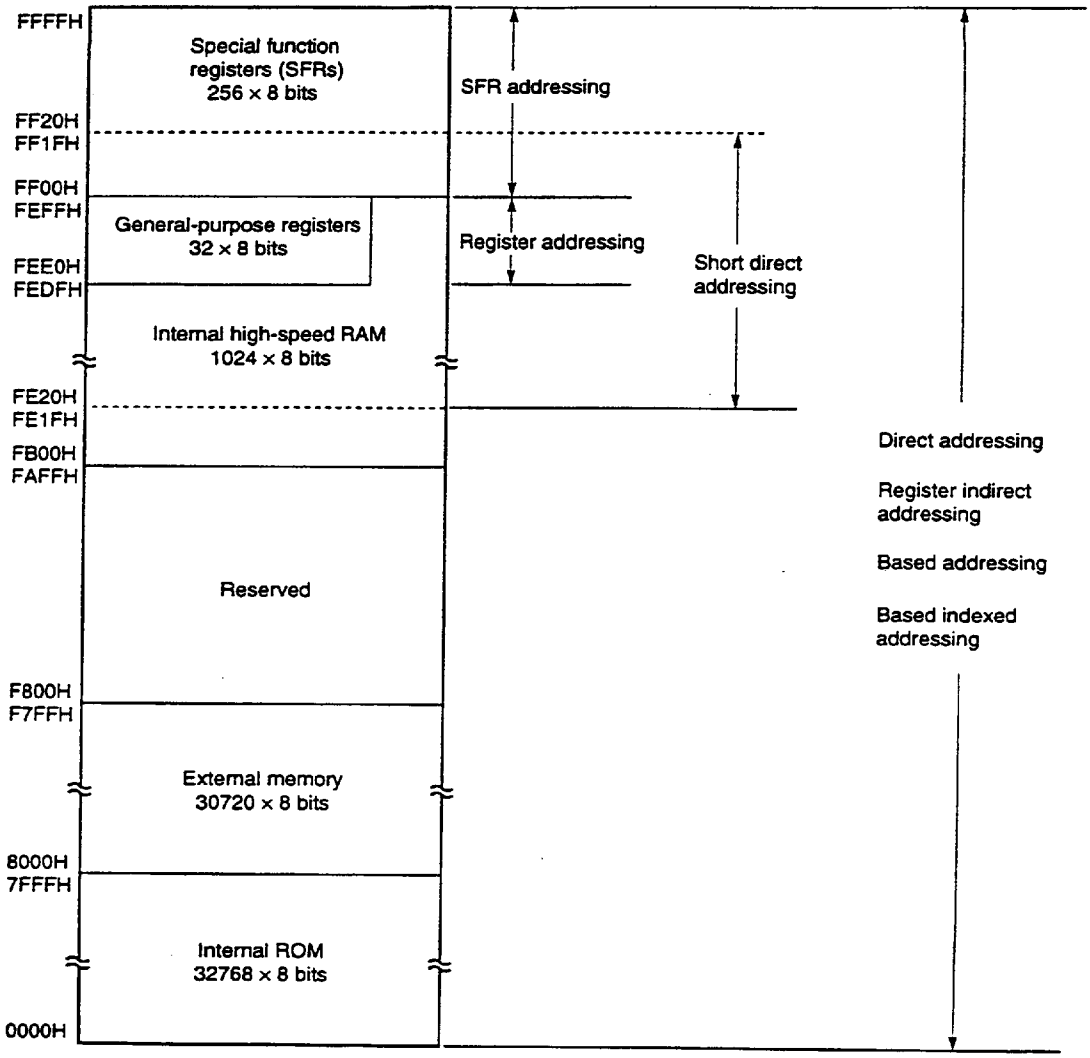
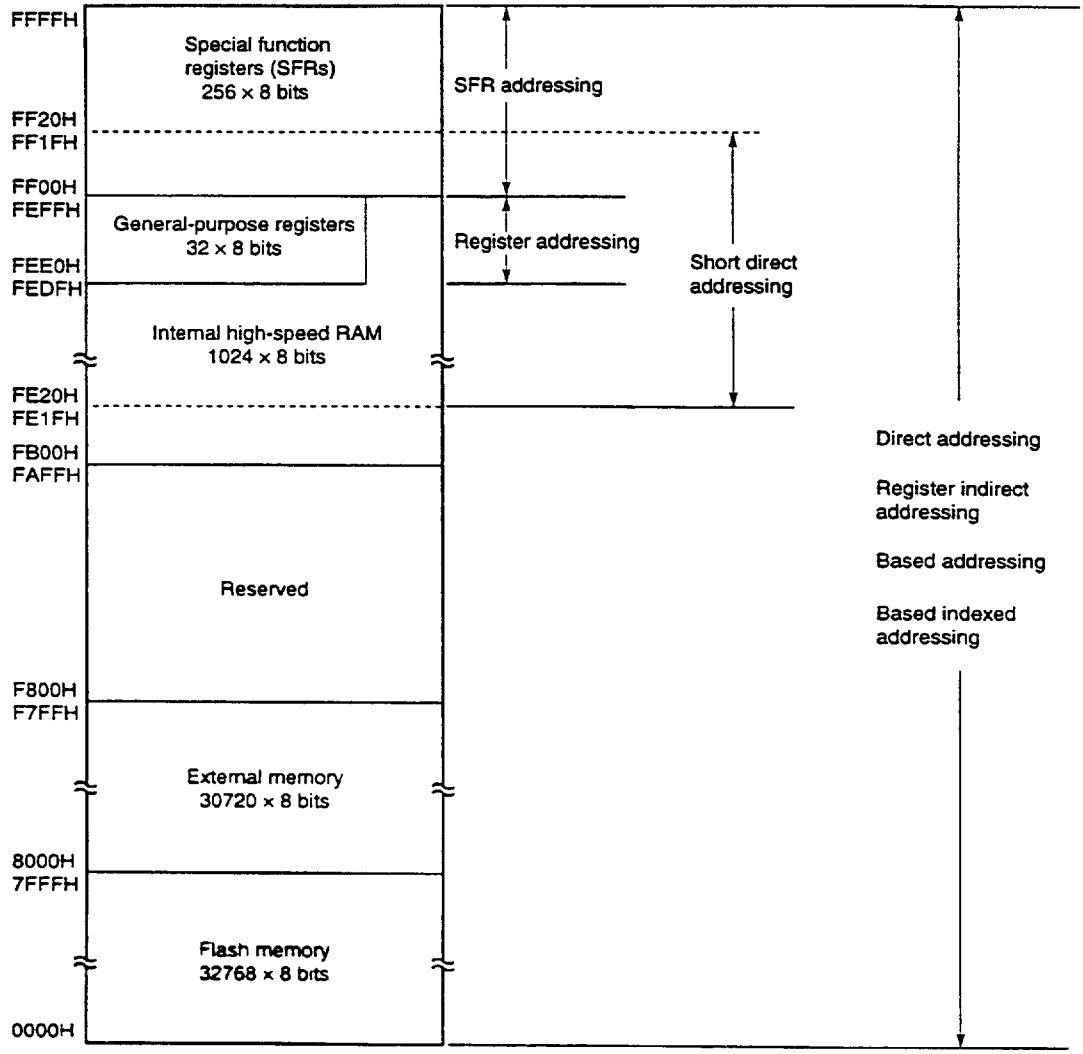


Figure 3-16. Data Memory Addressing (μPD78F0924, 78F0964)



3.4.2 Implied addressing

[Function]

This addressing is to automatically address a register that functions as an accumulator (A or AX) in the general-purpose register area.

Of the instruction words of the μPD780924 and 780964 subseries, those that use implied addressing are as follows:

Instruction	Register Specified by Implied Addressing
MULU	Register A to store multiplicand and register AX to store product
DIVUW	Register AX to store dividend and quotient
ADJBA/ADJBS	Register A to store numeric value subject to decimal adjustment
ROR4/ROL4	Register A to store digit data subject to digit rotation

[Operand Format]

No specific operand format is used because the operand format is automatically determined by an instruction.

[Example]

MULU X

The product between registers A and X is stored in register AX as a result of executing a multiply instruction of 8 bits x 8 bits. In this operation, registers A and AX are specified by implied addressing.

3.4.3 Register addressing

[Function]

This addressing accesses as an operand a general-purpose register selected by the register specification code (Rn, RPn) in an instruction word from the register bank specified by the register bank select flags (RBS0 and RBS1).

Register addressing is used when an instruction that has the following operand format is executed. When an 8-bit register is specified, one of the eight registers is specified by 3 bits in the instruction code.

[Operand Format]

Representation	Description
r	X, A, C, B, E, D, L, H
rp	AX, BC, DE, HL

r and rp can be described not only in function name (X, A, C, B, E, D, L, H, AX, BC, DE, or HL) but also in absolute name (R0 to R7, RP0 to RP3).

[Example]

MOV A, C; To select C register as r

Instruction code | 0 1 1 0 0 0 1 0 |

INCW DE; To select DE register pair as rp

Instruction code | 1 0 0 0 0 1 0 0 |

3.4.4 Direct addressing

[Function]

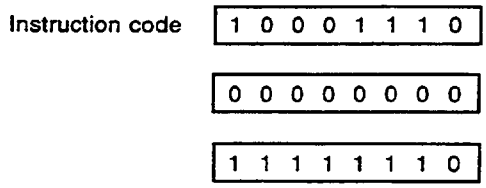
This addressing is to address a memory area to be manipulated by using the immediate data in an instruction word as an operand address.

[Operand Format]

Representation	Description
addr16	Label or 16-bit immediate data

[Example]

MOV A, !FE00H; To specify FE00H as !addr16



3.4.5 Short direct addressing

[Function]

This addressing directly addresses a memory area to be manipulated from a fixed space by using the 8-bit data in an instruction word.

This addressing is applicable to a 256-byte space of FE20H to FF1FH. The internal high-speed RAM is mapped to addresses FE20H to FEFFH, and special function registers (SFRs) are mapped to addresses FF00H to FF1FH. To the SFR area (FF00H to FF1FH) to which short direct addressing is applied, ports, and compare and capture registers of timer/event counters that are frequently accessed on program are mapped. These SFRs can be manipulated with a few bytes and clocks.

Bit 8 of the effective address is 0 if the 8-bit immediate data is in a range of 20H to FFH, and 1 if the data is in a range of 00H to 1FH.

[Operand Format]

Representation	Description
saddr	Label or immediate data FE20H to FF1FH
saddrp	Label or immediate data FE20H to FF1FH (even address only)

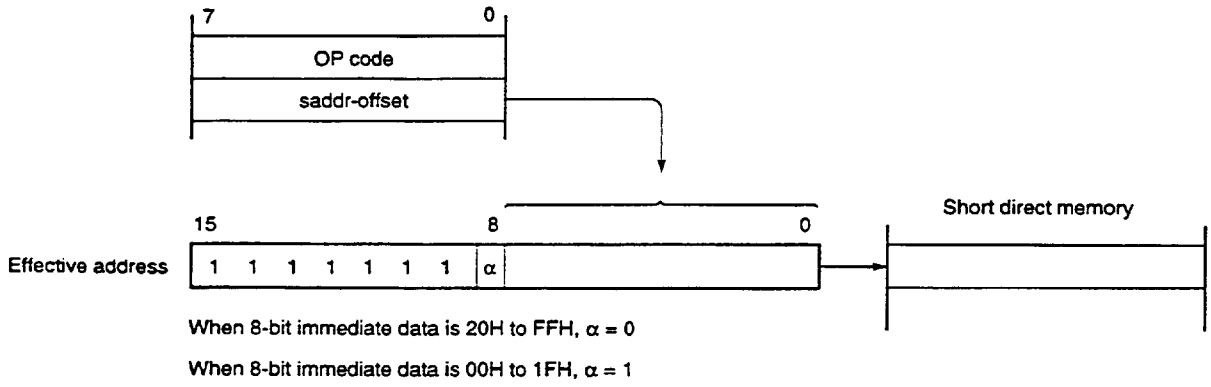
[Example]

MOV FE30H, #50H; To specify FE30H as saddr and 50H as immediate data

Instruction code

0	0	0	1	0	0	0	1	
0	0	1	1	0	0	0	0	
0	1	0	1	0	0	0	0	

[Operation]



When 8-bit immediate data is 20H to FFH, α = 0

When 8-bit immediate data is 00H to 1FH, α = 1

3.4.6 Special function register (SFR) addressing

[Function]

This addressing is to address special function registers (SFRs) mapped to the memory by using an 8-bit immediate data in an instruction word.

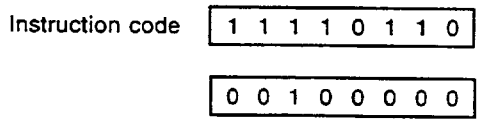
This addressing is applied to a 240-byte space of FF00H to FFCFH and FFE0H to FFFFH. However, the SFRs mapped to an area of FF00H to FF1FH can also be accessed by means of short direct addressing.

[Operand Format]

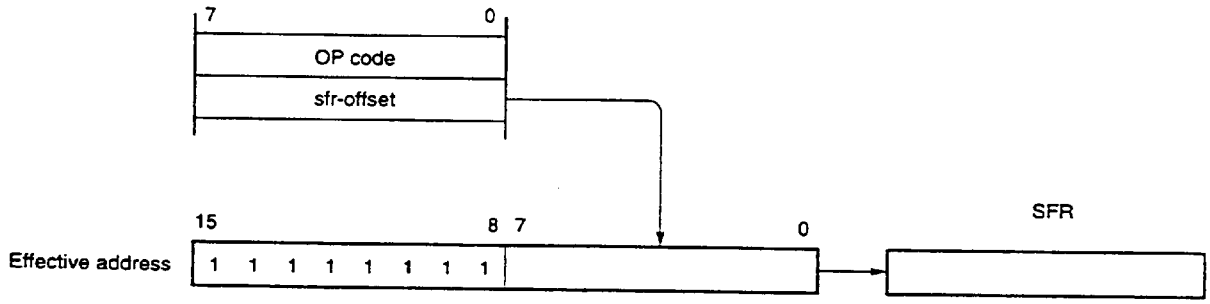
Representation	Description
sfr	Special function register name
sfrp	Name of special function register that can be manipulated in 16-bit units (even address only)

[Example]

MOV PM0, A: To select PM0 as sfr



[Operation]



3.4.7 Register indirect addressing

[Function]

This addressing is to address a memory area to be manipulated by using as an operand address the contents of a register pair specified by the register pair specification code in an instruction word from the register bank specified by the register bank select flags (RBS0 and RBS1). This addressing can address the entire memory space.

[Operand Format]

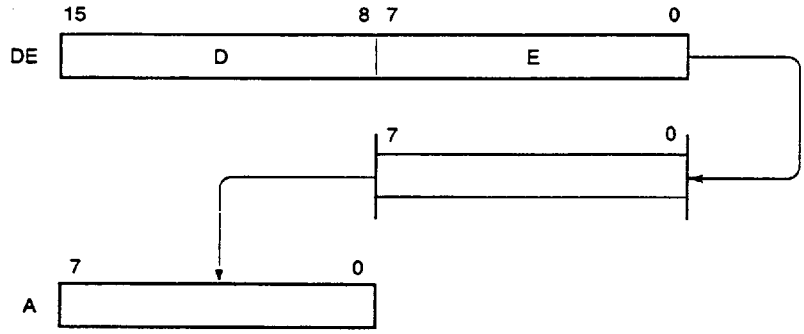
Representation	Description
—	[DE], [HL]

[Example]

MOV A, [DE]; To select [DE] as register pair

Instruction code | 1 0 0 0 0 1 0 1 |

[Operation]



3.4.8 Based addressing

[Function]

This addressing is to address a memory area by using the result of adding 8-bit immediate data as offset data, to the contents of the HL register pair in an instruction word as a base register. The base register is selected from the register bank specified by the register bank select flags (RBS0 and RBS1). The addition is executed by extending the offset data to 16 bits as a positive number. A carry from the 16th bit is ignored. This addressing is used to address the entire memory space.

[Operand Format]

Representation	Description
—	[HL + byte]

[Example]

MOV A, [HL+10H]; To specify 10H as byte

Instruction code | 1 0 1 0 1 1 1 0 |

| 0 0 0 1 0 0 0 0 |

3.4.9 Based indexed addressing

[Function]

This addressing is to address a memory area by using the result of adding the contents of the B or C register specified in an instruction word to the HL register pair in an instruction word as a base register. The base register is selected from the register bank specified by the register bank select flags (RBS0 and RBS1). The addition is executed by extending the offset data to 16 bits as a positive number. A carry from the 16th bit is ignored. This addressing is used to address the entire memory space.

[Operand Format]

Representation	Description
—	[HL + B], [HL + C]

[Example]

When MOV A, [HL+B]

Instruction code | 1 0 1 0 1 0 1 1 |

3.4.10 Stack addressing

[Function]

This addressing is to indirectly address the stack area by using the contents of the stack pointer (SP).
This addressing is automatically used to save/restore register contents when the PUSH, POP, subroutine call, or return instruction is executed, or when an interrupt request is generated.
The stack addressing can access the internal high-speed RAM area only.

[Example]

When PUSH DE is executed

Instruction code	1	0	1	1	0	1	0	1

[MEMO]

PAGE(S) INTENTIONALLY BLANK

4.1 Functions of Ports

The μPD780924, 780964 subseries is provided with eight input port pins and 39 I/O port pins. Figure 4-1 shows these port pins. Each port can be manipulated in 1-bit or 8-bit units and controlled in various ways. Moreover, some port pins also serve as the I/O pins of the internal hardware.

Figure 4-1. Types of Ports

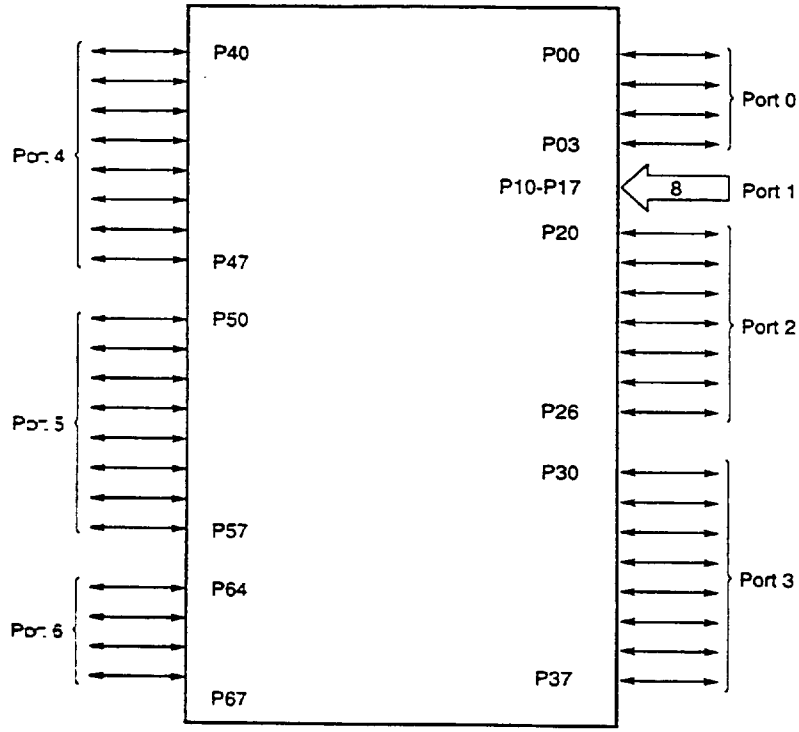


Table 4-1. Port Functions

Pin Name		Function	Shared by:
Port 0	P00	4-bit I/O port Can be specified for input/output bitwise. When used as input port, internal pull-up resistor can be connected by software.	INTP0/TOFF7
	P01		INTP1
	P02		INTP2
	P03		INTP3/ADTRG
Port 1	P10 to P17	8-bit input port	ANI0 to ANI7
Port 2	P20	7-bit I/O port Can be specified for input/output bitwise. When used as input port, internal pull-up resistor can be connected by software.	RxD00
	P21		TxD00
	P22		RxD01
	P23		TxD01
	P24		TI50/TO50
	P25		TI51/TO51
	P26		TI52/TO52
Port 3	P30 to P37	8-bit I/O port Can be specified for input/output bitwise. When used as input port, internal pull-up resistor can be connected by software.	RTP0 to RTP7
Port 4	P40 to P47	8-bit I/O port Can be specified for input/output bitwise. When used as input port, internal pull-up resistor can be connected by software.	AD0 to AD7
Port 5	P50 to P57	8-bit I/O port Can directly drive LED. Can be specified for input/output bitwise. When used as input port, internal pull-up resistor can be connected by software.	A8 to A15
Port 6	P64	4-bit I/O port Can be specified for input/output bitwise. When used as input port, internal pull-up resistor can be connected by software.	$\overline{RD}$
	P65		$\overline{WR}$
	P66		$\overline{WAIT}$
	P67		ASTB

4.2 Port Configuration

A port consists of the following hardware:

Table 4-2. Port Configuration

Item		Configuration
Control register		Port mode register (PMm: m = 0, 2 to 6) Pull-up resistor option register (PUm: m = 0, 2 to 6)
Port	Total	47 lines
	Input	8 lines
	I/O	39 lines
Pull-up resistor		39 lines (software control)

4.2.1 Port 0

This is a 4-bit I/O port with output latch. Port 0 can be specified in the input or output mode in 1-bit units by using the port mode register 0. When P00 to P03 pins are used as input port pins, internal pull-up resistors can be connected in bit-wise by using the pull-up resistor option register 0.

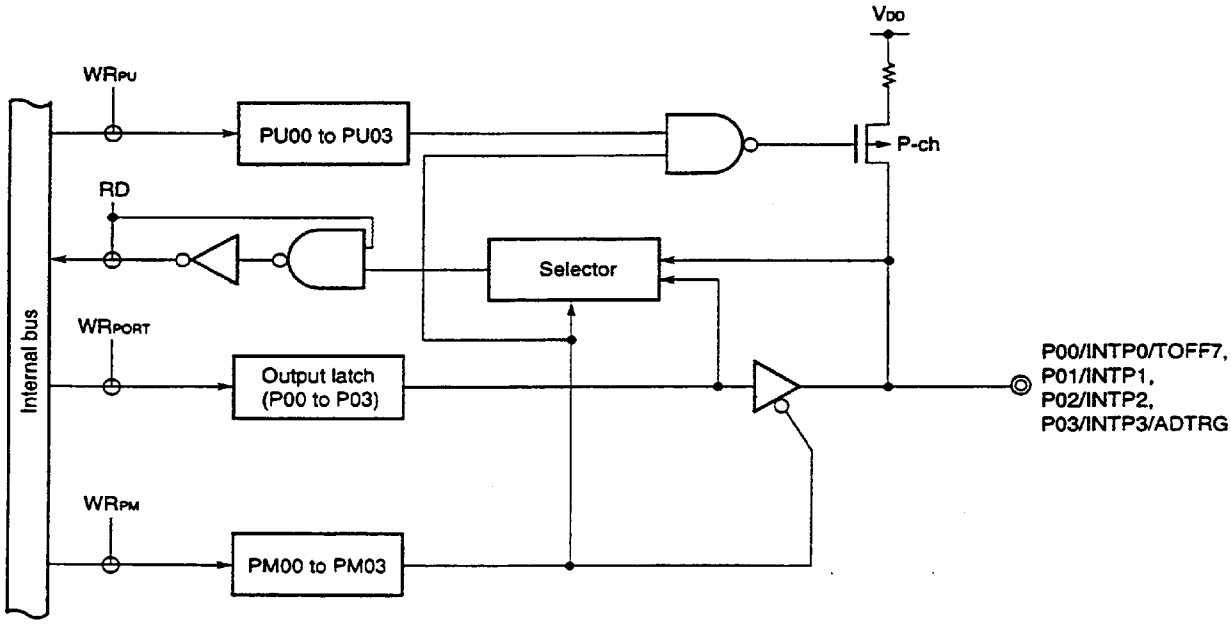
The five port pins are also used to input external interrupts, a timer output stop interrupt, and an external trigger signal for the A/D converter.

Port 0 is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

Figure 4-2 shows the block diagram of port 0.

Caution Because port 0 is also used as an external interrupt input pin, an interrupt request flag is set when the port is specified in the output mode and its output level is changed. When using port 0 in the output mode, therefore, set the interrupt mask flag to 1.

Figure 4-2. Block Diagram of P00 to P03

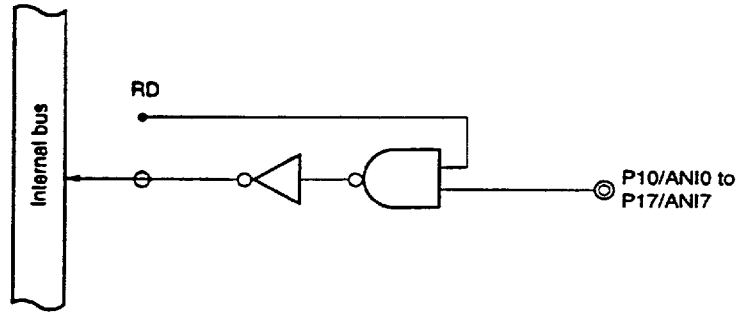


PU : pull-up resistor option register
PM : port mode register
RD : read signal of port 0
WR : write signal of port 0

4.2.2 Port 1

This is an 8-bit input port. The pins of this port are used as the analog input pins of the A/D converter.
Figure 4-3 shows the block diagram of port 1.

Figure 4-3. Block Diagram of P10 to P17



4.2.3 Port 2

This is an 7-bit I/O port with output latch. P20 to P26 pins can be specified in the input or output mode in 1-bit units by using the port mode register 2. When using P20 to P26 pins as input port pins, internal pull-up resistors can be connected bit-wise by using the pull-up resistor option register 2.

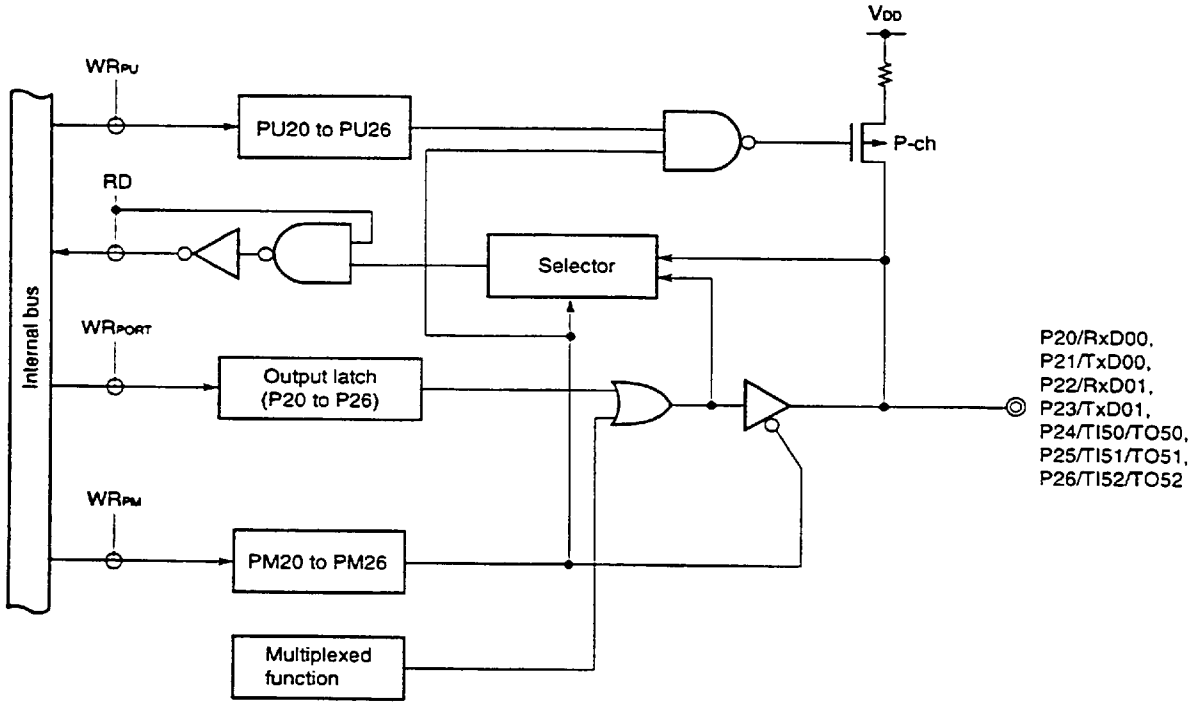
The pins of this port are also used as the data I/O pin and timer I/O pin of the serial interface.

This port is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

Figure 4-4 shows the block diagram of port 2.

Caution When performing transmission using the serial interface or timer output, set the pins to be used to the output mode, and set the output latch to 0.
When performing reception or timer input, set the pins to be used to the input mode.

Figure 4-4. Block Diagram of P20 to P26



PU : pull-up resistor option register
PM : port mode register
RD : read signal of port 2
WR : write signal of port 2

4.2.4 Port 3

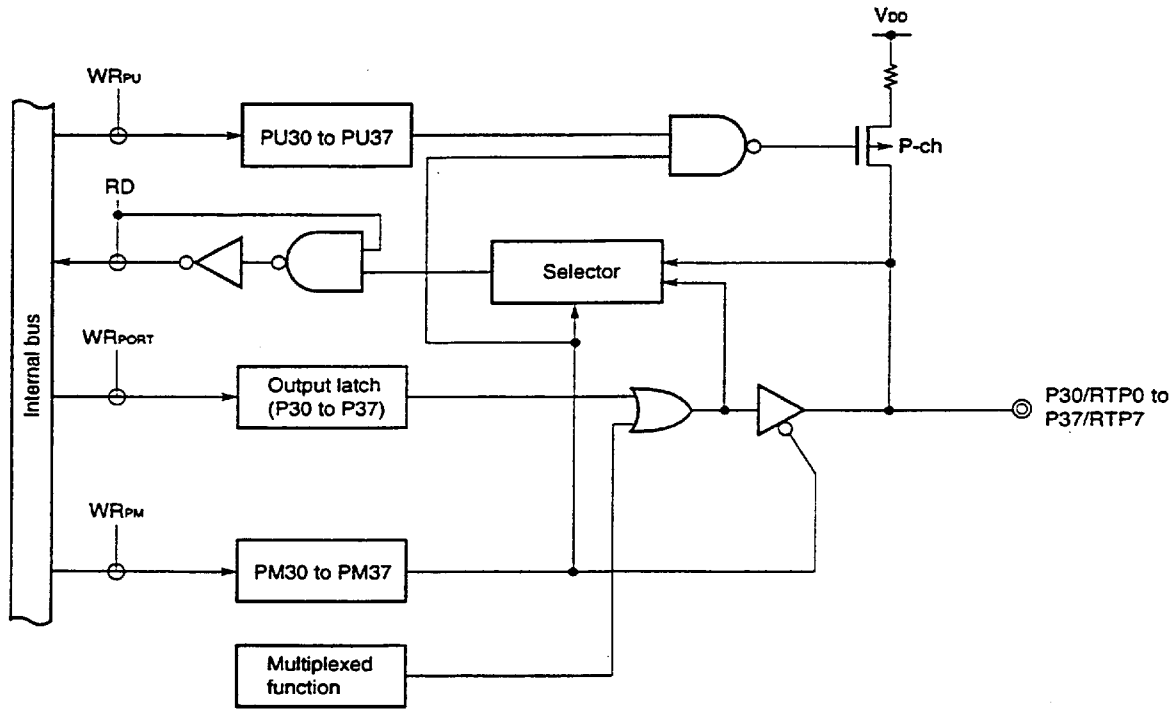
This is an 8-bit I/O port with output latch. P30 to P37 pins can be specified in the input or output mode in 1-bit units by using the port mode register 3. When using P30 to P37 pins as input port pins, internal pull-up resistors can be connected bit-wise by using the pull-up resistor option register 3.

The pins of this port are also used as the real-time output port.

This port is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

Figure 4-5 shows the block diagram of port 3.

Figure 4-5. Block Diagram of P30 to P37



PU : pull-up resistor option register
PM : port mode register
RD : read signal of port 3
WR : write signal of port 3

4.2.5 Port 4

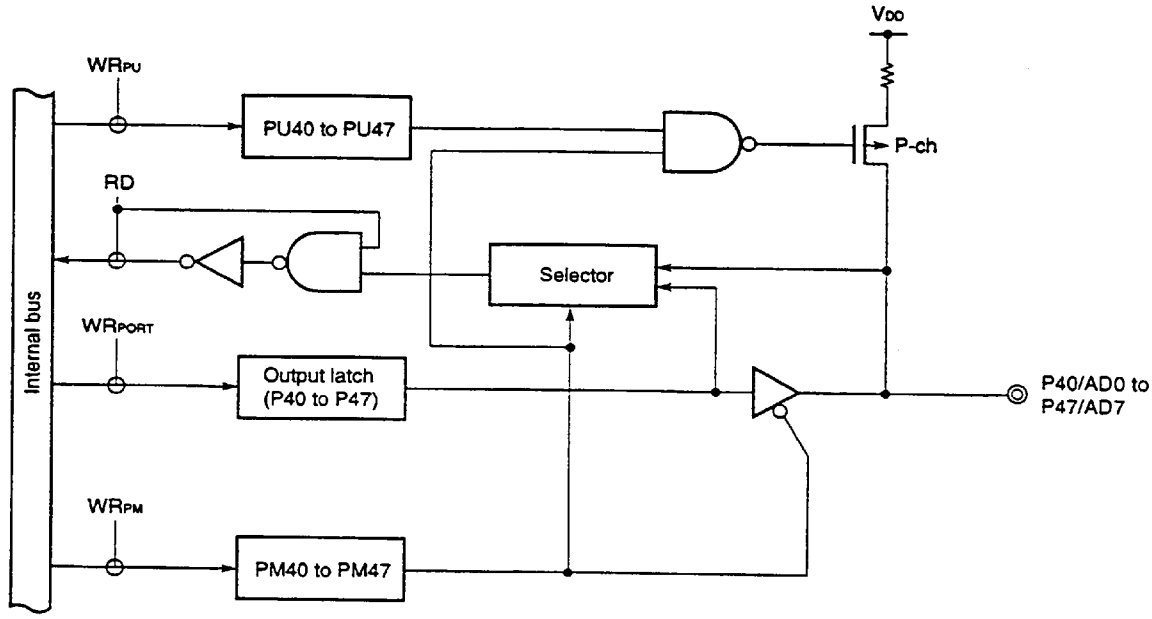
This is an 8-bit I/O port with output latch. P40 to P47 pins can be specified in the input or output mode bit-wise by using the port mode register 4. When using P40 to P47 pins as input port pins, internal pull-up resistors can be connected bit-wise by using the pull-up resistor option register 4.

These port pins are also multiplexed with an address/data bus that is used in the external memory extension mode.

This port is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

Figure 4-6 shows the block diagram of port 4.

Figure 4-6. Block Diagram of P40 to P47



PU : pull-up resistor option register
PM : port mode register
RD : read signal of port 4
WR : write signal of port 4

4.2.6 Port 5

This is an 8-bit I/O port with output latch. P50 to P57 pins can be specified in the input or output mode in 1-bit units by using the port mode register 5. When using P50 to P57 pins as input port pins, internal pull-up resistors can be connected bit-wise by using the pull-up resistor option register 5.

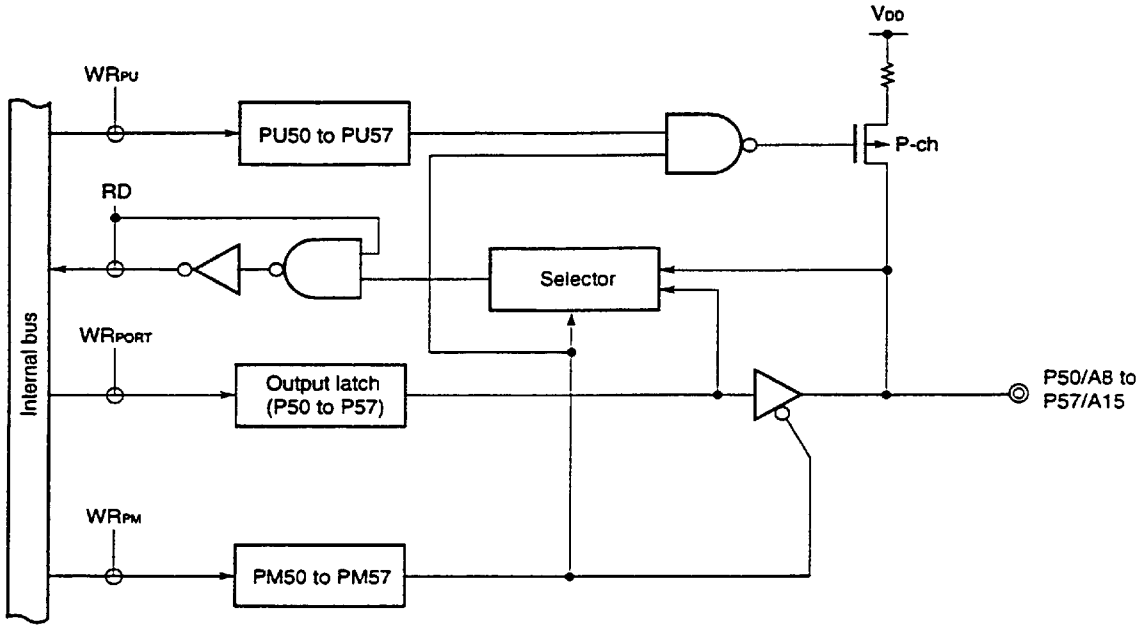
Port 5 can directly drive an LED.

These port pins are also multiplexed with an address bus that is used in the external memory extension mode.

This port is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

Figure 4-7 shows the block diagram of port 5.

Figure 4-7. Block Diagram of P50 to P57



PU : pull-up resistor option register
PM : port mode register
RD : read signal of port 5
WR : write signal of port 5

4.2.7 Port 6

This is an 4-bit I/O port with output latch. Pins P64 to P67 can be specified in the input or output mode in 1-bit units by using the port mode register 6. When pins P64 to P67 are used as input ports, an on-chip pull-up resistor can be connected to them bit-wise with a pull-up resistor option register 6.

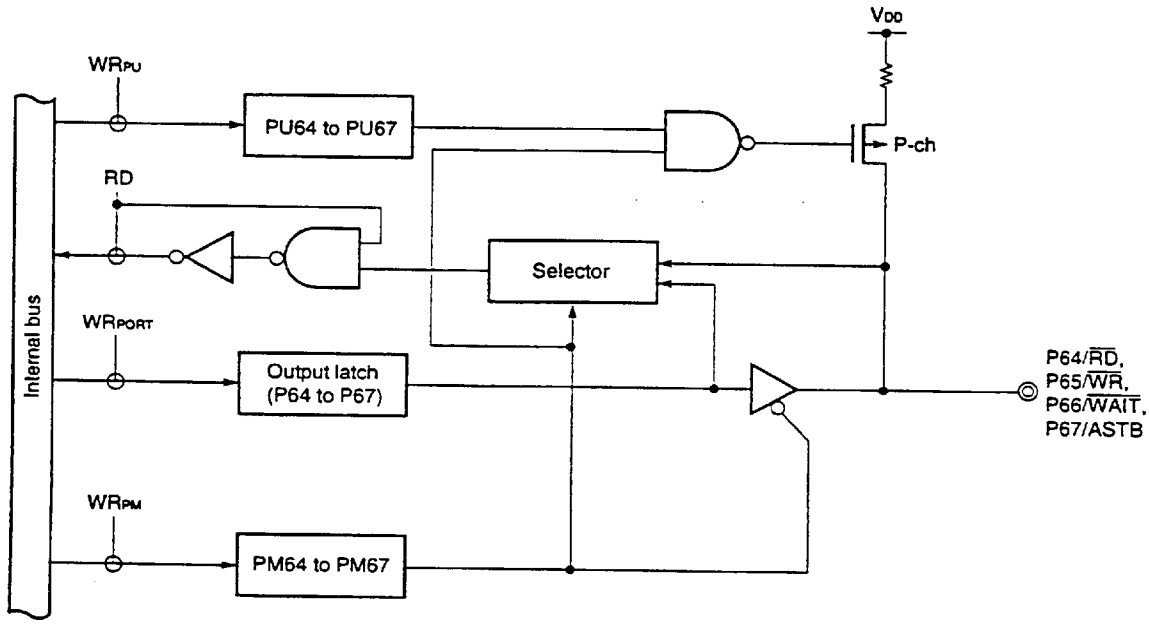
These port pins are also used to output control signals in the external memory extension mode.

This port is set in the input mode when the $\overline{\text{RESET}}$ signal is input.

Figure 4-8 shows the block diagram of port 6.

Caution P66 can be used as an I/O port pins when no external wait state is used in the external memory extension mode.

Figure 4-8. Block Diagram of P64 to P67



PU : pull-up resistor option register
PM : port mode register
RD : read signal of port 6
WR : write signal of port 6

4.3 Registers Controlling Port Functions

The following two types of registers control the ports:

- Port mode registers (PM0, PM2, PM3, PM4, PM5, PM6)
- Pull-up resistor option register (PU0, PU2, PU3, PU4, PU5, PU6)

(1) Port mode registers (PM0, PM2, PM3, PM4, PM5, PM6)

These registers set the corresponding ports in the input or output mode in 1-bit units.

PM0, PM2, PM3, PM4, PM5, and PM6 are manipulated by a 1-bit or 8-bit memory manipulation instruction.

When the $\overline{\text{RESET}}$ signal is input, these registers are set to FFH.

Caution Because port 0 is multiplexed with external interrupt input pins, interrupt request flags are set when the output mode of the port function is specified and the output level is changed. To use this port in the output mode, therefore, set 1 to the interrupt mask flags in advance.

Figure 4-9. Format of Port Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
PM0	1	1	1	1	PM03	PM02	PM01	PM00	FF20H	FFH	R/W
PM2	PM27	PM26	PM25	PM24	PM23	PM22	PM21	PM20	FF22H	FFH	R/W
PM3	PM37	PM36	PM35	PM34	PM33	PM32	PM31	PM30	FF23H	FFH	R/W
PM4	PM47	PM46	PM45	PM44	PM43	PM42	PM41	PM40	FF24H	FFH	R/W
PM5	PM57	PM56	PM55	PM54	PM53	PM52	PM51	PM50	FF25H	FFH	R/W
PM6	PM67	PM66	PM65	PM64	1	1	1	1	FF26H	FFH	R/W

PMmn	Selects I/O mode of Pmn pin (m = 0: n = 0 to 3) (m = 2: n = 0 to 6) (m = 3, 4, 5: n = 0 to 7) (m = 6: n = 4 to 7)
0	Output mode (output buffer ON)
1	Input mode (output buffer OFF)

(2) Pull-up resistor option register (PU0, PU2, PU3, PU4, PU5, PU6)

This register sets whether the internal pull-up resistor is connected to each port. PUm (m = 0, 2 to 6) can specify connection of a pull-up resistor to the pins of each port.

PUm (m = 0, 2 to 6) is set by a 1-bit or 8-bit memory manipulation instruction.

This register is set to 00H when the $\overline{\text{RESET}}$ signal is input.

Caution Port 1 is not provided with an internal pull-up resistor.

Figure 4-10. Format of Pull-Up Resistor Option Register

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
PU0	0	0	0	0	PU03	PU02	PU01	PU00	FF30H	00H	R/W
PU2	0	PU26	PU25	PU24	PU23	PU22	PU21	PU20	FF32H	00H	R/W
PU3	0	PU36	PU35	PU34	PU33	PU32	PU31	PU30	FF33H	00H	R/W
PU4	PU47	PU46	PU45	PU44	PU43	PU42	PU41	PU40	FF34H	00H	R/W
PU5	PU57	PU56	PU55	PU54	PU53	PU52	PU51	PU50	FF35H	00H	R/W
PU6	PU67	PU66	PU65	PU64	0	0	0	0	FF36H	00H	R/W

PUmn	Selects internal pull-up resistor of Pmn pin (m = 0: n = 0 to 3) (m = 2: n = 0 to 6) (m = 3, 4, 5: n = 0 to 7) (m = 6: n = 4 to 7)
0	Internal pull-up resistor is not used
1	Internal pull-up resistor is used

4.4 Operation of Port Functions

The operation of a port differs depending on whether the port is set in the input or output mode, as described below.

4.4.1 Writing to I/O port

(1) In output mode

A value can be written to the output latch of a port by using a transfer instruction. The contents of the output latch can be output from the pins of the port.

The data once written to the output latch is retained until new data is written to the output latch.

(2) In input mode

A value can be written to the output latch by using a transfer instruction. However, the status of the port pin is not changed because the output buffer is OFF.

The data once written to the output latch is retained until new data is written to the output latch.

Caution A 1-bit memory manipulation instruction is executed to manipulate 1 bit of a port. However, this instruction accesses the port in 8-bit units. When this instruction is executed to manipulate a bit of an input/output port, therefore, the contents of the output latch of the pin that is set in the input mode and not subject to manipulation become undefined.

4.4.2 Reading from I/O port

(1) In output mode

The contents of the output latch can be read by using a transfer instruction. The contents of the output latch are not changed.

(2) In input mode

The status of a pin can be read by using a transfer instruction. The contents of the output latch are not changed.

4.4.3 Arithmetic operation of I/O port

(1) In output mode

An arithmetic operation can be performed with the contents of the output latch. The result of the operation is written to the output latch. The contents of the output latch are output from the port pins.

The data once written to the output latch is retained until new data is written to the output latch.

(2) In input mode

The contents of the output latch become undefined. However, the status of the pin is not changed because the output buffer is OFF.

Caution A 1-bit memory manipulation instruction is executed to manipulate 1 bit of a port. However, this instruction accesses the port in 8-bit units. When this instruction is executed to manipulate a bit of an input/output port, therefore, the contents of the output latch of the pin that is set in the input mode and not subject to manipulation become undefined.

5.1 Function of Clock Generation Circuit

The clock generation circuit generates the clock to be supplied to the CPU and peripheral hardware.

The system oscillation circuit oscillates a frequency of 1.0 to 8.38 MHz. Oscillation can be stopped by executing the STOP instruction.

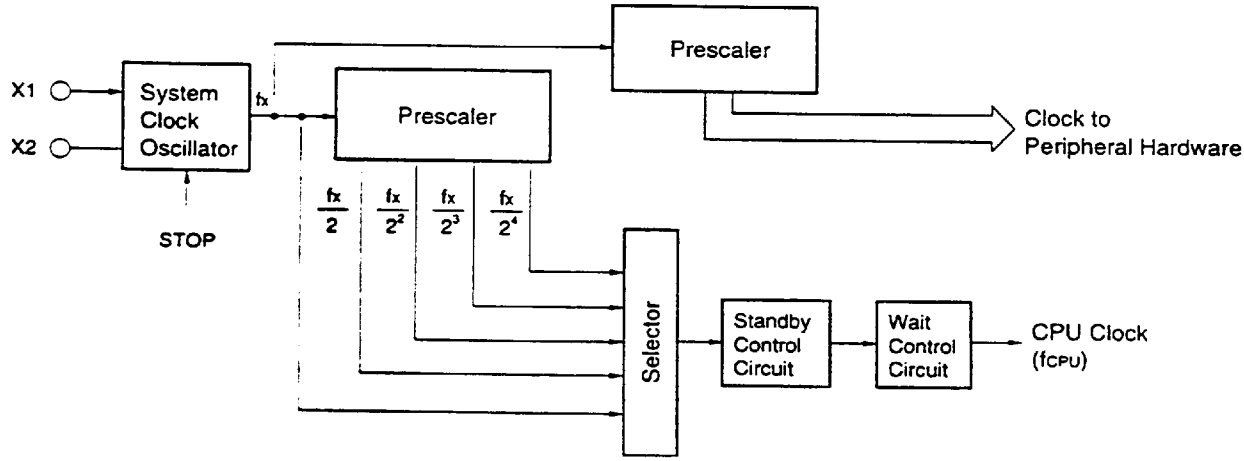
5.2 Configuration of Clock Generation Circuit

The clock generation circuit consists of the following hardware:

Table 5-1. Configuration of Clock Generation Circuit

Item	Configuration
Control register	Processor clock control register (PCC)
Oscillation circuit	System clock oscillation circuit

Figure 5-1. Clock Generator Block Diagram



5.3 Register Controlling Clock Generation Circuit

The clock generation circuit is controlled by the processor clock control register (PCC). This register selects the CPU clock.

PCC is set by a 1-bit or 8-bit memory manipulation instruction.

This register is set to 04H when the $\overline{\text{RESET}}$ signal is input.

Figure 5-2. Format of Processor Clock Control Register

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
PCC	0	0	0	0	0	PCC2	PCC1	PCC0	FFFBH	04H	R/W

PCC2	PCC1	PCC0	Selects CPU clock (f_{CPU})
0	0	0	f_x (0.24 μs)
0	0	1	$f_x/2$ (0.48 μs)
0	1	0	$f_x/2^2$ (0.96 μs)
0	1	1	$f_x/2^3$ (1.9 μs)
1	0	0	$f_x/2^4$ (3.8 μs)
Others			Setting prohibited

Caution Be sure to clear bit 3 to bit 7 to 0.

Remarks 1. fx : system clock oscillation frequency

2. ( ): at fx = 8.38 MHz operation

5.4 System Clock Oscillation Circuits

5.4.1 System clock oscillation circuit

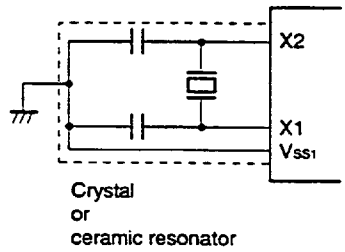
The main system clock oscillation circuit is oscillated by the crystal or ceramic resonator (8.38 MHz TYP.) connected across the X1 and X2 pins.

An external clock can also be input to the circuit. In this case, input the clock signal to the X1 pin, and input the reversed signal to the X2 pin.

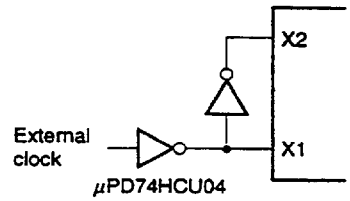
Figure 5-3 shows the external circuit of the system clock oscillation circuit.

Figure 5-3. External Circuit of System Clock Oscillation Circuit

(a) Crystal or ceramic oscillation



(b) External clock



Cautions 1. The STOP instruction cannot be excuted when the external clock is input because the X2 pin is pulled up to V_{DD1}.

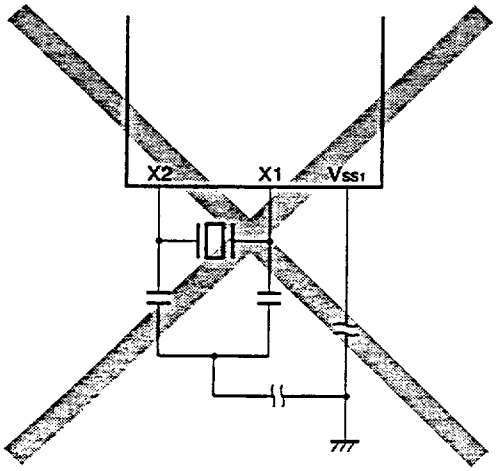
2. When using the subsystem clock oscillation circuit, to avoid influence of wiring capacity, etc. wire the portion enclosed by dotted line in Figure 5-3 as follows:

- **Keep the wiring length as short as possible.**
- **Do not cross the wiring with any other signal lines. Do not route the wiring in the vicinity of a line through which a high alternating current flows.**
- **Always keep the ground of the capacitor of the oscillation circuit at the same potential as V_{SS1}. Do not ground the capacitor to a ground pattern through which a high current flows.**
- **Do not extract a signal from the oscillation circuit.**

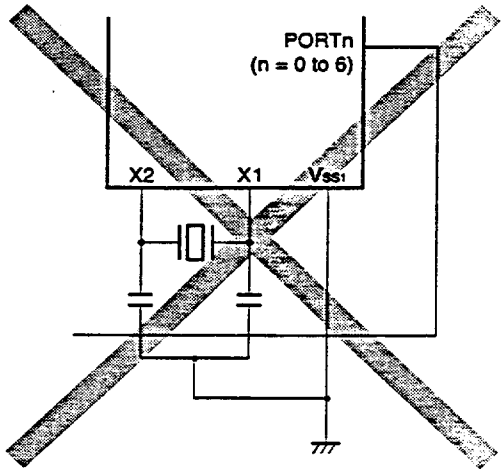
Figure 5-4 shows incorrect examples of resonator connection.

Figure 5-4. Incorrect Examples of Resonator Connection (1/2)

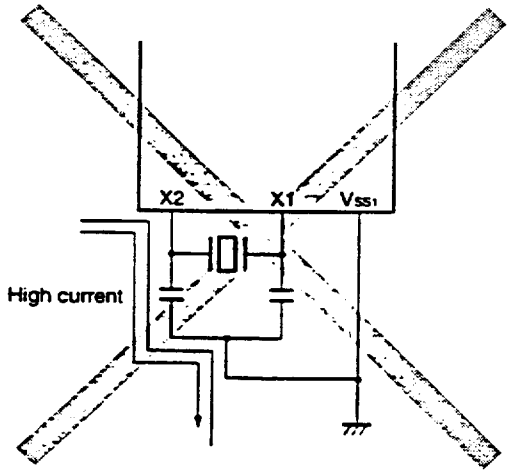
(a) Too long wiring



(b) Crossed signal line



(c) Wiring near high alternating current



(d) Current flowing through ground line of oscillation circuit (potential at points A, B, and C fluctuates)

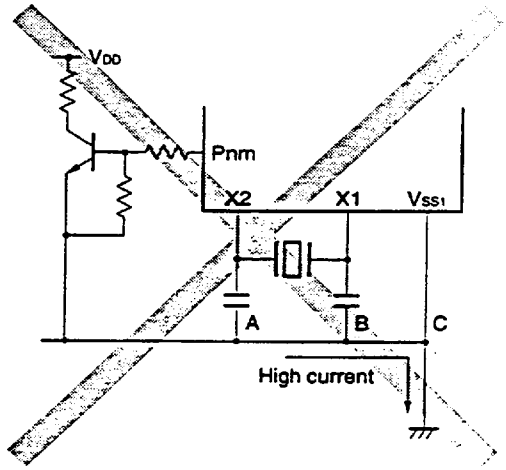
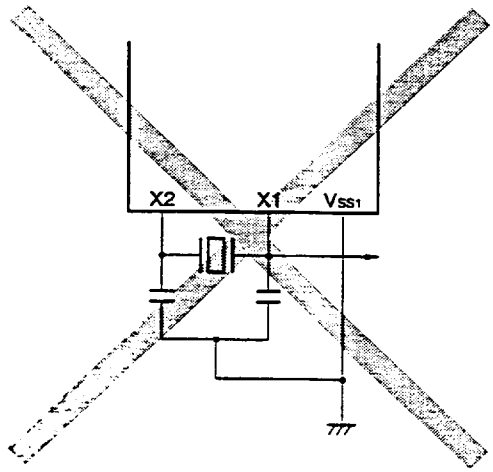


Figure 5-4. Incorrect Examples of Resonator Connection (2/2)

(e) Signal are fetched



5.4.2 Divider circuit

The divider circuit divides the output of the system clock oscillation circuit (fx) to generate various clocks.

5.5 Operation of Clock Generation Circuit

The clock generation circuit generates the following clocks and control the operation modes of the CPU, such as the standby mode:

- System clock fx
- CPU clock f_{CPU}
- Clock to peripheral hardware

The operation of the clock generation circuit is determined by the processor clock control register (PCC), as follows:

(a) The slowest mode (3.8 μs: at 8.38-MHz operation) of the system clock is selected when the $\overline{\text{RESET}}$ signal is generated (PCC = 04H). While a low level is input to the $\overline{\text{RESET}}$ pin, oscillation of the system clock is stopped.

(b) Five types of CPU clocks (0.24 μs, 0.48 μs, 0.96 μs, 1.9 μs, and 3.8 μs: at 8.38-MHz operation) can be selected by the PCC setting with the system clock selected.

(c) Two standby modes, STOP and HALT, can be used.

(d) The clock to the peripheral hardware is supplied by dividing the system clock. Therefore, the other peripheral hardware is stopped when the system clock is stopped (except, however, the external clock input operation).

5.6 Changing Setting of CPU Clock

5.6.1 Time required for switching CPU clock

The CPU clock can be selected by using bits 0 to 2 (PCC0 to PCC2) of the processor clock control register (PCC).

Actually, the specified clock is not selected immediately after the setting of PCC has been changed, and the old clock is used for the duration of several instructions after that (refer to **Table 5-2**).

Table 5-2. Maximum Time Required for Switching CPU Clock

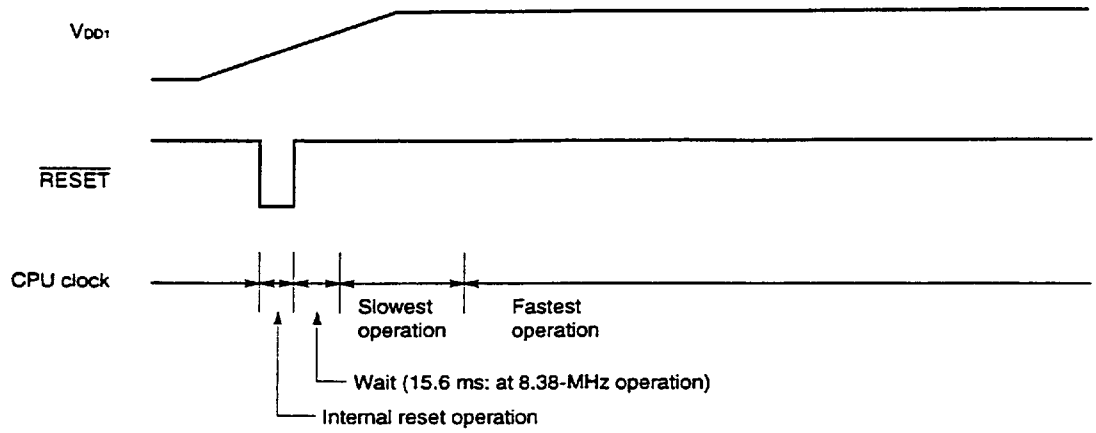
Set Value before Switching			Set Value after Switching													
PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0	PCC2	PCC1	PCC0
			0	0	0	0	0	1	0	1	0	0	1	1	1	0	0
0	0	0				16 instructions			16 instructions			16 instructions			16 instructions		
0	0	1	8 instructions						8 instructions			8 instructions			8 instructions		
0	1	0	4 instructions			4 instructions						4 instructions			4 instructions		
0	1	1	2 instructions			2 instructions			2 instructions						2 instructions		
1	0	0	1 instruction			1 instruction			1 instruction			1 instruction					

Remark One instruction is the minimum instruction execution time of the CPU clock before switching.

5.6.2 Switching CPU clock

The following figure illustrates how the CPU clock is switched.

Figure 5-5. Switching between System Clock and CPU Clock



<1> The CPU is reset when the $\overline{\text{RESET}}$ pin is made low on power application. The effect of resetting is released when the $\overline{\text{RESET}}$ pin is later made high, and the system clock starts oscillating. At this time, the time during which oscillation stabilizes (2^{17}/fx) is automatically secured.
After that, the CPU starts instruction execution at the slowest speed of the system clock (3.8 μs: at 8.38-MHz operation).

<2> After the time during which the V_{DD1} voltage rises to the level at which the CPU can operate at the highest speed has elapsed, processor clock control register (PCC) is rewritten so that the highest speed can be selected.

CHAPTER 6 REAL-TIME OUTPUT PORT

6.1 Real-Time Output Port Functions

Data set previously in the real-time output buffer register can be transferred to the output latch by hardware concurrently with timer interrupts or external interrupt generation, then output externally. This is called the real-time output function. The pins that output data externally are called real-time output ports.

By using a real-time output, a signal which has no jitter can be output. This port is therefore suitable for control of stepping motors, etc.

Port mode/real-time output port mode can be specified bit-wise.

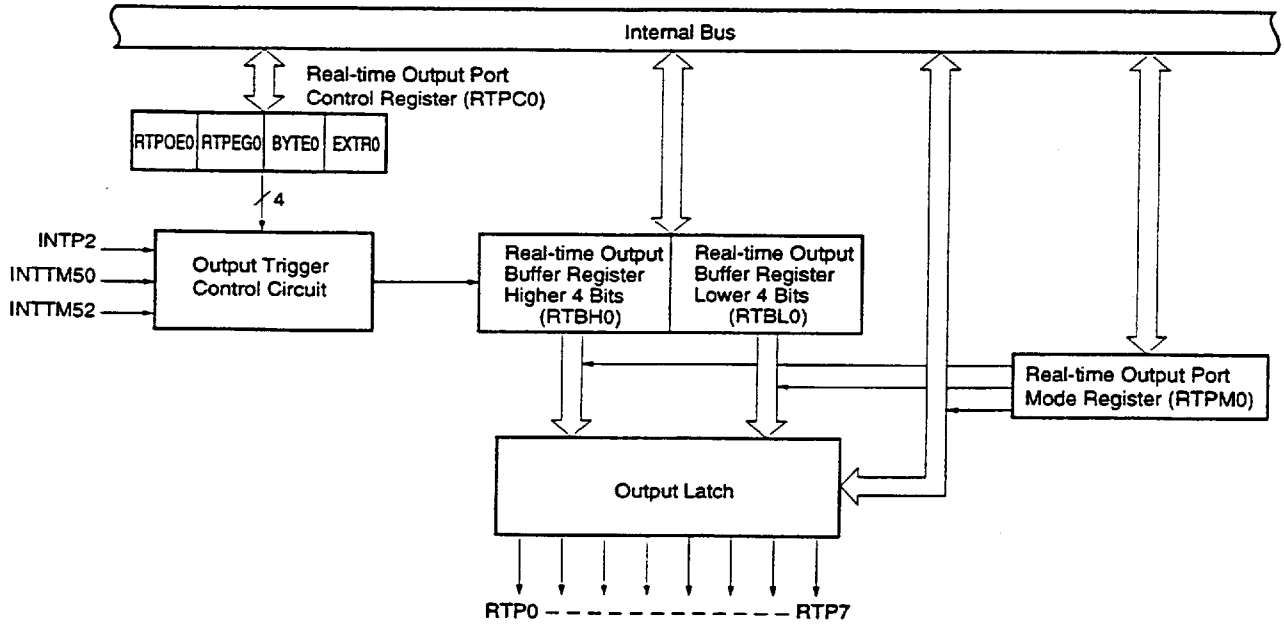
6.2 Real-Time Output Port Configuration

The real-time output port consists of the following hardware.

Table 6-1. Real-Time Output Port Configuration

Item	Configuration
Register	Real-time output buffer register (RTBL0, RTBH0)
Control register	Real-time output port mode register (RTPM0) Real-time output port control register (RTPC0)

Figure 6-1. Real-Time Output Port Block Diagram



(1) Real-time output buffer register (RTBL0, RTBH0)

These are 4-bit registers that hold output data in advance.

Addresses of RTBL0 and RTBH0 are mapped individually in the Special function register (SFR) area as shown in Figure 6-2.

When specifying 4 bits × 2 channels as the operating mode, data are set individually in RTBL0 and RTBH0. The data of both RTBL0 and RTBH0 can be read all at once regardless of which address is specified.

When specifying 8 bits × 1 channel as the operating mode, data are set to both RTBL0 and RTBH0 by writing 8-bit data to either RTBL0 or RTBH0. The data of both RTBL0 and RTBH0 can be read all at once regardless of which address is specified.

Table 6-2 shows operations during manipulation of RTBL0 and RTBH0.

Figure 6-2. Real-Time Output Buffer Register Configuration

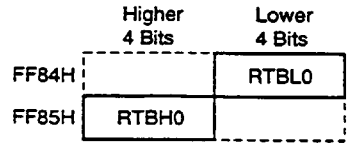


Table 6-2. Operation in Real-Time Output Buffer Register Manipulation

Operating Mode	Register to be Manipulated	In Read Note1		In Write Note2	
		Higher 4 Bits	Lower 4 Bits	Higher 4 Bits	Lower 4 Bits
4 Bits × 2 Channels	RTBL0	RTBH0	RTBL0	Invalid	RTBL0
	RTBH0	RTBH0	RTBL0	RTBH0	Invalid
8 Bits × 1 Channel	RTBL0	RTBH0	RTBL0	RTBH0	RTBL0
	RTBH0	RTBH0	RTBL0	RTBH0	RTBL0

Notes 1. Only the bits set in the real-time output port mode can be read. When a bit set in the port mode is read, 0 is read.

2. After setting data in the real-time output port, output data should be set in RTBL0 and RTBH0 by the time a real-time output trigger is generated.

6.3 Registers Controlling Real-Time Output Port

The following two types of registers control the real-time output ports.

- Real-time output port mode register (RTPM0)
- Real-time output port control register (RTPC0)

(1) Real-time output port mode register (RTPM0)

This register sets the real-time output port mode or port mode in 1-bit units.

RTPM0 is manipulated by a 1-bit or 8-bit memory manipulation instruction.

When the $\overline{\text{RESET}}$ signal is input, this register is set to 00H.

Figure 6-3. Format of Real-Time Output Port Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
RTPM0	RTPM07	RTPM06	RTPM05	RTPM04	RTPM03	RTPM02	RTPM01	RTPM00	FF86H	00H	R/W

RTPM0m	Real-Time Output Port Selection (m = 0 to 7)
0	Port mode
1	Real-time output port mode

Cautions 1. When using a real-time output port, set the port to be used as a real-time output port in the output mode.

2. The port specified as a real-time output port cannot set data to its output latch. To set an initial value, therefore, set data to the output latch before setting the real-time output port mode.

(2) Real-time output port control register (RTPC0)

This register sets an operating mode of real-time output port and output trigger.

The relation between the operating mode of the real-time output port and output trigger is as shown in Table 6-3.

RTPC0 is manipulated by a 1-bit or 8-bit memory manipulation instruction.

When the $\overline{\text{RESET}}$ signal is input, this register is set to 00H.

Figure 6-4. Format of Real-Time Output Port Control Register

Symbol	⑦	⑥	⑤	④	3	2	1	0	Address	On reset	R/W
RTPC0	RTPOE0	RTPEG0	BYTE0	EXTR0	0	0	0	0	FF87H	00H	R/W

RTPOE0	Controls operation of real-time output port.
0	Disables operation[Note].
1	Enables operation.

RTPEG0	Specifies valid edge of INTP2.
0	Falling edge
1	Rising edge

BYTE0	Operating mode of real-time output port
0	4 bits x 2 channels
1	8 bits x 1 channel

EXTR0	Controls real-time output by INTP2.
0	Does not use INTP2 as real-time output trigger.
1	Uses INTP2 as real-time output trigger.

Note When the real-time output operation is disabled (RTPOE0 = 0), RTP0 to RTP7 output "0".

Table 6-3. Real-Time Output Port Operating Mode and Output Trigger

BYTE0	EXTR0	Operating Mode	RTBH0 → Port Output	RTBL0 → Port Output
0	0	4 bits x 2 channels	INTTM52	INTTM50
0	1		INTTM50	INTP2
1	0	8 bits x 1 channel	INTTM50	
1	1		INTP2	

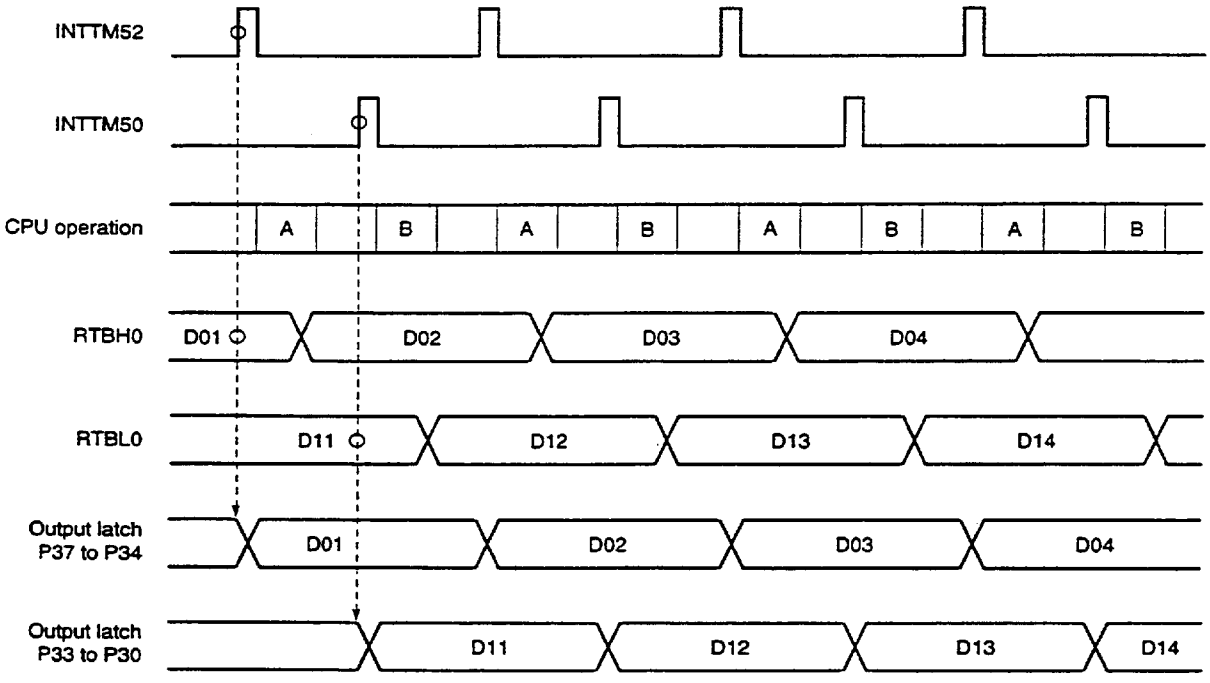
6.4 Operation of Real-Time Output Port

If the real-time output operation is enabled by setting bit 7 (RTPOE0) of the real-time output port control register (RTPC0) to 1, the data of the real-time output buffer registers (RTBH0 and RTBL0) are transferred to the output latch in synchronization with the generation of a transfer trigger (which is selected by EXTR0 and BYTE0[Note]). Of the transferred data, only the data of the bits specified as the real-time output port by the real-time output port mode register (RTPM0) are output from bits RTP0 to RTP7. The port pin set by RTPM0 in the port mode can be used as a general-purpose I/O port pin.

If the real-time output operation is disabled by RTPOE0 = 0, RTP0 to RTP7 output 0 regardless of the setting of RTPM0.

Note EXTR0: Bit 4 of real-time output port control register (RTPC0)
BYTE0: Bit 5 of real-time output port control register (RTPC0)

Figure 6-5. Example of Operation Timing of Real-Time Output Port (when EXTR0 = 0, BYTE0 = 0)



A: Software processing by INTTM52 (RTBH0 write)
B: Software processing by INTTM50 (RTBL0 write)

6.5 Using Real-Time Output Port

When using the real-time output port, perform the setting in the following procedure.

(1) Disable real-time output operation.
Clear bit 7 (RTPOE0) of the real-time output port control register (RTPC0) to 0.

(2) Initial setting
- Set the initial value to the output latch.
- Specify the real-time output port mode or port mode bitwise.
  Set the real-time output port mode register (RTPM0).
- Select a trigger and a valid edge.
  Set bits 4, 5, and 6 (EXTR0, BYTE0, and RTPEG0) of RTPC0.
- Set the same initial value as the output latch to the real-time output buffer registers (RTBH0 and RTBL0).

(3) Enable the real-time output operation.
RTPOE0 = 1

(4) Set the next output to RTBH0 and RTBL0 until the selected transfer trigger is generated.

(5) Sequentially set the next real-time output value to RTBH0 and RTBL0 by using the interrupt processing corresponding to the selected trigger.

6.6 Notes on Real-Time Output Port

(1) Before performing the initial setting, disable the real-time output operation by clearing bit 7 (RTPOE0) of the real-time output port control register (RTPC0) to 0.

(2) Once the real-time output operation has been disabled (RTPOE0 = 0), be sure to set the same initial value as the output latch to the real-time output buffer registers (RTBH0 and RTBL0) before enabling the real-time output operation (RTPOE0 = 0 -> 1).

[MEMO]

PAGE(S) INTENTIONALLY BLANK

CHAPTER 7 10-BIT INVERTER CONTROL TIMER

7.1 Functions of 10-Bit Inverter Control Timer

The 10-bit inverter control timer (TM7) realizes inverter control. It incorporates an 8-bit timer for dead time generation and can output waveforms that don't overlap active levels. A total of six positive phase and negative phase channels are output. In addition, an active level change function and output stop function by external interrupt or watchdog timer interrupt are provided.

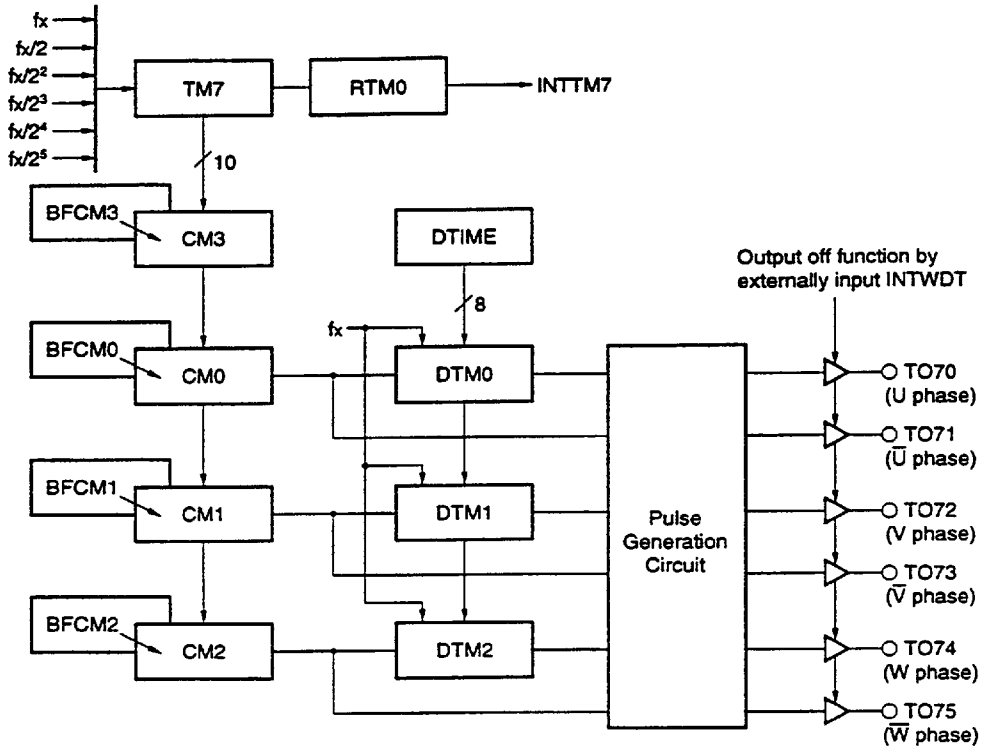
7.2 Configuration of 10-Bit Inverter Control Timer

The 10-bit inverter control timer (TM7) consists of the following hardware.

Table 7-1. Configuration of 10-Bit Inverter Control Timer

Item	Function
Timer counter	10-bit up/down counter (TM7) x 1 8-bit down counters (DTM6, DTM1, DTM2) x 3 3-bit down counter (RTM0) x 1
Register	10-bit compare registers (CM0, CM1, CM2, CM3) x 4 10-bit buffer registers (BFCM0, BFCM1, BFCM2, BFCM3) x 4 8-bit reload register (DTIME) x 1
Timer output	6 (TO70, TO71, TO72, TO73, TO74, TO75)
Control register	Inverter timer control register 7 (TMC7) Inverter timer mode register 7 (TMM7)

Figure 7-1. Block Diagram of 10-Bit Inverter Control Timer



(1) 10-bit inverter control timer (TM7)

TM7 is a 10-bit up/down counter that counts count pulses in synchronization with the rising edge of the count clock. When the timer starts, the number of count pulse count is incremented from 0, while when the value of the preset compare register 3 (CM3) and the TM7 counting match, it is switched to the count down operation.

An underflow signal is generated if the value becomes 000H during the count down operation and an interrupt request signal INTTM7 is generated. When underflow occurs, it is switched from the count down operation to the count up operation. INTTM7 is normally generated at every underflow but the number of occurrences can be divided by the IDEV0 to IDEV2 bits of inverter timer control register 7 (TMC7).

TM7 cannot be read/written.

The cycle of TM7 is controlled by CM3.

The count clock can be selected among 6 types: fx, fx/2, fx/4, fx/8, fx/16, fx/32.

$\overline{\text{RESET}}$ input or clearing the CE7 bit of the TMC7 register sets TM7 to 000H.

(2) 10-bit compare registers 0 to 2 (CM0 to CM2)

CM0 to CM2 are 10-bit compare registers that always compare their own value with that of TM7, and if they coincide the contents of the flip-flops are changed.

Each of CM0 to CM2 are provided with a buffer register (BFCM0 to BFCM2), so that the contents of the buffer can be transferred to CM0 to CM2 with the timing of interrupt request signal INTTM7 generation.

Write operation to CM0 to CM2 is possible only during TM7 stop.

To set the output timing, perform data write to BFCM0 to BFCM3.

$\overline{\text{RESET}}$ input or clearing the CE7 bit of the TMC7 register sets these registers to 000H.

(3) 10-bit compare register 3 (CM3)

CM3 is a 10-bit compare register that controls the high limit value of TM7. If the count value of TM7 coincides with the value CM3 or 0, count up/down is switched at the next count clock.

CM3 provides a buffer register (BFCM3) whose contents are transferred to CM3 with the timing of interrupt request signal INTTM7 generation.

CM3 can be written to only during TM7 stop.

To set the cycle to TM7, write data to BFCM3.

$\overline{\text{RESET}}$ input sets CM3 to 0FFH.

Do not set CM3 to 000H.

(4) 10-bit buffer registers 0 to 3 (BFCM0 to BFCM3)

BFCM0 to BFCM3 are 10-bit registers. They transfer data to the compare register (CM0 to CM3) corresponding to each buffer with the timing of the interrupt request signal INTTM7 generation.

BFCM0 to BFCM3 can be read/written irrespective of TM7 count stop or operation.

$\overline{\text{RESET}}$ input sets BFCM0 to BFCM2 and BFCM3, to 000H and 0FFH, respectively.

These registers can be read/written in word and byte units. For read/write operations of less than 8 bits, BFCM0L to BFCM3L are used.

(5) Dead time reload register (DTIME)

DTIME is an 8-bit register to set dead time and is common to three dead time timers (DTM0 to DTM2). However, the data load timing from DTIME to DTM0, DTM1 and DTM2 is independent.

DTIME can be written only during TM7 count stop. Data does not change even if an instruction to rewrite DTIME is executed during timer operation.

$\overline{\text{RESET}}$ input sets DTIME to FFH.

Even if 00H is set to DTIME, an output with dead time fx is performed.

(6) Dead time timers 0 to 2 (DTM0 to DTM2)

DTM0 to DTM2 are 8-bit down counters that generate dead time.

Count down is performed after the value of the dead time reload register (DTIME) is reloaded with the timing of compare match between CM0 to CM2 and TM7. DTM0 to DTM2 generate an underflow signal when 00H changes to FFH and stop with FFH.

The count clock is fx.

DTM0 to DTM2 cannot be read/written.

$\overline{\text{RESET}}$ input or clearing the CE7 bit of the TMC7 register sets these registers to FFH.

(7) Buffer transfer control timer (RTM0)

RTM0 is a 3-bit up counter. It has the function to divide interrupt request signal INTTM7.

The number of division that is set with bits IDEV2 to IDEV0 of the TCM7 register is loaded at TM7 count start and at the generation timing of INTTM7. Incrementing is performed with the TM7 underflow signal and INTTM7 is generated when the value matches with the loaded value.

RTM0 cannot be read/written.

$\overline{\text{RESET}}$ input sets RTM0 to 7H. The division number that is set with bits IDEV2 to IDEV0 is set by generating INTTM7 and clearing the CE7 bit of the TMC7 register.

7.3 Registers Controlling 10-bit Inverter Control Timer

The following two types of registers are available to control the 10-bit inverter control timer.

- Inverter timer control register 7 (TMC7)
- Inveter timer mode register 7 (TMM7)

(1) Inverter timer control register 7 (TMC7)

TMC7 controls the operation of TM7, DTM0 to DTM2, and RTM0, specifies the count clock of TM7, and selects the compare register transfer cycle.

TMC7 is set by a 1-bit or 8-bit memory manipulation instruction.

$\overline{\text{RESET}}$ input sets TMC7 to 00H.

Figure 7-2. Inverter Timer Control Register 7 Format

Symbol	⑦	6	5	4	3	2	1	0	Address	On reset	R/W
TMC7	CE7	0	TCL72	TCL71	TCL70	IDEV2	IDEV1	IDEV0	FF90H	00H	R/W

CE7	TM7, DTM0 to DTM2, RTM0 operation control
0	Clear and stop (TO70 to TO75 are Hi-z)
1	Count enable

TCL72	TCL71	TCL70	Count clock selection
0	0	0	fx (8.38 MHz)
0	0	1	fx/2 (4.19 MHz)
0	1	0	$fx/2^2$ (2.1 MHz)
0	1	1	$fx/2^3$ (1.05 MHz)
1	0	0	$fx/2^4$ (524 kHz)
1	0	1	$fx/2^5$ (262 kHz)
Other than the above			Setting prohibited

IDEV2	IDEV1	IDEV0	INTTM7 occurrence frequency selection
0	0	0	Occurs once every TM7 underflow.
0	0	1	Occurs once every two TM7 underflows.
0	1	0	Occurs once every three TM7 underflows.
0	1	1	Occurs once every four TM7 underflows.
1	0	0	Occurs once every five TM7 underflows.
1	0	1	Occurs once every six TM7 underflows.
1	1	0	Occurs once every seven TM7 underflows.
1	1	1	Occurs once every eight TM7 underflows.

Remark Figures in parentheses apply to operation with fx = 8.38 MHz.

(2) Inverter timer mode register 7 (TMM7)

TMM7 controls the operation of and specifies active level of TO70 to TO75 outputs, and sets the valid edge of TOFF7.

TMM7 is set by a 1-bit or 8-bit memory manipulation instruction. $\overline{\text{RESET}}$ input sets TMM7 to 00H.

Figure 7-3. Inverter Timer Mode Register 7 Format

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
TMM7	0	0	0	PNOFFB Note	ALV	TOEDG	TOSPP	TOSPW	FF91H	00H	R/W

PNOFFB Note	Control status flag of TM7 output to TO70 to TO75
0	TM7 output disable status (TO70 to TO75 are Hi-z)
1	TM7 output enable status

ALV	TO70 to TO75 output active level specification
0	Low level
1	High level

TOEDG	TOFF7 valid edge specification
0	Falling edge
1	Rising edge

TOSPP	TO70 to TO75 output stop control by valid edge
0	Does not stop output
1	Stops output (TO70 to TO75 are Hi-z)

TOSPW	TO70 to TO75 output stop control by INTWDT
0	Does not stop output
1	Stops output (TO70 to TO75 are Hi-z)

Note The PNOFFB bit is a read-only flag. This bit cannot be set or reset by software. PNOFFB is reset when an output stop is caused by TOFF7 and INTWDT during TM7 stop (CE7 = 0) or operation (CE7 = 1).

Caution Always set bits 5 to 7 of TMM7 to 0.

Remarks 1. TO70 to TO75 become Hi-z state in the following cases. However, the TM7, DTM0 to DTM2, and RTM0 timers do not stop if CE7 = 1 is set.

- A valid edge is input to the TOFF7 pin while TOSPP = 1.
- A specified interrupt is generated while TOSPW = 1.

To restore the output of TO70 to TO75, perform the procedure below.

<1> Write 0 to CE7 and stop the timer.
<2> Write 0 to the output stop function flag that is used.
<3> Reset the registers to their default values.

2. PNOFFB, ALV, CE7, and TO70 to TO75 are related as follows.

PNOFFB	ALV	CE7	TO70, TO72, TO74	TO71, TO73, TO75
0	0	0	Hi-z	Hi-z
0	1	0	Hi-z	Hi-z
0	0/1	1	Hi-z	Hi-z
1	0/1	1	PWM wave output	PWM wave output

7.4 Operation of 10-Bit Inverter Control Timer

(1) Setting procedure

(a) The TM7 count clock is set with the TCL70 to TCL72 bits of inverter timer control register 7 (TMC7) and the occurrence frequency of INTTM7 is set with the IDEV0 to IDEV2 bits.

(b) The active level of the TO70 to TO75 pins is set with the ALV bit of inverter timer mode register 7 (TMM7).

(c) Set the half width of the first PWM cycle to CM3.
- PWM cycle = CM3 value × 2 × TM7 clock rate
  (The clock rate of TM7 is set with the TMC7 register)

(d) Set the half width of the second PWM cycle to BFCM3.

(e) Set the dead time width to DTIME.
- Dead time width = (DTIME + 1) × fx
  fx: internal system clock

(f) Set F/F set/reset timing that is used during the first cycle to CM0 to CM2.

(g) Set F/F set/reset timing that is used during the second cycle to BFCM3.

(h) After the CE7 bit of the TMC7 register is set (1), the operation of TM7, DTM0 to DTM2, and RTM0 is enabled.

Caution Always use a bit manipulation instruction to set the CE7 bit.

(i) Set the F/F set/reset timing that is used for the next cycle to BFCM0 to BFCM3 during TM7 operation.

(j) To stop the TM7 operation, set the CE7 bit of the TMC7 register to 0.

Caution Another bit cannot be rewritten at the same time that the CE7 bit is being rewritten.

(2) Output waveform widths corresponding to set values

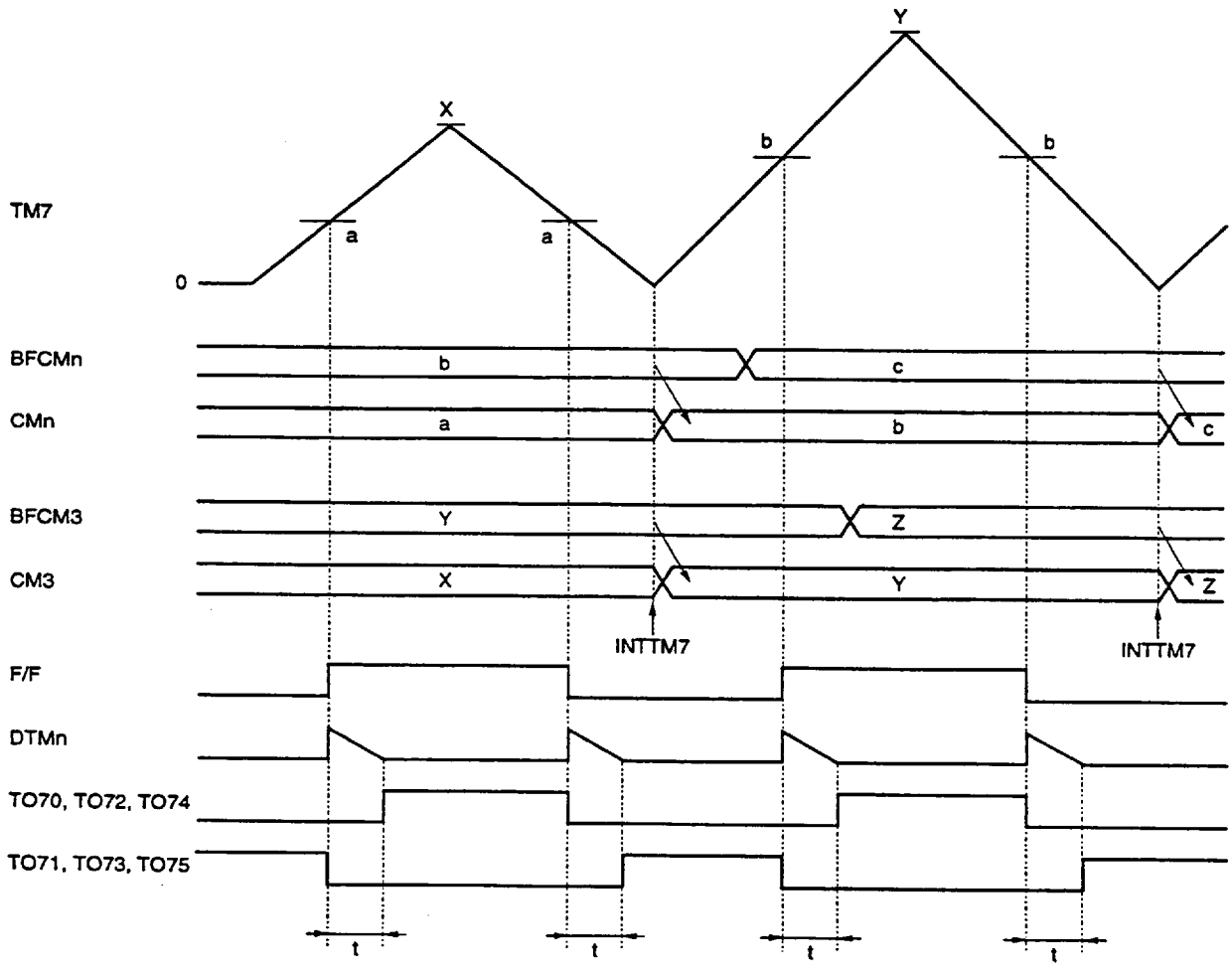
- PWM cycle = CM3 × 2 × T_{TM7}
- Dead time width = T_{DTM} = (DTIME + 1) × fx
- Active width of positive phase (TO70, TO72, TO74 pin)
  = {(CM3 − CM_{up}) + (CM3 − CM_{down})} × T_{TM7} − T_{DTM}
- Active width of negative phase (TO71, TO73, TO75 pin)
  = (CM_{down} + CM_{up}) × T_{TM7} − T_{DTM}

fx : System clock oscillation frequency
T_{TM7} : TM7 count clock
CM_{up} : Set value of CM0 to CM2 during TM7 count up
CM_{down} : Set value of CM0 to CM2 during TM7 count down

Caution If a value whose active width in the positive phase or negative phase becomes 0 or negative via the above calculation, TO70 to TO75 output a waveform fixed at inactive level during active width 0.

(3) Operation timing

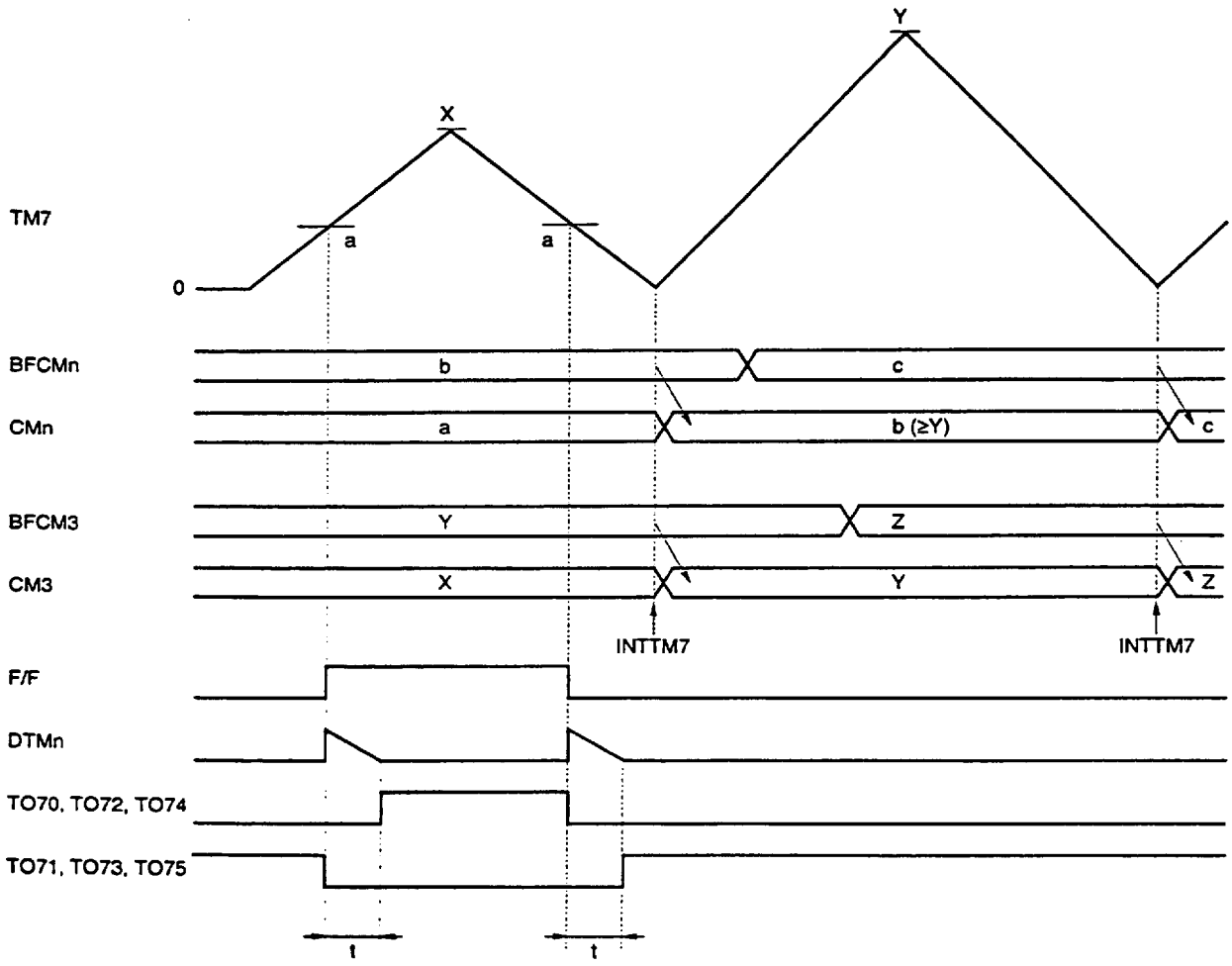
Figure 7-4. TM7 Operation Timing (Basic Operation)



Remarks 1. n = 0 to 2

2. t: Dead time = (DTIME + 1) × fx
   (fx: System clock oscillation frequency)
3. The above figure assumes an active high and undivided INTTM7 occurrence.

Figure 7-5. TM7 Operation Timing (CMn (BFCMn) ≥ CM3 (BFCM3))

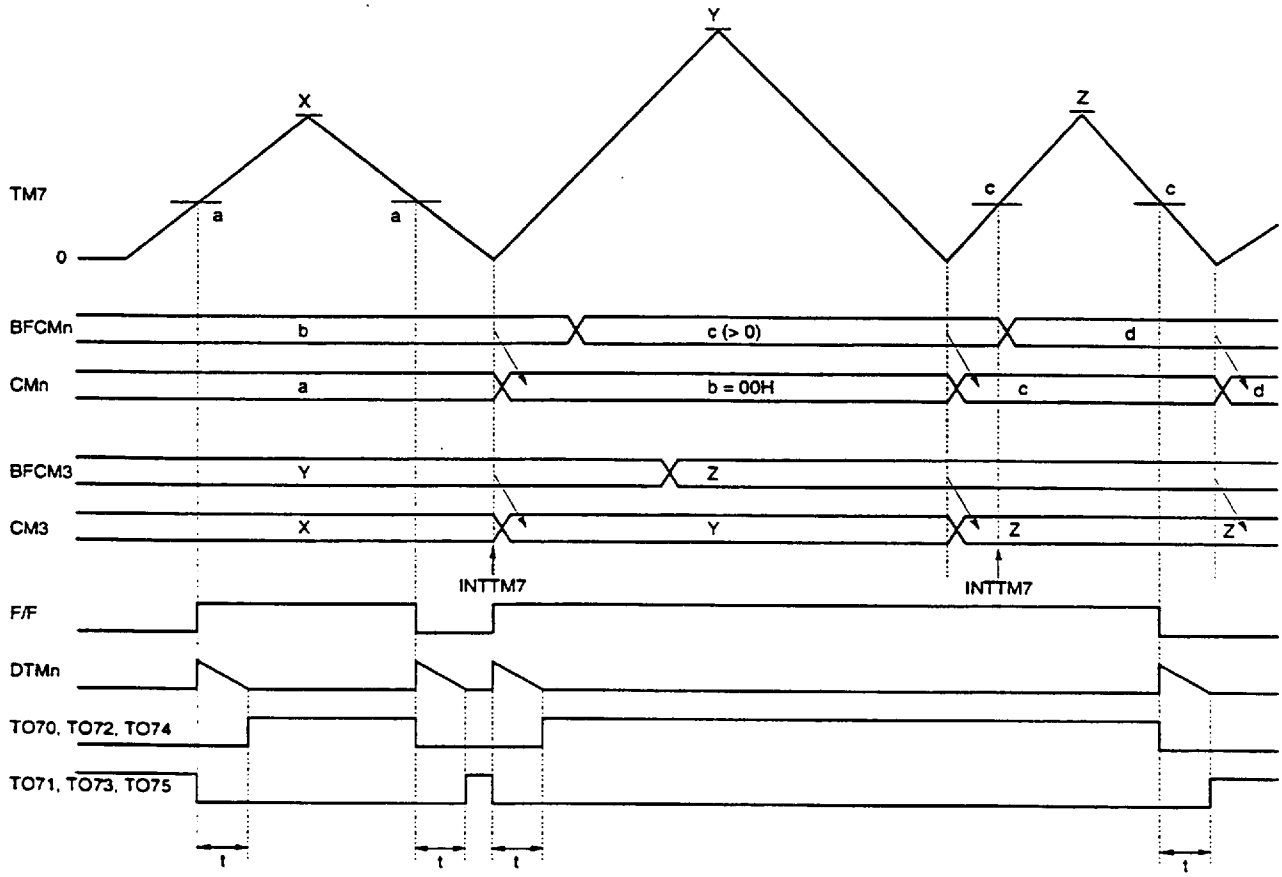


Remarks 1. n = 0 to 2

2. t: Dead time = (DTIME + 1) × fx
   (fx: System clock oscillation frequency)
3. The above figure assumes an active high and undivided INTTM7 occurrence.

If a value higher than CM3 is set to BFCMn, high-level output in the positive phases (TO70, TO72, TO74 pins), and low-level output in the negative phases (TO71, TO73, TO75 pins) are continued. This setting is effective to output signals whose low and high widths are longer than the PWM cycle when controlling the inverter, etc.

Figure 7-6. TM7 Operation Timing (CMn (BFCMn) = 000H)

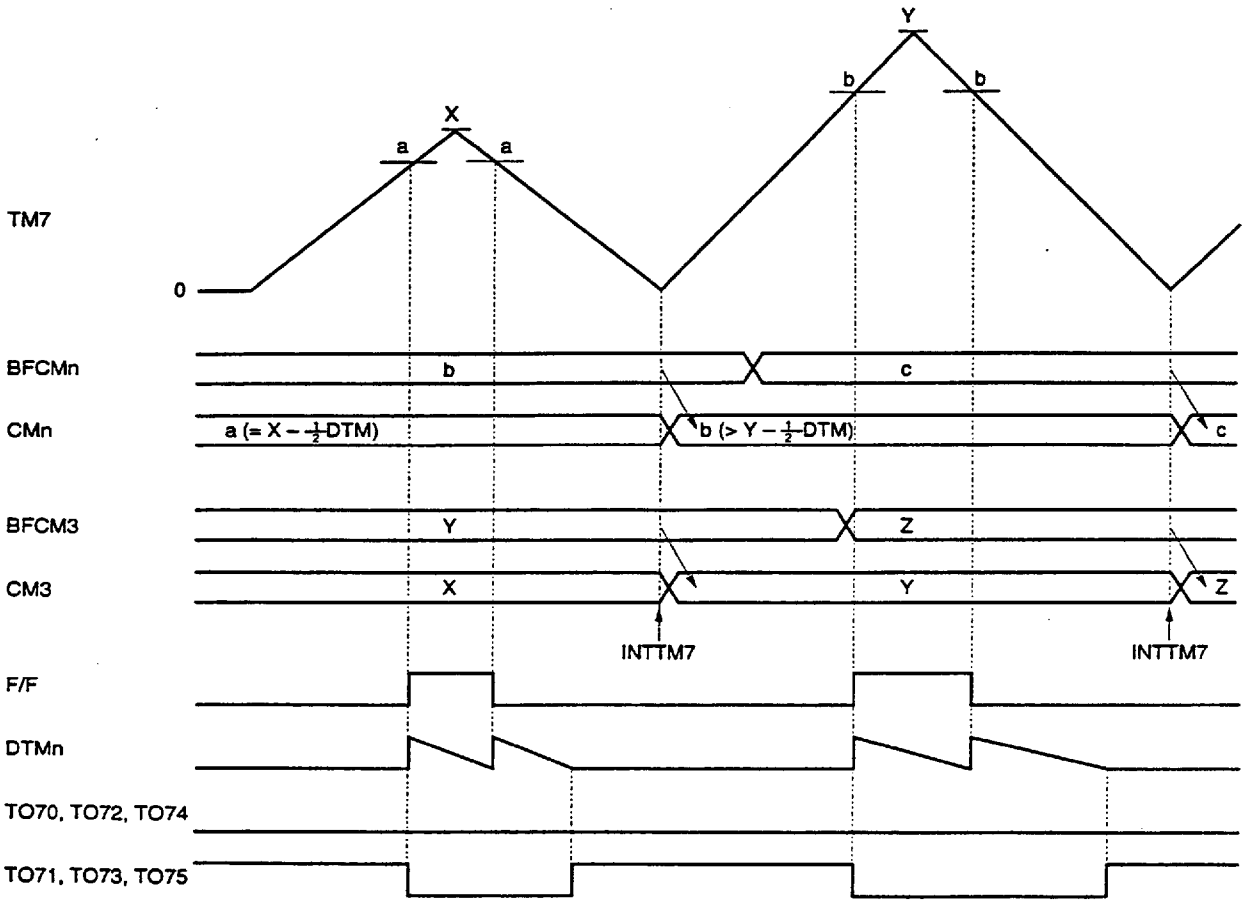


Remarks 1. n = 0 to 2

2. t: Dead time = (DTIME + 1) × fx
   (fx: System clock oscillation frequency)

3. The above figure assumes an active high and undivided INTTM7 occurrence.

Figure 7-7. TM7 Operation Timing (CMn (BFCMn) = CM3 – 1/2DTM, CMn (BFCMn) > CM3 – 1/2DTM)



Remarks 1. n = 0 to 2

2. The above figure assumes an active high and undivided INTTM7 occurrence.

CHAPTER 8 8-BIT TIMER/EVENT COUNTER

8.1 Function of 8-Bit Timer/Event Counter

The 8-bit timer/event counters (TM50, TM51, and TM52) have the following two modes:

- Mode in which an 8-bit timer/event counter is used alone (single mode)
- Mode in which two or more 8-bit timer/event counters are connected in cascade (16-bit resolution: cascade mode)

These two modes are explained below.

(1) Mode in which an 8-bit timer/event counter is used alone (single mode)
In this mode, the following functions can be used.

- Interval timer
- External event counter
- Square wave output
- PWM output

(2) Mode in which TM50 and TM51, or TM51 and TM52 are connected in cascade (16-bit resolution: cascade mode)
By connecting 8-bit timer/event counters in cascade, they can be used as a 16-bit timer/event counter. In the cascade mode, the following functions can be used.

- 16-bit resolution interval timer
- 16-bit resolution external event counter
- 16-bit resolution square wave output

Figures 8-1 to 8-3 show the block diagrams of 8-bit timer/event counters.

8.2 Configuration of 8-Bit Timer/Event Counter

An 8-bit timer/event counter consists of the following hardware:

Table 8-1. Configuration of 8-Bit Timer/Event Counter

Item	Configuration
Timer register	8-bit counter 5n (TM5n)
Register	8-bit compare register 5n (CR5n)
Timer output	TO5n
Control register	8-bit timer mode control register 5n (TMC5n) Timer clock select register 5n (TCL5n)

Remark n = 0 to 2

Figure 8-1. 8-Bit Timer/Event Counter 50 (TM50) Block Diagram

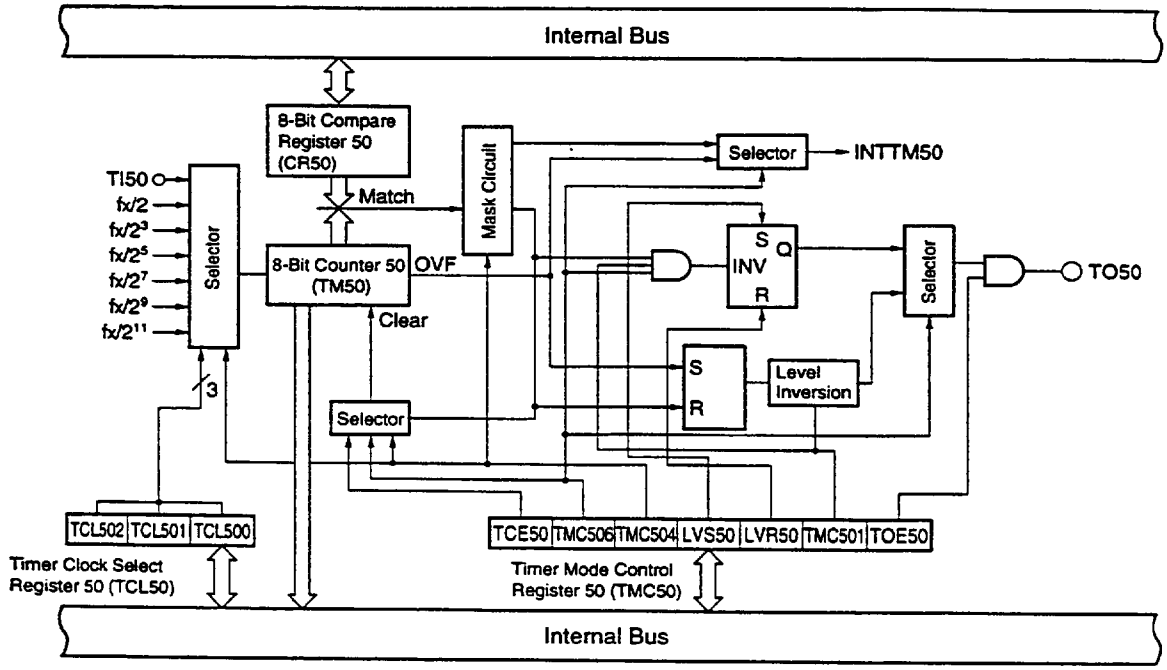


Figure 8-2. 8-Bit Timer/Event Counter 51 (TM51) Block Diagram

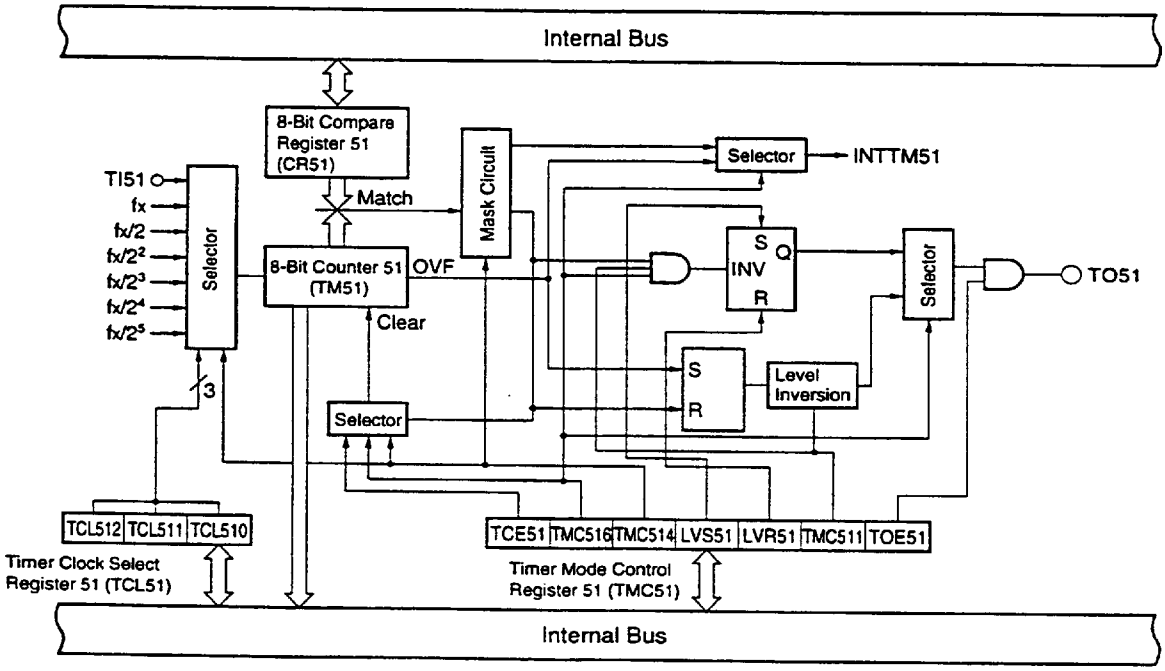
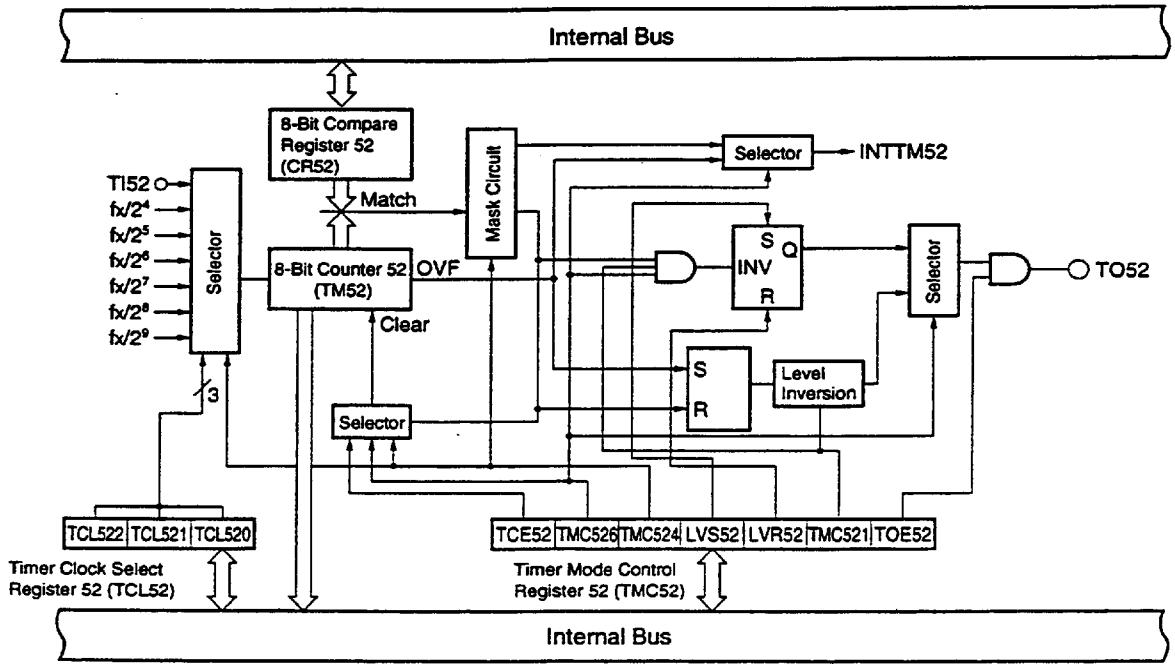


Figure 8-3. 8-Bit Timer/Event Counter 52 (TM52) Block Diagram



(1) 8-bit timer counters 50, 51, and 52 (TM50, TM51, and TM52)

TM50, TM51, and TM52 are 8-bit read-only registers that count count pulses.

These counters are incremented in synchronization with the rising edge of the count clock.

TM50 and TM51, or TM51 and TM52 can be connected in cascade and can be used as a 16-bit timer.

When TM50 and TM51 are connected in cascade and used as a 16-bit timer, the values of these timers/counters can be read by using a 16-bit manipulation instruction. TM50 and TM51 are connected with an internal 8-bit bus, and TM50 and TM51 are read one at a time. This means that the value of, say, TM50 may change while that of TM51 is read. Therefore, read TM50 and TM51 two times to compare their first and second values for the sake of accuracy.

When TM51 and TM52 are connected in cascade and used as a 16-bit timer, they cannot be read by using a 16-bit manipulation instruction. When reading the TM51 and TM52, read the TM51 and TM52 separately by using an 8-bit manipulation instruction.

If the count value is read during operation, input of the count clock is temporarily stopped, and the count value at that point is read. The count value is cleared to 00H in the following cases:

<1> $\overline{\text{RESET}}$ input

<2> Clearing TCE5n

<3> Coincidence between TM5n and CR5n in clear & start mode

Caution In the cascade mode, the 16-bit timer is cleared to 00H regardless of whether TCE51 of TM51 or TCE52 of TM52 is cleared.

Remark n = 0 to 2

(2) 8-bit compare registers 50, 51, and 52 (CR50, CR51, and CR52)

The value set to CR5n is always compared with the count value of 8-bit timer counter 5n (TM5n). When the value of the compare register coincides with the value of the timer counter, an interrupt request (INTTM5n) is generated (in a mode other than the PWM mode).

The value of CR5n can be set in a range of 00H to FFH and can be rewritten during counting.

If TM50 and TM51 are connected in cascade and used as a 16-bit timer, CR50 and CR51 operate as a 16-bit compare register. Therefore, the count value and register value are compared in 16-bit units, and if the two values coincide with each other, an interrupt request (INTTM50) is generated. At this time, INTTM51 interrupt request is also generated. When connecting TM50 and TM51 in cascade, therefore, mask the INTTM51 interrupt request.

The same applies when TM51 and TM52 are connected in cascade. If the value of the 16-bit timer coincides with that of the 16-bit compare register, the INTTM51 interrupt request is generated (mask the INTTM52 interrupt request).

Caution When setting data to the timers connected in cascade, be sure to stop the timer operation.

Remark n = 0 to 2

8.3 Registers Controlling 8-Bit Timer/Event Counters

The following six registers control 8-bit timer/event counters 50, 51, and 52.

- 8-bit timer mode control registers 50, 51, and 52 (TMC50, TMC51, and TMC52)
- Timer clock select registers 50, 51, and 52 (TCL50, TCL51, and TCL52)

(1) 8-bit timer mode control registers 50, 51, and 52 (TMC50, TMC51, and TMC52)

TMC50, TMC51, and TMC52 perform the following six operations:

<1> Control of count operation of 8-bit timer counters 50, 51, and 52 (TM50, TM51, and TM52)
<2> Selection of operation mode of 8-bit timer counters 50, 51, and 52 (TM50, TM51, and TM52)
<3> Selection of single mode or cascade mode
<4> Setting of status of timer output F/F (flip-flop)
<5> Control of timer F/F or selection of active level in PWM (free running) mode
<6> Control of timer output

TMC50, TMC51, and TMC52 are set by using a 1-bit or 8-bit memory manipulation instruction.
The values of these registers are set to 04H at $\overline{\text{RESET}}$.
Figures 8-4 to 8-6 show the formats of TMC50, TMC51, and TMC52.

Figure 8-4. Format of 8-Bit Timer Mode Control Register 50

Symbol	⑦	6	5	4	③	②	1	⓪	Address	On reset	R/W
TMC50	TCE50	TMC506	0	0	LVS50	LVR50	TMC501	TOE50	FF68H	04H	R/W

TCE50	Controls count operation of TM50.
0	Disables count operation after clearing counter to 0 (disables prescaler).
1	Starts counting.

TMC506	Selects operating mode of TM50.
0	Clears and starts on coincidence between TM50 and CR50.
1	PWM (free running) mode

LVS50	LVR50	Sets status of timer output F/F.
0	0	Not affected
0	1	Resets timer output F/F (to 0).
1	0	Sets timer output F/F (to 1).
1	1	Setting prohibited

TMC501	Other than PWM mode (TMC506 = 0)	PWM mode (TMC506 = 1)
	Controls timer F/F.	Selects active level.
0	Disables inverted operation.	High active
1	Enables inverted operation.	Low active

TOE50	Controls timer output.
0	Disables output (port mode).
1	Enables output.

Remarks 1. The PWM output is at the inactive level in the PWM mode because TCE50 = 0.

2. If LVS50 and LVR50 are read immediately after data has been set, these bits are 0.

Figure 8-5. Format of 8-Bit Timer Mode Control Register 51

Symbol	⑦	6	5	4	③	②	1	⓪
TMC51	TCE51	TMC516	0	TMC514	LVS51	LVR51	TMC511	TOE51

Address	On reset	R/W
FF70H	04H	R/W

TCE51	Controls count operation of TM51.
0	Disables count operation after clearing counter to 0 (disables prescaler).
1	Starts counting.

TMC516	Selects operating mode of TM51.
0	Clears and starts on coincidence between TM51 and CR51.
1	PWM (free running) mode

TMC514	Selects single mode or cascade mode.
0	Single mode
1	Cascade mode (connected to TM50)

LVS51	LVR51	Sets status of timer output F/F.
0	0	Not affected
0	1	Resets timer output F/F (to 0).
1	0	Sets timer output F/F (to 1).
1	1	Setting prohibited

TMC511	Other than PWM mode (TMC516 = 0)	PWM mode (TMC516 = 1)
	Controls timer F/F.	Selects active level.
0	Disables inverted operation.	High active
1	Enables inverted operation.	Low active

TOE51	Controls timer output.
0	Disables output (port mode).
1	Enables output.

Remarks 1. The PWM output is at the inactive level in the PWM mode because TCE51 = 0.

2. If LVS51 and LVR51 are read immediately after data has been set, these bits are 0.

Figure 8-6. Format of 8-Bit Timer Mode Control Register 52

Symbol	7	6	5	4	3	2	1	0
TMC52	TCE52	TMC526	0	TMC524	LVS52	LVR52	TMC521	TOE52

Address	On reset	R/W
FF78H	04H	R/W

TCE52	Controls count operation of TM52.
0	Disables count operation after clearing counter to 0 (disables prescaler).
1	Starts counting.

TMC526	Selects operating mode of TM52.
0	Clears and starts on coincidence between TM52 and CR52.
1	PWM (free running) mode

TMC524	Selects single mode or cascade mode.
0	Single mode
1	Cascade mode (connected to TM51)

LVS52	LVR52	Sets status of timer output F/F.
0	0	Not affected
0	1	Resets timer output F/F (to 0).
1	0	Sets timer output F/F (to 1).
1	1	Setting prohibited

TMC521	Other than PWM mode (TMC526 = 0)	PWM mode (TMC526 = 1)
	Controls timer F/F.	Selects active level.
0	Disables inverted operation.	High active
1	Enables inverted operation.	Low active

TOE52	Controls timer output.
0	Disables output (port mode).
1	Enables output.

Remarks 1. The PWM output is at the inactive level in the PWM mode because TCE52 = 0.

2. If LVS52 and LVR52 are read immediately after data has been set, these bits are 0.

(2) Timer clock select registers 50, 51, and 52 (TCL50, TCL51, and TCL52)

These registers specify the count clock of the 8-bit timer counters 50, 51, and 52 (TM50, TM51, and TM52) and the valid edges of TI50, TI51, and TI52 inputs.

TCL50, TCL51, and TCL52 are set by using an 8-bit memory manipulation instruction.

When the $\overline{\text{RESET}}$ signal is input, these registers are set to 00H.

Figures 8-7 to 8-9 show the formats of TCL50, TCL51, and TCL52.

Figure 8-7. Format of Timer Clock Select Register 50

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
TCL50	0	0	0	0	0	TCL502	TCL501	TCL500	FF69H	00H	R/W

TCL502	TCL501	TCL500	Selects count clock.
0	0	0	Falling edge of TI50
0	0	1	Rising edge of TI50
0	1	0	fx/2 (4.19 MHz)
0	1	1	$fx/2^3$ (1.05 MHz)
1	0	0	$fx/2^5$ (262 kHz)
1	0	1	$fx/2^7$ (65.5 kHz)
1	1	0	$fx/2^9$ (16.4 kHz)
1	1	1	$fx/2^{11}$ (4.09 kHz)

Cautions 1. Before rewriting the data of TCL50, stop the timer operation once.

2. Be sure to clear bits 3 to 7 of TCL50 to 0.

Remark ( ): at fx = 8.38-MHz operation

Figure 8-8. Format of Timer Clock Select Register 51

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
TCL51	0	0	0	0	0	TCL512	TCL511	TCL510	FF71H	00H	R/W

TCL512	TCL511	TCL510	Selects count clock.
0	0	0	Falling edge of TI51
0	0	1	Rising edge of TI51
0	1	0	fx (8.38 MHz)
0	1	1	fx/2 (4.19 MHz)
1	0	0	$fx/2^2$ (2.1 MHz)
1	0	1	$fx/2^3$ (1.05 MHz)
1	1	0	$fx/2^4$ (524 kHz)
1	1	1	$fx/2^5$ (262 kHz)

Cautions 1. Before rewriting the data of TCL51, stop the timer operation once.
2. Be sure to clear bits 3 to 7 of TCL51 to 0.

Remark ( ): at fx = 8.38-MHz operation

Figure 8-9. Format of Timer Clock Select Register 52

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
TCL52	0	0	0	0	0	TCL522	TCL521	TCL520	FF79H	00H	R/W

TCL522	TCL521	TCL520	Selects count clock.
0	0	0	Falling edge of TI52
0	0	1	Rising edge of TI52
0	1	0	$fx/2^4$ (524 kHz)
0	1	1	$fx/2^5$ (262 kHz)
1	0	0	$fx/2^6$ (131 kHz)
1	0	1	$fx/2^7$ (65.5 kHz)
1	1	0	$fx/2^8$ (32.7 kHz)
1	1	1	$fx/2^9$ (16.4 kHz)

Cautions 1. Before rewriting the data of TCL52, stop the timer operation once.
2. Be sure to clear bits 3 to 7 of TCL52 to 0.

Remark ( ): at fx = 8.38-MHz operation

8.4 Operation of 8-Bit Timer/Event Counter

8.4.1 Operation as interval timer (8-bit operation)

The 8-bit timer/event counters operate as interval timers that repeatedly generate an interrupt at time intervals specified by the count values set to the corresponding 8-bit compare registers (CR5n) in advance.

When the count values of the 8-bit timer counter (TM5n) coincide with the values set to the corresponding compare register CR5n, the value of TM5n is cleared to 0, TM5n continues counting, and at the same time, interrupt request signal (INTTM5n) is generated.

The count clock of the TM5n can be selected by bits 0 to 2 (TCL5n to TCL5n2) of the timer clock select register (TCL5n).

<Setting>

<1> Set each register.
- TCL5n: Selects count clock.
- CR5n: Compare value
- TMC5n: Selects clear and start mode on coincidence between TM5n and CR5n (TMC5n = 0000xxx0B x = don't care).

<2> The count operation is started when TCE5n is set to 1.

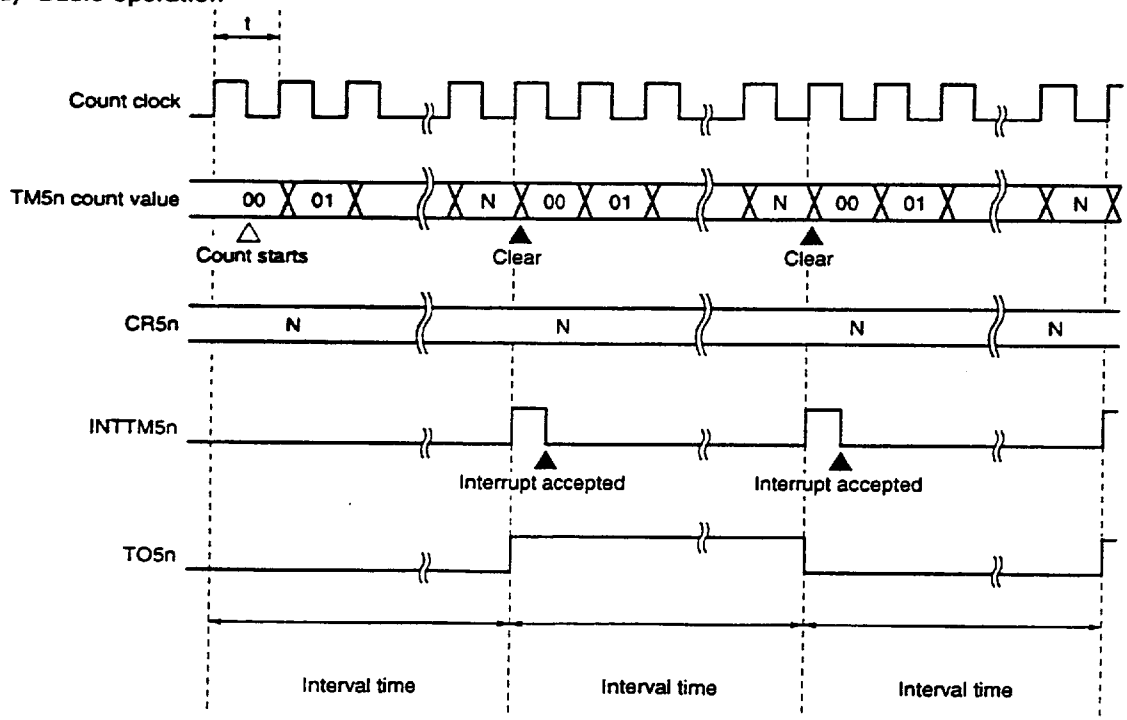
<3> INTTM5n occurs when the values of TM5n and CR5n coincide (TM5n is cleared to 00H).

<4> After that, INTTM5n repeatedly occurs at the same interval. To stop the count operation, clear TCE5n to 0.

Remark n = 0 to 2

Figure 8-10. Interval Timer Operation Timing (1/3)

(a) Basic operation

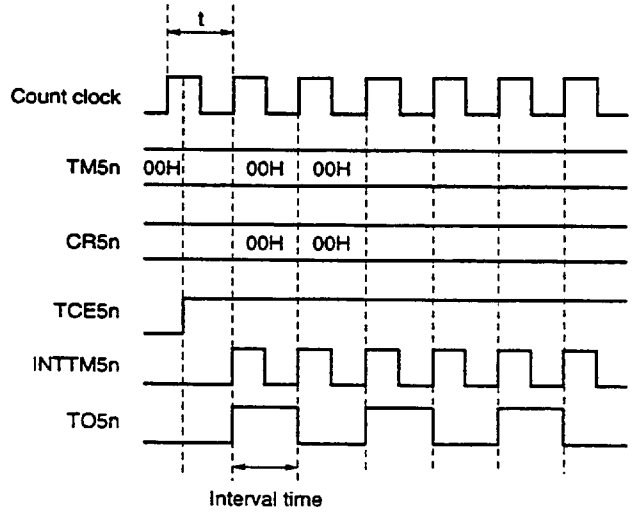


Remarks 1. Interval time = (N + 1) × t: N = 00H to FFH

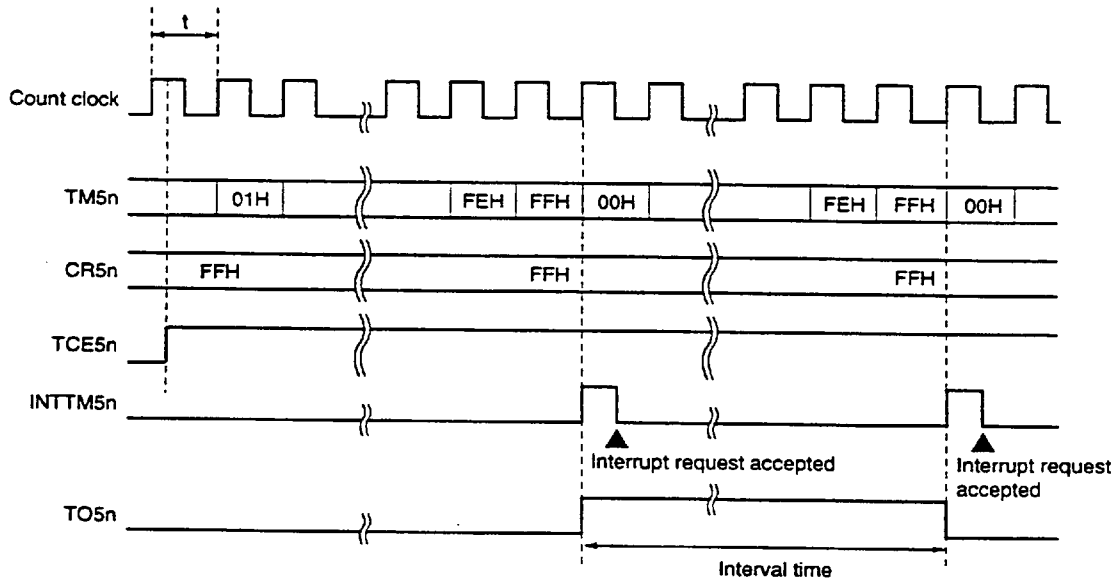
2. n = 0 to 2

Figure 8-10. Interval Timer Operation Timing (2/3)

(b) When CR5n = 00H



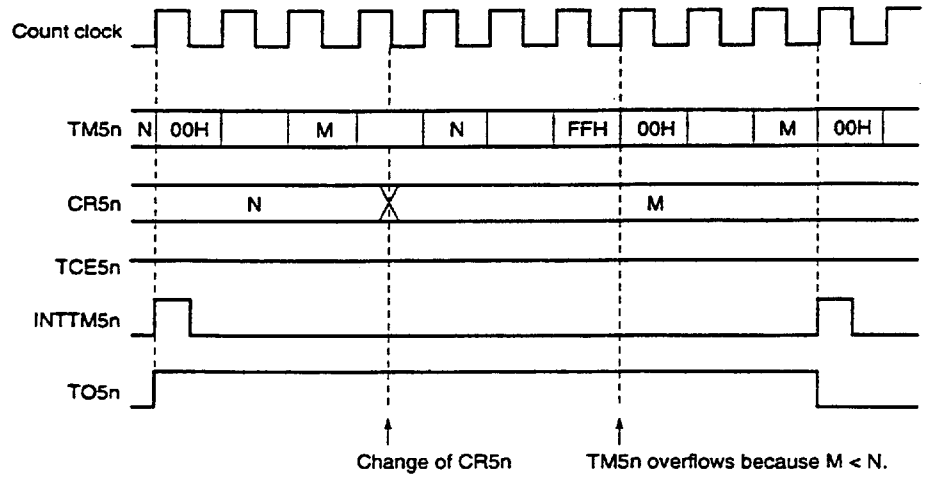
(c) When CR5n = FFH



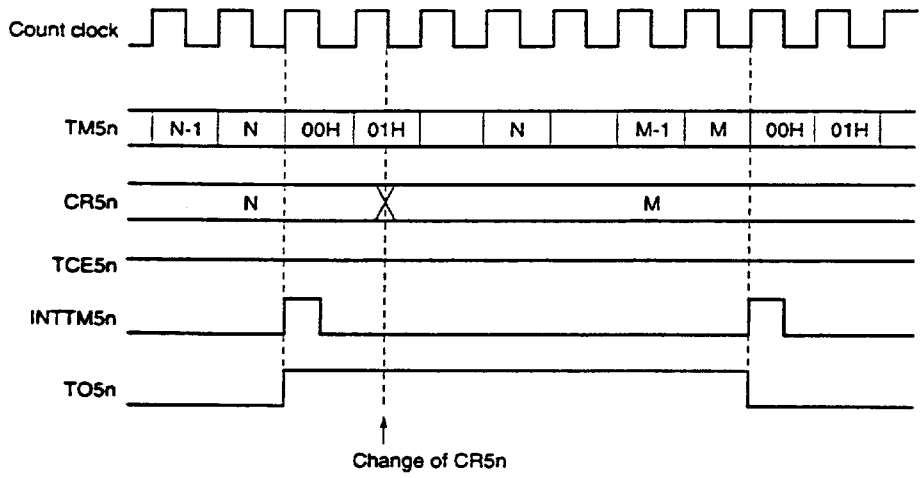
Remark n = 0 to 2

Figure 8-10. Interval Timer Operation Timing (3/3)

(d) Operation when CR5n is changed (M < N)



(e) Operation when CR5n is changed (M > N)



Remark n = o to 2

8.4.2 Operation as external event counter

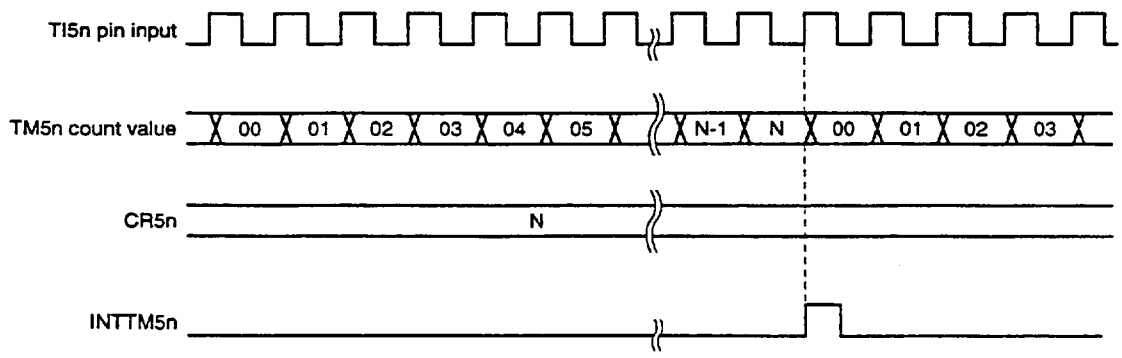
The external event counter counts the number of clock pulses externally input to the TI50/P24 to TI52/P26 pins by using the 8-bit timer counter (TM5n).

Each time the valid edge specified by the timer clock select register (TCL5n) is input, the value of TM5n is incremented. Either the rising edge or falling edge can be specified as the valid edge.

When the count value of TM5n coincide with the values of the corresponding 8-bit compare register (CR5n), TM5n is cleared to 0, and interrupt request signal (INTTM5n) is generated.

Remark n = 0 to 2

Figure 8-11. External Event Counter Operation Timing (with rising edge specified)



Remark N = 00H to FFH
n = 0 to 2

8.4.3 Square wave output operation (8-bit resolution)

Square wave of any frequency can be output at the interval set in advance to an 8-bit compare register (CR5n).

When bit 0 (TOE5n) of the 8-bit timer mode control register (TMC5n) is set to 1, the output status of TO5n is inverted at the interval time secified by the count value set in advance to CR5n. In this way, square wave of any frequency (duty factor = 50%) can be output.

<Setting>

<1> Set each register.

- Clear the port latch and port mode register 2 to "0".
- TCL5n: Selects count clock
- CR5n: Compare value
- TMC5n: Clear and start mode on coincidence between TM5n and CR5n

LVS5n	LVR5n	Sets status of timer output F/F
1	0	High-level output
0	1	Low-level output

Enables inverting timer output F/F

Timer output enabled → TOE5n = 1

<2> The count operation is started if TCE5n is set to 1.

<3> The timer output F/F is inverted if the values of TM5n and CR5n coincide.

INTTM5n occurs and TM5n is cleared to 00H.

<4> After that, the timer output F/F is inverted at the same interval, and square wave is output from TO5n.

Remark n = 0 to 2

8.4.4 8-bit PWM output operation

The PWM output operation is performed when bit 6 (TMC5n6) of the 8-bit timer mode control register (TMC5n) is set to "1".

The pulse with a duty factor determined by the value set to the 8-bit compare register (CR5n) is output from TO5n.

Set the width of the active level of the PWM pulse to CR5n. The active level can be selected from bit 1 (TMC5n1) of TMC5n.

The count clock can be selected by bits 0 to 2 (TCL5n0 to TCL5n2) of the timer clock select register (TCL5n).

The PWM output can be enabled or disabled by bit 0 (TOE5n) of TMC5n.

(1) Basic operation of PWM output

<Setting>

- **<1>** Clear the port latch and port mode register 2 to "0".
- **<2>** Set an active level width with the 8-bit compare register (CR5n).
- **<3>** Select a count clock with the timer clock select register (TCL5n).
- **<4>** Set an active level by using the bit 1 (TMC5n1) of TMC5n.
- **<5>** The count operation is started when bit 7 (TCE5n) of TMC5n is set to "1".
  To stop the count operation, clear TCE5n to "0".

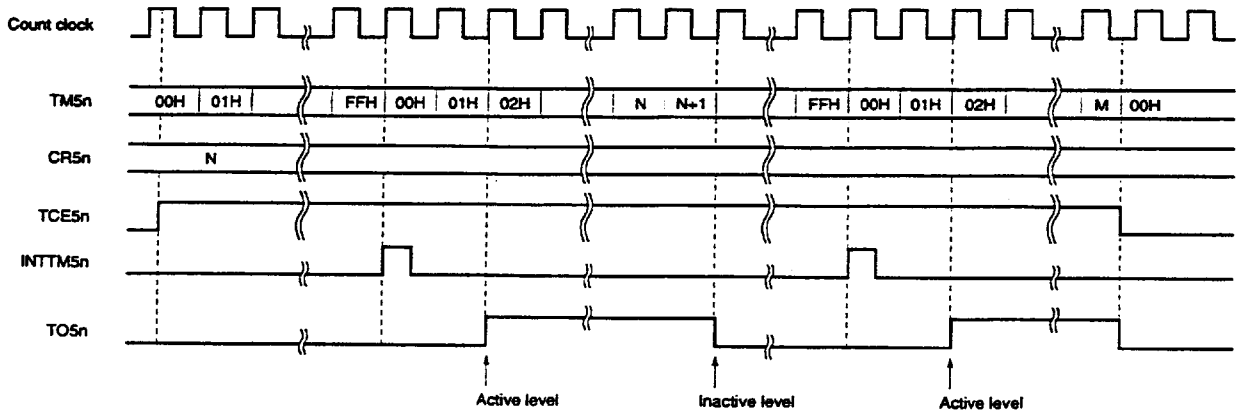
<PWM output operation>

- **<1>** When the count operation is started, an inactive level is output as the PWM output (output from TO5n) until an overflow occurs.
- **<2>** When an overflow occurs, the active level set in step **<1>** above is output. This active level is continuously output until the value of CR5n coincides with the count value of an 8-bit timer register (TM5n).
- **<3>** An inactive level is output as the PWM output after the value of CR5n has coincided with the count value of TM5n, until an overflow occurs again.
- **<4>** After that, **<2>** and **<3>** are repeated until the count operation is stopped.
- **<5>** When the count operation is stopped by clearing TCE5n to 0, the PWM output becomes inactive.

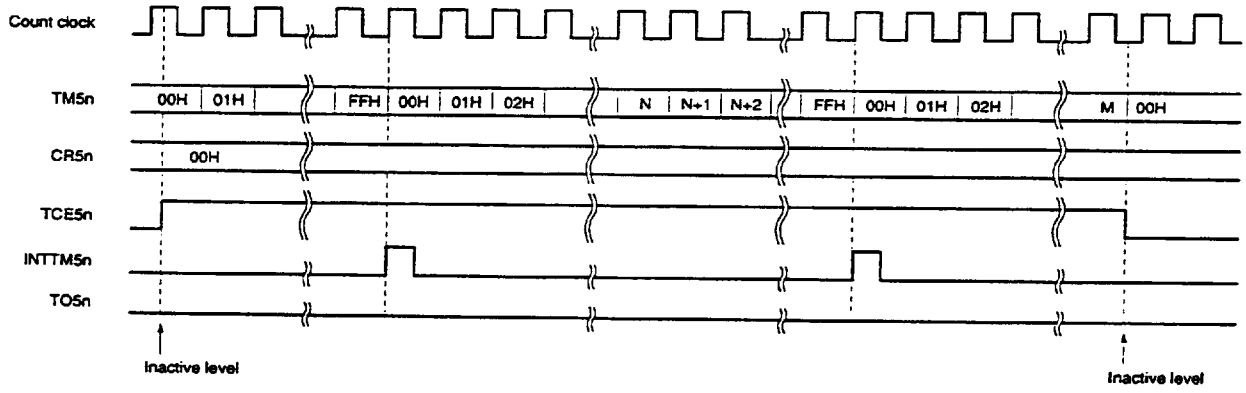
Remark n = 0 to 2

Figure 8-12. PWM Output Operation Timing

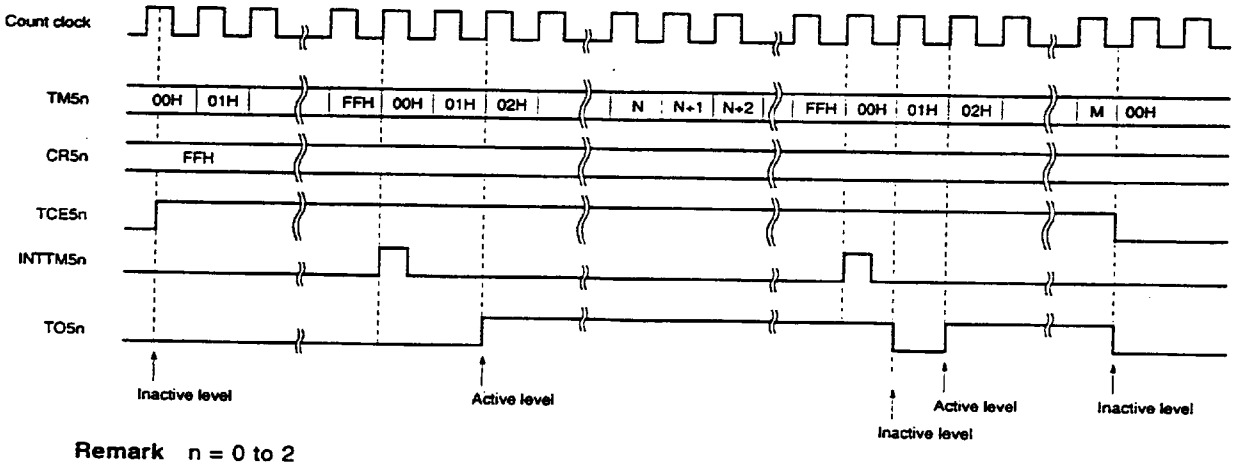
(a) Basic operation (when active level = H)



(b) When CR5n = 0



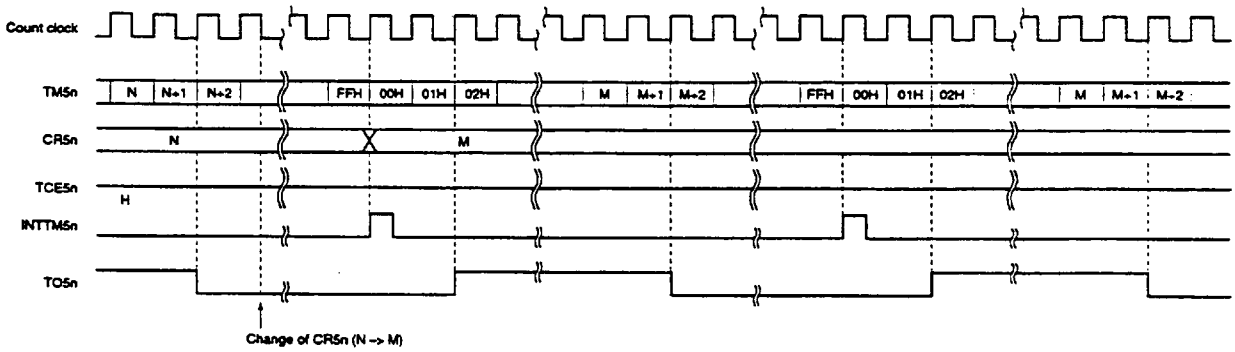
(c) When CR5n = FFH



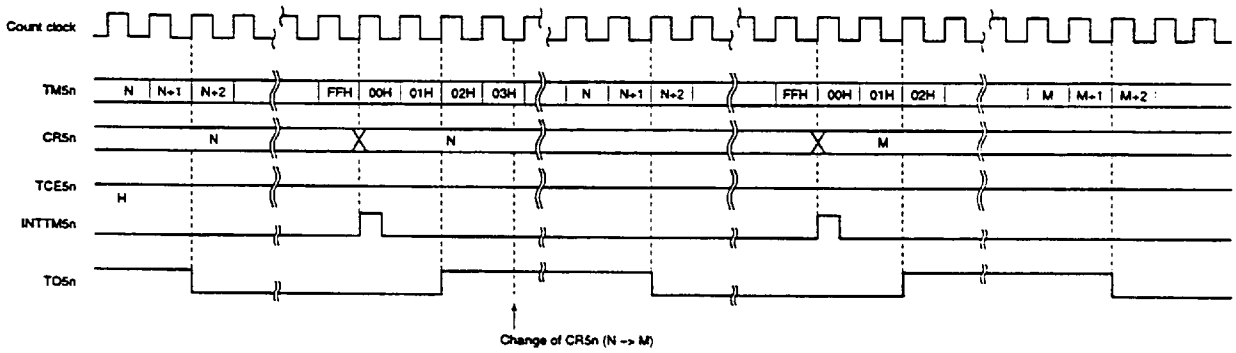
Remark n = 0 to 2

Figure 8-13. Operation Timing When CR5n is Changed

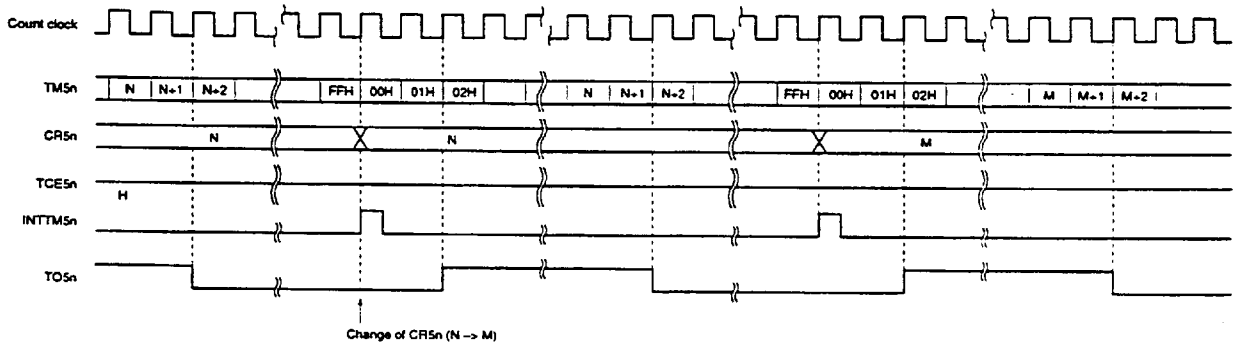
(a) If value of CR5n is changed from N to M before TM5n overflows



(b) If value of CR5n is changed from N to M after TM5n overflows



(c) If value of CR5n is changed from N to M during 2 clocks (00H, 01H) immediately after TM5n overflows



Remark n = 0 to 2

8.4.5 Operation as interval timer (16-bit operation)

(1) Cascade (16-bit timer) mode (TM50 and TM51)

The 16-bit resolution timer/counter mode is set by setting bit 4 (TMC514) of the 8-bit timer mode control register 51 (TMC51) to "1".

In this mode, TM50 and TM51 operate as a 16-bit interval timer that repeatedly generates an interrupt at intervals specified by the count value set in advance to 8-bit compare registers (CR50 and CR51).

<Setting>

<1> Set each register.

- TCL50: TM50 selects a count clock.
  TM51, which is connected in cascade, does not have to be set.
- CR50 and CR51: Compare values (Each compare value can be set in a range of 00H to FFH).
- TMC50 and TMC51: Select the mode that clears and starts the timer on coincidence between TM50 and CR50 (TM51 and CR51).

  [TM50 → TMC50 = 0000xxx0B x: don't care
  TM51 → TMC51 = 0001xxx0B x: don't care]

<2> By setting TCE51 to 1 for TMC51 first, and then setting TCE50 to 1 for TMC50, the count operation is started.

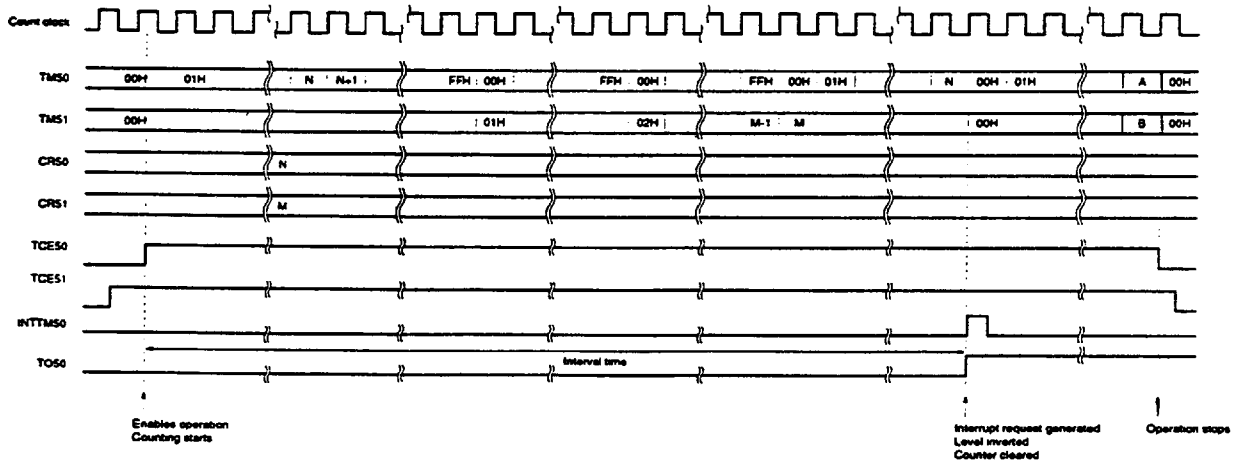
<3> When the value of TM50 connected in cascade coincides with the value of CR50, TM50 generates INTTM50 (TM50 and TM51 are cleared to 00H).

<4> After that, INTTM50 is repeatedly generated at the same interval.

Cautions
1. **Be sure to set the compare registers (CR50 and CR51) after stopping the timer operation.**
2. **Even if the timers are connected in cascade, TM51 generates INTTM51 when the count value of TM51 coincides with the value of CR51. Be sure to mask TM51 to disable it from generating an interrupt.**
3. **To set TCE50 and TCE51, first manipulate TM51 and then TM50.**
4. **Counting can be started or stopped by setting or clearing only TCE50 of TM50 to 1 or 0.**

Figure 8-14 shows an example of timing in the 16-bit resolution cascade mode.

Figure 8-14. 16-Bit Resolution Cascade Mode (with TM50 and TM51)



(2) Cascade (16-bit timer) mode (TM51 and TM52)

The 16-bit resolution timer/counter mode is set by setting bit 4 (TMC524) of the 8-bit timer mode control register 52 (TMC52) to "1".

In this mode, TM51 and TM52 operate as a 16-bit interval timer that repeatedly generates an interrupt at intervals specified by the count value set in advance to 8-bit compare registers (CR51 and CR52).

<Setting>

<1> Set each register.

- TCL51: TM51 selects a count clock.
  TM52, which is connected in cascade, does not have to be set.
- CR51 and CR52: Compare values (Each compare value can be set in a range of 00H to FFH).
- TMC51 and TMC52: Select the mode that clears and starts the timer on coincidence between TM51 and CR51 (TM52 and CR52).

  [TM51 → TMC51 = 0000xxx0B x: don't care
  TM52 → TMC52 = 0001xxx0B x: don't care]

<2> By setting TCE52 to 1 for TMC52 first, and then setting TCE51 to 1 for TMC51, the count operation is started.

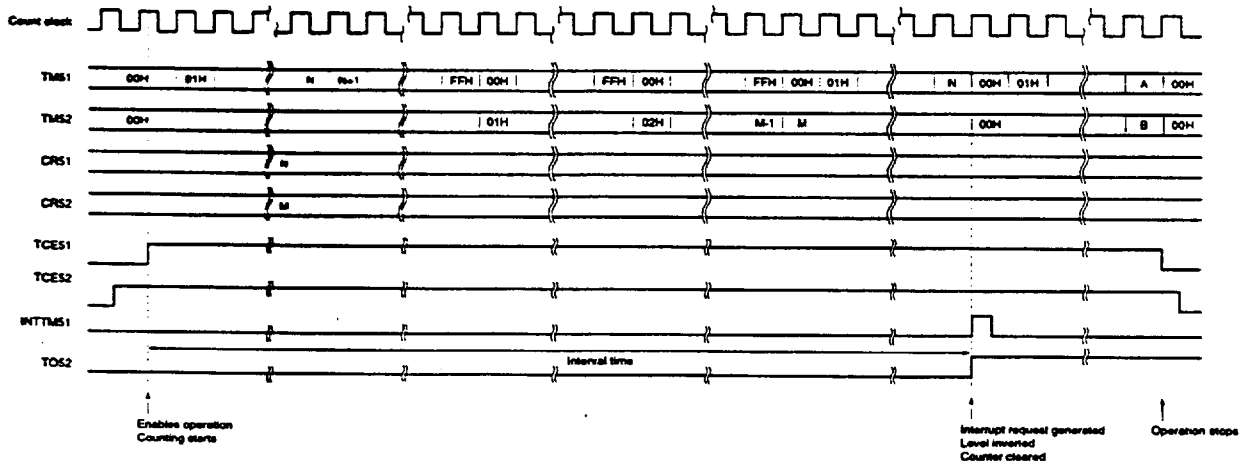
<3> When the value of TM51 connected in cascade coincides with the value of CR51, TM51 generates INTTM51 (TM51 and TM52 are cleared to 00H).

<4> After that, INTTM51 is repeatedly generated at the same interval.

Cautions
1. **Be sure to set the compare registers (CR51 and CR52) after stopping the timer operation.**
2. **Even if the timers are connected in cascade, TM52 generates INTTM52 when the count value of TM52 coincides with the value of CR52. Be sure to mask TM52 to disable it from generating an interrupt.**
3. **To set TCE51 and TCE52, first manipulate TM52 and then TM51.**
4. **Counting can be started or stopped by setting or clearing only TCE51 of TM51 to 1 or 0.**

Figure 8-15 shows an example of timing in the 16-bit resolution cascade mode.

Figure 8-15. 16-Bit Resolution Cascade Mode (with TM51 and TM52)

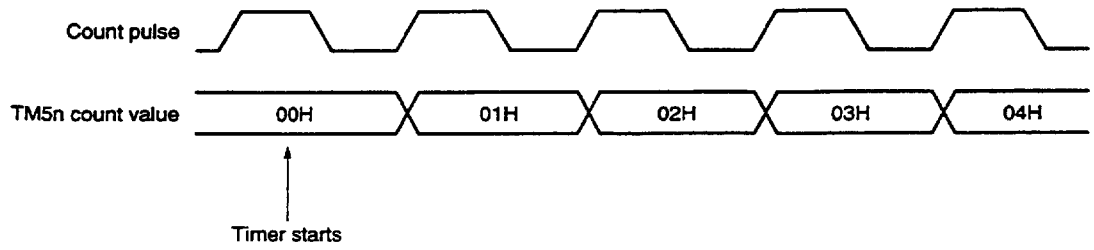


8.5 Notes on Using 8-Bit Timer/Event Counters

(1) Error on starting timer

An error of up to 1 clock occurs after the timer has been started until a coincidence signal is generated. This is because the 8-bit timer counter (TM5n) is started in asynchronization with the count pulse.

Figure 8-16. Start Timing of 8-Bit Timer Counter

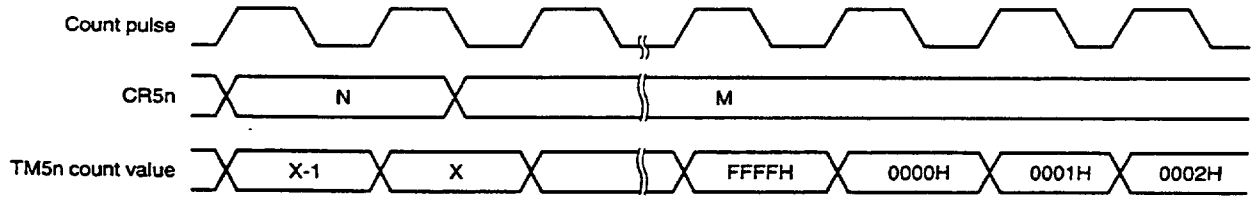


Remark n = 0 to 2

(2) Operation after changing value of compare register during timer count operation

If a new value of an 8-bit compare register (CR5n) is less than the value of the corresponding 8-bit timer counter (TM5n), TM5n continues counting, overflows, and restarts counting from 0. Therefore, if the new value of CR5n (M) is less than their old value (N), it is necessary to restart the timers after changing the value of CR5n.

Figure 8-17. Timing after Changing Values of Compare Registers during Timer Count Operation



Caution Except when TI5n input is selected, be sure to clear TCE5n to 0 before setting the STOP mode.

Remark N > X > M
n = 0 to 2

(3) Reading TM5n during timer operation

Because the count clock is stopped when TM5n is read during operation, select a count clock with a waveform whose high-/low-level are longer than two CPU clock cycles. For example, in the case of a CPU clock (f_{CPU}) equal to fx, TM5n can be read as long as the selected count clock is fx/4 or lower.

Remark n = 0 to 2

9.1 Functions of Watchdog Timer

The watchdog timer has the following functions:

- Watchdog timer
- Interval timer
- Oscillation stabilization time specification

Caution Select the watchdog timer mode or interval timer mode by using the watchdog timer mode register (WDTM). (The watchdog timer and interval timer cannot be used simultaneously.)

(1) Watchdog timer mode

The watchdog timer is used to detect inadvertent program loop. When the inadvertent loop is detected, a non-maskable interrupt or the $\overline{\text{RESET}}$ signal can be generated.

Table 9-1. Inadvertent Loop Detection Time of Watchdog Timer

Inadvertent Loop Detection Time	At fx = 8.38 MHz	Inadvertent Loop Detection Time	At fx = 8.38 MHz
$2^{12} \times 1/fx$	488.8 μs	$2^{16} \times 1/fx$	7.82 ms
$2^{13} \times 1/fx$	977.6 μs	$2^{17} \times 1/fx$	15.6 ms
$2^{14} \times 1/fx$	1.96 ms	$2^{18} \times 1/fx$	31.3 ms
$2^{15} \times 1/fx$	3.91 ms	$2^{20} \times 1/fx$	125.1 ms

Remark fx: system clock oscillation frequency

(2) Interval timer mode

When the watchdog timer is used as an interval timer, it generates an interrupt at time intervals set in advance.

Table 9-2. Interval Time

Interval Time	At fx = 8.38 MHz	Interval Time	At fx = 8.38 MHz
$2^{12} \times 1/fx$	488.8 μs	$2^{16} \times 1/fx$	7.82 ms
$2^{13} \times 1/fx$	977.6 μs	$2^{17} \times 1/fx$	15.6 ms
$2^{14} \times 1/fx$	1.96 ms	$2^{18} \times 1/fx$	31.3 ms
$2^{15} \times 1/fx$	3.91 ms	$2^{20} \times 1/fx$	125.1 ms

Remark fx: system clock oscillation frequency

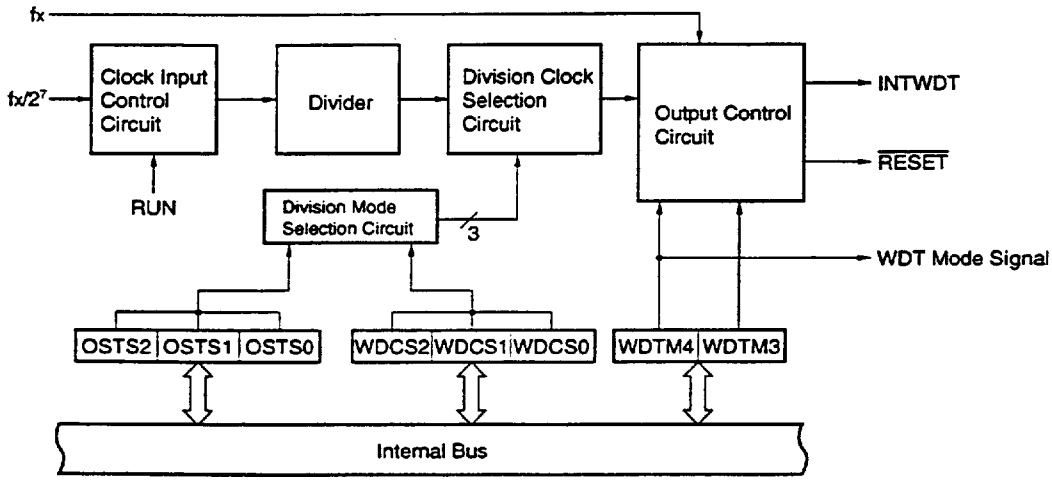
9.2 Configuration of Watchdog Timer

The watchdog timer consists of the following hardware:

Table 9-3. Configuration of Watchdog Timer

Item	Configuration
Control register	Watchdog timer clock select register (WDCS)
	Watchdog timer mode register (WDTM)
	Oscillation stabilization time select register (OSTS)

Figure 9-1. Watchdog Timer Block Diagram



9.3 Registers Controlling Watchdog Timer

The following three registers control the watchdog timer:

- Watchdog timer clock select register (WDCS)
- Watchdog timer mode register (WDTM)
- Oscillation stabilization time specification register (OSTS)

(1) Watchdog timer clock select register (WDCS) (refer to Figure 9-2)

This register sets the watchdog timer and overflow time of interval timer.

WDCS is set by an 8-bit memory manipulation instruction.

This register is set to 00H when the $\overline{\text{RESET}}$ signal is input.

Figure 9-2. Format of Watchdog Timer Clock Select Register

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
WDCS	0	0	0	0	0	WDCS2	WDCS1	WDCS0	FF42H	00H	R/W

WDCS2	WDCS1	WDCS0	Selects watchdog timer/overflow time of interval timer.
0	0	0	2^{12}/fx (488.8 μs)
0	0	1	2^{13}/fx (977.6 μs)
0	1	0	2^{14}/fx (1.96 ms)
0	1	1	2^{15}/fx (3.91 ms)
1	0	0	2^{16}/fx (7.82 ms)
1	0	1	2^{17}/fx (15.6 ms)
1	1	0	2^{18}/fx (31.3 ms)
1	1	1	2^{20}/fx (125.1 ms)

Remarks 1. fx : system clock oscillation frequency

2. ( ): at fx = 8.38-MHz operation

(2) Watchdog timer mode register (WDTM)

This register sets an operation mode of the watchdog timer, and enables/disables counting of the watchdog timer.

WDTM is set by a 1-bit or 8-bit memory manipulation instruction.

This register is set to 00H when the $\overline{\text{RESET}}$ signal is input.

Figure 9-3. Format of Watchdog Timer Mode Register

Symbol	⑦	6	5	4	3	2	1	0	Address	On reset	R/W
WDTM	RUN	0	0	WDTM4	WDTM3	0	0	0	FFF9H	00H	R/W

WDTM4	WDTM3	Selects operation mode of watchdog timer[Note 1] and controls reset by watchdog timer and interrupt of timer.
0	×	Interval timer mode[Note 2] (overflow and maskable interrupt occur)/PWM output off function of TM7 by INTWDT can be used.
1	0	Watchdog timer mode 1 (overflow and non-maskable interrupt occur)/PWM output off function of TM7 by INTWDT can be used.
1	1	Watchdog timer mode 2 (overflow occurs and reset operation started)

RUN	Selects operation of watchdog timer[Note 3].
0	Stops counting.
1	Clears counter and starts counting.

Notes
1. Once WDTM3 and WDTM4 have been set to 1, they cannot be cleared to 0 by software.
2. The watchdog timer starts operating as an interval timer as soon as the RUN bit has been set to 1.
3. Once RUN has been set to 1, it cannot be cleared to 0 by software. Therefore, when counting is started, it cannot be stopped by any means other than $\overline{\text{RESET}}$ input.

Caution When the watchdog timer is cleared by setting 1 to RUN, the actual overflow time is up to 0.5% shorter than the time set by the watchdog timer clock select register.

Remark ×: don't care

(3) Oscillation stabilization time select register (OSTS)

This register selects the oscillation stabilization time that elapses since the $\overline{\text{RESET}}$ signal has been applied or the STOP mode has been released, until oscillation is stabilized.

OSTS is set by an 8-bit memory manipulation instruction.

This register is set to 04H when the $\overline{\text{RESET}}$ signal is input. Therefore, to release the STOP mode by inputting the $\overline{\text{RESET}}$ signal, the ime required to release the mode is 2^{17}/fx.

Figure 9-4. Format of Oscillation Stabilization Time Select Register

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0	FFFAH	04H	R/W

OSTS2	OSTS1	OSTS0	Selects oscillation stabilization time when released the STOP mode.
0	0	0	2^{12}/fx (488.8 μs)
0	0	1	2^{14}/fx (1.96 ms)
0	1	0	2^{15}/fx (3.91 ms)
0	1	1	2^{16}/fx (7.82 ms)
1	0	0	2^{17}/fx (15.6 ms)
Others			Setting prohibited

Remarks 1. fx: system clock oscillation frequency

2. ( ): at fx = 8.38-MHz operation

9.4 Operation of Watchdog Timer

9.4.1 Operation as watchdog timer

The watchdog timer detects an inadvertent program loop when bit 4 (WDTM4) of the watchdog timer mode register (WDTM) is set to 1.

The inadvertent loop detection time interval of the watchdog timer can be selected by bits 0 to 2 (WDCS0 to WDCS2) of the watchdog timer clock select register (WDCS). By setting bit 7 (RUN) of WDTM to 1, the watchdog timer is started. Set RUN to 1 within the set inadvertent loop detection time interval after the watchdog timer has been started. By setting RUN to 1, the watchdog timer can be cleared and started counting. If RUN is not set to 1, and the inadvertent loop detection time is exceeded, the system is reset or a non-maskable interrupt is generated by the value of bit 3 (WDTM3) of WDTM.

The watchdog timer continues operation in the HALT mode, but stops in the STOP mode. Therefore, set RUN to 1 before entering the STOP mode to clear the watchdog timer, and then execute the STOP instruction.

Cautions 1. The actual inadvertent loop detection time may be up to 0.5% shorter than the set time.

2. The count operation of the watchdog timer is stopped when the subsystem clock is selected as the CPU clock.

Table 9-4. Inadvertent Loop Detection Time of Watchdog Timer

TCL22	TCL21	TCL20	Inadvertent Loop Detection Time	At fx = 8.38 MHz
0	0	0	$2^{12} \times 1/fx$	488.8 μs
0	0	1	$2^{13} \times 1/fx$	977.6 μs
0	1	0	$2^{14} \times 1/fx$	1.96 ms
0	1	1	$2^{15} \times 1/fx$	3.91 ms
1	0	0	$2^{16} \times 1/fx$	7.82 ms
1	0	1	$2^{17} \times 1/fx$	15.6 ms
1	1	0	$2^{18} \times 1/fx$	31.3 ms
1	1	1	$2^{20} \times 1/fx$	125.1 ms

Remark fx: system clock oscillation frequency

9.4.2 Operation as interval timer

When bit 4 (WDTM4) of the watchdog timer mode register (WDTM) is set to 0, the watchdog timer also operates as an interval timer that repeatedly generates an interrupt at time intervals specified by a count value set in advance.

Bits 0 to 2 (WDCS0 to WDCS2) of the watchdog timer clock select register (WDCS) can be used to select interval time of interval timer. When bit 7 (RUN) of WDTM is set to 1, the watchdog timer starts operating as an interval timer.

In the interval timer mode, the interrupt mask flag (WDTMK) and priority specification flag (WDTPR) are valid, and a maskable interrupt (INTWDT) can be generated. The default priority of INTWDT is set the highest of all the maskable interrupts.

The interval timer continues operation in the HALT mode, but stops in the STOP mode. Therefore, set bit 7 of WDTM (RUN) to 1 before entering the STOP mode to clear the interval timer, and then execute the STOP instruction.

Cautions: 1. Once bit 4 (WDTM4) of WDTM has been set to 1 (when the watchdog timer mode is selected), the interval timer mode is not set, unless the $\overline{\text{RESET}}$ signal is input.

2. The interval time immediately after it has been set by WDTM may be up to 0.5% shorter than the set time.

Table 9-5. Interval Time of Interval Timer

WDCS2	WDCS1	WDCS0	Interval Time	At fx = 8.38 MHz
0	0	0	$2^{12} \times 1/fx$	488.8 μs
0	0	1	$2^{13} \times 1/fx$	977.6 μs
0	1	0	$2^{14} \times 1/fx$	1.96 ms
0	1	1	$2^{15} \times 1/fx$	3.91 ms
1	0	0	$2^{16} \times 1/fx$	7.82 ms
1	0	1	$2^{17} \times 1/fx$	15.6 ms
1	1	0	$2^{18} \times 1/fx$	31.3 ms
1	1	1	$2^{20} \times 1/fx$	125.1 ms

Remark fx: system clock oscillation frequency

[MEMO]

PAGE(S) INTENTIONALLY BLANK

10.1 Function of A/D Converter

The A/D converter converts an analog input signal into a digital value, and consists of eight channels (ANI0 to ANI7) with a resolution of 8 bits.

This A/D converter is of successive approximation type, and the result of conversion is held by an 8-bit A/D conversion result register (ADCR0).

A/D conversion can be started in the following two ways:

(1) Hardware start

Conversion is started by trigger input (P03) (rising, falling, both rising and falling edges can be specified).

(2) Software start

Conversion is started by setting the A/D converter mode register 0.

One channel of analog input is selected from ANI0 to ANI7 and A/D conversion is executed. The A/D conversion is stopped, if it has been started by means of hardware start, after the conversion has been completed, and an interrupt request (INTAD0) is generated. When A/D conversion has been started by means of software start, conversion is repeatedly performed. Each time conversion has been completed once, an interrupt request (INTAD0) is generated.

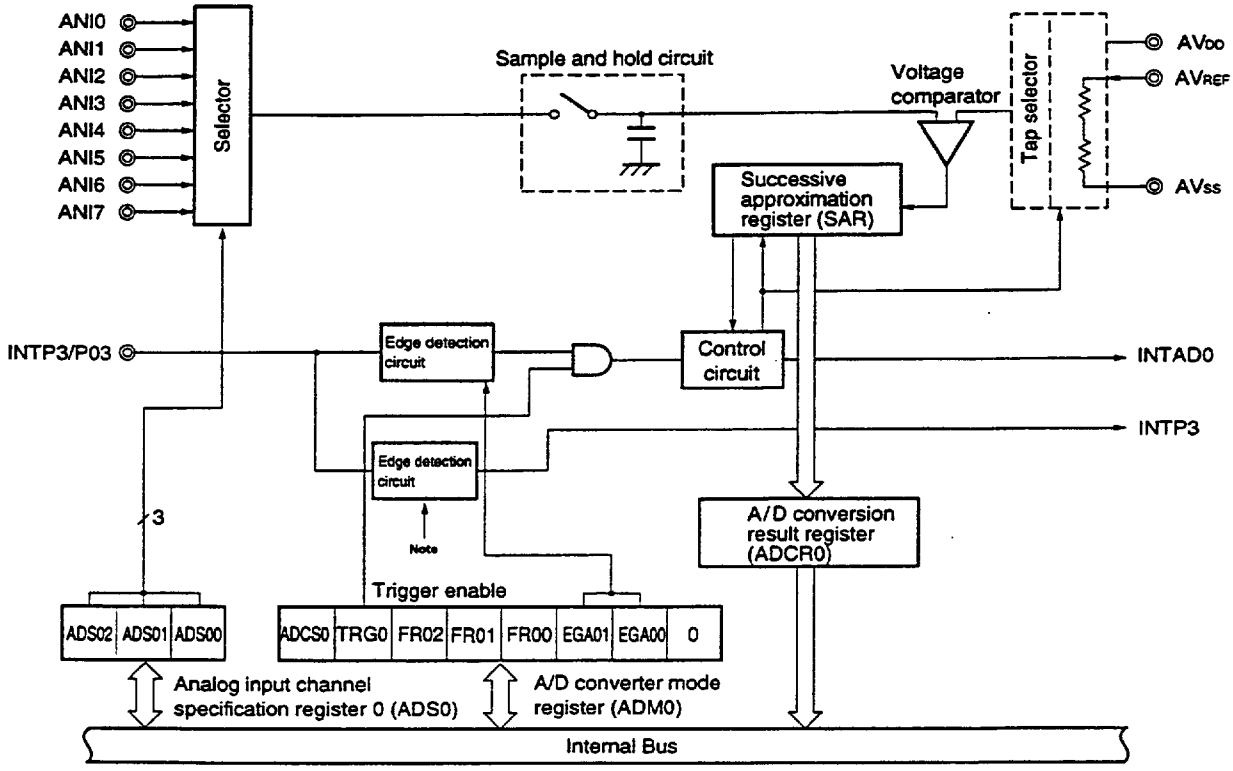
10.2 Configuration of A/D Converter

The A/D converter consists of the following hardware:

Table 10-1. Configuration of 8-Bit A/D Converter

Item	Configuration
Analog input	8 channels (ANI0 to ANI7)
Control register	A/D converter mode register 0 (ADM0) Analog input channel specification register 0 (ADS0)
Register	Successive approximation register (SAR) A/D conversion result register (ADCR0)

Figure 10-1. Block Diagram of 8-Bit A/D Converter



Note Specify the valid edge by using bit 3 of the EGP and EGN registers (refer to **Figure 13-5 Formats of External Interrupt Rising Edge Enable Register and External Interrupt Falling Edge Enable Register**).

(1) Successive approximation register (SAR)

This register compares the voltage value of analog input with the value of a voltage tap (compare voltage) from the series resistor string, and holds the result of the comparison starting from the most significant bit (MSB).

When the result is held down to the least significant bit (LSB) (end of A/D conversion), the contents of SAR are transferred to the A/D conversion result register.

(2) A/D conversion result register (ADCR0)

This register holds the result of A/D conversion result. Each time A/D conversion has been completed, the result of the conversion is loaded to this register from the successive approximation register.

ADCR0 can be read by an 8-bit memory manipulation instruction.

The contents of this register become undefined when the $\overline{\text{RESET}}$ signal is input.

(3) Sample and hold circuit

The sample and hold circuit samples analog input signals sequentially sent from the input circuit on a one-by-one basis, and sends the sampled signals to the voltage comparator. This circuit holds the sampled analog input voltage value during A/D conversion.

(4) Voltage comparator

The voltage comparator compares the analog input with the output voltage of the series resistor string.

(5) Series resistor string

The series resistor string is connected between AV_{REF} and AVss and generates a voltage to be compared with an analog input.

(6) ANI0 to ANI7 pins

These are eight channels of analog input pins of the A/D converter. They input analog signals that are converted to digital values.

Caution Observe the specified input voltage range of ANI0 to ANI7. If a voltage of AV_{REF} or higher, or AVss or lower (even within the range of absolute maximum ratings) is applied to a channel, the converted value of that channel becomes undefined, or the converted value of the other channels may be affected.

(7) AV_{REF} pin

This pin inputs a reference voltage to the A/D converter.

Based on the voltage applied between AV_{REF} and AVss, the signal input to ANI0 to ANI7 is converted into a digital signal.

(8) AVss pin

This is a ground pin of the A/D converter. Be sure to use this pin at the same voltage as that on the V_{SS0} pin always even when the A/D converter is not used.

(9) AV_{DD} pin

This is an analog pin of the A/D converter. Be sure to use this pin at the same voltage as that on the V_{DDO} pin always even when the A/D converter is not used.

10.3 Registers Controlling A/D Converter

The following two registers control the A/D converter:

- A/D converter mode register 0 (ADM0)
- Analog input channel specification register 0 (ADS0)

(1) A/D converter mode register 0 (ADM0)

This register sets conversion time of an analog input to be converted into a digital value, starts/stops conversion operation, and sets an external trigger.

ADM0 is set by a 1-bit or 8-bit memory manipulation instruction.

This register is set to 00H when the $\overline{\text{RESET}}$ signal is input.

Figure 10-2. Format of A/D Converter Mode Register 0

Symbol	⑦	⑥	5	4	3	2	1	0	Address	On reset	R/W
ADM0	ADCS0	TRG0	FR02	FR01	FR00	EGA01	EGA00	0	FF80H	00H	R/W

ADCS0	Controls A/D conversion operation.
0	Stops operation.
1	Enables operation.

TRG0	Selects software/hardware start.
0	Software start
1	Hardware start

FR02	FR01	FR00	Selects A/D conversion time.
0	0	0	144/fx (17.2 μs)
0	0	1	120/fx (14.3 μs)
1	0	0	288/fx (34.4 μs)
1	0	1	240/fx (28.6 μs)
1	1	0	192/fx (22.9 μs)
Others			Setting prohibited

EGA01	EGA00	Specifies valid edge of external trigger.
0	0	No edge is detected.
0	1	Detects falling edge.
1	0	Detects rising edge.
1	1	Detects both rising and falling edges.

Caution Do not set the A/D conversion time to less than 14 μs.

Remark ( ): at fx = 8.38-MHz operation

(2) Analog input channel specification register 0 (ADS0)

This register sets the input port of analog voltage to be converted into a digital value.

ADS0 is set by an 8-bit memory manipulation instruction.

This register is set to 00H when the $\overline{\text{RESET}}$ signal is input.

Figure 10-3. Format of Analog Input Channel Specification Register 0

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
ADS0	0	0	0	0	0	ADS02	ADS01	ADS00	FF81H	00H	R/W

ADS02	ADS01	ADS00	Specifies analog input channel.
0	0	0	ANI0
0	0	1	ANI1
0	1	0	ANI2
0	1	1	ANI3
1	0	0	ANI4
1	0	1	ANI5
1	1	0	ANI6
1	1	1	ANI7

10.4 Operation of A/D Converter

10.4.1 Basic operation of A/D converter

<1> Select one channel for A/D conversion by using the analog input channel specification register 0 (ADS0).

<2> The voltage input to the selected analog input channel is sampled by the sample and hold circuit.

<3> When the voltage has been sampled for a specific time, the sample and hold circuit enters the hold status, and holds the input analog voltage until A/D conversion is completed.

<4> Set bit 7 of the successive approximation register (SAR). The tap selector selects (1/2)AV_{REF} as the voltage tap of the series resistor string.

<5> The voltage difference between the voltage tap of the series resistor string and the analog input is compared by the voltage comparator. If the analog input is higher than (1/2)AV_{REF}, the MSB of SAR remains set. If it is less than (1/2)AV_{REF}, the MSB is reset.

<6> Next, bit 6 of SAR is automatically set, and the next voltage difference is compared. Here the voltage tap of the series resistor string is selected as follows, according to the value of bit 7 to which the result of the first comparison has been already set.

- Bit 7 = 1 : (3/4)AV_{REF}
- Bit 7 = 0 : (1/4)AV_{REF}

This voltage tap and analog input voltage are compared, and bit 6 of SAR is manipulated as follows, according to the result of the comparison:

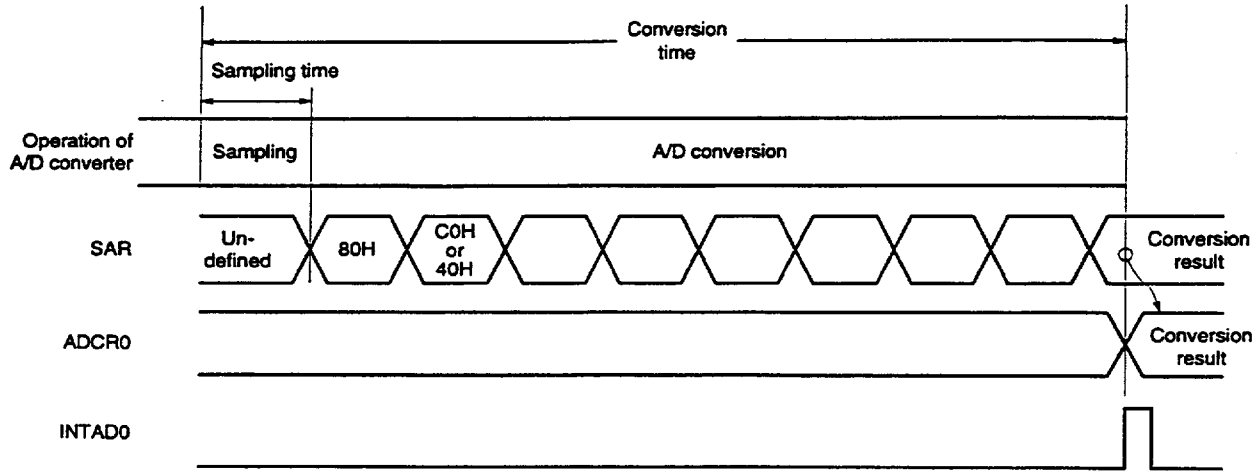
- Analog input voltage ≥ voltage tap : bit 6 = 1
- Analog input voltage ≤ voltage tap : bit 6 = 0

<7> In this way, all the bits of SAR, including bit 0, are compared.

<8> When all the 8 bits of SAR have been compared, SAR holds the valid digital result whose values are transferred and latched to the A/D conversion result register (ADCR0).

At the same time, an A/D conversion end interrupt request (INTAD0) can be generated.

Figure 10-4. Basic Operation of A/D Converter



The A/D conversion is performed continuously, until the bit 7 of ADM0 (ADCS0) is reset to 0 by software.

If the data of the ADM0 register or ADS0 register is rewritten during the A/D conversion, the conversion is initialized. If the ADCS0 bit is set to 1 at this time, conversion is performed again from the start.

The contents of the ADCR0 register become undefined when the $\overline{\text{RESET}}$ signal is input.

10.4.2 Input voltage and conversion result

The relation between the analog voltage input to the analog input pins (ANI0 to ANI7) and A/D conversion result (value stored to A/D conversion result register (ADCR0)) is as follows:

$$ADCR0 = INT\left(\frac{V_{IN}}{AV_{REF}} \times 256 + 0.5\right)$$

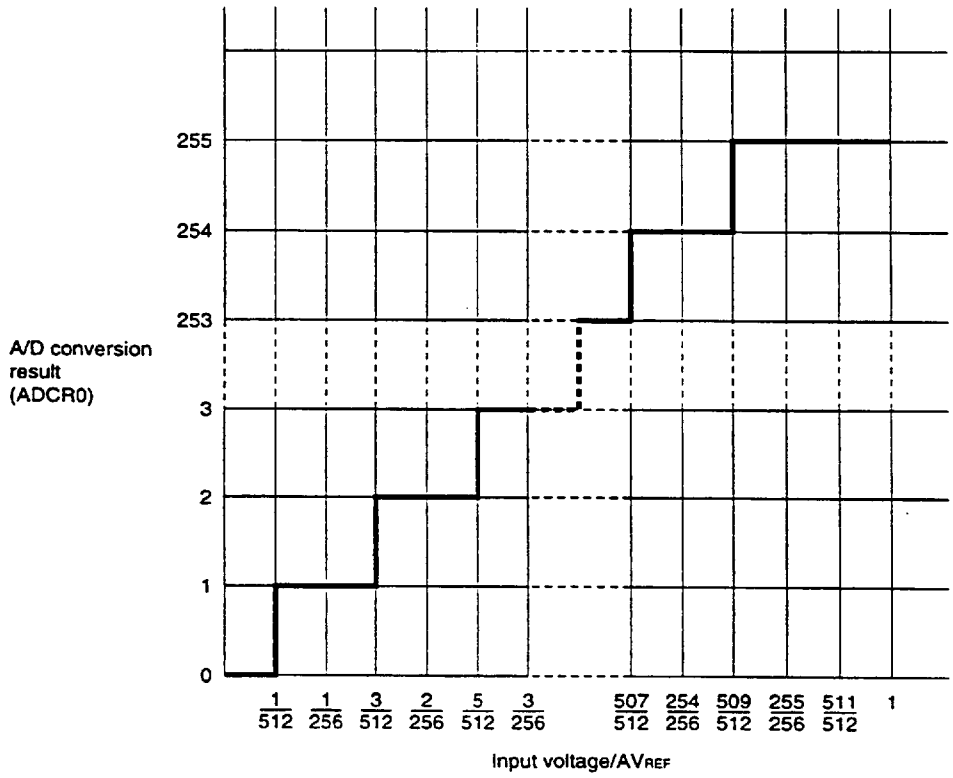
or,

$$(ADCR0 - 0.5) \times \frac{AV_{REF}}{256} \leq V_{IN} < (ADCR0 + 0.5) \times \frac{AV_{REF}}{256}$$

Remark INT( ) : function returning integer of value in ( )
V_{IN} : analog input voltage
AV_{REF} : AV_{REF} pin voltage
ADCR0 : value of A/D conversion result register (ADCR0)

Figure 10-5 shows the relations between the analog input voltage and A/D conversion result.

Figure 10-5. Relations between Analog Input Voltage and A/D Conversion Result



10.4.3 Operation mode of A/D converter

The A/D converter operates in the select mode only. One analog input channel is selected from ANI0-ANI7 for A/D conversion by using the analog input channel specification register 0 (ADS0).

The A/D conversion can be started in the following two ways:

- Hardware start : Conversion is started by trigger input (P03).
- Software start : Conversion is started by setting ADM0.

The result of the A/D conversion is stored in the A/D conversion result register (ADCR0), and at the same time, an interrupt request signal (INTAD0) is generated.

(1) A/D conversion operation by hardware start

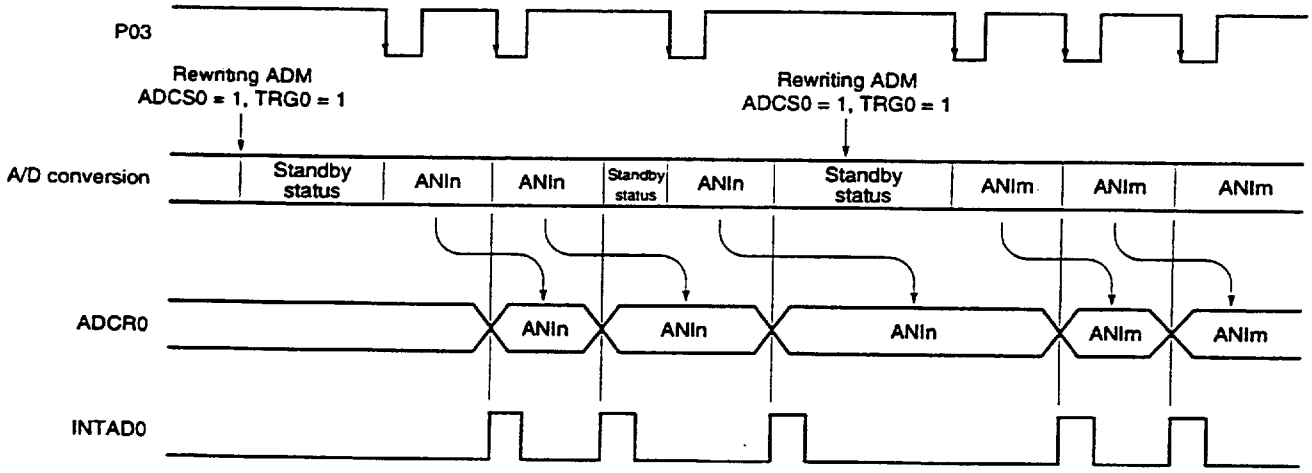
The A/D conversion stands by when both bits 6 (TRG0) and 7 (ADCS0) of A/D converter mode register (ADM0) are set to 1. When an external trigger signal (P03) is input, the voltage applied to the analog input pin specified by ADS0 is converted into a digital value.

When the A/D conversion has been completed, the result of the conversion is stored in the A/D conversion result register (ADCR0), and an interrupt request signal (INTAD0) is generated. Once the A/D conversion has been started and when one A/D conversion has been completed, the next A/D conversion is not started unless a new external trigger signal is input.

If data whose ADCS0 is 1 is written again to ADM0 during A/D conversion, the AD conversion under execution is stopped, and stands by until a new external trigger signal is input. When the external trigger signal is input, A/D conversion is performed again from the start.

When 0 is written to the ADCS0 bit of ADM0 during A/D conversion, the conversion is immediately stopped.

Figure 10-6. A/D Conversion by Hardware Start (When falling edge is specified)



Remark n = 0, 1, ..., 7
m = 0, 1, ..., 7

(2) A/D conversion by software start

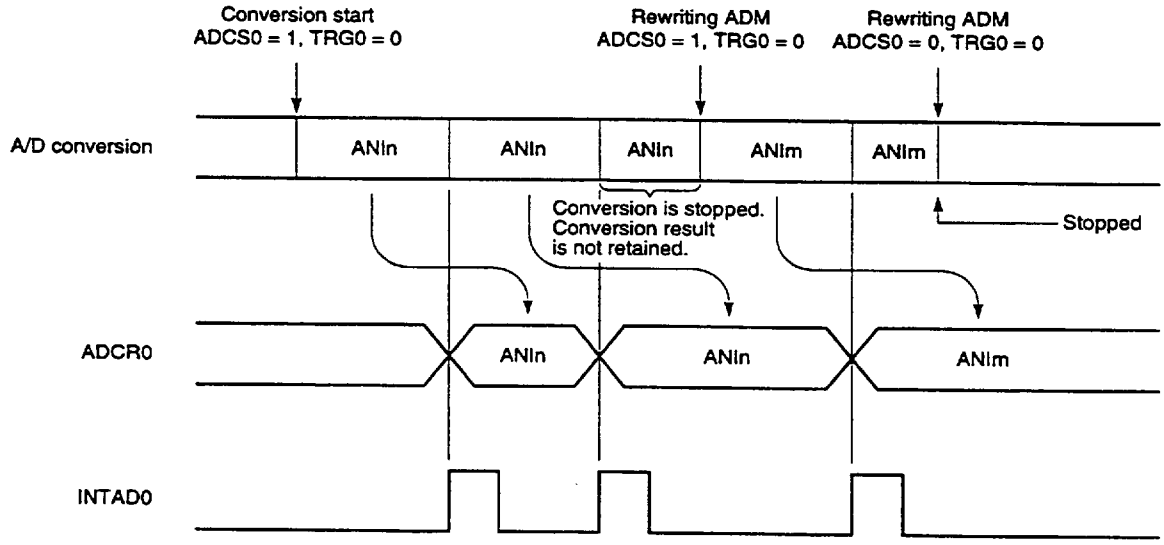
By setting bit 6 (TRG0) of the A/D converter mode register (ADM0) to 0 and setting bit 7 (ADCS0) to 1, the voltage applied to the analog input pin specified by ADS0 is converted into digital values.

When the A/D conversion has been completed, the result of the conversion is stored in the A/D conversion result register (ADCR0), and an interrupt request signal (INTAD0) is generated. When the A/D conversion has been started once, and one A/D conversion has been completed, the next A/D conversion is immediately started. In this way, A/D conversion is repeatedly executed until new data is written to ADM0.

If data whose ADCS0 is 1 is written again to ADM0 during A/D conversion, the conversion under execution is stopped, and the A/D conversion of the newly written data is started.

If data whose ADCS0 is 0 is written to ADM0 during A/D conversion, the conversion is immediately stopped.

Figure 10-7. A/D Conversion by Software Start



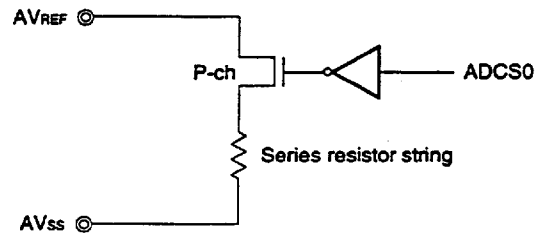
Remark n = 0, 1, ..., 7
m = 0, 1, ..., 7

10.5 Notes on A/D Converter

(1) Current consumption in standby mode

The A/D converter stops operating in the standby mode. At this time, the current consumption can be reduced by stopping the conversion operation (by clearing bit 7 (ADCS0) of the A/D converter mode register 0 (ADM0) to 0). An example of reducing the current consumption in standby mode is shown in Figure 10-8.

Figure 10-8. Example of Reducing Current Consumption in Standby Mode



(2) ANI0 to ANI7 input range

Observe the rated range of the ANI0 to ANI7 input voltage. If a voltage of AVREF or higher, or AVss or lower (even in the range of absolute maximum ratings) is input to an analog input channel, the converted value of that channel becomes undefined. In addition, the converted values of the other channels may also be affected.

(3) Conflicting operations

<1> Conflict between writing A/D conversion result register (ADCR0) on completion of conversion and reading ADCR0 by instruction

Reading ADCR0 takes precedence. After it has been read, a new conversion result is written to ADCR0.

<2> Conflict between writing ADCR0 on completion of conversion and external trigger signal input

The external trigger signal is not accepted during A/D conversion. Therefore, the external trigger signal is not accepted while ADCR0 is written.

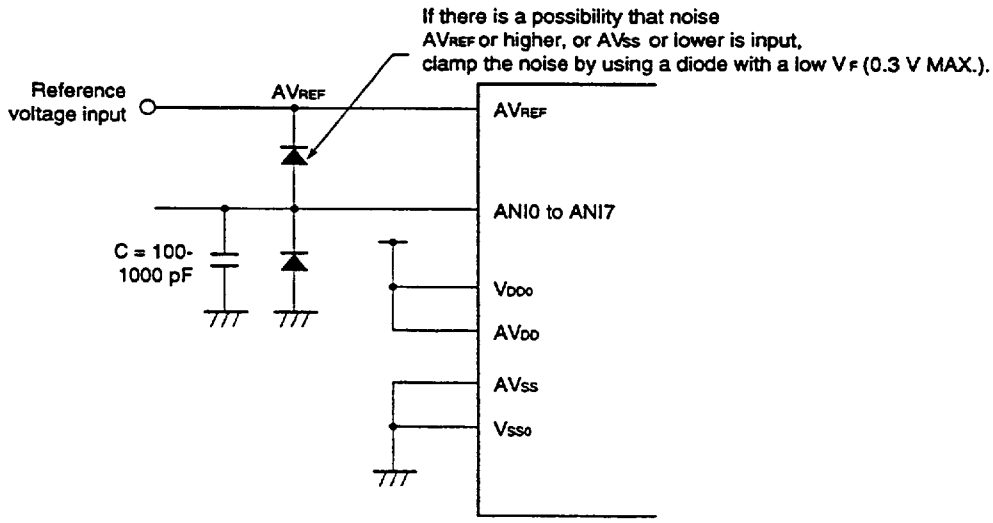
<3> Conflict between writing ADCR0 on completion of conversion and writing A/D converter mode register 0 (ADM0) or writing analog input channel specification register 0 (ADS0)

Writing ADM0 or ADS0 takes precedence. ADCR0 is not written. The conversion end interrupt request signal (INTAD0) is not generated.

(4) Countermeasures against noise

To keep the resolution of 8 bits, noise superimposed on the AV_{REF} and ANI0 to ANI7 pins must be suppressed as much as possible. The higher the output impedance of the analog input source, the greater the effect. To suppress noise, connecting an external capacitor as shown in Figure 10-9 is recommended.

Figure 10-9. Processing Analog Input Pin



(5) ANI0/P10 to ANI7/P17

The analog input pins (ANI0 to ANI7) are also used as input port pins (PORT 1).

When A/D conversion is performed with any of ANI0 to ANI7 selected, do not execute the input instruction of PORT 1 while conversion is in progress; otherwise, the conversion resolution may be degraded.

If a digital pulse is applied to the pins adjacent to the pins currently used for A/D conversion, the expected value of the A/D conversion may not be obtained due to coupling noise. Therefore, do not apply a pulse to the adjacent pins to the pin under A/D conversion.

(6) Input impedance to AV_{REF} pin

A series resistor string of about 21 kΩ is connected between the AV_{REF} and AVss pins.

If the output impedance of the reference voltage source is high, therefore, an error of the reference voltage increases by connecting the impedance in parallel with the series resistor string between the AV_{REF} and AVss pins.

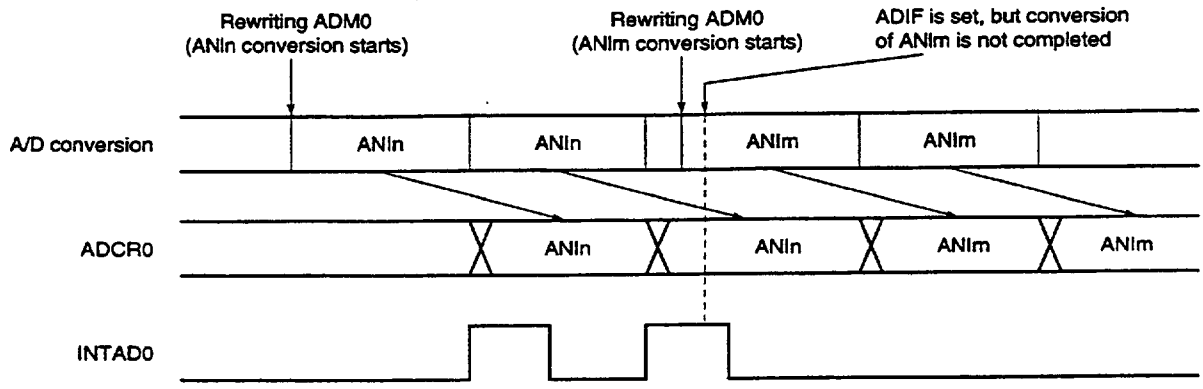
(7) Interrupt request flag (ADIF)

The interrupt request flag (ADIF) is not cleared even when the contents of the A/D converter mode register 0 (ADM0) are changed.

When the analog input pin is changed during A/D conversion, therefore, the chances are that the A/D conversion result of the old analog input and interrupt request flags are set immediately before the contents of ADM0 are rewritten. Consequently, ADIF may be set even if A/D conversion for the newly specified analog input pin has not yet been completed when ADIF is read immediately after ADM0 has been rewritten (refer to **Figure 10-10**).

To resume A/D conversion that has been once stopped, clear ADIF before resuming the conversion.

Figure 10-10. A/D Conversion End Interrupt Generation Timing



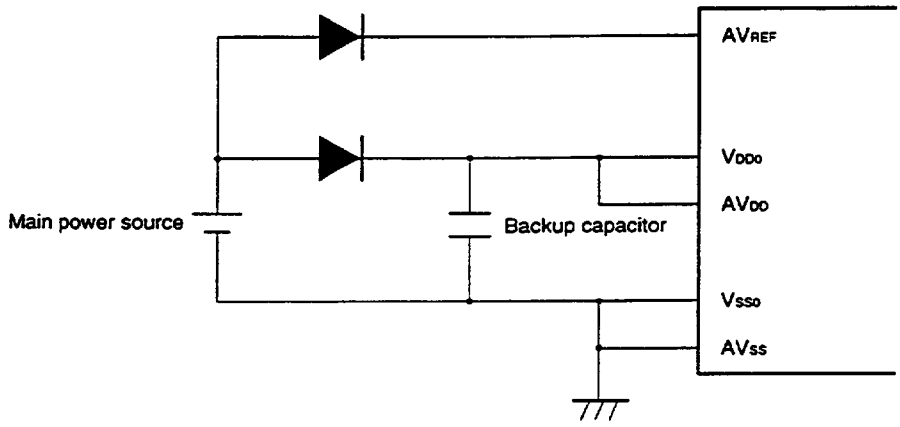
Remark n = 0, 1, ..., 7
n = 0, 1, ..., 7

(8) AV_{DD} pin

The AV_{DD} pin is the power supply pin to the analog circuit and supplies power to the input circuit of ANI0/P10 to ANI7/P17.

Therefore, even in the application which can be switched over to backup power source, be sure to apply the same voltage as V_{DDO} as shown in Figure 10-11.

Figure 10-11. Processing of AV_{DD} Pin



CHAPTER 11 10-BIT A/D CONVERTER (μPD780964 SUBSERIES)

11.1 Function of A/D Converter

The A/D converter converts an analog input signal into a digital value, and consists of eight channels (ANI0 to ANI7) with a resolution of 10 bits.

This A/D converter is of successive approximation type, and the result of conversion is held by a 10-bit A/D conversion result register (ADCR0).

A/D conversion can be started in the following two ways:

(1) **Hardware start**
Conversion is started by trigger input (P03) (rising edge, falling edge, or both rising and falling edges can be specified).

(2) **Software start**
Conversion is started by setting the A/D converter mode register 0.

One channel of analog input is selected from ANI0 to ANI7 and A/D conversion is executed. The A/D conversion is stopped, if it has been started by means of hardware start, after the conversion has been completed, and an interrupt request (INTAD0) is generated. When A/D conversion has been started by means of software start, conversion is repeatedly performed. Each time conversion has been completed once, an interrupt request (INTAD0) is generated.

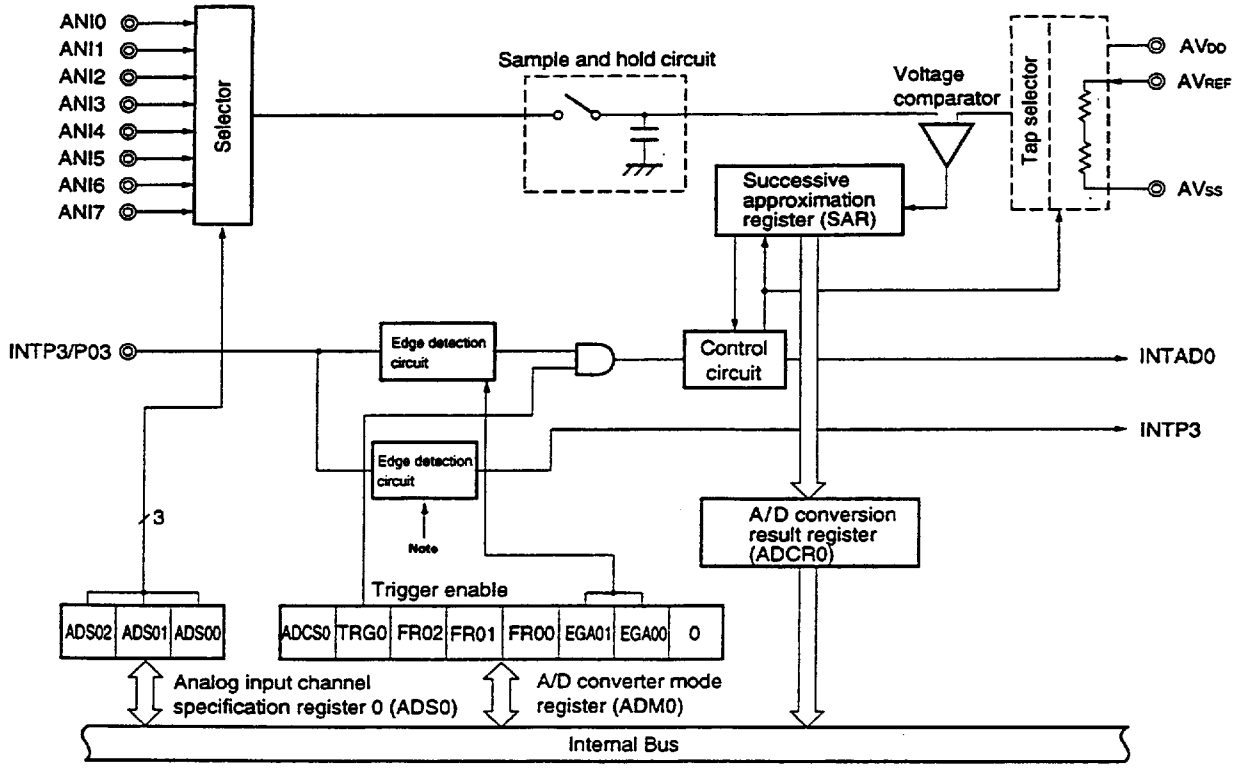
11.2 Configuration of A/D Converter

The A/D converter consists of the following hardware:

Table 11-1. Configuration of 10-Bit A/D Converter

Item	Configuration
Analog input	8 channels (ANI0 to ANI7)
Control register	A/D converter mode register 0 (ADM0) Analog input channel specification register 0 (ADS0)
Register	Successive approximation register (SAR) A/D conversion result registers (ADCR0H, ADCR0L)

Figure 11-1. Block Diagram of 10-Bit A/D Converter



Note Specify the valid edge by using bit 3 of the EGP and EGN registers (refer to **Figure 13-5 Formats of External Interrupt Rising Edge Enable Register and External Interrupt Falling Edge Enable Register**).

(1) Successive approximation register (SAR)

This register compares the voltage value of analog input with the value of a voltage tap (compare voltage) from the series resistor string, and holds the result of the comparison starting from the most significant bit (MSB).

When the result is held down to the least significant bit (LSB) (end of A/D conversion), the contents of SAR are transferred to the A/D conversion result register.

(2) A/D conversion result register (ADCR0)

This register holds the result of A/D conversion result. Each time A/D conversion has been completed, the result of the conversion is loaded to this register from the successive approximation register. The higher 8 bits of the conversion result are stored to ADCR0H, and the lower 2 bits are stored to bits 6 and 7 of ADCR0L. ADCR0H and ADCR0L are used in combination as a 16-bit register, ADCR0, which is read by using a 16-bit memory manipulation instruction.

The contents of this register become undefined when the $\overline{\text{RESET}}$ signal is input.

(3) Sample and hold circuit

The sample and hold circuit samples analog input signals sequentially sent from the input circuit on a one-by-one basis, and sends the sampled signals to the voltage comparator. This circuit holds the sampled analog input voltage value during A/D conversion.

(4) Voltage comparator

The voltage comparator compares the analog input with the output voltage of the series resistor string.

(5) Series resistor string

The series resistor string is connected between AV_{REF} and AVss and generates a voltage to be compared with an analog input.

(6) ANI0 to ANI7 pins

These are eight channels of analog input pins of the A/D converter. They input analog signals that are converted to digital values.

Caution Observe the specified input voltage range of ANI0 to ANI7. If a voltage of AV_{REF} or higher, or AVss or lower (even within the range of absolute maximum ratings) is applied to a channel, the converted value of that channel becomes undefined, or the converted value of the other channels may be affected.

(7) AV_{REF} pin

This pin inputs a reference voltage to the A/D converter.

Based on the voltage applied between AV_{REF} and AVss, the signal input to ANI0 to ANI7 is converted into a digital signal.

(8) AVss pin

This is a ground pin of the A/D converter. Be sure to use this pin at the same voltage as that on the V_{SS0} pin always even when the A/D converter is not used.

(9) AV_{DD} pin

This is an analog pin of the A/D converter. Be sure to use this pin at the same voltage as that on the V_{DDO} pin always even when the A/D converter is not used.

11.3 Registers Controlling A/D Converter

The following two registers control the A/D converter:

- A/D converter mode register 0 (ADM0)
- Analog input channel specification register 0 (ADS0)

(1) A/D converter mode register 0 (ADM0)

This register sets conversion time of an analog input to be converted into a digital value, starts/stops conversion operation, and sets an external trigger.

ADM0 is set by a 1-bit or 8-bit memory manipulation instruction.

This register is set to 00H when the $\overline{\text{RESET}}$ signal is input.

Figure 11-2. Format of A/D Converter Mode Register 0

Symbol	⑦	⑥	5	4	3	2	1	0	Address	On reset	R/W
ADM0	ADCS0	TRG0	FR02	FR01	FR00	EGA01	EGA00	0	FF80H	00H	R/W

ADCS0	Controls A/D conversion operation.
0	Stops operation.
1	Enables operation.

TRG0	Selects software/hardware start.
0	Software start
1	Hardware start

FR02	FR01	FR00	Selects A/D conversion time.
0	0	0	144/fx (17.2 μs)
0	0	1	120/fx (14.3 μs)
1	0	0	288/fx (34.4 μs)
1	0	1	240/fx (28.6 μs)
1	1	0	192/fx (22.9 μs)
Others			Setting prohibited

EGA01	EGA00	Specifies valid edge of external trigger.
0	0	No edge is detected.
0	1	Detects falling edge.
1	0	Detects rising edge.
1	1	Detects both rising and falling edges.

Caution **Do not set the A/D conversion time to less than 14 μs.**

Remark ( ): at fx = 8.38-MHz operation

(2) Analog input channel specification register 0 (ADS0)

This register sets the input port of analog voltage to be converted into a digital value.

ADS0 is set by an 8-bit memory manipulation instruction.

This register is set to 00H when the $\overline{\text{RESET}}$ signal is input.

Figure 11-3. Format of Analog Input Channel Specification Register 0

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
ADS0	0	0	0	0	0	ADS02	ADS01	ADS00	FF81H	00H	R/W

ADS02	ADS01	ADS00	Specifies analog input channel.
0	0	0	ANI0
0	0	1	ANI1
0	1	0	ANI2
0	1	1	ANI3
1	0	0	ANI4
1	0	1	ANI5
1	1	0	ANI6
1	1	1	ANI7

11.4 Operation of A/D Converter

11.4.1 Basic operation of A/D converter

<1> Select one channel for A/D conversion by using the analog input channel specification register 0 (ADS0).

<2> The voltage input to the selected analog input channel is sampled by the sample and hold circuit.

<3> When the voltage has been sampled for a specific time, the sample and hold circuit enters the hold status, and holds the input analog voltage until A/D conversion is completed.

<4> Set bit 9 of the successive approximation register (SAR). The tap selector selects (1/2)AV_{REF} as the voltage tap of the series resistor string.

<5> The voltage difference between the voltage tap of the series resistor string and the analog input is compared by the voltage comparator. If the analog input is higher than (1/2)AV_{REF}, the MSB of SAR remains set. If it is less than (1/2)AV_{REF}, the MSB is reset.

<6> Next, bit 8 of SAR is automatically set, and the next voltage difference is compared. Here the voltage tap of the series resistor string is selected as follows, according to the value of bit 9 to which the result of the first comparison has been already set.

- Bit 9 = 1 : (3/4)AV_{REF}
- Bit 9 = 0 : (1/4)AV_{REF}

This voltage tap and analog input voltage are compared, and bit 8 of SAR is manipulated as follows, according to the result of the comparison:

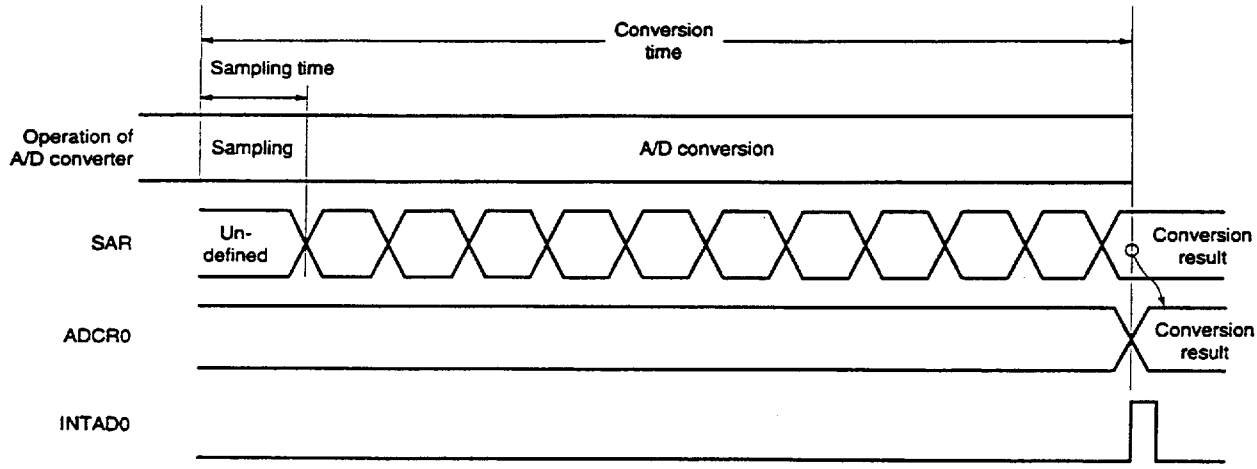
- Analog input voltage ≥ voltage tap : bit 8 = 1
- Analog input voltage ≤ voltage tap : bit 8 = 0

<7> In this way, all the bits of SAR, including bit 0, are compared.

<8> When all the 10 bits of SAR have been compared, SAR holds the valid digital result whose values are transferred and latched to the A/D conversion result register (ADCR0).

At the same time, an A/D conversion end interrupt request (INTAD0) can be generated.

Figure 11-4. Basic Operation of A/D Converter



The A/D conversion is performed continuously, until the bit 7 of ADM0 (ADCS0) is reset to 0 by software.

If the data of the ADM0 register or ADS0 register is rewritten during the A/D conversion, the conversion is initialized. If the ADCS0 bit is set to 1 at this time, conversion is performed again from the start.

The contents of the ADCR0 register become undefined when the $\overline{\text{RESET}}$ signal is input.

11.4.2 Input voltage and conversion result

The relation between the analog voltage input to the analog input pins (ANI0 to ANI7) and A/D conversion result (value stored to A/D conversion result register (ADCR0)) is as follows:

$$ADCR0 = INT\left(\frac{V_{IN}}{AV_{REF}} \times 1024 + 0.5\right)$$

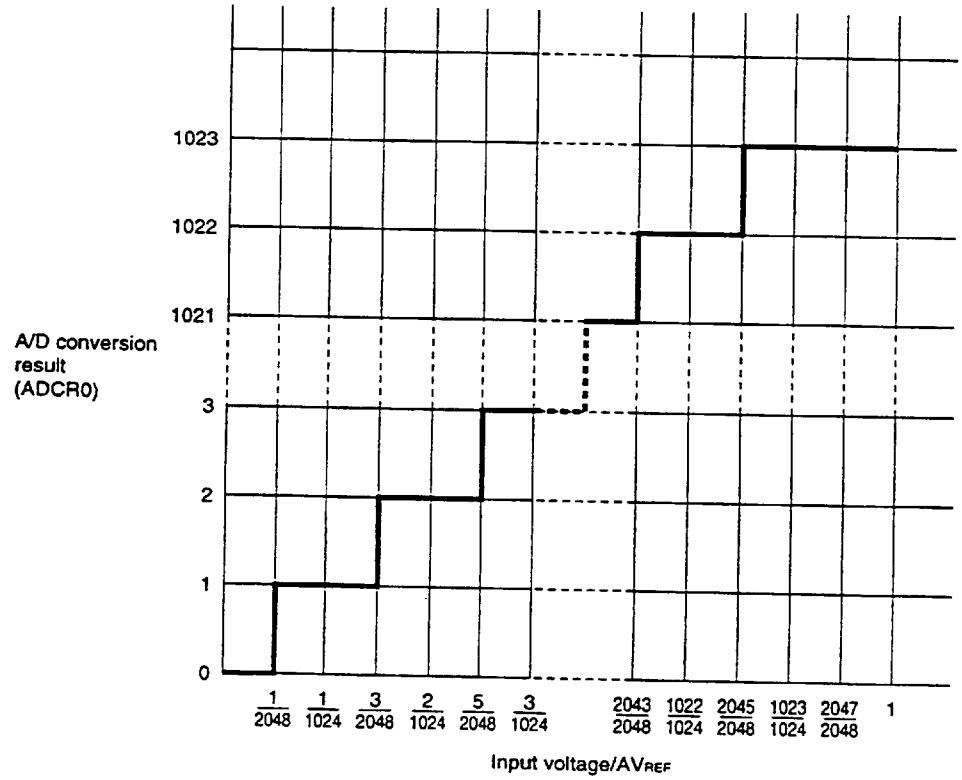
or,

$$(ADCR0 - 0.5) \times \frac{AV_{REF}}{1024} \leq V_{IN} < (ADCR0 + 0.5) \times \frac{AV_{REF}}{1024}$$

Remark INT( ) : function returning integer of value in ( )
V_{IN} : analog input voltage
AV_{REF} : AV_{REF} pin voltage
ADCR0 : value of A/D conversion result register (ADCR0)

Figure 11-5 shows the relations between the analog input voltage and A/D conversion result.

Figure 11-5. Relations between Analog Input Voltage and A/D Conversion Result



11.4.3 Operation mode of A/D converter

The A/D converter operates in the select mode only. One analog input channel is selected from ANI0 to ANI7 for A/D conversion by using the analog input channel specification register 0 (ADS0).

The A/D conversion can be started in the following two ways:

- Hardware start : Conversion is started by trigger input (P03).
- Software start : Conversion is started by setting ADM0.

The result of the A/D conversion is stored in the A/D conversion result register (ADCR0), and at the same time, an interrupt request signal (INTAD0) is generated.

(1) A/D conversion operation by hardware start

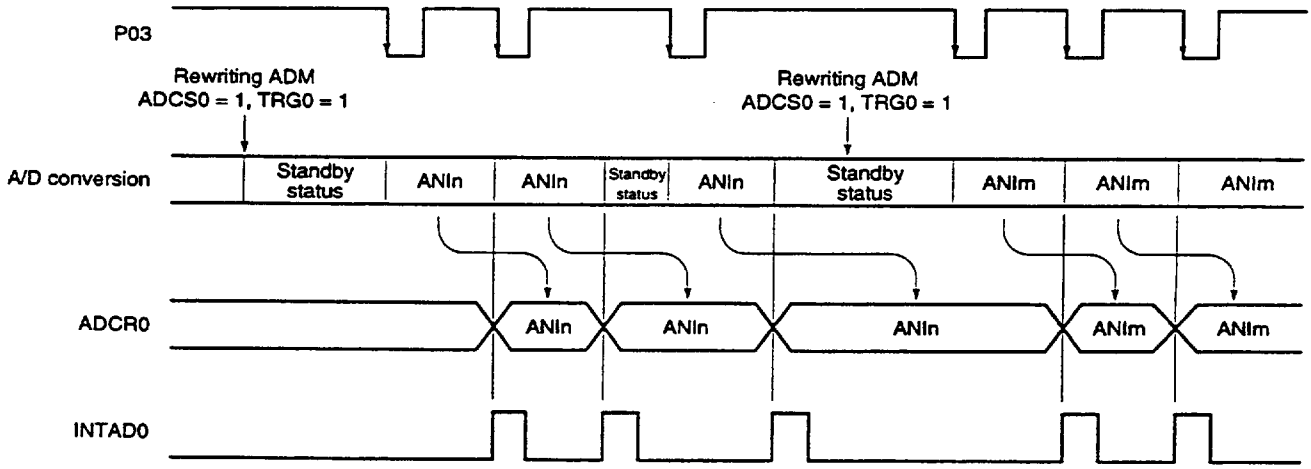
The A/D conversion stands by when both bits 6 (TRG0) and 7 (ADCS0) of A/D converter mode register (ADM0) are set to 1. When an external trigger signal (P03) is input, the voltage applied to the analog input pin specified by ADS0 is converted into a digital value.

When the A/D conversion has been completed, the result of the conversion is stored in the A/D conversion result register (ADCR0), and an interrupt request signal (INTAD0) is generated. Once the A/D conversion has been started and when one A/D conversion has been completed, the next A/D conversion is not started unless a new external trigger signal is input.

If data whose ADCS0 is 1 is written again to ADM0 during A/D conversion, the AD conversion under execution is stopped, and stands by until a new external trigger signal is input. When the external trigger signal is input, A/D conversion is performed again from the start.

When 0 is written to the ADCS0 bit of ADM0 during A/D conversion, the conversion is immediately stopped.

Figure 11-6. A/D Conversion by Hardware Start (When falling edge is specified)



Remark n = 0, 1, ..., 7
m = 0, 1, ..., 7

(2) A/D conversion by software start

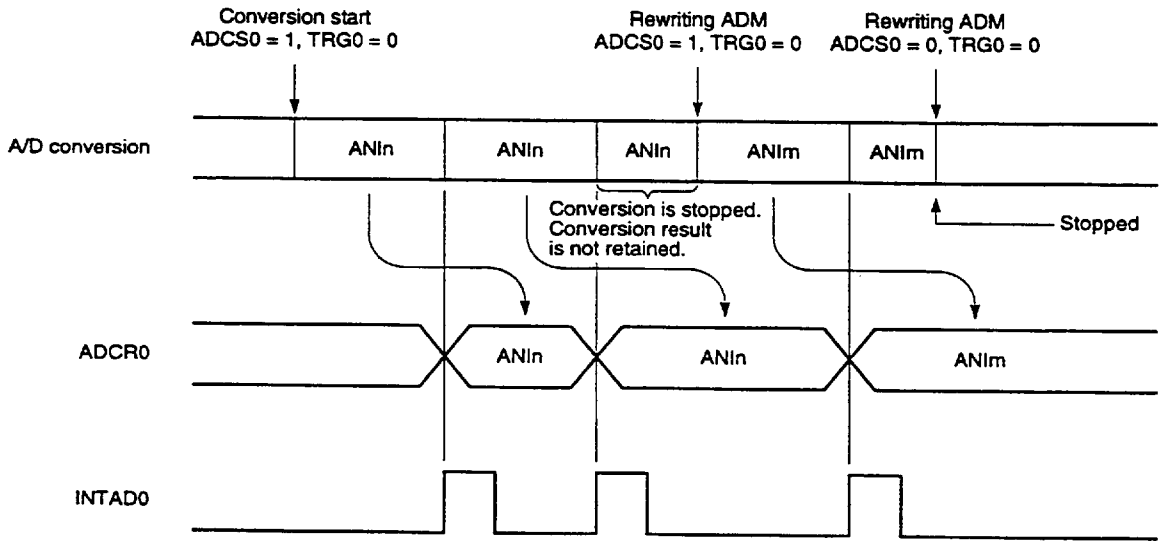
By setting bit 6 (TRG0) of the A/D converter mode register (ADM0) to 0 and setting bit 7 (ADCS0) to 1, the voltage applied to the analog input pin specified by ADS0 is converted into digital values.

When the A/D conversion has been completed, the result of the conversion is stored in the A/D conversion result register (ADCR0), and an interrupt request signal (INTAD0) is generated. When the A/D conversion has been started once, and one A/D conversion has been completed, the next A/D conversion is immediately started. In this way, A/D conversion is repeatedly executed until new data is written to ADM0.

If data whose ADCS0 is 1 is written again to ADM0 during A/D conversion, the conversion under execution is stopped, and the A/D conversion of the newly written data is started.

If data whose ADCS0 is 0 is written to ADM0 during A/D conversion, the conversion is immediately stopped.

Figure 11-7. A/D Conversion by Software Start



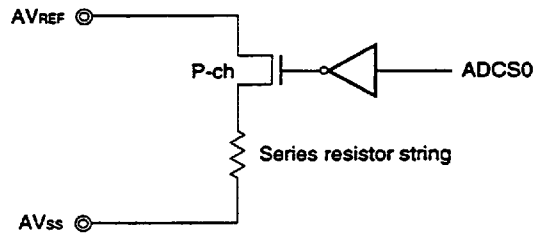
Remark n = 0, 1, ..., 7
m = 0, 1, ..., 7

11.5 Notes on A/D Converter

(1) Current consumption in standby mode

The A/D converter stops operating in the standby mode. At this time, the current consumption can be reduced by stopping the conversion operation (by clearing bit 7 (ADCS0) of the A/D converter mode register 0 (ADM0) to 0). An example of reducing the current consumption in standby mode is shown in Figure 11-8.

Figure 11-8. Example of Reducing Current Consumption in Standby Mode



(2) ANI0 to ANI7 input range

Observe the rated range of the ANI0 to ANI7 input voltage. If a voltage of AVREF or higher, or AVss or lower (even in the range of absolute maximum ratings) is input to an analog input channel, the converted value of that channel becomes undefined. In addition, the converted values of the other channels may also be affected.

(3) Conflicting operations

<1> Conflict between writing A/D conversion result register (ADCR0) on completion of conversion and reading ADCR0 by instruction

Reading ADCR0 takes precedence. After it has been read, a new conversion result is written to ADCR0.

<2> Conflict between writing ADCR0 on completion of conversion and external trigger signal input

The external trigger signal is not accepted during A/D conversion. Therefore, the external trigger signal is not accepted while ADCR0 is written.

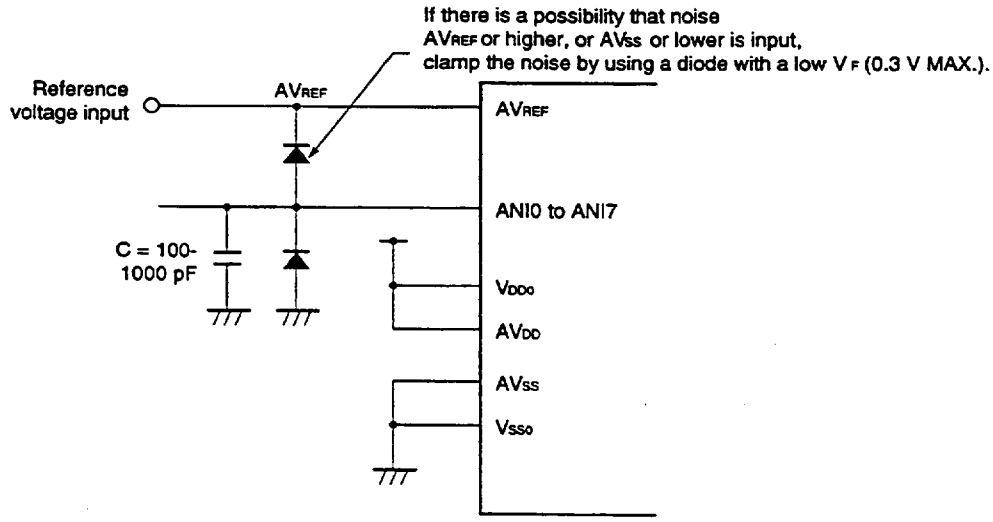
<3> Conflict between writing ADCR0 on completion of conversion and writing A/D converter mode register 0 (ADM0) or writing analog input channel specification register 0 (ADS0)

Writing ADM0 or ADS0 takes precedence. ADCR0 is not written. The conversion end interrupt request signal (INTAD0) is not generated.

(4) Countermeasures against noise

To keep the resolution of 10 bits, noise superimposed on the AVREF and ANI0 to ANI7 pins must be suppressed as much as possible. The higher the output impedance of the analog input source, the greater the effect. To suppress noise, connecting an external capacitor as shown in Figure 11-9 is recommended.

Figure 11-9. Processing Analog Input Pin



(5) ANI0/P10 to ANI7/P17

The analog input pins (ANI0 to ANI7) are also used as input port pins (PORT 1).

When A/D conversion is performed with any of ANI0 to ANI7 selected, do not execute the input instruction of PORT 1 while conversion is in progress; otherwise, the conversion resolution may be degraded.

If a digital pulse is applied to the pins adjacent to the pins currently used for A/D conversion, the expected value of the A/D conversion may not be obtained due to coupling noise. Therefore, do not apply a pulse to the adjacent pins to the pin under A/D conversion.

(6) Input impedance to AVREF pin

A series resistor string of about 47 kΩ is connected between the AVREF and AVss pins.

If the output impedance of the reference voltage source is high, therefore, an error of the reference voltage increases by connecting the impedance in parallel with the series resistor string between the AVREF and AVss pins.

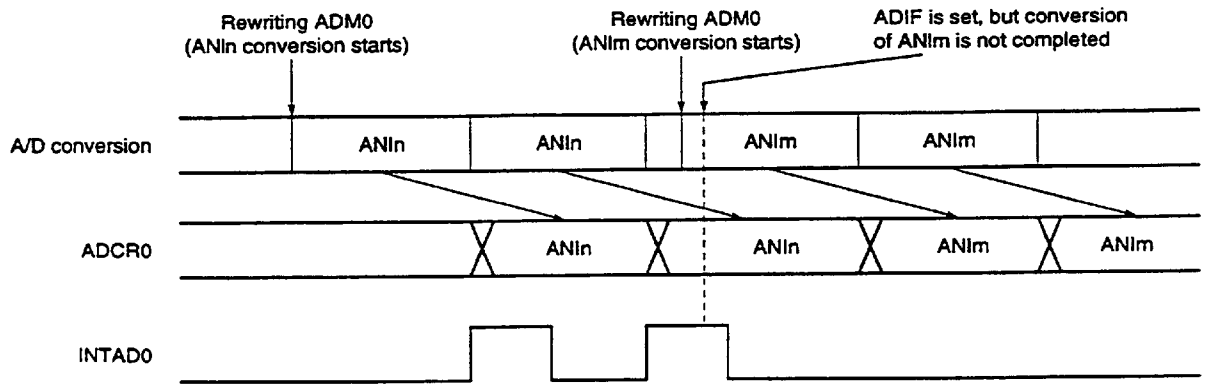
(7) Interrupt request flag (ADIF)

The interrupt request flag (ADIF) is not cleared even when the contents of the A/D converter mode register 0 (ADM0) are changed.

When the analog input pin is changed during A/D conversion, therefore, the chances are that the A/D conversion result of the old analog input and interrupt request flags are set immediately before the contents of ADM0 are rewritten. Consequently, ADIF may be set even if A/D conversion for the newly specified analog input pin has not yet been completed when ADIF is read immediately after ADM0 has been rewritten (refer to **Figure 11-10**).

To resume A/D conversion that has been once stopped, clear ADIF before resuming the conversion.

Figure 11-10. A/D Conversion End Interrupt Generation Timing



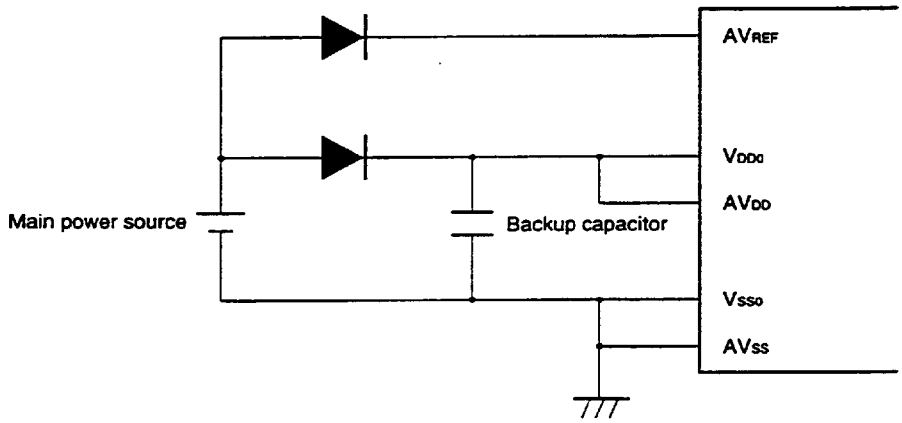
Remark n = 0, 1, ..., 7
n = 0, 1, ..., 7

(8) AV_{DD} pin

The AV_{DD} pin is the power supply pin to the analog circuit and supplies power to the input circuit of ANI0/P10 to ANI7/P17.

Therefore, even in the application which can be switched over to backup power source, be sure to apply the same voltage as V_{DDO} as shown in Figure 11-11.

Figure 11-11. Processing of AV_{DD} Pin



12.1 Serial Interface Functions

Serial interface has the following three modes.

- Operation stop mode
- Asynchronous serial interface (UART) mode
- Infrared data transfer (IrDA) mode (UART00 only)

(1) Operation stop mode

This mode is used when serial transfer is not carried out to reduce power consumption.

(2) Asynchronous serial interface (UART) mode

In this mode, one byte of data is transmitted/received following the start bit, and full-duplex operation is possible.

A dedicated UART baud rate generator is incorporated, allowing communication over a wide range of baud rates. In addition, the baud rate can be defined by scaling the input clock to the ASCK pin.

The MIDI standard baud rate (31.25 kbps) can be used by employing the dedicated UART baud rate generator.

(3) Infrared data transfer (IrDA) mode

12.2 Serial Interface Configuration

Serial interface consists of the following hardware.

Table 12-1. Serial Interface Configuration

Item	Configuration
Register	Transmit shift registers (TXS00, TXS01) Receive shift registers (RX00, RX01) Receive buffer registers (RXB00, RXB01)
Control register	Asynchronous serial interface mode registers (ASIM00, ASIM01) Asynchronous serial interface status registers (ASIS00, ASIS01) Baud rate generator control registers (BRGC00, BRGC01)

Figure 12-1. Serial Interface (UART00) Block Diagram

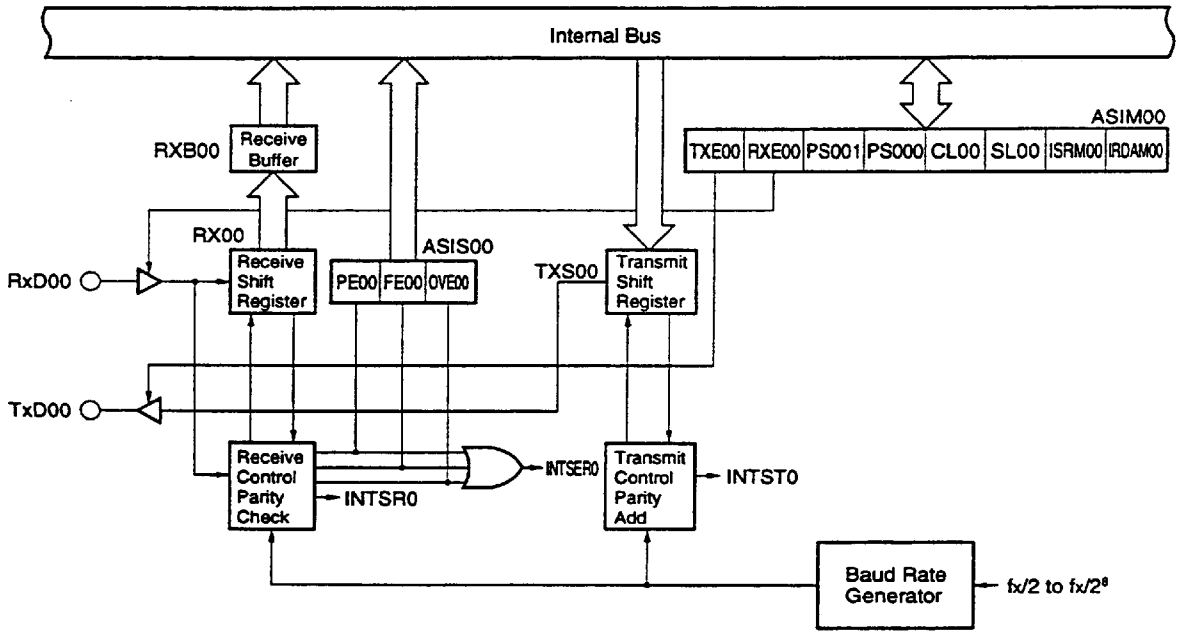
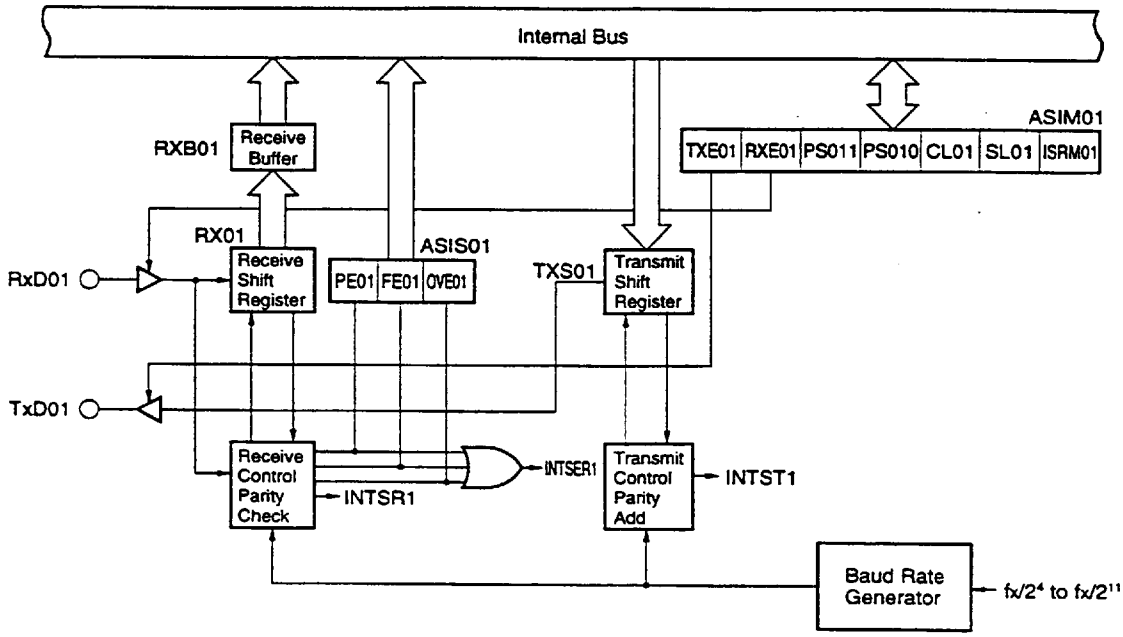


Figure 12-2. Serial Interface (UART01) Block Diagram



(1) Transmit shift register (TXS0n)

This register is used to set the transmit data. The data written in TXS0n is transmitted as serial data.
If the data length is specified as 7 bits, bits 0 to 6 of the data written in TXS0n are transferred as transmit data.
Writing data to TXS0n starts the transmit operation.
TXS0n is written to with an 8-bit memory manipulation instruction. It cannot be read.
TXS0n value is FFH after $\overline{\text{RESET}}$ input.

Caution TXS0n must not be written to during a transmit operation. TXS0n and the receive buffer register (RXB0n) are allocated to the same address, and when a read is performed, the value of RXB0n is read.

(2) Receive shift register (RX0n)

This register is used to convert serial data input to the RxD0n pin to parallel data. When one byte of data is received, the receive data is transferred to the receive buffer register (RXB0n).
RXS0n cannot be directly manipulated by a program.

(3) Receive buffer register (RXB0n)

This register holds receive data. Each time one byte of data is received, new receive data is transferred from the receive shift register (RXS0n).
If the data length is specified as 7 bits, the receive data is transferred to bits 0 to 6 of RXB0n, and the MSB of RXB0n is always set to 0.
RXB0n is read with an 8-bit memory manipulation instruction. It cannot be written to.
RXB0n value is FFH after $\overline{\text{RESET}}$ input.

Caution RXB0n and the transmit shift register (TXS0n) are allocated to the same address, and when a write is performed, the value is written to TXS0n.

(4) Transmission control circuit

This circuit performs transmit operation control such as the addition of a start bit, parity bit and stop bit to data written in the transmit shift register (TXS0n) in accordance with the contents set in the asynchronous serial interface mode register (ASIM0n).

(5) Reception control circuit

This circuit controls receive operations in accordance with the contents set in the asynchronous serial interface mode register (ASIM0n). It performs error checks for parity errors, etc., during a receive operation, and if an error is detected, sets a value in the asynchronous serial interface status register (ASIS0n) in accordance with the error contents.

Remark n = 0, 1

12.3 Registers Controlling Serial Interface

The following six registers control the serial interface.

- Asynchronous serial interface mode registers (ASIM00, ASIM01)
- Asynchronous serial interface status registers (ASIS00, ASIS01)
- Baud rate generator control registers (BRGC00, BRGC01)

(1) Asynchronous serial interface mode registers (ASIM00, ASIM01)

ASIM00 and ASIM01 are 8-bit registers that control the serial transmit operation of the asynchronous serial interface.

These registers are set by a 1-bit or 8-bit memory manipulation instruction.

When the $\overline{\text{RESET}}$ signal is input, these registers are set to 00H.

Figure 12-3. Format of Asynchronous Serial Interface Mode Register 0

Symbol	⑦	⑥	5	4	3	2	1	0	Address	On reset	R/W
ASIM00	TXE00	RXE00	PS001	PS000	CL00	SL00	ISRM00	IRDAM00	FFA0H	00H	R/W

TXE00	RXE00	Operating Mode	Function of RxD00/P20 pin	Function of TxD00/P21 pin
0	0	Stop operation	Port function	Port function
0	1	UART mode (reception only)	Serial function	Port function
1	0	UART mode (transmission only)	Port function	Serial function
1	1	UART mode (transmission/reception)	Serial function	Serial function

PS001	PS000	Parity Bit Specification
0	0	No parity
0	1	Transmission = Always 0 parity addition Reception = Parity not checked (parity error not generated)
1	0	Odd parity
1	1	Even parity

CL00	Character Length Specification
0	7 bits
1	8 bits

SL00	Transmit Data Stop Bit Length Specification
0	1 bit
1	2 bits

ISRM00	Controls Reception End Interrupt on Occurrence of Error.
0	Generates reception end interrupt on occurrence of error.
1	Does not generate reception end interrupt on occurrence of error.

IRDAM00	Specifies Operation of IrDA Mode[Note1].
0	UART (transmission/reception) mode
1	IrDA (transmission/reception) mode[Note2]

Notes 1. The UART or IrDA mode is specified by TXE00 and RXE00.

2. When using the IrDA mode, be sure to clear bits 3 to 0 (MLD003 to MLD000) of the baud rate generator control register (BRGC00) to 0000 (set the clock to fsck/16).

Caution Before changing the operating mode, be sure to stop the serial transmission/reception.

Figure 12-4. Format of Asynchronous Serial Interface Mode Register 1

Symbol	⑦	⑥	5	4	3	2	1	0	Address	On reset	R/W
ASIM01	TXE01	RXE01	PS011	PS010	CL01	SL01	ISRM01	0	FFA4H	00H	R/W

TXE01	RXE01	Operating Mode	Function of RxD01/P22 pin	Function of TxD01/P23 pin
0	0	Stop operation	Port function	Port function
0	1	UART mode (reception only)	Serial function	Port function
1	0	UART mode (transmission only)	Port function	Serial function
1	1	UART mode (transmission/reception)	Serial function	Serial function

PS011	PS010	Parity Bit Specification
0	0	No parity
0	1	Transmission = Always 0 parity addition Reception = Parity not checked (parity error not generated)
1	0	Odd parity
1	1	Even parity

CL01	Character Length Specification
0	7 bits
1	8 bits

SL01	Transmit Data Stop Bit Length Specification
0	1 bit
1	2 bits

ISRM01	Controls Reception End Interrupt on Occurrence of Error.
0	Generates reception end interrupt on occurrence of error.
1	Does not generate reception end interrupt on occurrence of error.

Caution Before changing the operating mode, be sure to stop the serial transmission/reception.

(2) Asynchronous serial interface status registers (ASIS00, ASIS01)

ASIS00 and ASIS01 are the registers that indicate the error contents when a receive error occurs.

These registers can be read by an 8-bit memory manipulation instruction.

When the $\overline{\text{RESET}}$ *signal is input, these registers are set to 00H.*

Figure 12-5. Format of Asynchronous Serial Interface Status Register 0

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
ASIS00	0	0	0	0	0	PE00	FE00	OVE00	FFA1H	00H	R

PE00	Parity Error Flag
0	Parity error does not occur.
1	Parity error occurs (transmit data parity specification and receive data parity do not match).

FE00	Framing Error Flag
0	Framing error does not occur.
1	Framing error occurs[Note1]. (stop bit not detected)

OVE00	Overrun Error Flag
0	Overrun error does not occur.
1	Overrun error occurs[Note2] (Next receive completed before data is read from receive buffer).

Notes 1. Even if the stop bit length is set to 2 bits by using bit 2 (SL00) of the asynchronous serial interface mode register (ASIM00), only 1 stop bit is detected during reception.

2. If an overrun error occurs, be sure to read the receive buffer register (RXB00). Until RXB00 is read, an overrun error persistently occurs each time data has been received.

Figure 12-6. Format of Asynchronous Serial Interface Status Register 1

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
ASIS01	0	0	0	0	0	PE01	FE01	OVE01	FFA5H	00H	R

PE01	Parity Error Flag
0	Parity error does not occur.
1	Parity error occurs (transmit data parity specification and receive data parity do not match).

FE01	Framing Error Flag
0	Framing error does not occur.
1	Framing error occurs[Note1]. (stop bit not detected)

OVE01	Overrun Error Flag
0	Overrun error does not occur.
1	Overrun error occurs[Note2] (Next receive completed before data is read from receive buffer).

Notes 1. Even if the stop bit length is set to 2 bits by using bit 2 (SL01) of the asynchronous serial interface mode register (ASIM01), only 1 stop bit is detected during reception.

2. If an overrun error occurs, be sure to read the receive buffer register (RXB01). Until RXB01 is read, an overrun error persistently occurs each time data has been received.

(3) Baud rate generator control registers (BRGC00, BRGC01)

BRGC00 and BRGC01 are the registers that set the serial clock of the asynchronous serial interface. These registers are set by an 8-bit memory manipulation instruction.

When the $\overline{\text{RESET}}$ signal is input, these registers are set to 00H.

Figure 12-7. Format of Baud Rate Generator Control Register 0

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
BRGC00	0	TPS002	TPS001	TPS000	MDL003	MDL002	MDL001	MDL000	FFA2H	00H	R/W

TPS002	TPS001	TPS000	5-Bit Counter Source Clock Selection
0	0	0	fx/2 (4.19 MHz)
0	0	1	$fx/2^2$ (2.1 MHz)
0	1	0	$fx/2^3$ (1.05 MHz)
0	1	1	$fx/2^4$ (524 kHz)
1	0	0	$fx/2^5$ (262 kHz)
1	0	1	$fx/2^6$ (131 kHz)
1	1	0	$fx/2^7$ (65.5 kHz)
1	1	1	$fx/2^8$ (32.7 kHz)

MDL003	MDL002	MDL001	MDL000	Baud Rate Generator Input Clock Selection	k
0	0	0	0	fsck/16	0
0	0	0	1	fsck/17	1
0	0	1	0	fsck/18	2
0	0	1	1	fsck/19	3
0	1	0	0	fsck/20	4
0	1	0	1	fsck/21	5
0	1	1	0	fsck/22	6
0	1	1	1	fsck/23	7
1	0	0	0	fsck/24	8
1	0	0	1	fsck/25	9
1	0	1	0	fsck/26	10
1	0	1	1	fsck/27	11
1	1	0	0	fsck/28	12
1	1	0	1	fsck/29	13
1	1	1	0	fsck/30	14
1	1	1	1	Setting prohibited	–

Caution If a write to the BRGC00 is performed during communication, the output of baud rate generator may be disrupted, preventing normal communication from continuing. The BRGC00 should therefore not be written to during communication.

Remarks 1. ( ): at fx = 8.38-MHz operation
2. fsck: 5-bit counter source clock
3. k: Value set in bits MDL000 to MDL003 ($0 \leq k \leq 14$)

Figure 12-8. Format of Baud Rate Generator Control Register 1

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
BRGC01	0	TPS012	TPS011	TPS010	MDL013	MDL012	MDL011	MDL010	FFA6H	00H	R/W

TPS012	TPS011	TPS010	5-Bit Counter Source Clock Selection
0	0	0	$f_x/2^4$ (524 kHz)
0	0	1	$f_x/2^5$ (262 kHz)
0	1	0	$f_x/2^6$ (131 kHz)
0	1	1	$f_x/2^7$ (65.5 kHz)
1	0	0	$f_x/2^8$ (32.7 kHz)
1	0	1	$f_x/2^9$ (16.4 kHz)
1	1	0	$f_x/2^{10}$ (8.2 kHz)
1	1	1	$f_x/2^{11}$ (4.1 kHz)

MDL013	MDL012	MDL011	MDL010	Baud Rate Generator Input Clock Selection	k
0	0	0	0	f_{SCK}/16	0
0	0	0	1	f_{SCK}/17	1
0	0	1	0	f_{SCK}/18	2
0	0	1	1	f_{SCK}/19	3
0	1	0	0	f_{SCK}/20	4
0	1	0	1	f_{SCK}/21	5
0	1	1	0	f_{SCK}/22	6
0	1	1	1	f_{SCK}/23	7
1	0	0	0	f_{SCK}/24	8
1	0	0	1	f_{SCK}/25	9
1	0	1	0	f_{SCK}/26	10
1	0	1	1	f_{SCK}/27	11
1	1	0	0	f_{SCK}/28	12
1	1	0	1	f_{SCK}/29	13
1	1	1	0	f_{SCK}/30	14
1	1	1	1	Setting prohibited	–

Caution **If a write to the BRGC01 is performed during communication, the output of baud rate generator may be disrupted, preventing normal communication from continuing. The BRGC01 should therefore not be written to during communication.**

Remarks 1. ( ): at f_x = 8.38-MHz operation
2. f_{SCK}: 5-bit counter source clock
3. k: Value set in bits MDL010 to MDL013 ($0 \leq k \leq 14$)

12.4 Operation of Serial Interface

The following three operating modes are available to the serial interface.

- Operation stop mode
- Asynchronous serial interface (UART) mode
- Infrared data transfer (IrDA) mode (UART00 only)

12.4.1 Operation stop mode

Serial transfer is not executed in this mode. Consequently, the power dissipation can be reduced.
In the operation stop mode, the pins can be used as ordinary port pins.

(1) Register setting

The operation stop mode is set by using the asynchronous serial interface mode register (ASIM00, ASIM01).
ASIM00 and ASIM01 are set by using a 1-bit or 8-bit memory manipulation instruction.
These registers are set to 00H when the $\overline{\text{RESET}}$ signal is input.

Symbol	⑦	⑥	5	4	3	2	1	0	Address	On reset	R/W
ASIM0n	TXE0n	RXE0n	PS0n1	PS0n0	CL0n	SL0n	ISRM0n	IRDAM0n	FFA0H, FFA4H	00H	R/W

TXE0n	RXE0n	Operating Mode	Function of RxD00/P20, RxD01/P22 pins	Function of TxD00/P21, TxD01/P23 pins
0	0	Stop operation	Port function	Port function
0	1	UART mode (reception only)	Serial function	Port function
1	0	UART mode (transmission only)	Port function	Serial function
1	1	UART mode (transmission/reception)	Serial function	Serial function

Caution Before changing the operating mode, be sure to stop the serial transmission/reception.

Remark n = 0, 1

12.4.2 Asynchronous serial interface (UART) mode

In this mode, one byte of data is transmitted/received following a start bit, and full-duplex operation is possible.
A baud rate generator is incorporated, enabling communication to be performed at any of a wide range of baud rates.
The baud rate (31.25 kbps) conforming to the MIDI standard can be used by using the UART-dedicated baud rate generator.

(1) Register setting

The UART mode is set by using the asynchronous serial interface mode registers (ASIM00 and ASIM01), asynchronous serial interface status registers (ASIS00 and ASIS01), and baud rate generator control registers (BRGC00 and BRGC01).

(a) Asynchronous serial interface mode registers (ASIM00, ASIM01)

ASIM00 and ASIM01 are set by a 1-bit or 8-bit memory manipulation instruction.

These registers are set to 00H when the $\overline{\text{RESET}}$ signal is input.

Symbol	⑦	⑥	5	4	3	2	1	0	Address	On reset	R/W
ASIM00	TXE00	RXE00	PS001	PS000	CL00	SL00	ISRM00	IRDAM00	FFA0H	00H	R/W

TXE00	RXE00	Operating Mode	Function of RxD00/P20 pin	Function of TxD00/P21 pin
0	0	Stop operation	Port function	Port function
0	1	UART mode (reception only)	Serial function	Port function
1	0	UART mode (transmission only)	Port function	Serial function
1	1	UART mode (transmission/reception)	Serial function	Serial function

PS001	PS000	Parity Bit Specification
0	0	No parity
0	1	Transmission = Always 0 parity addition Reception = Parity not checked (parity error not generated)
1	0	Odd parity
1	1	Even parity

CL00	Character Length Specification
0	7 bits
1	8 bits

SL00	Transmit Data Stop Bit Length Specification
0	1 bit
1	2 bits

ISRM00	Controls Reception End Interrupt on Occurrence of Error.
0	Generates reception end interrupt on occurrence of error.
1	Does not generate reception end interrupt on occurrence of error.

IRDAM00	Specifies Operation of IrDA Mode[Note1].
0	UART (transmission/reception) mode
1	IrDA (transmission/reception) mode[Note2]

Notes 1. The UART or IrDA mode is specified by TXE00 and RXE00.

2. When using the IrDA mode, be sure to clear bits 3 to 0 (MLD003 to MLD000) of the baud rate generator control register (BRGC00) to 0000 (set the clock to fsck/16).

Symbol	⑦	⑥	5	4	3	2	1	0	Address	On reset	R/W
ASIM01	TXE01	RXE01	PS011	PS010	CL01	SL01	ISRM01	0	FFA4H	00H	R/W

TXE01	RXE01	Operating Mode	Function of RxD01/P22 pin	Function of TxD01/P23 pin
0	0	Stop operation	Port function	Port function
0	1	UART mode (reception only)	Serial function	Port function
1	0	UART mode (transmission only)	Port function	Serial function
1	1	UART mode (transmission/reception)	Serial function	Serial function

PS011	PS010	Parity Bit Specification
0	0	No parity
0	1	Transmission = Always 0 parity addition Reception = Parity not checked (parity error not generated)
1	0	Odd parity
1	1	Even parity

CL01	Character Length Specification
0	7 bits
1	8 bits

SL01	Transmit Data Stop Bit Length Specification
0	1 bit
1	2 bits

ISRM01	Controls Reception End Interrupt on Occurrence of Error.
0	Generates reception end interrupt on occurrence of error.
1	Does not generate reception end interrupt on occurrence of error.

Caution Before changing the operating mode, be sure to stop the serial transmission/reception.

(b) Asynchronous serial interface status registers (ASIS00, ASIS01)

ASIS00 and ASIS01 can be read by an 8-bit memory manipulation instruction.

When the $\overline{\text{RESET}}$ signal is input, these registers are set to 00H.

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
ASIS0n	0	0	0	0	0	PE0n	FE0n	OVE0n	FFA1H, FFA5H	00H	R

PE0n	Parity Error Flag
0	Parity error does not occur.
1	Parity error occurs (transmit data parity specification and receive data parity do not match).

FE0n	Framing Error Flag
0	Framing error does not occur.
1	Framing error occurs[Note1]. (stop bit not detected)

OVE0n	Overrun Error Flag
0	Overrun error does not occur.
1	Overrun error occurs[Note2] (Next receive completed before data is read from receive buffer).

Notes 1. Even if the stop bit length is set to 2 bits by using bit 2 (SL0n) of the asynchronous serial interface mode register (ASIM0n), only 1 stop bit is detected during reception.

2. If an overrun error occurs, be sure to read the receive buffer register (RXB0n). Until RXB0n is read, an overrun error persistently occurs each time data has been received.

Remark n = 0, 1

(c) Baud rate generator control registers (BRGC00, BRGC01)

BRGC00 and BRGC01 are set by an 8-bit memory manipulation instruction.

These registers are set to 00H when the $\overline{\text{RESET}}$ signal is input.

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
BRGC00	0	TPS002	TPS001	TPS000	MDL003	MDL002	MDL001	MDL000	FFA2H	00H	R/W

TPS002	TPS001	TPS000	5-Bit Counter Source Clock Selection
0	0	0	fx/2 (4.19 MHz)
0	0	1	$fx/2^2$ (2.1 MHz)
0	1	0	$fx/2^3$ (1.05 MHz)
0	1	1	$fx/2^4$ (524 kHz)
1	0	0	$fx/2^5$ (262 kHz)
1	0	1	$fx/2^6$ (131 kHz)
1	1	0	$fx/2^7$ (65.5 kHz)
1	1	1	$fx/2^8$ (32.7 kHz)

MDL003	MDL002	MDL001	MDL000	Baud Rate Generator Input Clock Selection	k
0	0	0	0	fsck/16	0
0	0	0	1	fsck/17	1
0	0	1	0	fsck/18	2
0	0	1	1	fsck/19	3
0	1	0	0	fsck/20	4
0	1	0	1	fsck/21	5
0	1	1	0	fsck/22	6
0	1	1	1	fsck/23	7
1	0	0	0	fsck/24	8
1	0	0	1	fsck/25	9
1	0	1	0	fsck/26	10
1	0	1	1	fsck/27	11
1	1	0	0	fsck/28	12
1	1	0	1	fsck/29	13
1	1	1	0	fsck/30	14
1	1	1	1	Setting prohibited	–

Caution If a write to the BRGC00 is performed during communication, the output of baud rate generator may be disrupted, preventing normal communication from continuing. The BRGC00 should therefore not be written to during communication.

Remarks 1. ( ): at fx = 8.38-MHz operation

2. fsck: 5-bit counter source clock

3. k: Value set in bits MDL000 to MDL003 ($0 \leq k \leq 14$)

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
BRGC01	0	TPS012	TPS011	TPS010	MDL013	MDL012	MDL011	MDL010	FFA6H	00H	R/W

TPS012	TPS011	TPS010	5-Bit Counter Source Clock Selection
0	0	0	$f_X/2^4$ (524 kHz)
0	0	1	$f_X/2^5$ (262 kHz)
0	1	0	$f_X/2^6$ (131 kHz)
0	1	1	$f_X/2^7$ (65.5 kHz)
1	0	0	$f_X/2^8$ (32.7 kHz)
1	0	1	$f_X/2^9$ (16.4 kHz)
1	1	0	$f_X/2^{10}$ (8.2 kHz)
1	1	1	$f_X/2^{11}$ (4.1 kHz)

MDL013	MDL012	MDL011	MDL010	Baud Rate Generator Input Clock Selection	k
0	0	0	0	f_{SCK}/16	0
0	0	0	1	f_{SCK}/17	1
0	0	1	0	f_{SCK}/18	2
0	0	1	1	f_{SCK}/19	3
0	1	0	0	f_{SCK}/20	4
0	1	0	1	f_{SCK}/21	5
0	1	1	0	f_{SCK}/22	6
0	1	1	1	f_{SCK}/23	7
1	0	0	0	f_{SCK}/24	8
1	0	0	1	f_{SCK}/25	9
1	0	1	0	f_{SCK}/26	10
1	0	1	1	f_{SCK}/27	11
1	1	0	0	f_{SCK}/28	12
1	1	0	1	f_{SCK}/29	13
1	1	1	0	f_{SCK}/30	14
1	1	1	1	Setting prohibited	–

Caution **If a write to the BRGC01 is performed during communication, the output of baud rate generator may be disrupted, preventing normal communication from continuing. The BRGC01 should therefore not be written to during communication.**

Remarks 1. ( ): at fx = 8.38-MHz operation
2. f_{SCK}: 5-bit counter source clock
3. k: Value set in bits MDL010 to MDL013 ($0 \leq k \leq 14$)

The transmit/receive clock for the baud rate to be generated is obtained by dividing the system clock.

- **Generating transmit/receive clock for baud rate from system clock**
  The transmit/receive clock is generated by dividing the system clock. The baud rate generated from the system clock can be calculated from the following expression:

$$[\text{Baud rate}] = \frac{fx}{2^{m+1}(k+16)} \text{ [Hz]}$$

fx: System clock oscillation frequency
m: Value set by TPS0n0 to TPS0n2
(1 ≤ m ≤ 8 for UART00, 4 ≤ m ≤ 11 for UART01)
k: Value set by MDL0n0 to MDL0n3 (0 ≤ k ≤ 14)

Tables 12-2 and 12-3 show the relations between the source clock of the 5-bit counter and the value of m. Table 12-4 shows the relation between the system clock and baud rate.

Table 12-2. Relations between Source Clock of 5-Bit Counter and Value of m (with UART00)

TPS002	TPS001	TPS000	5-Bit Counter Source Clock Selection	m
0	0	0	fx/2 (4.19 MHz)	1
0	0	1	$fx/2^2$ (2.1 MHz)	2
0	1	0	$fx/2^3$ (1.05 MHz)	3
0	1	1	$fx/2^4$ (524 kHz)	4
1	0	0	$fx/2^5$ (262 kHz)	5
1	0	1	$fx/2^6$ (131 kHz)	6
1	1	0	$fx/2^7$ (65.5 kHz)	7
1	1	1	$fx/2^9$ (32.7 kHz)	8

Remark ( ): at fx = 8.38-MHz operation

Table 12-3. Relations between Source Clock of 5-Bit Counter and Value of m (with UART01)

TPS012	TPS011	TPS010	5-Bit Counter Source Clock Selection	m
0	0	0	$fx/2^4$ (524 kHz)	4
0	0	1	$fx/2^5$ (262 kHz)	5
0	1	0	$fx/2^6$ (131 kHz)	6
0	1	1	$fx/2^7$ (65.5 kHz)	7
1	0	0	$fx/2^8$ (32.7 kHz)	8
1	0	1	$fx/2^9$ (16.4 kHz)	9
1	1	0	$fx/2^{10}$ (8.2 kHz)	10
1	1	1	$fx/2^{11}$ (4.1 kHz)	11

Remark ( ): at fx = 8.38-MHz operation

Table 12-4. Relation between the System Clock and Baud Rate

System Clock fx (MHz)	8.386			8.000			7.3728			5.000			4.1943		
Baud rate (bps)	BRGC00	BRGC01	Error (%)	BRGC00	BRGC01	Error (%)	BRGC00	BRGC01	Error (%)	BRGC00	BRGC01	Error (%)	BRGC00	BRGC01	Error (%)
75	–	7BH	1.10	–	7AH	0.16	–	78H	0	–	70H	1.73	–	6BH	1.14
110	–	73H	–2.0	–	72H	0.16	–	70H	2.27	–	66H	0.88	–	63H	–2.0
150	–	6BH	1.10	–	6AH	0.16	–	68H	0	–	60H	1.73	–	5BH	1.14
300	–	5BH	1.10	–	5AH	0.16	–	58H	0	–	50H	1.73	7BH	4BH	1.14
600	7BH	4BH	1.10	7AH	4AH	0.16	78H	48H	0	70H	40H	1.73	6BH	3BH	1.14
1200	6BH	3BH	1.10	6AH	3AH	0.16	68H	38H	0	60H	30H	1.73	5BH	2BH	1.14
2400	5BH	2BH	1.10	5AH	2AH	0.16	58H	28H	0	50H	20H	1.73	4BH	1BH	1.14
4800	4BH	1BH	1.10	4AH	1AH	0.16	48H	18H	0	40H	10H	1.73	3BH	0BH	1.14
9600	3BH	0BH	1.10	3AH	0AH	0.16	38H	08H	0	30H	00H	1.73	2BH	–	1.14
19200	2BH	–	1.10	2AH	–	0.16	28H	–	0	20H	–	1.73	1BH	–	1.14
31250	21H	–	–1.3	20H	–	0	1DH	–	1.70	14H	–	0	11H	–	–1.3
38400	1BH	–	1.10	1AH	–	0.16	18H	–	0	10H	–	1.73	0BH	–	1.14
76800	0BH	–	1.10	0AH	–	0.16	08H	–	0	00H	–	1.73	–	–	–
115200	02H	–	1.10	01H	–	2.12	00H	–	0	–	–	–	–	–	–

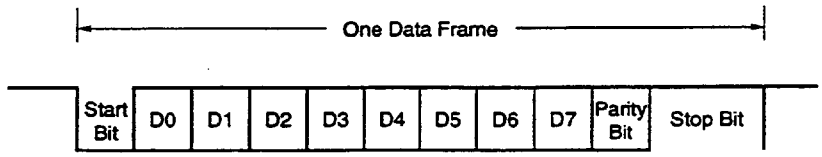
(2) Communication operation

(a) Data format

The transmit/receive data format is as shown in Figure 12-9. One data frame consists of a start bit, character bits, parity bit and stop bit(s).

The specification of character bit length, parity selection, and specification of stop bit length for each data frame is carried out with asynchronous serial interface mode register (ASIM0n).

Figure 12-9. Asynchronous Serial Interface Transmit/Receive Data Format



- Start bits ..... 1 bit
- Character bits ..... 7 bits/8 bits
- Parity bits ..... Even parity/odd parity/0 parity/no parity
- Stop bit(s) ..... 1 bit/2 bits

When 7 bits are selected as the number of character bits, only the lower 7 bits (bits 0 to 6) are valid; in transmission the most significant bit (bit 7) is ignored, and in reception the most significant bit (bit 7) is always "0".

The serial transfer rate is selected by means of the asynchronous serial interface mode register (ASIM0n) and the baud rate generator control register (BRGC0n).

If a serial data receive error is generated, the receive error contents can be determined by reading the status of the asynchronous serial interface status register (ASIS0n).

Remark n = 0, 1

(b) Parity types and operation

The parity bit is used to detect a bit error in the communication data. Normally, the same kind of parity bit is used on the transmitting side and the receiving side. With even parity and odd parity, a one-bit (odd number) error can be detected. With 0 parity and no parity, an error cannot be detected.

(i) Even parity

- **Transmission**

  The number of bits with a value of "1", including the parity bit, in the transmit data is controlled to be even.

  The value of the parity bit is as follows:

  Number of bits with a value of "1" in transmit data is odd: 1
  Number of bits with a value of "1" in transmit data is even: 0

- **Reception**

  The number of bits with a value of "1", including the parity bit, in the receive data is counted. If it is odd, a parity error occurs.

(ii) Odd parity

- **Transmission**

  Conversely to the situation with even parity, the number of bits with a value of "1", including the parity bit, in the transmit data is controlled to be odd. The value of the parity bit is as follows:

  Number of bits with a value of "1" in transmit data is odd: 0
  Number of bits with a value of "1" in transmit data is even: 1

- **Reception**

  The number of bits with a value of "1", including the parity bit, in the receive data is counted. If it is even, a parity error occurs.

(iii) 0 Parity

When transmitting, the parity bit is set to "0" irrespective of the transmit data.

At reception, a parity bit check is not performed. Therefore, a parity error is not generated, irrespective of whether the parity bit is set to "0" or "1".

(iv) No parity

A parity bit is not added to the transmit data. At reception, data is received assuming that there is no parity bit. Since there is no parity bit, a parity error is not generated.

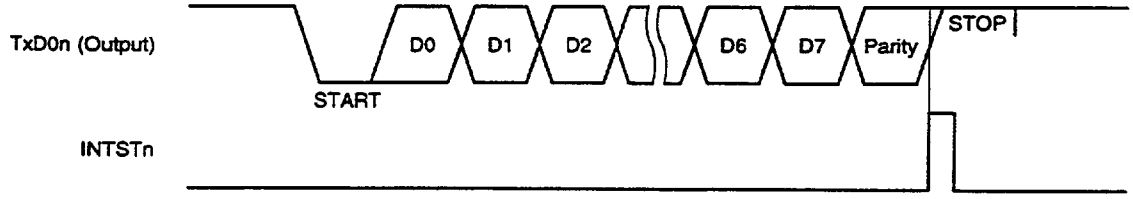
(c) Transmission

A transmit operation is started by writing transmit data to the transmit shift register (TXS0n). The start bit, parity bit and stop bit(s) are added automatically.

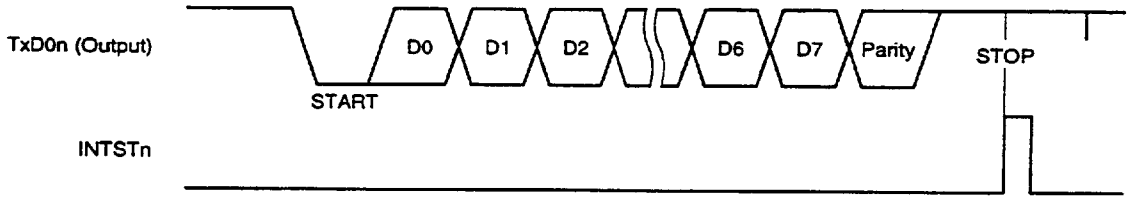
When the transmit operation starts, the data in the transmit shift register (TXS0n) is shifted out, and when the transmit shift register (TXS0n) is empty, a transmission completion interrupt (INTSTn) is generated.

Figure 12-10. Asynchronous Serial Interface Transmission Completion Interrupt Timing

(a) Stop bit length: 1



(b) Stop bit length: 2



Caution **Rewriting of the asynchronous serial interface mode register (ASIM0n) should not be performed during a transmit operation. If rewriting of the ASIM0n register is performed during transmission, subsequent transmit operations may not be possible (the normal state is restored by $\overline{\text{RESET}}$ input).**

It is possible to determine whether transmission is in progress by software by using a transmission completion interrupt (INTSTn) or the interrupt request flag (STIFn) set by the INTSTn.

Remark n = 0, 1

(d) Reception

When the RXE0n bit of the asynchronous serial interface mode register (ASIM0n) is set (1), a receive operation is enabled and sampling of the RxD0n pin input is performed.

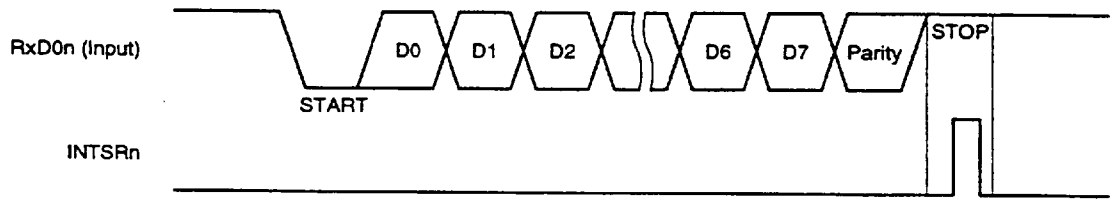
RxD0n pin input sampling is performed using the serial clock specified by ASIM0n.

When the RxD0n pin input becomes low, the 5-bit counter starts counting, and at the time when the half time determined by specified baud rate has passed, the data sampling start timing signal is output. If the RxD0n pin input sampled again as a result of this start timing signal is low, it is identified as a start bit, the 5-bit counter is initialized and starts counting, and data sampling is performed. When character data, a parity bit and one stop bit are detected after the start bit, reception of one frame of data ends. When one frame of data has been received, the receive data in the shift register is transferred to the receive buffer register (RXB0n), and a reception completion interrupt (INTSR0n) is generated.

If an error is generated, the receive data in which the error was generated is still transferred to RXB0n, and INTSRn is generated.

If the RXE0n bit is reset (0) during the receive operation, the receive operation is stopped immediately. In this case, the contents of RXB0n and ASIS0n are not changed, and INTSRn and INTSERn are not generated.

Figure 12-11. Asynchronous Serial Interface Reception Completion Interrupt Timing



Caution **The receive buffer register (RXB0n) must be read even if a receive error is generated. If RXB0n is not read, an overrun error will be generated when the next data is received, and the receive error state will continue indefinitely.**

Remark n = 0, 1

(e) Receive errors

Three kinds of errors can occur during a receive operation: a parity error, framing error, or overrun error. As a result of data reception, if the error flag is set in the asynchronous serial interface status register (ASIS0n), a receive error interrupt (INTSERn) is generated. The receive error interrupt occurs before a receive end interrupt (INTSRn). Receive error causes are shown in Table 12-5.

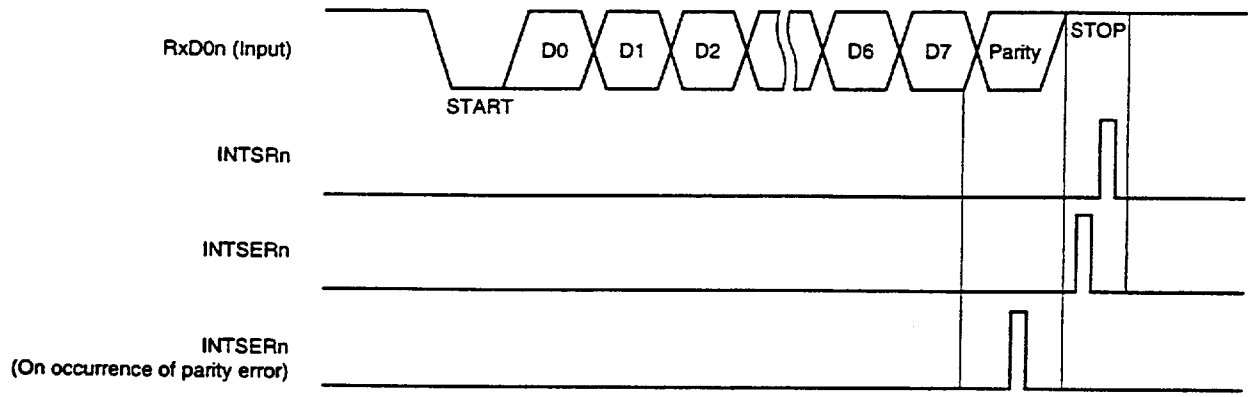
It is possible to determine what kind of error was generated during reception by reading the contents of the asynchronous serial interface status register (ASIS0n) in the reception error interrupt servicing (INTSERn) (see **Figures 12-11** and **12-12**).

The contents of ASIS0n are reset (0) by reading the receive buffer register (RXB0n) or receiving the next data (if there is an error in the next data, the corresponding error flag is set).

Table 12-5. Receive Error Causes

Receive Errors	Cause
Parity error	Transmission-time parity specification and reception data parity do not match
Framing error	Stop bit not detected
Overrun error	Reception of next data is completed before data is read from receive buffer register

Figure 12-12. Receive Error Timing



Cautions 1. The contents of the ASIS0n register are reset (0) by reading the receive buffer register (RXB0n) or receiving the next data. To ascertain the error contents, ASIS must be read before reading RXB0n.

2. The receive buffer register (RXB0n) must be read even if a receive error is generated. If RXB0n is not read, an overrun error will be generated when the next data is received, and the receive error state will continue indefinitely.

Remark n = 0, 1

12.4.3 Infrared data transfer (IrDA) mode

Caution The infrared data transfer (IrDA) mode can be used only with UART00.

(1) Data format

Figure 12-13 shows the format of the IrDA mode in comparison with the data format in UART mode. The IR frame corresponds the bit string of the UART frame that consists of 8 data bits and 1 stop bit. The length of the electrical pulse transmitted or received in the IR frame is 3/16 of a 1-bit cycle. The pulse 3/16 of a 1-bit cycle rises in the middle of the bit cycle (refer to the figure below).

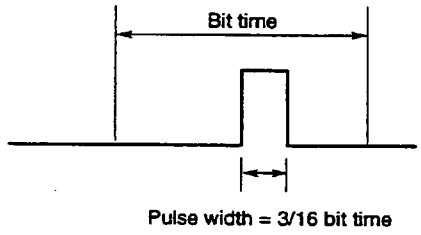
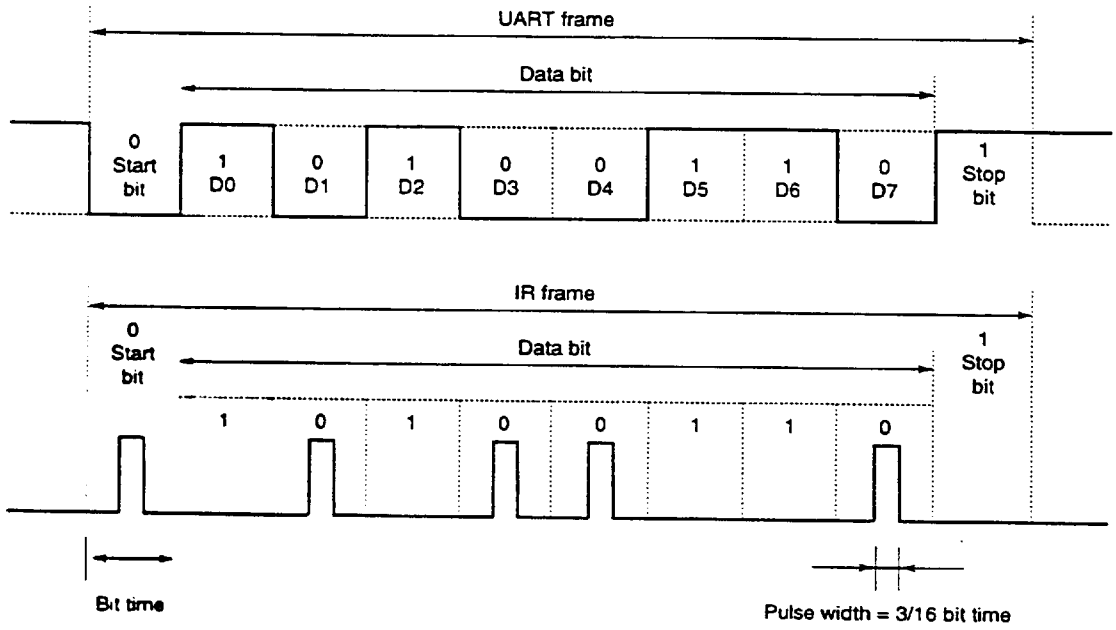


Figure 12-13. Comparison of Data Formats in IrDA Mode and UART Mode



(2) Bit rate and pulse width

Table 12-6 shows the values of the bit rate, permissible bit rate error, and pulse width.

Because the minimum pulse width may be 3/16 of the bit cycle or the minimum pulse width of the signal of 115.2 kbps (permissible error: 1.63 μs to 22 μs), the value is the same at any bit rate.

The maximum pulse width is the sum of bit time 3/16 and 2.5% of the bit cycle or 1.08 μs whichever greater.

Table 12-6. Bit Rate and Pulse Width

Bit Rate (kbits/s)	Permissible Bit Rate Error (% of bit rate)	Minimum Pulse Width (μs)Note	Nominal Value of Pulse Width 3/16 (μs)	Maximum Pulse Width (μs)
2.4	+/– 0.87	1.41	78.13	88.55
9.6	+/– 0.87	1.41	19.53	22.13
19.2	+/– 0.87	1.41	9.77	11.07
38.4	+/– 0.87	1.41	4.88	5.96
57.6	+/– 0.87	1.41	3.26	4.34
115.2	+/– 0.87	1.41	1.63	2.71

Note Where a digital noise elimination circuit is used with the microcontroller at a frequency of 1.41 MHz or higher

The maximum pulse width can be calculated as follows where the bit rate is 2.4 kbps.

$$78.13 + (78.13 \times \frac{16}{3} \times 0.025) = 88.55\ [\mu s]$$

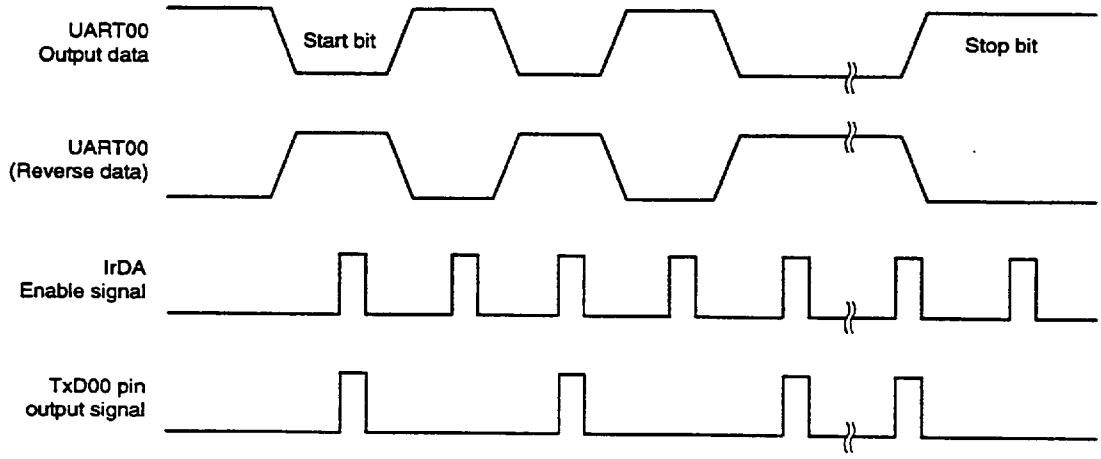
(3) Baud rate that can be set in IrDA mode

Table 12-7. Baud Rate That Can Be Set in IrDA Mode

System Clock fx (MHz)	Baud Rate (bps)
8.386	131031
8.000	125000
7.3728	115200
5.000	78125
4.1943	65536

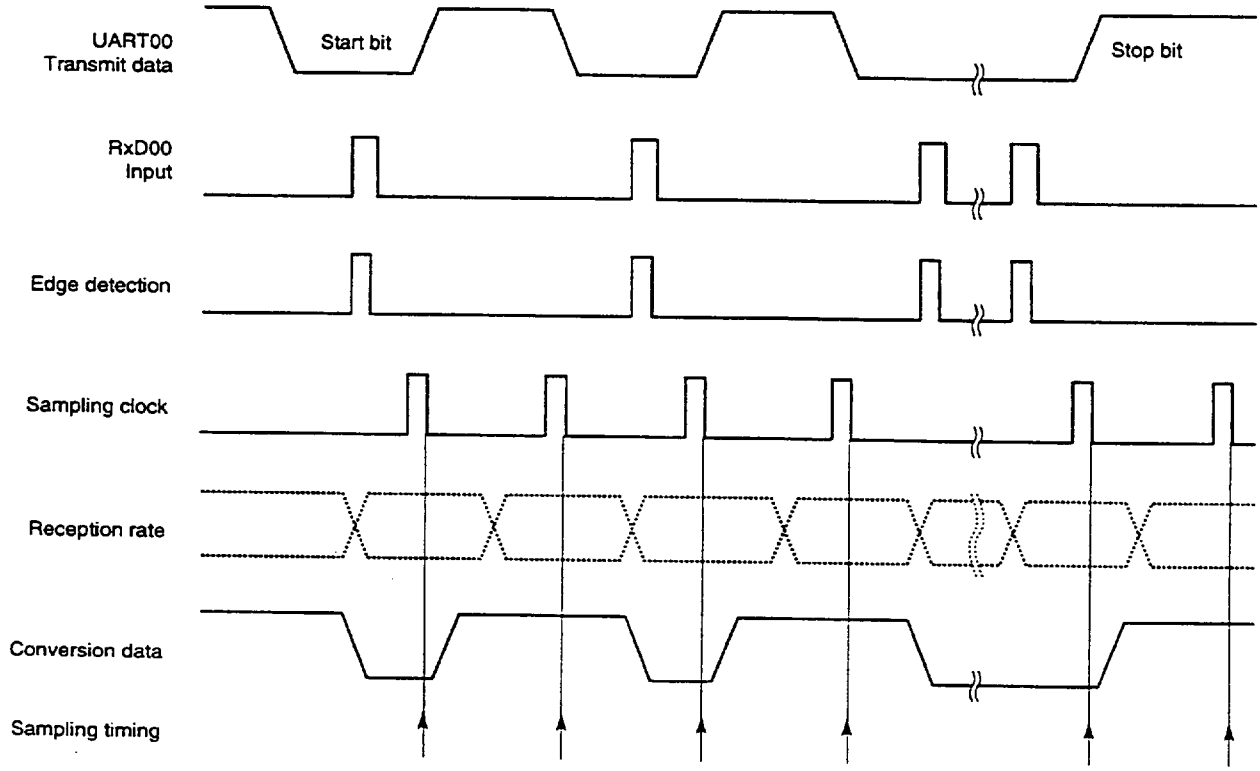
(4) I/O data and internal signal

- **Transmission timing**



- **Reception timing**

Data reception is delayed by half the set baud rate.



[MEMO]

PAGE(S) INTENTIONALLY BLANK

CHAPTER 13 INTERRUPT FUNCTIONS

13.1 Types of Interrupt Functions

The following three types of interrupt functions are available:

(1) Non-maskable interrupts

This interrupt is unconditionally accepted even in the interrupt disabled status. It is not subject to interrupt priority control and therefore takes precedence over all interrupt requests.
This interrupt generates a standby release signal.
The non-maskable interrupts have one interrupt from the watchdog timer.

(2) Maskable interrupts

These interrupts are subject to mask control, and can be divided into two groups according to the setting of the priority specification flag register (PR0L, PR0H): one with higher priority and the other with lower priority. Higher-priority interrupts can nest lower-priority interrupts. The priority when two or more interrupt requests with the same priority occur at the same time is predetermined (refer to **Table 13-1**).
This interrupt generates a standby release signal.
As the maskable interrupts, four external interrupts and twelve internal interrupts are available.

(3) Software interrupts

This is a vectored interrupt generated when the BRK instruction is executed and can be accepted even in the interrupt disabled status. This interrupt is not subject to interrupt priority control.

13.2 Interrupt Sources and Configuration

A total of 18 interrupt sources including non-maskable, maskable, and software interrupt sources are available (refer to **Table 13-1**).

Table 13-1. Interrupt Source List

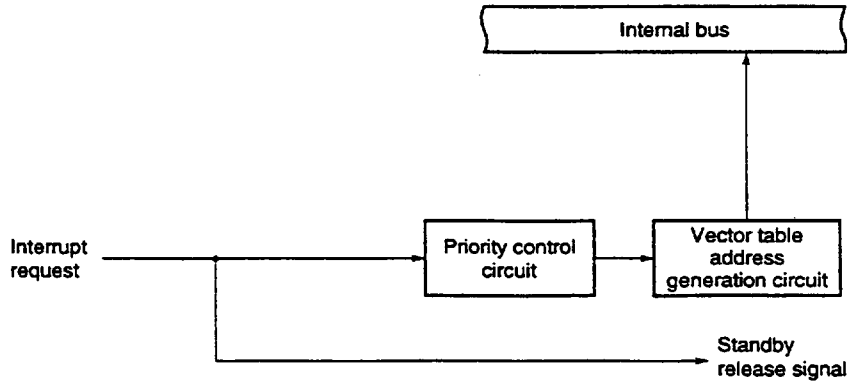
Interrupt Type	Note 1 Default Priority	Interrupt Source		Internal/ External	Vector Table Address	Note 2 Basic Configuration Type
		Name	Trigger			
Non-maskable	—	INTWDT	Watchdog timer overflow (when non-maskable interrupt is selected)	Internal	0004H	(A)
Maskable	0	INTWDT	Watchdog timer overflow (when interval timer mode is selected)			(B)
	1	INTP0	Pin input edge detection	External	0006H	(C)
	2	INTP1			0008H	
	3	INTP2			000AH	
	4	INTP3			000CH	
	5	INTTM7	TM7 under flow	Internal	000EH	(B)
	6	INTSER0	UART00 receive error generation		0010H	
	7	INTSR0	UART00 receive termination		0012H	
	8	INTST0	UART00 transmit termination		0014H	
	9	INTSER1	UART01 receive error generation		0016H	
	10	INTSR1	UART01 receive termination		0018H	
	11	INTST1	UART01 transmit termination		001AH	
	12	INTTM50	TM50 and CR50 match signal generation		001CH	
	13	INTTM51	TM51 and CR51 match signal generation		001EH	
	14	INTTM52	TM52 and CR52 match signal generation		0020H	
	15	INTAD0	A/D conversion termination		0022H	
Software	—	BRK	Execution of BRK instruction	—	003EH	(D)

Notes 1. The default priority is the priority applicable when more than one maskable interrupt is generated. 0 is the highest priority and 15, the lowest.

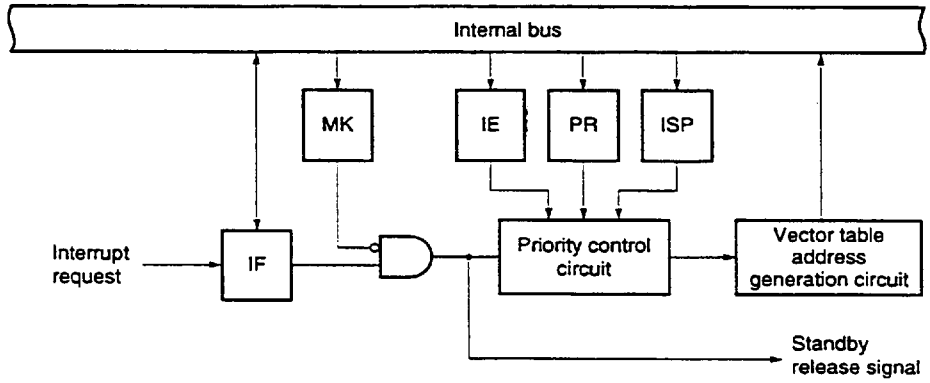
2. Basic configuration types (A) to (D) correspond to (A) to (D) on the next pages.

Figure 13-1. Basic Configuration of Interrupt Function (1/2)

(A) Internal non-maskable interrupt



(B) Internal maskable interrupt



(C) External maskable interrupt

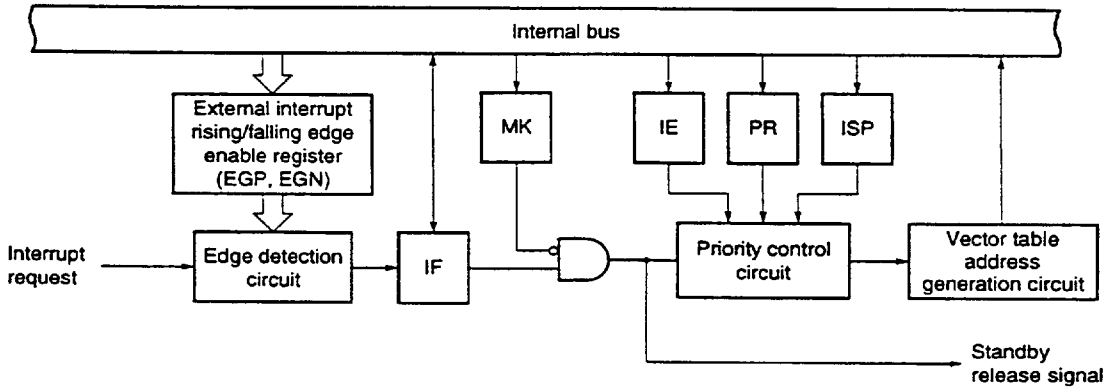
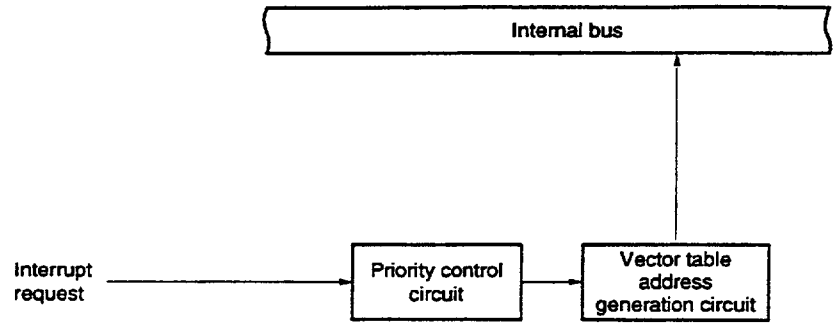


Figure 13-1. Basic Configuration of Interrupt Function (2/2)

(D) Software interrupt



IF : interrupt request flag
IE : interrupt enable flag
ISP : in-service priority flag
MK : interrupt mask flag
PR : priority specification flag

13.3 Registers Controlling Interrupt Function

The following six types of registers control the interrupt function:

- Interrupt request flag registers (IF0L, IF0H)
- Interrupt mask flag registers (MK0L, MK0H)
- Priority specification flag registers (PR0L, PR0H)
- External interrupt rising edge enable register (EGP)
- External interrupt falling edge enable register (EGN)
- Program status word (PSW)

Table 13-2 shows the names of the interrupt request flags, interrupt mask flags, and priority specification flags corresponding to the respective interrupt request sources.

Table 13-2. Flags Corresponding to Respective Interrupt Request Sources

Interrupt Request Signal Name	Interrupt Request Flag	Interrupt Mask Flag	Priority Specification Flag
INTP0	PIF0	PMK0	PPR0
INTP1	PIF1	PMK1	PPR1
INTP2	PIF2	PMK2	PPR2
INTP3	PIF3	PMK3	PPR3
INTTM7	TMIF7	TMMK7	TMPR7
INTTM50	TMIF50	TMMK50	TMPR50
INTTM51	TMIF51	TMMK51	TMPR51
INTTM52	TMIF52	TMMK52	TMPR52
INTWDT	WDTIF[Note]	WDTMK[Note]	WDTPR[Note]
INTSER0	SERIF0	SERMK0	SERPR0
INTSR0	SRIF0	SRMK0	SRPR0
INTST0	STIF0	STMK0	STPR0
INTSER1	SERIF1	SERMK1	SERPR1
INTSR1	SRIF1	SRMK1	SRPR1
INTST1	STIF1	STMK1	STPR1
INTAD0	ADIF	ADMK	ADPR

Note Interrupt control flag when the watchdog timer is used as an interval timer

(1) Interrupt request flag registers (IF0L, IF0H)

An interrupt request flag is set to 1 when the corresponding interrupt request is generated or when an instruction is executed, and is cleared to 0 when the interrupt request is accepted, when the $\overline{\text{RESET}}$ signal is input, or when an instruction is executed.

IF0L and IF0H are set by a 1-bit or 8-bit memory manipulation instruction. When using IF0L and IF0H as a 16-bit register IF0, it is set by a 16-bit memory manipulation instruction.

These registers are set to 00H when the $\overline{\text{RESET}}$ signal is input.

Figure 13-2. Format of Interrupt Request Flag Registers

Symbol	⑦	⑥	⑤	④	③	②	①	⑩	Address	On reset	R/W
IF0L	SRIF0	SERIF0	TMIF7	PIF3	PIF2	PIF1	PIF0	WDTIF	FFE0H	00H	R/W
IF0H	ADIF	TMIF52	TMIF51	TMIF50	STIF1	SRIF1	SERIF1	STIF0	FFE1H	00H	R/W

××IF	Interrupt request flag
0	Interrupt request signal is not generated.
1	Interrupt request signal is generated and interrupt is requested.

Cautions 1. The WDTIF flag can be read/written only when the watchdog timer is used as an interval timer. Clear the WDTIF flag to 0 when the watchdog timer mode 1 is used.

2. Because port 0 is shared with external interrupt inputs, the corresponding interrupt request flag is set when the output mode is specified and output level of a port pin is changed.

To use the port in the output mode, therefore, set the corresponding interrupt mask flag to 1 in advance.

(2) Interrupt mask flag registers (MK0L, MK0H)

An interrupt mask flag enables or disables the corresponding maskable interrupt processing and releasing the standby mode.

MK0L and MK0H are set by a 1-bit or 8-bit memory manipulation instruction. When using MK0L and MK0H as a 16-bit register MK0, it is set by a 16-bit memory manipulation instruction.

These registers are reset to FFH when the $\overline{\text{RESET}}$ signal is input.

Figure 13-3. Format of Interrupt Mask Flag Register

Symbol	⑦	⑥	⑤	④	③	②	①	⑦	Address	On reset	R/W
MK0L	SRMK 0	SERMK 0	TMMK 7	PMK3	PMK2	PMK1	PMK0	WDTMK	FFE4H	FFH	R/W
MK0H	ADMK	TMMK 52	TMMK 51	TMMK 50	STMK 1	SRMK 1	SERMK 1	STMK0	FFE5H	FFH	R/W

xxMK	Controls interrupt processing.
0	Enables interrupt processing.
1	Disables interrupt processing.

Caution Because port 0 is shared with external interrupt inputs, the corresponding interrupt request flag is set when the output mode is specified and output level of a port pin is changed. To use the port in the output mode, therefore, set the corresponding interrupt mask flag to 1 in advance.

(3) Priority specification flag registers (PR0L, PR0H)

A priority specification flag sets the priority of the corresponding maskable interrupt.

PR0L and PR0H are set by a 1-bit or 8-bit memory manipulation instruction. When using PR0L and PR0H as a 16-bit register PR0, it is set by a 16-bit memory manipulation instruction.

These registers are set to FFH when the $\overline{\text{RESET}}$ signal is input.

Figure 13-4. Format of Priority Specification Flag Register

Symbol	⑦	⑥	⑤	④	③	②	①	⓪	Address	On reset	R/W
PR0L	SRPR 0	SERPR 0	TMPR 7	PPR3	PPR2	PPR1	PPR0	WDTPR	FFE8H	FFH	R/W
	⑦	⑥	⑤	④	③	②	①	⓪			
PR0H	ADPR	TMPR 52	TMPR 51	TMPR 50	STPR 1	SRPR 1	SERPR 1	STPR 0	FFE9H	FFH	R/W

××PR	Selects priority level.
0	High priority level
1	Low priority level

(4) External interrupt rising edge enable register (EGP), external interrupt falling edge enable register (EGN)

EGP and EGN specify the valid edge to be detected on pins P00 to P03.

EGP and EGN can be read or written to with a 1-bit or 8-bit memory manipulation instruction.

These registers are set to 00H when the $\overline{\text{RESET}}$ signal is input.

Figure 13-5. Formats of External Interrupt Rising Edge Enable Register and External Interrupt Falling Edge Enable Register

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
EGP	0	0	0	0	EGP3	EGP2	EGP1	EGP0	FF48H	00H	R/W

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
EGN	0	0	0	0	EGN3	EGN2	EGN1	EGN0	FF49H	00H	R/W

EGPn	EGNn	Valid edge of INTPn pin (n = 0-3)
0	0	Interrupt disable
0	1	Falling edge
1	0	Rising edge
1	1	Both rising and falling edges

(5) Program status word (PSW)

The program status word is a register that holds the instruction execution result and current status of interrupt request. An IE flag that enables/disables the maskable interrupts and an ISP flag that controls multiplexed interrupts processing are mapped to this register.

This register can be read or written in 8-bit units. In addition, it can also be manipulated by using a bit manipulation instruction or dedicated instructions (EI and DI). When a vectored interrupt is accepted, and when the BRK instruction is executed, PSW is automatically saved to the stack. At this time, the IE flag is reset to 0. If a maskable interrupt has been accepted the content of the priority flag of that interrupt is transferred to ISP flag. The contents of PSW can also be saved to the stack by the PUSH PSW instruction, and restored from the stack by RETI, RETB or POP PSW instruction.

PSW is set to 02H when the $\overline{\text{RESET}}$ signal is input.

Figure 13-6. Configuration of Program Status Word

Symbol	7	6	5	4	3	2	1	0	On reset
PSW	IE	Z	RBS1	AC	RBS0	0	ISP	CY	02H

Used when normal instruction is executed

ISP	Priority of interrupt currently processed
0	Interrupt with higher priority is processed (interrupt with lower priority is disabled).
1	Interrupt is not accepted, or interrupt with lower priority is processed (all maskable interrupts are enabled).

IE	Enables/disables accepting interrupt.
0	Disables.
1	Enables.

13.4 Interrupt Processing Operation

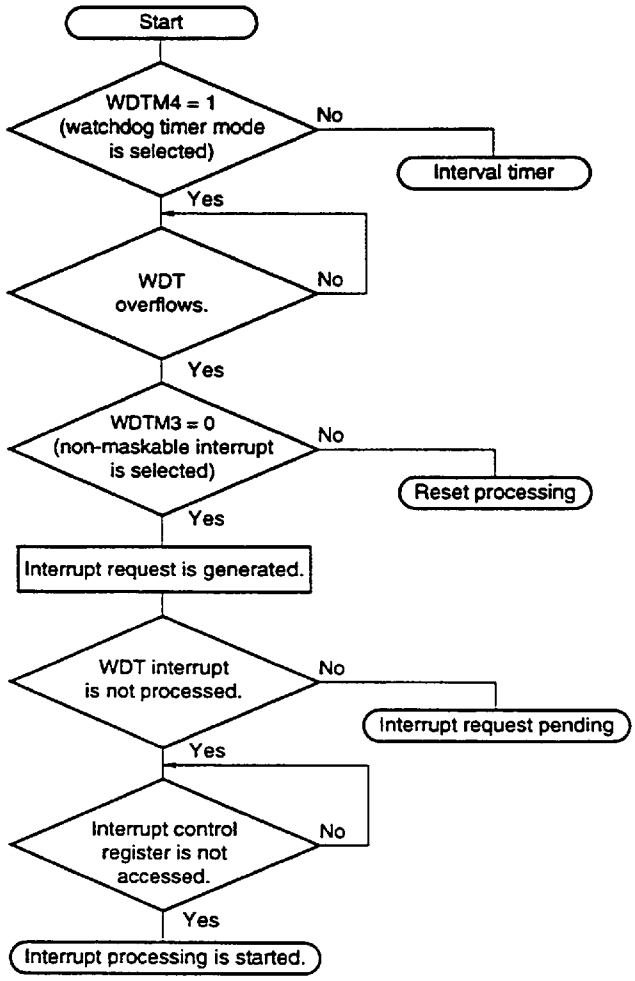
13.4.1 Non-maskable interrupt acceptance operation

The non-maskable interrupt is unconditionally accepted even when interrupts are disabled. It is not subject to interrupt priority control and takes precedence over all other interrupts.

When the non-maskable interrupt request is acknowledged, program status word (PSW) and program counter (PC) are saved to the stack in that order, the IE flag and ISP flag are reset to 0, the contents of the vector table are loaded to the PC, and then program execution branches.

If a new non-maskable interrupt request is generated while the non-maskable interrupt service program is executed, the interrupt request is accepted when the current execution of the non-maskable interrupt service program has been completed (after the RETI instruction has been executed) and one instruction in the main routine has been executed. If two or more new non-maskable interrupt requests are generated while the non-maskable interrupt service program is executed, only one non-maskable interrupt request is accepted after execution of the non-maskable interrupt service program has been completed.

Figure 13-7. Flowchart of Non-Maskable Interrupt Acceptance



WDTM : watchdog timer mode register
WDT : watchdog timer

Figure 13-8. Timing of Non-Maskable Interrupt Acceptance

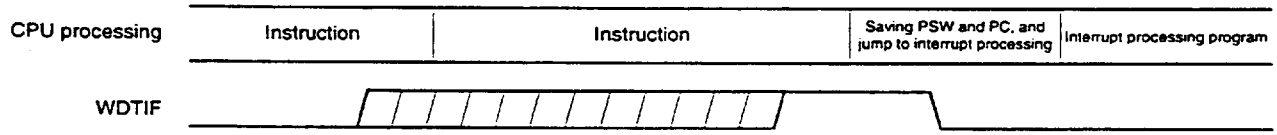
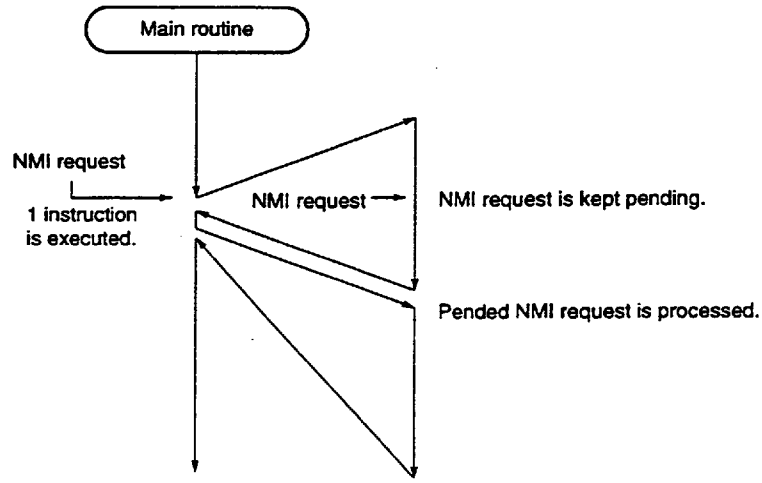
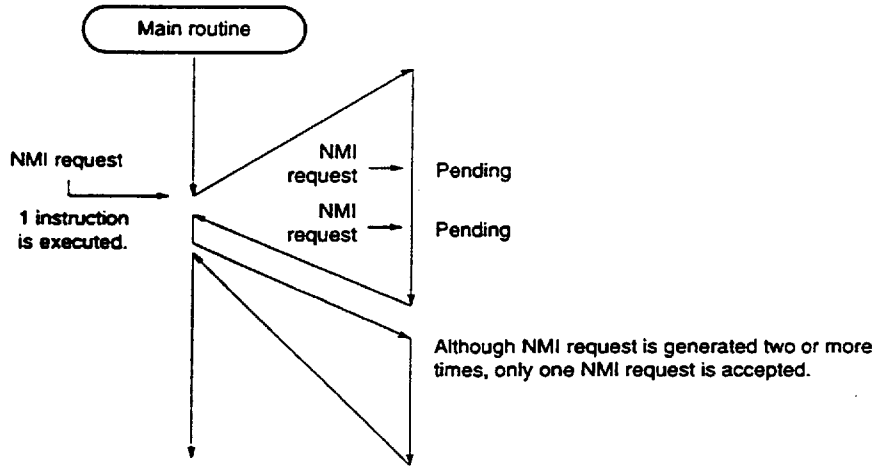


Figure 13-9. Accepting Non-Maskable Interrupt Request

(a) When new non-maskable interrupt request is generated while non-maskable interrupt service program is executed



(b) If two new non-maskable interrupt requests are generated while non-maskable interrupt service program is executed



13.4.2 Maskable interrupt acceptance operation

A maskable interrupt can be accepted when the interrupt request flag is set to 1 and the corresponding interrupt mask flag is cleared to 0. A vectored interrupt is accepted in the interrupt enabled status (when the IE flag is set to 1). However, an interrupt with a lower priority cannot be accepted while an interrupt with a higher priority is being processed (when the ISP flag is reset to 0).

The time required to start the interrupt processing after a maskable interrupt request has been generated is as follows:

Table 13-3. Time from Generation of Maskable Interrupt Request to Processing

	Minimum Time	Maximum Time[Note]
When xxPR = 0	7 clocks	32 clocks
When xxPR = 1	8 clocks	33 clocks

Note The wait time is maximum when an interrupt request is generated immediately before a division instruction.

Remark 1 clock : $\frac{1}{f_{CPU}}$ (f_{CPU} : CPU clock)

When two or more maskable interrupt requests are generated at the same time, they are accepted starting from the one assigned the highest priority by the priority specification flag. When interrupts are assigned the same priority, the default priority takes precedence.

A pended interrupt is accepted when the status where it can be accepted is set.

Figure 13-10 shows the algorithm of accepting interrupts.

When a maskable interrupt request is accepted, the program status word (PSW) and program counter (PC) are saved to the stack in that order, IE flag is reset to 0, and the content of the interrupt priority specification flag of the accepted interrupt is transferred to the ISP flag. In addition, the data in the vector table determined for each interrupt request is loaded to the PC, and execution branches.

To return from interrupt processing, use the RETI instruction.

Figure 13-10. Interrupt Acceptance Program Algorithm

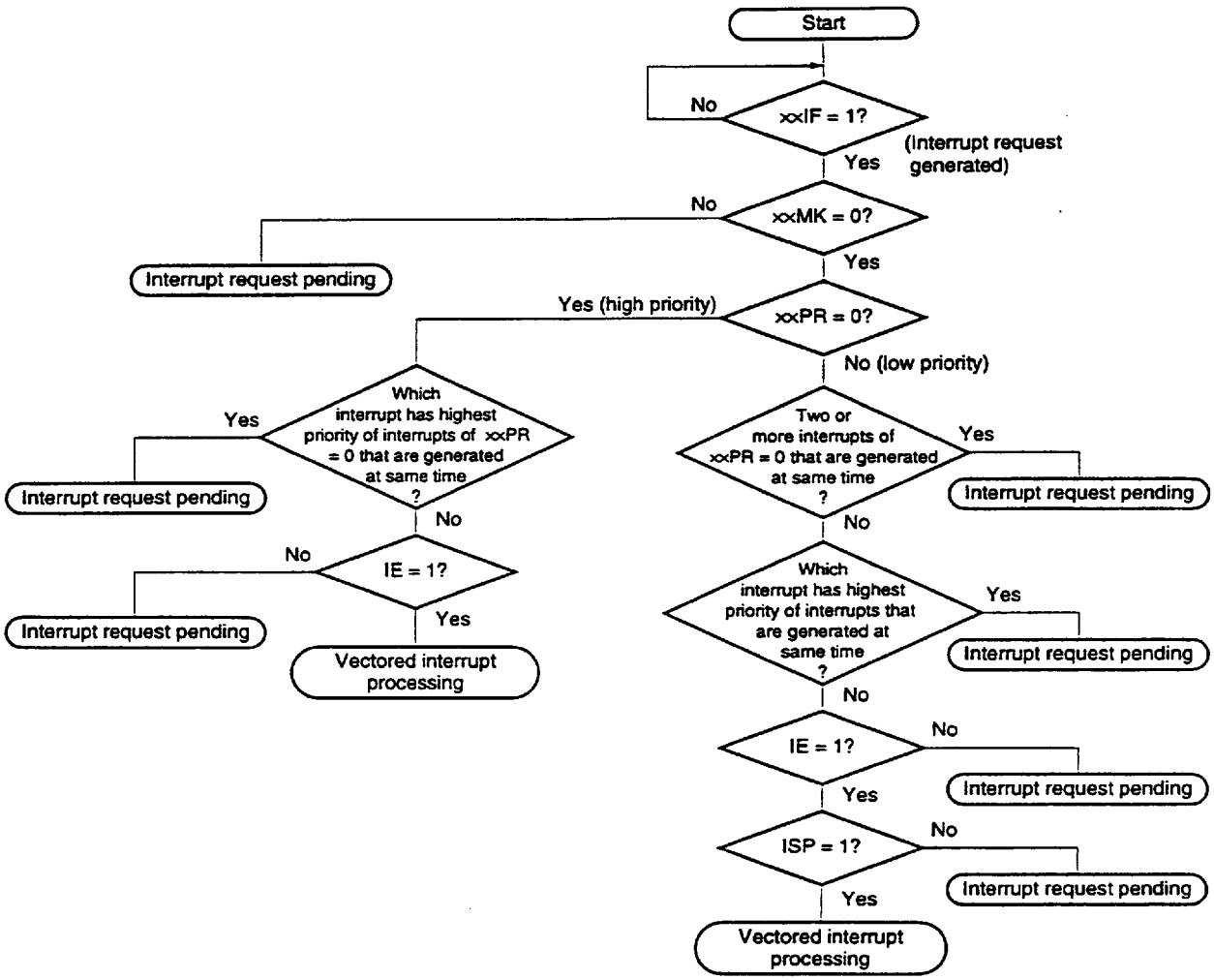
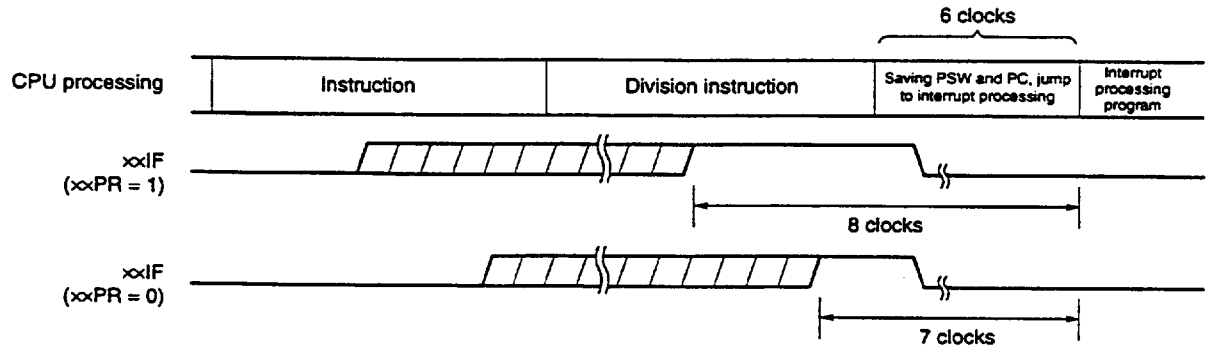
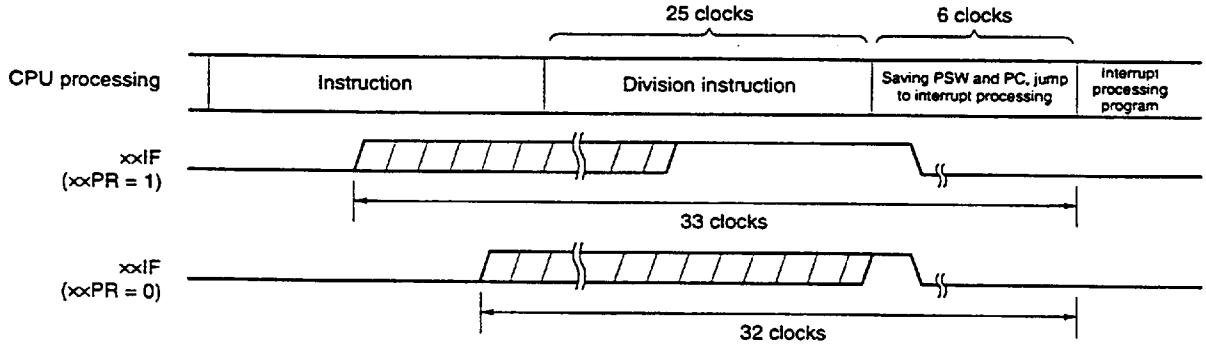


Figure 13-11. Interrupt Acceptance Timing (Minimum Time)



Remark 1 clock : $\frac{1}{f_{CPU}}$ (f_{CPU} : CPU clock)

Figure 13-12. Interrupt Acceptance Timing (Maximum Time)



Remark 1 clock : $\frac{1}{f_{CPU}}$ (f_{CPU} : CPU clock)

13.4.3 Software interrupt acceptance operation

The software interrupt can be accepted when the BRK instruction is executed. This interrupt cannot be disabled.

When the software interrupt is accepted, the program status word (PSW) and program counter (PC) are saved to the stack in that order, the IE flag is reset to 0, the contents of the vector table (003EH and 003FH) are loaded to the PC, and execution branches.

To return from the software interrupt processing, use the RETB instruction.

Caution Do not use the RETI instruction to return from the software interrupt.

13.4.4 Multiplexed interrupt processing

Multiplexed interrupt processing in which another interrupt is accepted while an interrupt is processed can be controlled by priority.

The priority is controlled in two ways: by using the default priority and by using programmable priority that is set by the priority specification flag register (PR0L, PR0H). When the priority is controlled by the default priority and two or more interrupts are generated at once, interrupt processing is performed according to the priority (default priority) assigned to each interrupt request in advance (refer to **Table 13-1**). The programmable priority control divides the interrupt requests into two groups by the setting of the bit corresponding to PR0L, PR0H: the one with the higher priority and the other with the lower priority. The interrupt requests that can be multiplexed are shown in the table below.

Table 13-4. Interrupt Requests that Can Be Multiplexed during Interrupt Processing

Multiplexed Interrupt Request / Interrupt being accepted		Non-Maskable Interrupt Request	Maskable Interrupt Request			
			××PR = 0		××PR = 1	
			IE = 1	IE = 0	IE = 1	IE = 0
Non-maskable interrupt processing		×	×	×	×	×
Maskable interrupt processing	ISP = 0	○	○	×	×	×
	ISP = 1	○	○	×	○	×
Software interrupt processing		○	○	×	○	×

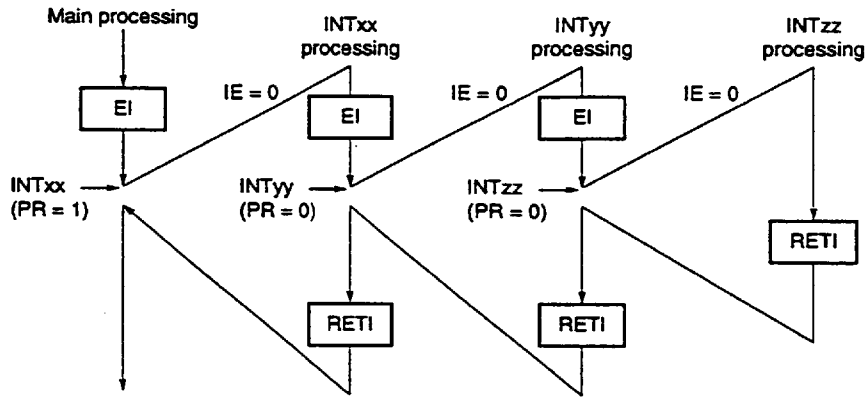
Remarks 1. ○ : can be multiplexed.
× : cannot be multiplexed.

2. ISP and IE are flags included in PSW.
ISP = 0 : interrupt with higher priority is processed.
ISP = 1 : interrupt is not accepted or interrupt with lower priority is processed.
IE = 0 : accepting interrupt is disabled.
IE = 1 : accepting interrupt is enabled.

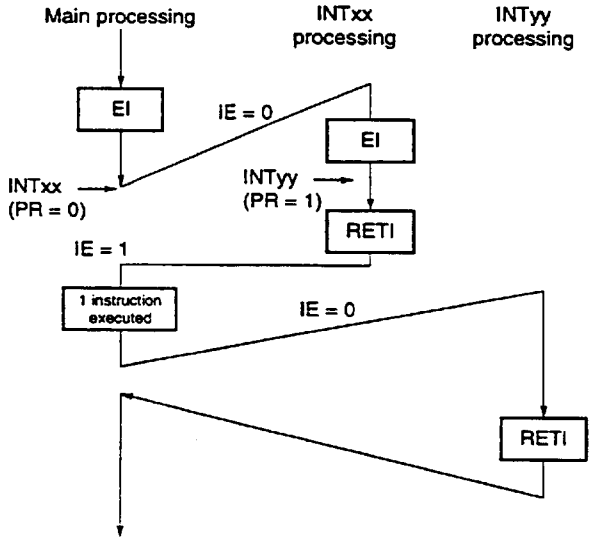
3. ××PR is flag included in PR0L, PR0H.
××PR = 0 : higher priority flag
××PR = 1 : lower priority flag

Figure 13-13. Example of Multiplexed Interrupt

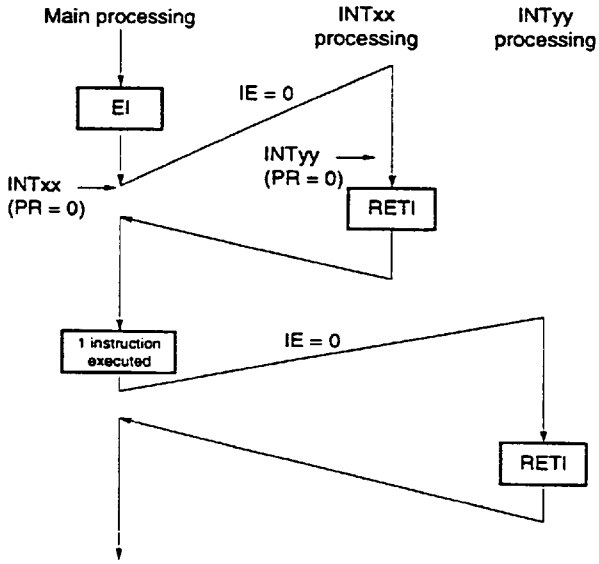
Example 1



Example 2



Example 3



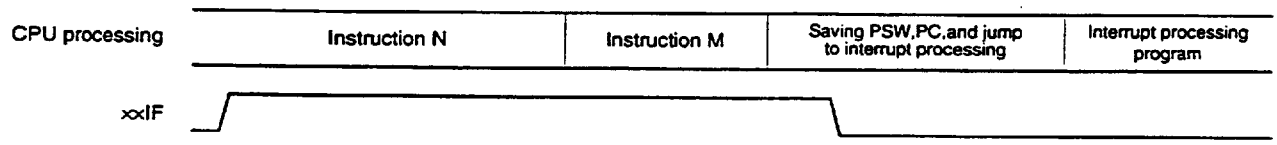
13.4.5 Pending interrupt

Interrupts are temporarily kept pending in between the following instructions and the instruction to be executed next.

- MOV PSW, #byte
- MOV A, PSW
- MOV PSW, A
- MOV1 PSW. bit, CY
- MOV1 CY, PSW. bit
- AND1 CY, PSW. bit
- OR1 CY, PSW. bit
- XOR1 CY, PSW. bit
- SET1 PSW. bit
- CLR1 PSW. bit
- RETB
- RETI
- PUSH PSW
- POP PSW
- BT PSW. bit, $addr16
- BF PSW. bit, $addr16
- BTCLR PSW. bit, $addr16
- EI
- DI
- Manipulation instruction to IF0L, IF0H, MK0L, MK0H, PR0L, PR0H, EGP, EGN registers

Caution The software interrupt (by execution of the BRK instruction) clears the IE flag to 0. Therefore, a maskable interrupt request is not accepted even if generated while the BRK instruction is executed. However, the non-maskable interrupt request is accepted.

Figure 13-14. Pending Interrupt Request



Remarks 1. Instruction N : instruction that keeps interrupt request pending
2. Instruction M : instruction other than that which keeps interrupt request pending
3. The operation of xxIF is not affected by the value of xxPR.

14.1 External Device Extension Function

The external device extension function is to connect an external device to areas other than the internal ROM, RAM, and SFR areas. To connect an external device, ports 4 to 6 are used. These ports control address/data, read/write strobe, wait, and address strobe signals.

Table 14-1. Pin Functions in External Memory Extension Mode

Pin Function when External Device is Connected		Shared by:
Name	Function	
AD0 to AD7	Multiplexed address/data bus	P40 to P47
A8 to A15	Address bus	P50 to P57
$\overline{RD}$	Read strobe signal	P64
$\overline{WR}$	Write strobe signal	P65
$\overline{WAIT}$	Wait signal	P66
ASTB	Address strobe signal	P67

Table 14-2. Status of Ports 4 to 6 in External Memory Extension Mode

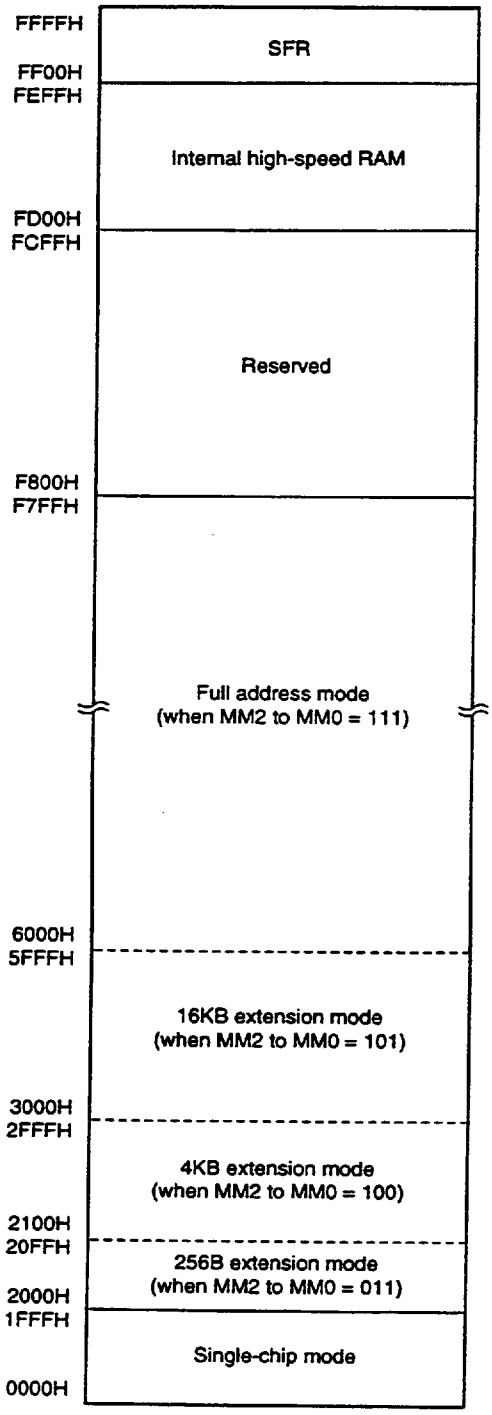
Port / External Extension Mode	Port 4	Port 5								Port 6			
	0-7	0	1	2	3	4	5	6	7	4	5	6	7
Single-chip mode	Port	Port								Port			
256 bytes extension mode	Address/data	Port								$\overline{RD}$, $\overline{WR}$, $\overline{WAIT}$, ASTB			
4 Kbytes extension mode	Address/data	Address						Port		$\overline{RD}$, $\overline{WR}$, $\overline{WAIT}$, ASTB			
16 Kbytes extension mode	Address/data	Address							Port	$\overline{RD}$, $\overline{WR}$, $\overline{WAIT}$, ASTB			
Full address mode	Address/data	Address								$\overline{RD}$, $\overline{WR}$, $\overline{WAIT}$, ASTB			

Caution When the external wait function is not used, the $\overline{WAIT}$ pin can be used as a port pin in all the modes.

The memory map is as follows when the external device extension function is used.

Figure 14-1. Memory Map when External Device Extension Function Is Used (1/2)

(a) Memory map of μPD780921, 780961



(b) Memory map of μPD780922, 780962

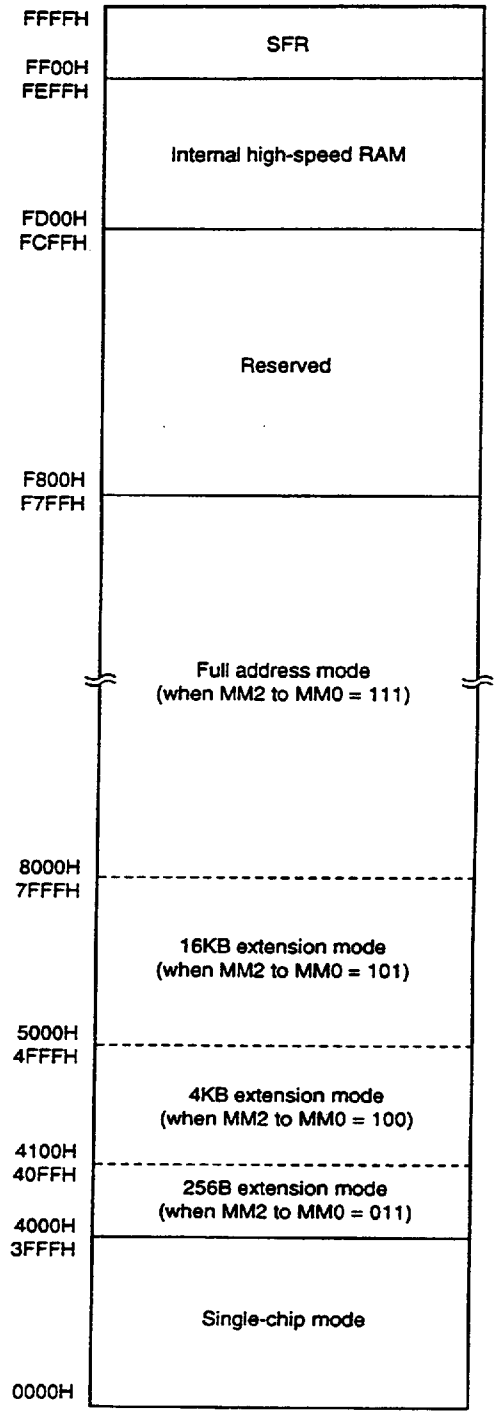
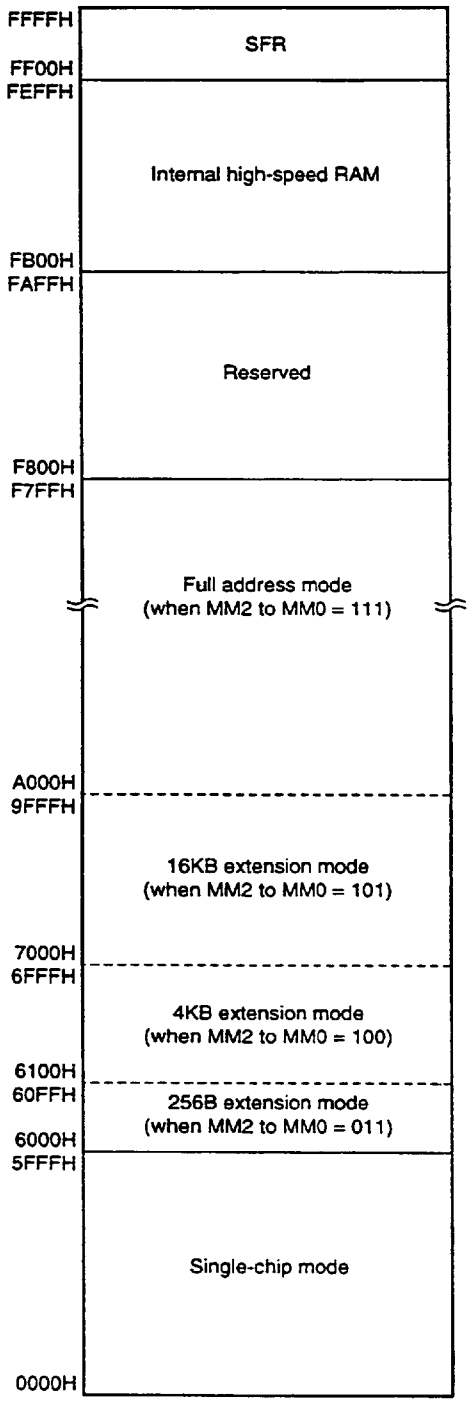
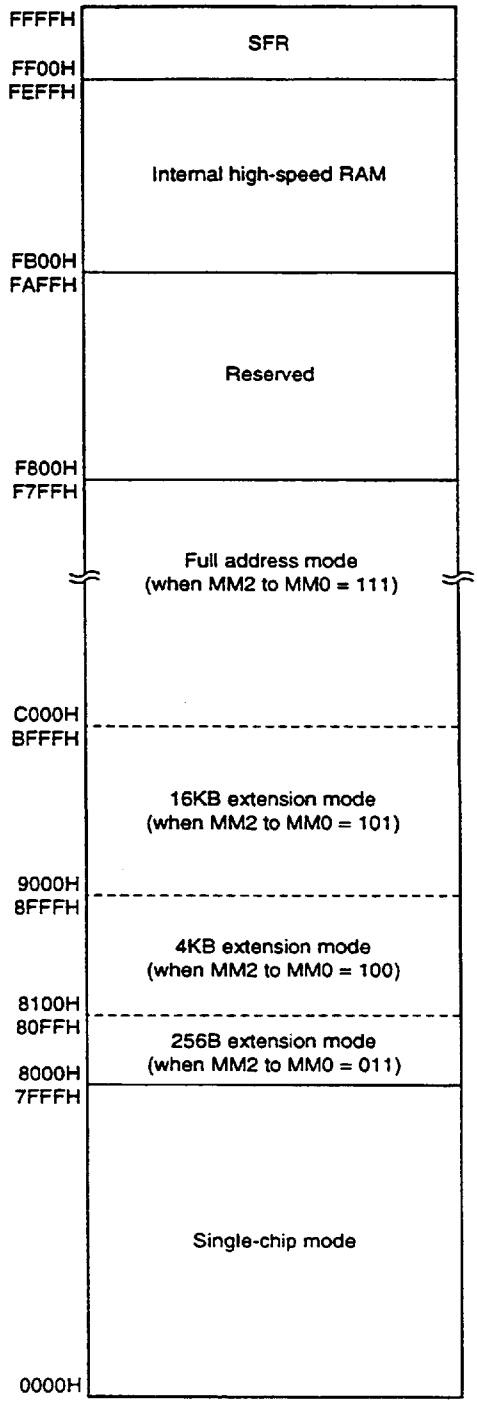


Figure 14-1. Memory Map when External Device Extension Function Is Used (2/2)

(c) Memory map of μPD780923, 780963



(d) Memory map of μPD780924, 780964, and μPD78F0924, 78F0964 with 32 KB flash memory



14.2 Registers Controlling External Device Extension Function

The external device expansion function is controlled by the following three registers.

- Memory extension mode register (MEM)
- Memory extension wait setting register (MM)
- Memory size select register (IMS)

(1) Memory extension mode register (MEM)

MEM is a register that sets an external expansion area.

MEM is set by using a 1-bit or 8-bit memory manipulation instruction.

Its value is set to 00H at $\overline{\text{RESET}}$.

Figure 14-2. Format of Memory Extension Mode Register

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
MEM	0	0	0	0	0	MM2	MM1	MM0	FF47H	00H	R/W

MM2	MM1	MM0	Selects single-chip/ memory extension mode		Status of P40 to P47, P50 to P57, P64 to P67 pins				
					P40 to P47	P50 to P53	P54, P55	P56, P57	P64 to P67
0	0	0	Single-chip mode		Port mode				
0	1	1	Memory extension mode	256B mode	AD0 to AD7	Port mode			P64 = $\overline{\text{RD}}$ P65 = $\overline{\text{WR}}$ P66 = $\overline{\text{WAIT}}$ P67 = ASTB
1	0	0		4KB mode		A8 to A11	Port mode		
1	0	1		16KB mode			A12, A13	Port mode	
1	1	1		Full[Note] address mode				A14, A15	
Others			Setting prohibited						

Note The full address mode is a mode in which the entire area of the 64K address space, except the internal ROM, RAM, SFR, and unused areas, can be externally extended.

(2) Memory extension wait setting register (MM)

MM is a register that sets the number of wait states.

MM is set by using a 1-bit or 8-bit memory manipulation instruction.

Its value is set to 10H at $\overline{\text{RESET}}$.

Figure 14-3. Format of Memory Extension Wait Setting Register

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
MM	0	0	PW1	PW0	0	0	0	0	FFF8H	10H	R/W

PW1	PW0	Controls wait state.
0	0	No wait
0	1	Wait (1 wait state is inserted)
1	0	Setting prohibited
1	1	Wait control by external wait pin

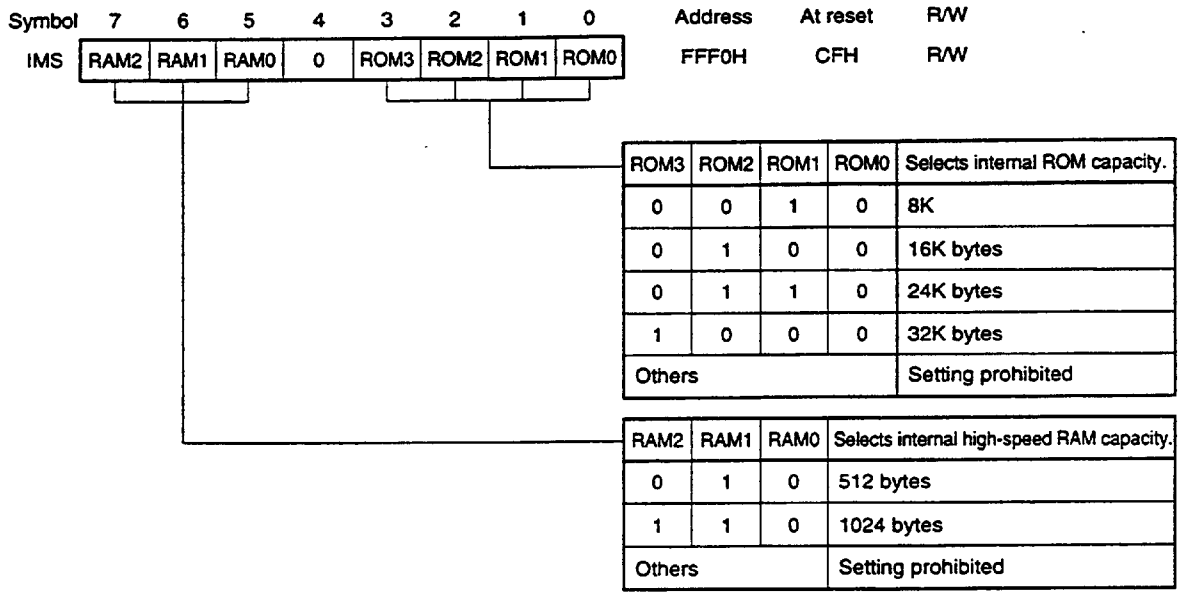
(3) Memory size select register (IMS)

This register sets the capacities of the internal ROM and internal high-speed RAM.

IMS is set by using an 8-bit memory manipulation instruction.

Its value is set to CFH at $\overline{\text{RESET}}$.

Figure 14-4. Format of Memory Size Select Register



ROM3	ROM2	ROM1	ROM0	Selects internal ROM capacity.
0	0	1	0	8K
0	1	0	0	16K bytes
0	1	1	0	24K bytes
1	0	0	0	32K bytes
Others				Setting prohibited

RAM2	RAM1	RAM0	Selects internal high-speed RAM capacity.
0	1	0	512 bytes
1	1	0	1024 bytes
Others			Setting prohibited

Cautions 1. The value of IMS at reset is the same (CFH) for all the models in the μPD780924 and 780964 subseries, regardless of the internal memory capacity. Therefore, be sure to set the value of IMS according to the internal memory capacity of the model used.

2. The external memory space can be expanded in a space other than that specified by IMS, regardless of the internal memory capacity.

Table 14-3. Set Value of Memory Size Select Register

Part Number	Set Value of IMS
μPD780921, 780961	42H
μPD780922, 780962	44H
μPD780923, 780963	C6H
μPD780924, 780964	C8H
μPD78F0924, 78F0964	Note

Note Set C8H, C6H, 44H, or 42H, according to the mask ROM model.

14.3 Timing of External Device Extension Function

The timing control signal output pins used in the external memory extension mode are as follows:

(1) $\overline{RD}$ pin (shared by P64)

This pin outputs a read strobe signal when an instruction is fetched or data is accessed from the external memory.

When the internal memory is accessed, the read strobe signal is not output (instead, this pin holds the high level).

(2) $\overline{WR}$ pin (shared by P65)

This pin outputs a write strobe signal when the external memory is accessed for data.

When the internal memory is accessed, the write strobe signal is not output (this pin holds the high level).

(3) $\overline{WAIT}$ pin (shared by P66)

This pin inputs an external wait signal.

When the external wait signal is not used, the $\overline{WAIT}$ pin can be used as an I/O port pin.

When the internal memory is accessed, the external wait signal is ignored.

(4) ASTB pin (shared by P67)

This pin outputs an address strobe signal which is always output regardless of instruction fetch or data access from the external memory.

The address strobe signal is also output when the internal memory is accessed.

(5) AD0 to AD7, A8 to A15 pins (shared by P40 to P47, P50 to P57)

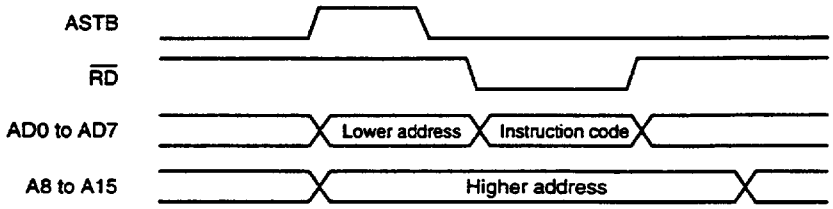
These pins output address and data signals. The valid signals are output or input when instructions are fetched or data is accessed from the external memory.

The status of the signal also changes when the internal memory is accessed (the output contents are undefined).

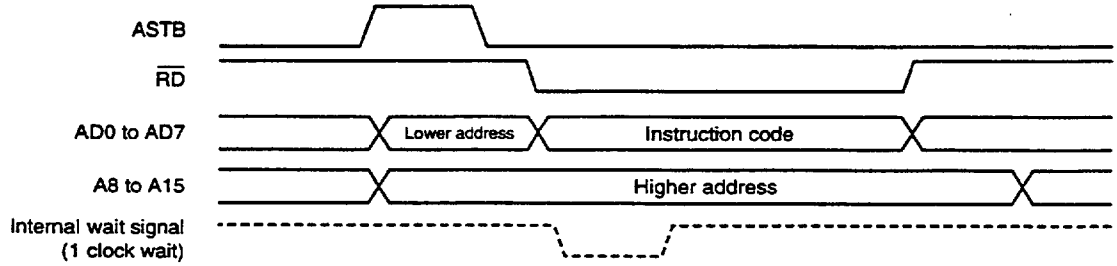
Figures 14-5 to 14-8 show the timing charts.

Figure 14-5. Instruction Fetch from External Memory

(a) When no wait state is set (PW1, PW0 = 0, 0)



(b) When wait state is set (PW1, PW0 = 0, 1)



(c) When external wait state is set (PW1, PW0 = 1, 1)

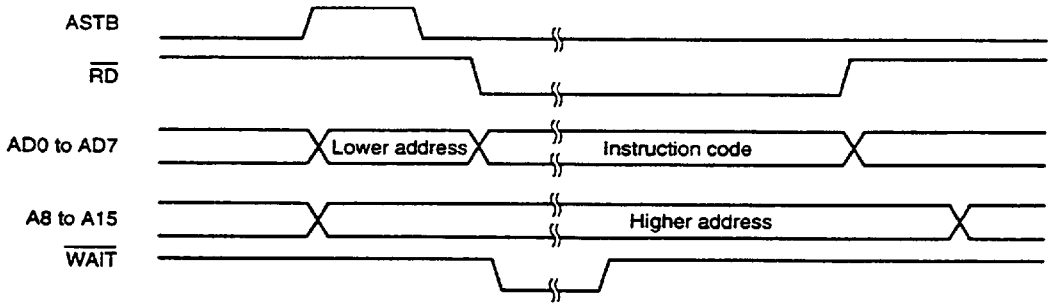
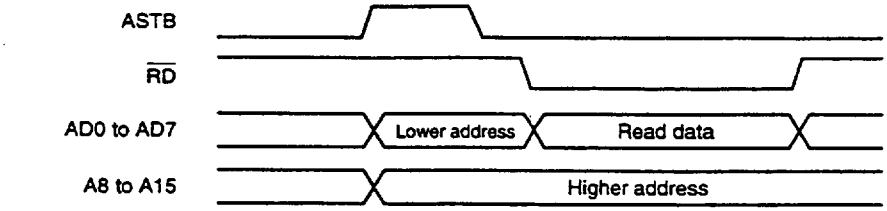
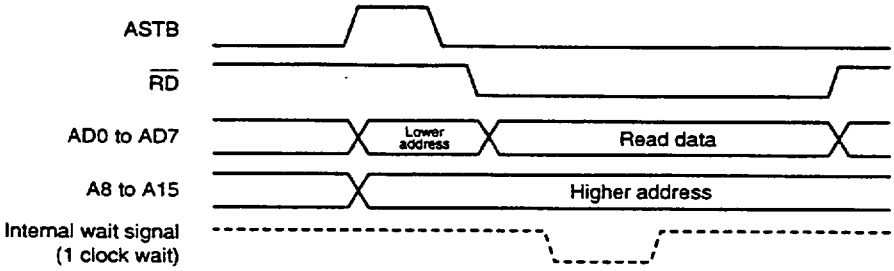


Figure 14-6. Read Timing of External Memory

(a) When no wait state is set (PW1, PW0 = 0, 0)



(b) When wait state is set (PW1, PW0 = 0, 1)



(c) When external wait state is set (PW1, PW0 = 1, 1)

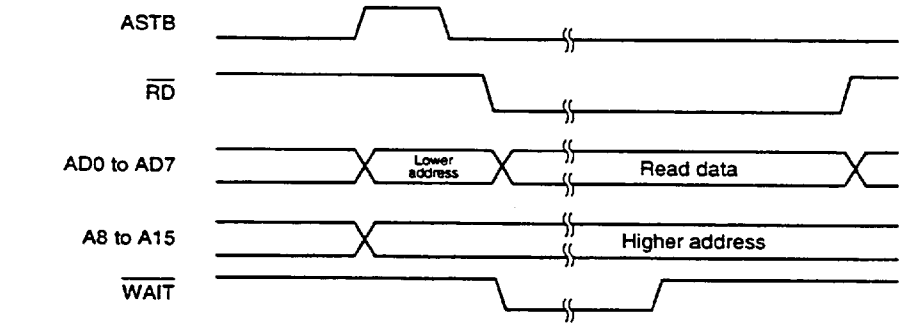
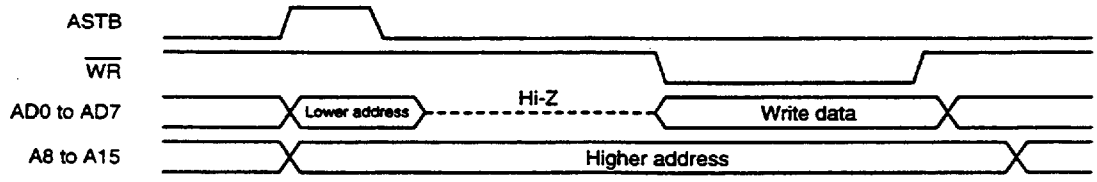
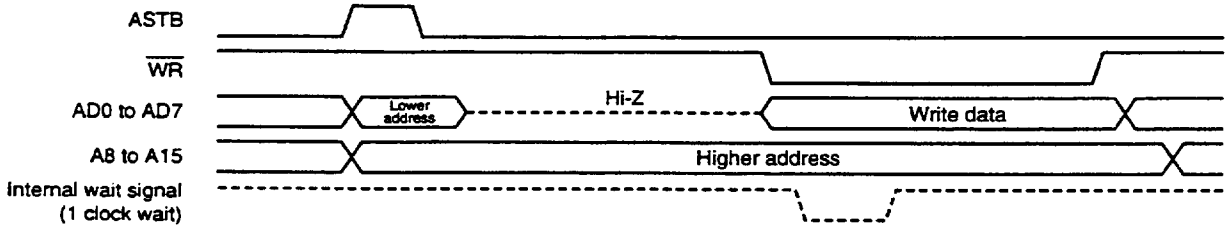


Figure 14-7. Write Timing of External Memory

(a) When no wait state is set (PW1, PW0 = 0, 0)



(b) When wait state is set (PW1, PW0 = 0, 1)



(c) When external wait state is set (PW1, PW0 = 1, 1)

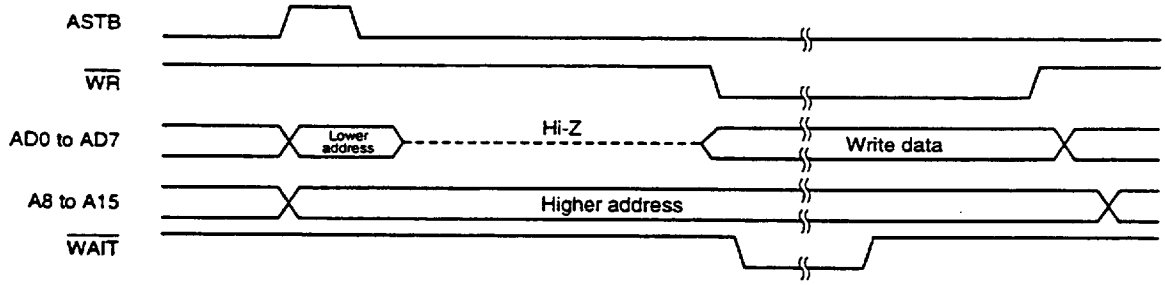
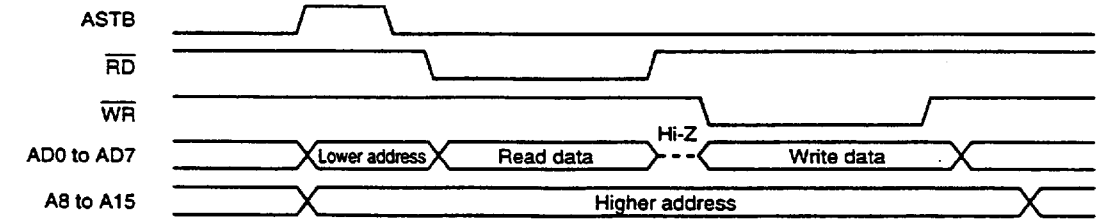
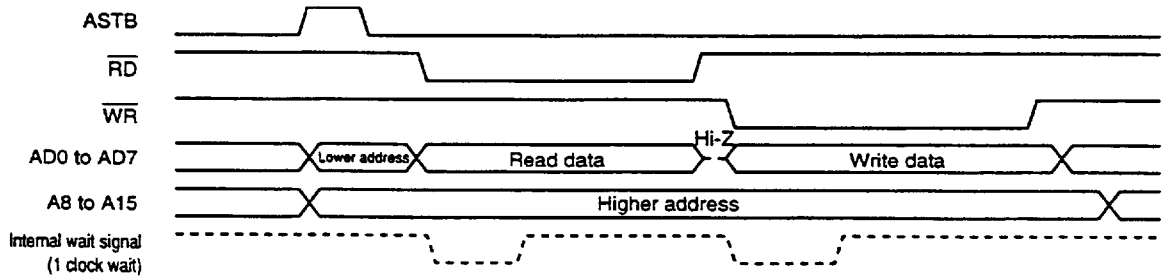


Figure 14-8. Read-Modify-Write Timing of External Memory

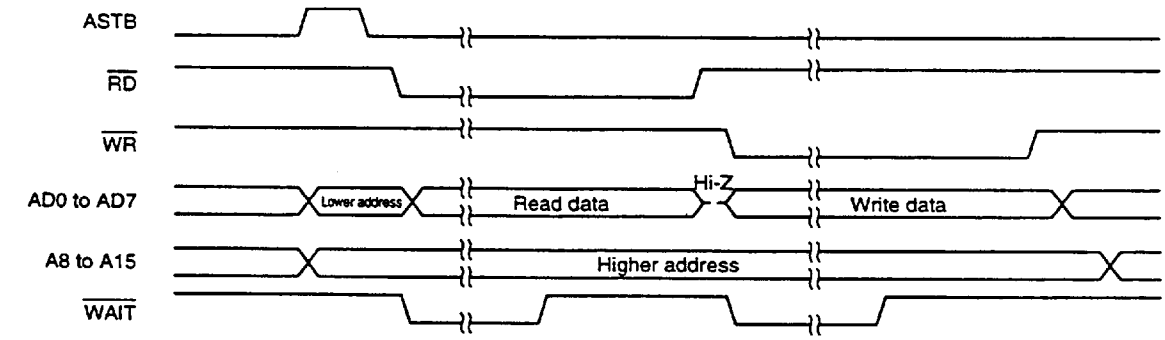
(a) When no wait state is set (PW1, PW0 = 0, 0)



(b) When wait state is set (PW1, PW0 = 0, 1)



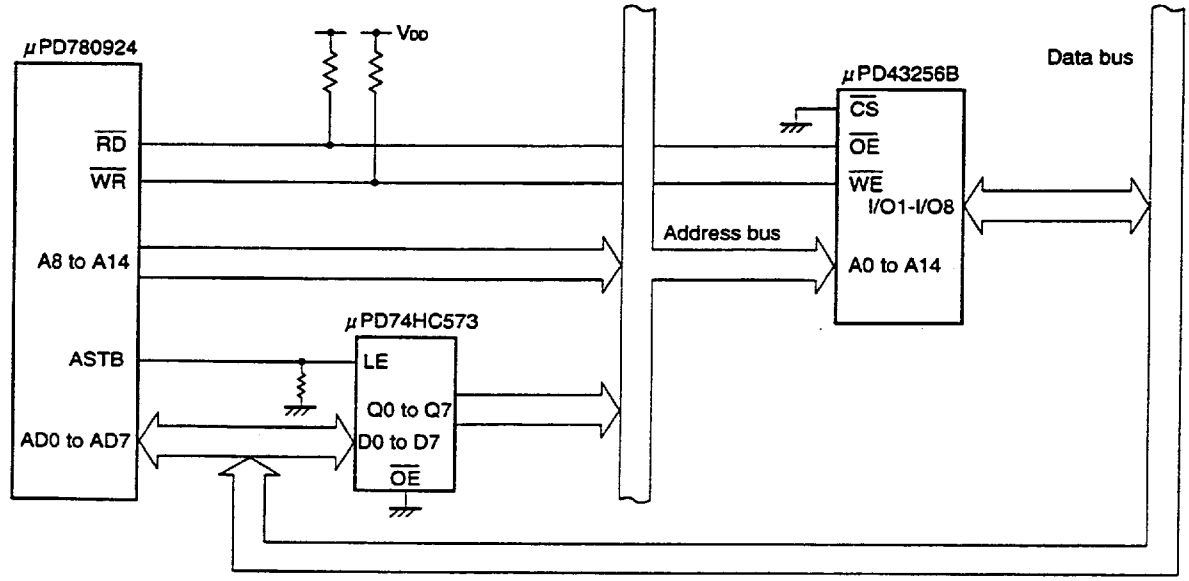
(c) When external wait state is set (PW1, PW0 = 1, 1)



14.4 Example of Connection with Memory

Figure 14-9 shows an example of connecting the μPD780924 and external memories. In this application example, SRAM is connected. In addition, the external device expansion function is used in the full address mode, and 32K bytes of addresses, 0000H to 7FFFH, are allocated to internal ROM; addresses 8000H and higher are allocated to SRAM.

Figure 14-9. Example of Connecting μPD780924 and Memories



15.1 Standby Function and Configuration

15.1.1 Standby function

The standby function is to reduce the power dissipation of the system and can be effected in the following two modes:

(1) HALT mode

This mode is set when the HALT instruction is executed. The HALT mode stops the operation clock of the CPU. The system clock oscillation circuit continues oscillating. This mode does not reduce the power dissipation as much as the STOP mode, but is useful for resuming processing immediately when an interrupt request is generated, or for intermittent operations such as a watch operation.

(2) STOP mode

This mode is set when the STOP instruction is executed. The STOP mode stops the system clock oscillation circuit and stops the entire system. The power dissipation of the CPU can be substantially reduced in this mode.

The low voltage (V_{DD} = 2.0 V) of the data memory can be retained. Therefore, this mode is useful for retaining the contents of the data memory at an extremely low current.

The STOP mode can be released by an interrupt request, so that this mode can be used for the intermittent operation. However, certain time is required until the system clock oscillation circuit stabilizes after the STOP mode has been released. If processing must be resumed immediately by using an interrupt request, therefore, use the HALT mode.

In both modes, the previous contents of the registers, flags, and data memory before setting the standby mode are all retained. In addition, the statuses of the output latch of the I/O ports and output buffer are also retained.

Cautions 1. To set the STOP mode, be sure to stop the operations of the peripheral hardware, and then execute the STOP instruction.

2. To reduce the power dissipation of the A/D converter, clear bit 7 (ADCS0) of A/D converter mode register (ADM0) to 0 to stop the A/D conversion, and then execute the HALT or STOP instruction.

15.1.2 Registers controlling standby function

The wait time during which oscillation is stabilized after the STOP mode has been released by an interrupt request is controlled by the oscillation stabilization time select register (OSTS).

OSTS is set by an 8-bit memory manipulation instruction.

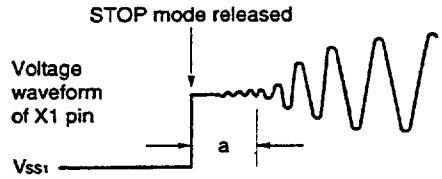
This register is set to 04H when the $\overline{\text{RESET}}$ signal is input. Therefore, to release the STOP mode by inputting the $\overline{\text{RESET}}$ signal, the time required to release the mode is 2^{17}/fx.

Figure 15-1. Format of Oscillation Stabilization Time Select Register

Symbol	7	6	5	4	3	2	1	0	Address	On reset	R/W
OSTS	0	0	0	0	0	OSTS2	OSTS1	OSTS0	FFFAH	04H	R/W

OSTS2	OSTS1	OSTS0	Selects oscillation stabilization time when released the STOP mode.
0	0	0	2^{12}/fx (488.8 μs)
0	0	1	2^{14}/fx (1.96 ms)
0	1	0	2^{15}/fx (3.91 ms)
0	1	1	2^{16}/fx (7.82 ms)
1	0	0	2^{17}/fx (15.6 ms)
Others			Setting prohibited

Caution The wait time when the STOP mode is released does not include the time required for the clock oscillation to start after the STOP mode has been released (see "a" in the figure below), regardless of whether the mode has been released by the $\overline{\text{RESET}}$ signal or an interrupt request.



Remarks 1. fx : system clock oscillation frequency

2. ( ) : at fx = 8.38-MHz operation

15.2 Operation of Standby Function

15.2.1 HALT mode

(1) Setting and operation status of HALT mode

The HALT mode is set by executing the HALT instruction.

The operation status in the HALT mode is shown in the table below.

Table 15-1. Operation Status in HALT Mode

Item		Operation Status
Clock generation circuit		Oscillatable Supply of clock to CPU is stopped.
CPU		Stops operation
Port (output latch)		Retains previous status before setting HALT mode.
10-bit inverter control timer		Operable
8-bit timer/event counter		
Watchdog timer		
A/D converter		
Real-time output port		
Serial interface		
External interrupt		
Externally extended bus line	AD0 to AD7	High impedance
	A8 to A15	Retains previous status before setting HALT mode.
	ASTB	Low level
	$\overline{WR}$, $\overline{RD}$	High level
	$\overline{WAIT}$	High impedance

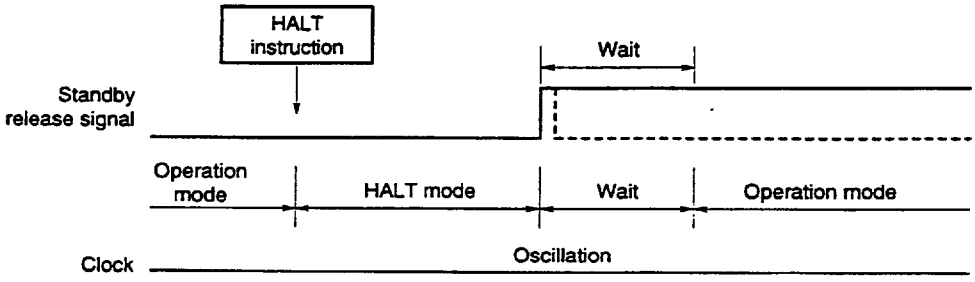
(2) Releasing HALT mode

The HALT mode can be released by the following three types of sources:

(a) Releasing by unmasked interrupt request

The HALT mode is released by an unmasked interrupt request. In this case, if the interrupt is enabled to be accepted, vectored interrupt processing is performed. If the interrupt is disabled, the instruction at the next address is executed.

Figure 15-2. Releasing HALT Mode by Interrupt



Remarks 1. The dotted line indicates the case where the interrupt request that has released the standby mode is accepted.

2. The wait time is as follows:

- When vectored interrupt processing is performed : 8 to 9 clocks
- When vectored interrupt processing is not performed : 2 to 3 clocks

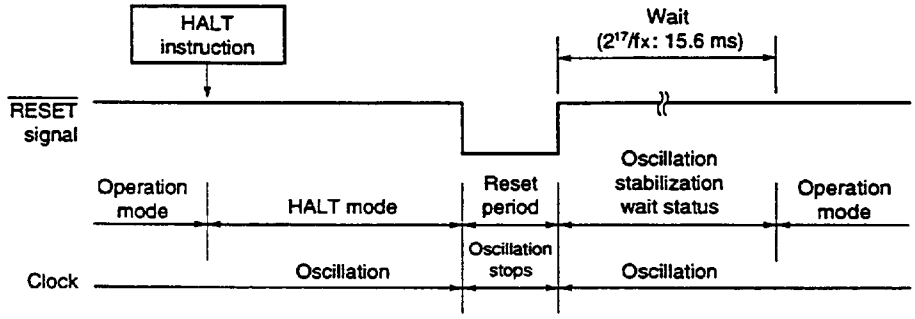
(b) Releasing by non-maskable interrupt request

The HALT mode is released regardless of whether the interrupt is enabled or disabled, and vectored interrupt processing is performed.

(c) Releasing by $\overline{\text{RESET}}$ input

When the HALT mode is released by the $\overline{\text{RESET}}$ signal, execution branches to the reset vector address in the same manner as the ordinary reset operation, and program execution is started.

Figure 15-3. Releasing HALT Mode by $\overline{\text{RESET}}$ Input



Remark ( ): at fx = 8.38 MHz operation

Table 15-2. Operation after Release of HALT Mode

Releasing Source	MKxx	PRxx	IE	ISP	Operation
Maskable interrupt request	0	0	0	×	Executes next address instruction.
	0	0	1	×	Executes interrupt processing.
	0	1	0	1	Executes next address instruction.
	0	1	×	0	
	0	1	1	1	Executes interrupt processing.
	1	×	×	×	Retains HALT mode.
Non-maskable interrupt request	–	–	×	×	Executes interrupt processing.
$\overline{\text{RESET}}$ input	–	–	×	×	Resets processing.

Remark ×: don't care

15.2.2 STOP mode

(1) Setting and operation status of STOP mode

The STOP mode is set by executing the STOP instruction.

Cautions 1. When the STOP mode is set, X2 pin is internally pulled up circuited to V_{DD1} to suppress the current leakage of the crystal oscillation circuit block. Therefore, do not use the STOP mode in a system where the external clock is used as the system clock.

2. Because the standby mode can be released by an interrupt request signal, the standby mode is released as soon as it is set if there is an interrupt source whose interrupt request flag is set and interrupt mask flag is reset. When the STOP mode is set, therefore, the HALT mode is set immediately after the STOP instruction has been executed, the wait times set by the oscillation stabilization time select register (OSTS) elapses, and then an operation mode is set.

The following table shows the operation status in the STOP mode.

Table 15-3. Operation Status in STOP Mode

Item		Operation Status
Clock generation circuit		Oscillation stopped
CPU		Stops operation.
Output port (output latch)		Retains previous status immediately before STOP instruction execution.
10-bit inverter control timer		Stops operation.
8-bit timer/event counter		Operable only when TI50, TI51, or TI52 is selected as count clock
Watchdog timer		Stops operation.
A/D converter		
Real-time output port		Operable when external trigger is used or when TI51 or TI52 is selected as count clock of 8-bit timer/event counter.
Serial interface		Stops operation.
External interrupt		Operable
Externally extended bus line	AD0 to AD7	High impedance
	A8 to A15	Retains previous status immediately before STOP instruction execution.
	ASTB	Low level
	$\overline{WR}$, $\overline{RD}$	High level
	$\overline{WAIT}$	High impedance

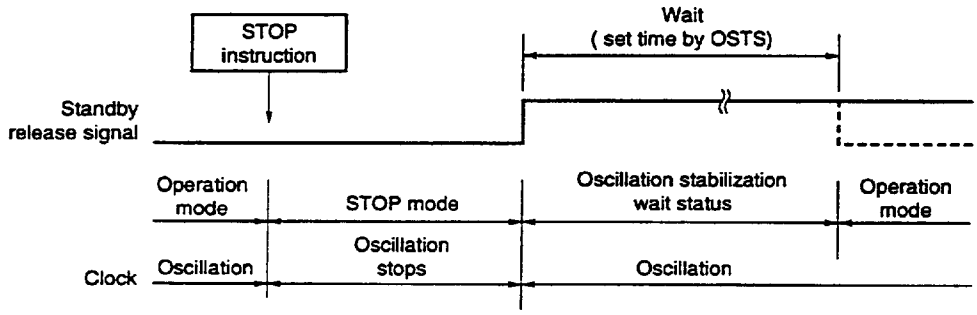
(2) Releasing STOP mode

The STOP mode can be released by the following two types of sources:

(a) Releasing by unmasked interrupt request

The STOP mode can be released by an unmasked interrupt request. In this case, if the interrupt is enabled to be accepted, vectored interrupt processing is performed, after the oscillation stabilization time has elapsed. If the interrupt is disabled to be accepted, the instruction at the next address is executed.

Figure 15-4. Releasing STOP Mode by Interrupt

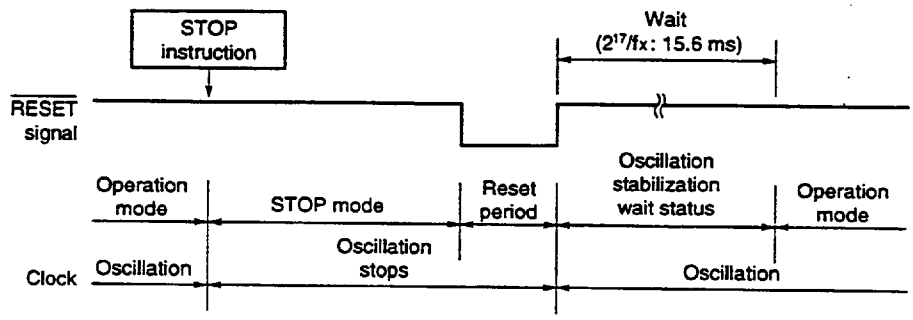


Remark The dotted line indicates the case where the interrupt request that has released the standby mode is accepted.

(b) Releasing by $\overline{\text{RESET}}$ input

When the STOP mode is released by the $\overline{\text{RESET}}$ signal, the reset operation is performed after the oscillation stabilization time has elapsed.

Figure 15-5. Releasing STOP Mode by $\overline{\text{RESET}}$ Input



Remark ( ) : at fx = 8.38-MHz operation

Table 15-4. Operation after Release of STOP Mode

Releasing Source	MKxx	PRxx	IE	ISP	Operation
Maskable interrupt request	0	0	0	×	Executes next address instruction.
	0	0	1	×	Executes interrupt processing.
	0	1	0	1	Executes next address instruction.
	0	1	×	0	
	0	1	1	1	Executes interrupt processing.
	1	×	×	×	Retains STOP mode.
$\overline{\text{RESET}}$ input	–	–	×	×	Resets processing.

Remark × : don't care

[MEMO]

PAGE(S) INTENTIONALLY BLANK

CHAPTER 16 RESET FUNCTION

The reset signal can be effected by the following two methods:

(1) External reset input from $\overline{\text{RESET}}$ pin
(2) Internal reset by inadvertent loop time detection by watchdog timer

There is no functional difference between the external reset and internal reset, and execution of the program is started from addresses written to addresses 0000H and 0001H when the $\overline{\text{RESET}}$ signal is input.

The reset function is effected when a low-level signal is input to the $\overline{\text{RESET}}$ pin or when an overflow occurs in the watchdog timer. As a result, each hardware enters the status shown in Table 16-1. Each pin goes into a high-impedance state while the $\overline{\text{RESET}}$ signal is input, and during the oscillation stabilization time immediately after the reset function has been released.

When a high-level signal is input to the $\overline{\text{RESET}}$ pin, the reset function is released, and program execution is started after oscillation stabilization time (2^{17}/fx) has elapsed. The reset function effected by an overflow in the watchdog timer is automatically released after reset, and program execution is started after the oscillation stabilization time (2^{17}/fx) has elapsed (refer to **Figures 16-2 to 16-4**).

Cautions 1. Input a low-level signal to the $\overline{\text{RESET}}$ pin for 10 μs or longer to execute external reset.
2. Oscillation of the system clock is stopped while the $\overline{\text{RESET}}$ signal is input.
3. To release the STOP mode by the $\overline{\text{RESET}}$ input, the contents in the STOP mode are retained while the $\overline{\text{RESET}}$ signal is input. However, the port pins go into a high-impedance state.

Figure 16-1. Block Diagram of Reset Function

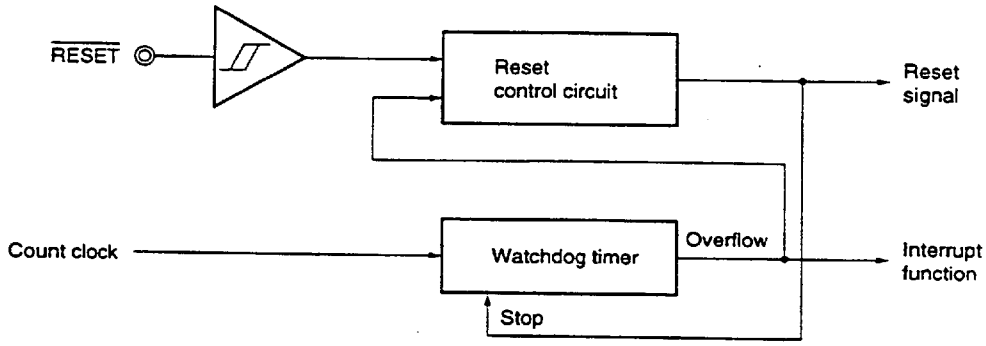


Figure 16-2. Reset Timing by $\overline{\text{RESET}}$ Input

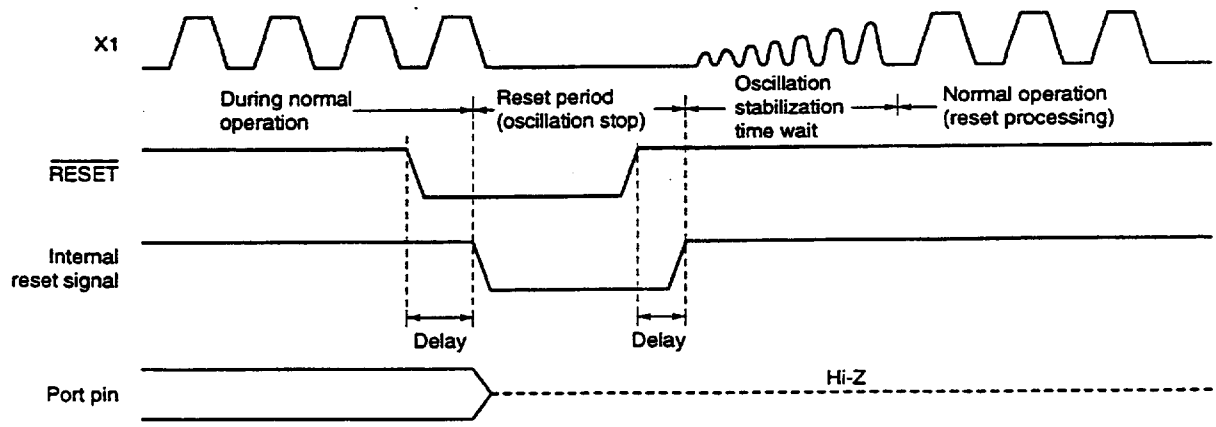


Figure 16-3. Reset Timing by Overflow in Watchdog Timer

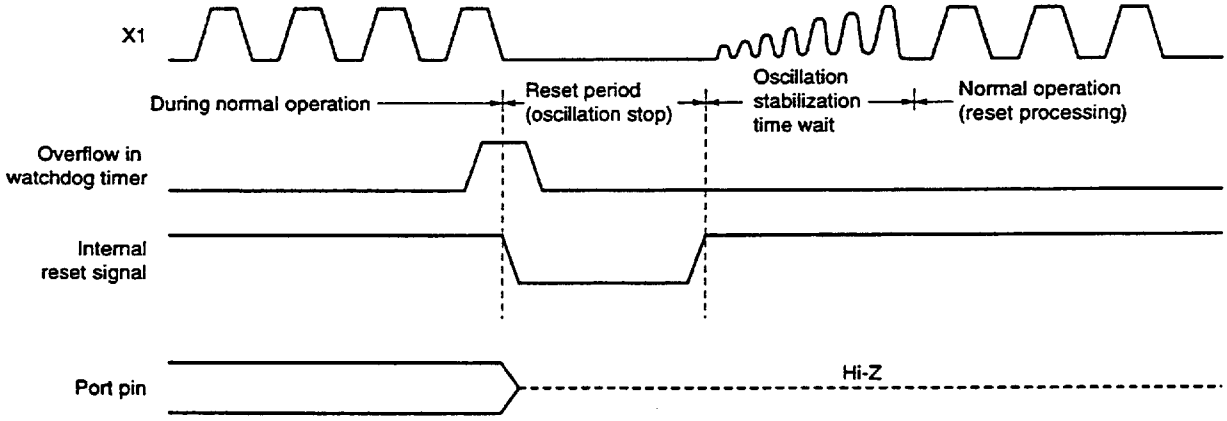


Figure 16-4. Reset Timing by $\overline{\text{RESET}}$ Input in STOP Mode

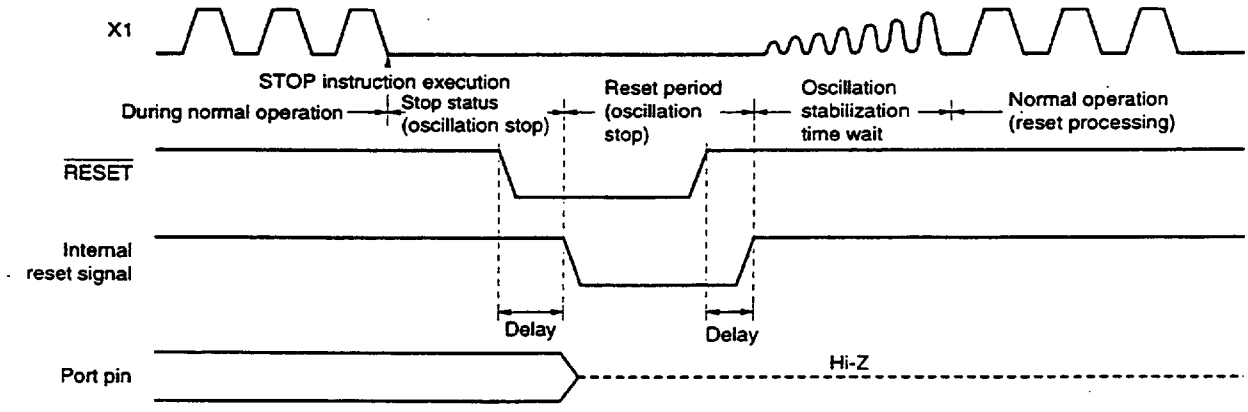


Table 16-1. Status of Each Hardware after Reset (1/2)

Hardware		Status after Reset
Program counter (PC)[Note 1]		Contents of reset vector table (0000H, 0001H) are set
Stack pointer (SP)		Undefined
Program status word (PSW)		02H
RAM	Data memory	Undefined[Note 2]
	General-purpose register	Undefined[Note 2]
Port (output latch)	Ports 0 to 3 (P0 to P3)	00H
	Ports 4 to 6 (P4 to P6)	Undefined
Port mode register (PM0, PM2 to PM6)		FFH
Pull-up resistor option register (PU0, PU2 to PU6)		00H
Processor clock control register (PCC)		04H
Memory extension mode register (MEM)		00H
Memory extension wait setting register (MM)		10H
Memory size select register (IMS)		CFH[Note 3]
Oscillation stabilization time select register (OSTS)		04H
Real-time output port	Mode register (RTPM0)	00H
	Control register (RTPC0)	00H
	Buffer register (RTBL0, RTBH0)	00H
10-bit inverter control timer	Compare register (CM0 to CM2)	00H
	Compare register (CM3)	FFH
	Buffer register (BFCM0 to BFCM2)	000H
	Buffer register (BFCM3)	0FFH
	Reload register (DTIME)	FFH
	Control register (TMC7)	00H
	Mode register (TMM7)	00H
8-bit timer/event counter	Timer register (TM50 to TM52)	00H
	Compare register (CM50 to CM52)	Undefined
	Clock select register (TCL50 to TCL52)	00H
	Mode control register (TMC50 to TMC52)	04H

Notes 1. Only the contents of the PC among hardware become undefined during reset input and oscillation stabilization time wait. The other status is not different from that after reset as above.

2. The status after reset is retained in the standby mode.

3. The initial value of this register is CFH. Set the following value to this register of each version.

   μPD780921, 780961: 42 H
   μPD780922, 780962: 44 H
   μPD780923, 780963: C6 H
   μPD780924, 780964: C8 H
   μPD78F0924, 78F0964: Value corresponding to those of mask ROM versions

Table 16-1. Status of Each Hardware after Reset (2/2)

Hardware		Status after Reset
Watchdog timer	Clock select register (WDCS)	00H
	Mode register (WDTM)	00H
Serial interface	Asynchronous serial interface mode register (ASIM00, ASIM01)	00H
	Asynchronous serial interface status register (ASIS00, ASIS01)	00H
	Transmit shift register (TXS00, TXS01)	FFH
	Receive buffer register (RXB00, RXB01)	FFH
	Baud rate generator control register (BRGC00, BRGC01)	00H
A/D converter	Mode register (ADM0)	00H
	Conversion result register (ADCR0)	Undefined
	Analog input channel specification register (ADS0)	00H
Interrupt	Request flag register (IF0L, IF0H)	00H
	Mask flag register (MK0L, MK0H)	FFH
	Priority specification flag register (PR0L, PR0H)	FFH
	External interrupt rising edge enable register (EGP)	00H
	External interrupt falling edge enable register (EGN)	00H

Data can be written to the flash memory with the device mounted on the target system (on-board). Write data to the flash memory by connecting the Flashpro to the host machine and target system.

Remark Flashpro is a product of Naitou Densei Machidaseisakusho Co., Ltd.

17.1 Selecting Communication Mode

Data are written to the flash memory by using the Flashpro and by means of serial communication. Select a communication mode from those listed in Table 17-1. To select a communication mode, use the format shown in Figure 17-1. Each communication mode is selected by the number of V_{PP} pulses shown in Table 17-1.

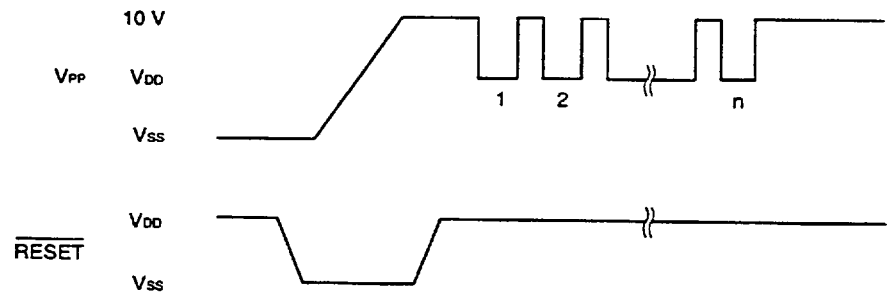
Table 17-1. Communication Modes

Communication Mode	Number of Channels	Pins Used	Number of V_{PP} Pulses
UART	2	RxD00/P20 TxD00/P21	8
		RxD01/P22 TxD01/P23	9
Pseudo 3-wire mode[Note]	2	P24/TI50/TO50 (serial data input) P25/TI51/TO51 (serial data output) P26/TI52/TO52 (serial clock input)	12
		P34/RTP4 (serial data input) P35/RTP5 (serial data output) P36/RTP6 (serial clock input)	13

Note Serial transfer is executed by controlling the port by software.

Caution Be sure to select a communication mode by specifying the number of V_{PP} pulses shown in Table 17-1.

Figure 17-1. Communication Mode Selecting Format



17.2 Function of Flash Memory Programming

Data is written to the flash memory by transmitting or receiving commands and data in the selected communication mode. The major functions are listed in Table 17-2.

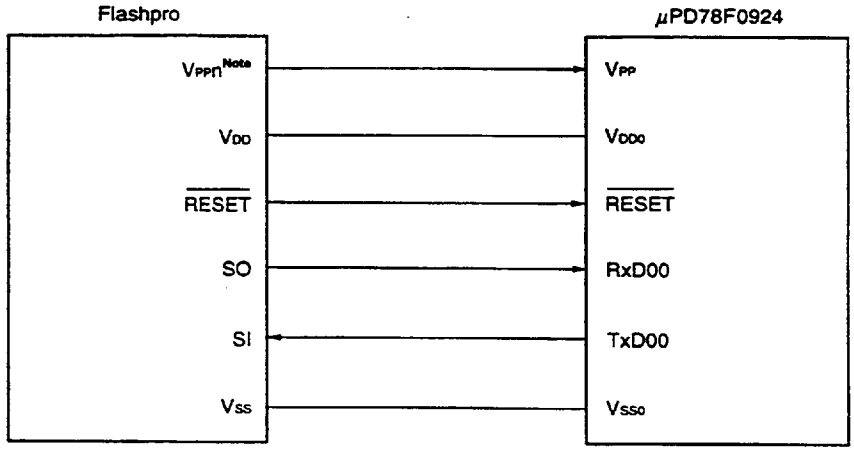
Table 17-2. Major Functions of Flash Memory Programming

Function	Description
Batch erase	Erases all contents of memory.
Batch blank check	Checks erased status of entire memory.
Data write	Writes data to flash memory based on write start address and number of data (bytes) to be written.
Batch verify	Compares all contents of memory with input data.

17.3 Connecting Flashpro

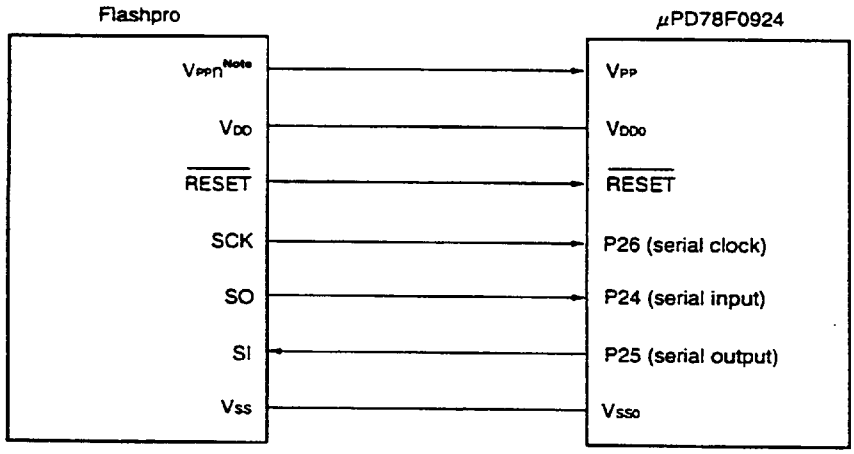
Connection between the Flashpro and μPD78F0924 (or μPD78F0964) differs depending on the communication mode. Figures 17-2 and 17-3 show the connections in the respective communication modes.

Figure 17-2. Connection of Flashpro in UART Mode (when UART00 is used)



Note n = 1, 2

Figure 17-3. Connection of Flashpro in Pseudo 3-Wire Mode (when port 2 is used)



Note n = 1, 2

[MEMO]

PAGE(S) INTENTIONALLY BLANK

CHAPTER 18 INSTRUCTION SET

This chapter lists the instruction set of the μPD780924 and 780964 subseries. For the details of the operation and machine language (instruction code) of each instruction, refer to **78K/0 Series User's Manual - Instruction (IEU-1372)**.

18.1 Legend

18.1.1 Operand representation and description formats

In the operand field of each instruction, an operand is described according to the description format for operand representation of that instruction (for details, refer to the assembler specifications). Some operands may be described in two or more description formats. In this case, select one of them. Uppercase characters, #, !, $, and [ ] are keywords and must be described as is. The meanings of the symbols are as follows:

- # : immediate data
- ! : absolute address
- $: relative address
- [ ] : indirect address

To describe immediate data, also describe an appropriate numeric value or label. To describe a label, be sure to describe #, !, $, or [ ].

Register description formats r or rp for an operand can be described as a function name (such as X, A, or C) or absolute name (the name in parentheses in the table below, such as R0, R1, or R2).

Table 18-1. Operand Representation and Description Formats

Representation	Description Format
r	X (R0), A (R1), C (R2), B (R3), E (R4), D (R5), L (R6), H (R7)
rp	AX (RP0), BC (RP1), DE (RP2), HL (RP3)
sfr	Special function register symbol[Note]
sfrp	Special function register symbol (only even address of register that can be manipulated in 16-bit units)[Note]
saddr	FE20H to FF1FH immediate data or label
saddrp	FE20H to FF1FH immediate data or label (even address only)
addr16	0000H to FFFFH immediate data or label (even address only for 16-bit data transfer instruction)
addr11	0800H to 0FFFH immediate data or label
addr5	0040H to 007FH immediate data or label (even address only)
word	16-bit immediate data or label
byte	8-bit immediate data or label
bit	3-bit immediate data or label
RBn	RB0 to RB3

Note FFD0H to FFDFH cannot be addressed.

Remark For the symbols of the special function registers, refer to **Table 3-6 Special Function Register List.**

18.1.2 Description of operation column

A	:	A register; 8-bit accumulator
X	:	X register
B	:	B register
C	:	C register
D	:	D register
E	:	E register
H	:	H register
L	:	L register
AX	:	AX register pair; 16-bit accumulator
BC	:	BC register pair
DE	:	DE register pair
HL	:	HL register pair
PC	:	program counter
SP	:	stack pointer
PSW	:	program status word
CY	:	carry flag
AC	:	auxiliary carry flag
Z	:	zero flag
RBS	:	register bank select flag
IE	:	interrupt request enable flag
NMIS	:	non-maskable interrupt processing flag
( )	:	memory contents indicated by contents of address or register in ( )
x_H, x_L	:	higher 8 bits and lower 8 bits of 16-bit register
∧	:	logical product (AND)
∨	:	logical sum (OR)
∀	:	exclusive logical sum (exclusive OR)
——	:	inverted data
addr16	:	16-bit immediate data or label
jdisp8	:	signed 8-bit data (displacement value)

18.1.3 Description in flag operation column

(Blank)	:	not affected
0	:	cleared to 0
1	:	set to 1
×	:	set/cleared according to result
R	:	value saved before is restored

18.2 Operation List

Instruction Group	Mnemonic	Operand	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
8-bit data transfer	MOV	r, #byte	2	4	–	r ← byte	
		saddr, #byte	3	6	7	(saddr) ← byte	
		sfr, #byte	3	–	7	sfr ← byte	
		A, r Note 3	1	2	–	A ← r	
		r, A Note 3	1	2	–	r ← A	
		A, saddr	2	4	5	A ← (saddr)	
		saddr, A	2	4	5	(saddr) ← A	
		A, sfr	2	–	5	A ← sfr	
		sfr, A	2	–	5	sfr ← A	
		A, !addr16	3	8	9 + n	A ← (addr16)	
		!addr16, A	3	8	9 + m	(addr16) ← A	
		PSW, #byte	3	–	7	PSW ← byte	× × ×
		A, PSW	2	–	5	A ← PSW	
		PSW, A	2	–	5	PSW ← A	× × ×
		A, [DE]	1	4	5 + n	A ← (DE)	
		[DE], A	1	4	5 + m	(DE) ← A	
		A, [HL]	1	4	5 + n	A ← (HL)	
		[HL], A	1	4	5 + m	(HL) ← A	
		A, [HL + byte]	2	8	9 + n	A ← (HL + byte)	
		[HL + byte], A	2	8	9 + m	(HL + byte) ← A	
		A, [HL + B]	1	6	7 + n	A ← (HL + B)	
		[HL + B], A	1	6	7 + m	(HL + B) ← A	
		A, [HL + C]	1	6	7 + n	A ← (HL + C)	
		[HL + C], A	1	6	7 + m	(HL + C) ← A	

Notes 1. When the internal high-speed RAM area is accessed or when an instruction that does not access data is executed
2. When an area other than the internal high-speed RAM area is accessed
3. Except r = A

Remarks 1. One clock of an instruction is equal to one CPU clock (f_{CPU}) selected by processor clock control register (PCC).
2. The number of clocks shown is when the program is stored in the internal ROM area.
3. n indicates the number of wait states when the external memory extension area is read.
4. m indicates the number of wait states when the external memory extension area is written.

Instruction Group	Mnemonic	Operand	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
8-bit data transfer	XCH	A, r Note 3	1	2	–	A ↔ r	
		A, saddr	2	4	6	A ↔ (saddr)	
		A, sfr	2	–	6	A ↔ sfr	
		A, !addr16	3	8	10+n+m	A ↔ (addr16)	
		A, [DE]	1	4	6+n+m	A ↔ (DE)	
		A, [HL]	1	4	6+n+m	A ↔ (HL)	
		A, [HL + byte]	2	8	10+n+m	A ↔ (HL + byte)	
		A, [HL + B]	2	8	10+n+m	A ↔ (HL + B)	
		A, [HL + C]	2	8	10+n+m	A ↔ (HL + C)	
16-bit data transfer	MOVW	rp, #word	3	6	–	rp ← word	
		saddrp, #word	4	8	10	(saddrp) ← word	
		sfrp, #word	4	–	10	sfrp ← word	
		AX, saddrp	2	6	8	AX ← (saddrp)	
		saddrp, AX	2	6	8	(saddrp) ← AX	
		AX, sfrp	2	–	8	AX ← sfrp	
		sfrp, AX	2	–	8	sfrp ← AX	
		AX, rp Note 4	1	4	–	AX ← rp	
		rp, AX Note 4	1	4	–	rp ← AX	
		AX, !addr16	3	10	12 + 2n	AX ← (addr16)	
		!addr16, AX	3	10	12 + 2m	(addr16) ← AX	
	XCHW	AX, rp Note 4	1	4	–	AX ↔ rp	
8-bit operation	ADD	A, #byte	2	4	–	A, CY ← A + byte	× × ×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) + byte	× × ×
		A, r Note 3	2	4	–	A, CY ← A + r	× × ×
		r, A	2	4	–	r, CY ← r + A	× × ×
		A, saddr	2	4	5	A, CY ← A + (saddr)	× × ×
		A, !addr16	3	8	9 + n	A, CY ← A + (saddr16)	× × ×
		A, [HL]	1	4	5 + n	A, CY ← A + (HL)	× × ×
		A, [HL + byte]	2	8	9 + n	A, CY ← A + (HL + byte)	× × ×
		A, [HL + B]	2	8	9 + n	A, CY ← A + (HL + B)	× × ×
		A, [HL + C]	2	8	9 + n	A, CY ← A + (HL + C)	× × ×

Notes 1. When the internal high-speed RAM area is accessed or when an instruction that does not access data is executed
2. When an area other than the internal high-speed RAM area is accessed
3. Except r = A
4. Only when rp = BC, DE, HL

Remarks 1. One clock of an instruction is equal to one CPU clock (f_{CPU}) selected by processor clock control register (PCC).
2. The number of clocks shown is when the program is stored in the internal ROM area.
3. n indicates the number of wait states when the external memory extension area is read.
4. m indicates the number of wait states when the external memory extension area is written.

Instruction Group	Mnemonic	Operand	Byte	Clock		Operation	Flag		
				Note 1	Note 2		Z	AC	CY
8-bit operation	ADDC	A, #byte	2	4	–	A, CY ← A + byte + CY	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) + byte + CY	×	×	×
		A, r Note 3	2	4	–	A, CY ← A + r + CY	×	×	×
		r, A	2	4	–	r, CY ← r + A + CY	×	×	×
		A, saddr	2	4	5	A, CY ← A + (saddr) + CY	×	×	×
		A, !addr16	3	8	9 + n	A, CY ← A + (addr16) + CY	×	×	×
		A, [HL]	1	4	5 + n	A, CY ← A + (HL) + CY	×	×	×
		A, [HL + byte]	2	8	9 + n	A, CY ← A + (HL + byte) + CY	×	×	×
		A, [HL + B]	2	8	9 + n	A, CY ← A + (HL + B) + CY	×	×	×
		A, [HL + C]	2	8	9 + n	A, CY ← A + (HL + C) + CY	×	×	×
	SUB	A, #byte	2	4	–	A, CY ← A – byte	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) – byte	×	×	×
		A, r Note 3	2	4	–	A, CY ← A – r	×	×	×
		r, A	2	4	–	r, CY ← r – A	×	×	×
		A, saddr	2	4	5	A, CY ← A – (saddr)	×	×	×
		A, !addr16	3	8	9 + n	A, CY ← A – (addr16)	×	×	×
		A, [HL]	1	4	5 + n	A, CY ← A – (HL)	×	×	×
		A, [HL + byte]	2	8	9 + n	A, CY ← A – (HL + byte)	×	×	×
		A, [HL + B]	2	8	9 + n	A, CY ← A – (HL + B)	×	×	×
		A, [HL + C]	2	8	9 + n	A, CY ← A – (HL + C)	×	×	×
	SUBC	A, #byte	2	4	–	A, CY ← A – byte – CY	×	×	×
		saddr, #byte	3	6	8	(saddr), CY ← (saddr) – byte – CY	×	×	×
		A, r Note 3	2	4	–	A, CY ← A – r – CY	×	×	×
		r, A	2	4	–	r, CY ← r – A – CY	×	×	×
		A, saddr	2	4	5	A, CY ← A – (saddr) – CY	×	×	×
		A, !addr16	3	8	9 + n	A, CY ← A – (addr16) – CY	×	×	×
		A, [HL]	1	4	5 + n	A, CY ← A – (HL) – CY	×	×	×
		A, [HL + byte]	2	8	9 + n	A, CY ← A – (HL + byte) – CY	×	×	×
		A, [HL + B]	2	8	9 + n	A, CY ← A – (HL + B) – CY	×	×	×
		A, [HL + C]	2	8	9 + n	A, CY ← A – (HL + C) – CY	×	×	×

Notes 1. When the internal high-speed RAM area is accessed or when an instruction that does not access data is executed
2. When an area other than the internal high-speed RAM area is accessed
3. Except r = A

Remarks 1. One clock of an instruction is equal to one CPU clock (f_{CPU}) selected by processor clock control register (PCC).
2. The number of clocks shown is when the program is stored in the internal ROM area.
3. n indicates the number of wait states when the external memory extension area is read.

Instruction Group	Mnemonic	Operand	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
8-bit operation	AND	A, #byte	2	4	–	A ← A∧byte	×
		saddr, #byte	3	6	8	(saddr) ← (saddr)∧byte	×
		A, r Note 3	2	4	–	A ← A∧r	×
		r, A	2	4	–	r ← r∧A	×
		A, saddr	2	4	5	A ← A∧(saddr)	×
		A, !addr16	3	8	9 + n	A ← A∧(addr16)	×
		A, [HL]	1	4	5 + n	A ← A∧(HL)	×
		A, [HL + byte]	2	8	9 + n	A ← A∧(HL + byte)	×
		A, [HL + B]	2	8	9 + n	A ← A∧(HL + B)	×
		A, [HL + C]	2	8	9 + n	A ← A∧(HL + C)	×
	OR	A, #byte	2	4	–	A ← A∨byte	×
		saddr, #byte	3	6	8	(saddr) ← (saddr)∨byte	×
		A, r Note 3	2	4	–	A ← A∨r	×
		r, A	2	4	–	r ← r∨A	×
		A, saddr	2	4	5	A ← A∨(saddr)	×
		A, !addr16	3	8	9 + n	A ← A∨(addr16)	×
		A, [HL]	1	4	5 + n	A ← A∨(HL)	×
		A, [HL + byte]	2	8	9 + n	A ← A∨(HL + byte)	×
		A, [HL + B]	2	8	9 + n	A ← A∨(HL + B)	×
		A, [HL + C]	2	8	9 + n	A ← A∨(HL + C)	×
	XOR	A, #byte	2	4	–	A ← A⊻byte	×
		saddr, #byte	3	6	8	(saddr) ← (saddr)⊻byte	×
		A, r Note 3	2	4	–	A ← A⊻r	×
		r, A	2	4	–	r ← r⊻A	×
		A, saddr	2	4	5	A ← A⊻(saddr)	×
		A, !addr16	3	8	9 + n	A ← A⊻(addr16)	×
		A, [HL]	1	4	5 + n	A ← A⊻(HL)	×
		A, [HL + byte]	2	8	9 + n	A ← A⊻(HL + byte)	×
		A, [HL + B]	2	8	9 + n	A ← A⊻(HL + B)	×
		A, [HL + C]	2	8	9 + n	A ← A⊻(HL + C)	×

Notes 1. When the internal high-speed RAM area is accessed or when an instruction that does not access data is executed
2. When an area other than the internal high-speed RAM area is accessed
3. Except r = A

Remarks 1. One clock of an instruction is equal to one CPU clock (f_{CPU}) selected by processor clock control register (PCC).
2. The number of clocks shown is when the program is stored in the internal ROM area.
3. n indicates the number of wait states when the external memory extension area is read.

Instruction Group	Mnemonic	Operand	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
8-bit operation	CMP	A, #byte	2	4	–	A – byte	× × ×
		saddr, #byte	3	6	8	(saddr) – byte	× × ×
		A, r Note 3	2	4	–	A – r	× × ×
		r, A	2	4	–	r – A	× × ×
		A, saddr	2	4	5	A – (saddr)	× × ×
		A, !addr16	3	8	9 + n	A – (addr16)	× × ×
		A, [HL]	1	4	5 + n	A – (HL)	× × ×
		A, [HL + byte]	2	8	9 + n	A – (HL + byte)	× × ×
		A, [HL + B]	2	8	9 + n	A – (HL + B)	× × ×
		A, [HL + C]	2	8	9 + n	A – (HL + C)	× × ×
16-bit operation	ADDW	AX, #word	3	6	–	AX, CY ← AX + word	× × ×
	SUBW	AX, #word	3	6	–	AX, CY ← AX – word	× × ×
	CMPW	AX, #word	3	6	–	AX – word	× × ×
Multiply/ divide	MULU	X	2	16	–	AX ← A × X	
	DIVUW	C	2	25	–	AX (quotient), C (remainder) ← AX ÷ C	
Increment/ decrement	INC	r	1	2	–	r ← r + 1	× ×
		saddr	2	4	6	(saddr) ← (saddr) + 1	× ×
	DEC	r	1	2	–	r ← r – 1	× ×
		saddr	2	4	6	(saddr) ← (saddr) – 1	× ×
	INCW	rp	1	4	–	rp ← rp + 1	
	DECW	rp	1	4	–	rp ← rp – 1	
Rotate	ROR	A, 1	1	2	–	(CY, A_7 ← A_0, A_{m-1} ← A_m) × 1 time	×
	ROL	A, 1	1	2	–	(CY, A_0 ← A_7, A_{m+1} ← A_m) × 1 time	×
	RORC	A, 1	1	2	–	(CY ← A_0, A_7 ← CY, A_{m-1} ← A_m) × 1 time	×
	ROLC	A, 1	1	2	–	(CY ← A_7, A_0 ← CY, A_{m+1} ← A_m) × 1 time	×
	ROR4	[HL]	2	10	12 + n + m	A_{3-0} ← $(HL)_{3-0}$, $(HL)_{7-4}$ ← A_{3-0}, $(HL)_{3-0}$ ← $(HL)_{7-4}$	
	ROL4	[HL]	2	10	12 + n + m	A_{3-0} ← $(HL)_{7-4}$, $(HL)_{3-0}$ ← A_{3-0}, $(HL)_{7-4}$ ← $(HL)_{3-0}$	

Notes 1. When the internal high-speed RAM area is accessed or when an instruction that does not access data is executed
2. When an area other than the internal high-speed RAM area is accessed
3. Except r = A

Remarks 1. One clock of an instruction is equal to one CPU clock (f_{CPU}) selected by processor clock control register (PCC).
2. The number of clocks shown is when the program is stored in the internal ROM area.
3. n indicates the number of wait states when the external memory extension area is read.
4. m indicates the number of wait states when the external memory extension area is written.

Instruction Group	Mnemonic	Operand	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
BCD adjustment	ADJBA		2	4	–	Decimal Adjust Accumulator after Addition	× × ×
	ADJBS		2	4	–	Decimal Adjust Accumulator after Subtract	× × ×
Bit manipulation	MOV1	CY, saddr.bit	3	6	7	CY ← (saddr.bit)	×
		CY, sfr.bit	3	–	7	CY ← sfr.bit	×
		CY, A.bit	2	4	–	CY ← A.bit	×
		CY, PSW.bit	3	–	7	CY ← PSW.bit	×
		CY, [HL].bit	2	6	7 + n	CY ← (HL).bit	×
		saddr.bit, CY	3	6	8	(saddr.bit) ← CY	
		sfr.bit, CY	3	–	8	sfr.bit ← CY	
		A.bit, CY	2	4	–	A.bit ← CY	
		PSW.bit, CY	3	–	8	PSW.bit ← CY	× ×
		[HL].bit, CY	2	6	8 + n + m	(HL).bit ← CY	
	AND1	CY, saddr.bit	3	6	7	CY ← CY ∧ (saddr.bit)	×
		CY, sfr.bit	3	–	7	CY ← CY ∧ sfr.bit	×
		CY, A.bit	2	4	–	CY ← CY ∧ A.bit	×
		CY, PSW.bit	3	–	7	CY ← CY ∧ PSW.bit	×
		CY, [HL].bit	2	6	7 + n	CY ← CY ∧ (HL).bit	×
	OR1	CY, saddr.bit	3	6	7	CY ← CY ∨ (saddr.bit)	×
		CY, sfr.bit	3	–	7	CY ← CY ∨ sfr.bit	×
		CY, A.bit	2	4	–	CY ← CY ∨ A.bit	×
		CY, PSW.bit	3	–	7	CY ← CY ∨ PSW.bit	×
		CY, [HL].bit	2	6	7 + n	CY ← CY ∨ (HL).bit	×
	XOR1	CY, saddr.bit	3	6	7	CY ← CY ∀ (saddr.bit)	×
		CY, sfr.bit	3	–	7	CY ← CY ∀ sfr.bit	×
		CY, A.bit	2	4	–	CY ← CY ∀ A.bit	×
		CY, PSW. bit	3	–	7	CY ← CY ∀ PSW.bit	×
		CY, [HL].bit	2	6	7 + n	CY ← CY ∀ (HL).bit	×

Notes 1. When the internal high-speed RAM area is accessed or when an instruction that does not access data is executed

2. When an area other than the internal high-speed RAM area is accessed

Remarks 1. One clock of an instruction is equal to one CPU clock (f_{CPU}) selected by processor clock control register (PCC).

2. The number of clocks shown is when the program is stored in the internal ROM area.
3. n indicates the number of wait states when the external memory extension area is read.
4. m indicates the number of wait states when the external memory extension area is written.

Instruction Group	Mnemonic	Operand	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
Bit manipulation	SET1	saddr.bit	2	4	6	(saddr.bit) ← 1	
		sfr.bit	3	–	8	sfr.bit ← 1	
		A.bit	2	4	–	A.bit ← 1	
		PSW.bit	2	–	6	PSW.bit ← 1	× × ×
		[HL].bit	2	6	8+n+m	(HL).bit ← 1	
	CLR1	saddr.bit	2	4	6	(saddr.bit) ← 0	
		sfr.bit	3	–	8	sfr.bit ← 0	
		A.bit	2	4	–	A.bit ← 0	
		PSW.bit	2	–	6	PSW.bit ← 0	× × ×
		[HL].bit	2	6	8+n+m	(HL).bit ← 0	
	SET1	CY	1	2	–	CY ← 1	1
	CLR1	CY	1	2	–	CY ← 0	0
	NOT1	CY	1	2	–	CY ← $\overline{CY}$	×
Call/return	CALL	!addr16	3	7	–	(SP – 1) ← (PC + 3)H, (SP – 2) ← (PC + 3)L, PC ← addr16, SP ← SP – 2	
	CALLF	!addr11	2	5	–	(SP – 1) ← (PC + 2)H, (SP – 2) ← (PC + 2)L, PC15-11 ← 00001, PC10-0 ← addr11, SP ← SP – 2	
	CALLT	[addr5]	1	6	–	(SP – 1) ← (PC + 1)H, (SP – 2) ← (PC + 1)L, PCH ← (00000000, addr5 + 1), PCL ← (00000000, addr5), SP ← SP – 2	
	BRK		1	6	–	(SP – 1) ← PSW, (SP – 2) ← (PC + 1)H, (SP – 3) ← (PC + 1)L, PCH ← (003FH), PCL ← (003EH), SP ← SP – 3, IE ← 0	
	RET		1	6	–	PCH ← (SP + 1), PCL ← (SP), SP ← SP + 2	
	RETI		1	6	–	PCH ← (SP + 1), PCL ← (SP), PSW ← (SP + 2), SP ← SP + 3, NMIS ← 0	R R R
	RETB		1	6	–	PCH ← (SP + 1), PCL ← (SP), PSW ← (SP + 2), SP ← SP + 3	R R R

Notes 1. When the internal high-speed RAM area is accessed or when an instruction that does not access data is executed

2. When an area other than the internal high-speed RAM area is accessed

Remarks 1. One clock of an instruction is equal to one CPU clock (fCPU) selected by processor clock control register (PCC).

2. The number of clocks shown is when the program is stored in the internal ROM area.
3. n indicates the number of wait states when the external memory extension area is read.
4. m indicates the number of wait states when the external memory extension area is written.

Instruction Group	Mnemonic	Operand	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
Stack manipulation	PUSH	PSW	1	2	–	(SP – 1) ← PSW, SP ← SP – 1	
		rp	1	4	–	(SP – 1) ← rpH, (SP – 2) ← rpL, SP ← SP – 2	
	POP	PSW	1	2	–	PSW ← (SP), SP ← SP + 1	R R R
		rp	1	4	–	rpH ← (SP + 1), rpL ← (SP), SP ← SP + 2	
	MOVW	SP, #word	4	–	10	SP ← word	
		SP, AX	2	–	8	SP ← AX	
		AX, SP	2	–	8	AX ← SP	
Unconditional branch	BR	!addr16	3	6	–	PC ← addr16	
		$addr16	2	6	–	PC ← PC + 2 + jdisp8	
		AX	2	8	–	PCH ← A, PCL ← X	
Conditional branch	BC	$addr16	2	6	–	PC ← PC + 2 + jdisp8 if CY = 1	
	BNC	$addr16	2	6	–	PC ← PC + 2 + jdisp8 if CY = 0	
	BZ	$addr16	2	6	–	PC ← PC + 2 + jdisp8 if Z = 1	
	BNZ	$addr16	2	6	–	PC ← PC + 2 + jdisp8 if Z = 0	
	BT	saddr.bit, $addr16	3	8	9	PC ← PC + 3 + jdisp8 if (saddr.bit) = 1	
		sfr.bit, $addr16	4	–	11	PC ← PC + 4 + jdisp8 if sfr.bit = 1	
		A.bit, $addr16	3	8	–	PC ← PC + 3 + jdisp8 if A.bit = 1	
		PSW.bit, $addr16	3	–	9	PC ← PC + 3 + jdisp8 if PSW.bit = 1	
		[HL].bit, $addr16	3	10	11 + n	PC ← PC + 3 + jdisp8 if (HL).bit = 1	
	BF	saddr.bit, $addr16	4	10	11	PC ← PC + 4 + jdisp8 if (saddr.bit) = 0	
		sfr.bit, $addr16	4	–	11	PC ← PC + 4 + jdisp8 if sfr.bit = 0	
		A.bit, $addr16	3	8	–	PC ← PC + 3 + jdisp8 if A.bit = 0	
		PSW.bit, $addr16	4	–	11	PC ← PC + 4 + jdisp8 if PSW. bit = 0	
		[HL].bit, $addr16	3	10	11 + n	PC ← PC + 3 + jdisp8 if (HL).bit = 0	

Notes 1. When the internal high-speed RAM area is accessed or when an instruction that does not access data is executed

2. When an area other than the internal high-speed RAM area is accessed

Remarks 1. One clock of an instruction is equal to one CPU clock (fCPU) selected by processor clock control register (PCC).

2. The number of clocks shown is when the program is stored in the internal ROM area.

3. n indicates the number of wait states when the external memory extension area is read.

Instruction Group	Mnemonic	Operand	Byte	Clock		Operation	Flag
				Note 1	Note 2		Z AC CY
Conditional branch	BTCLR	saddr.bit, $addr16	4	10	12	PC ← PC + 4 + jdisp8 if (saddr.bit) = 1 then reset (saddr.bit)	
		sfr.bit, $addr16	4	–	12	PC ← PC + 4 + jdisp8 if sfr.bit = 1 then reset sfr.bit	
		A.bit, $addr16	3	8	–	PC ← PC + 3 + jdisp8 if A.bit = 1 then reset A.bit	
		PSW.bit, $addr16	4	–	12	PC ← PC + 4 + jdisp8 if PSW.bit = 1 then reset PSW.bit	× × ×
		[HL].bit, $addr16	3	10	12+n+m	PC ← PC + 3 + jdisp8 if (HL).bit = 1 then reset (HL).bit	
	DBNZ	B, $addr16	2	6	–	B ← B – 1, then PC ← PC + 2 + jdisp8 if B ≠ 0	
		C, $addr16	2	6	–	C ← C –1, then PC ← PC + 2 + jdisp8 if C ≠ 0	
		saddr, $addr16	3	8	10	(saddr) ← (saddr) – 1, then PC ← PC + 3 + jdisp8 if (saddr) ≠ 0	
CPU control	SEL	RBn	2	4	–	RBS1, 0 ← n	
	NOP		1	2	–	No operation	
	EI		2	–	6	IE ← 1 (Enable interrupt)	
	DI		2	–	6	IE ← 0 (Disable interrupt)	
	HALT		2	6	–	Set HALT mode	
	STOP		2	6	–	Set STOP mode	

Notes 1. When the internal high-speed RAM area is accessed or when an instruction that does not access data is executed
2. When an area other than the internal high-speed RAM area is accessed

Remarks 1. One clock of an instruction is equal to one CPU clock (f_{CPU}) selected by processor clock control register (PCC).
2. The number of clocks shown is when the program is stored in the internal ROM area.
3. n indicates the number of wait states when the external memory extension area is read.
4. m indicates the number of wait states when the external memory extension area is written.

18.3 Instruction List by Addressing

(1) 8-bit instructions

MOV, XCH, ADD, ADDC, SUB, SUBC, AND, OR, XOR, CMP, MULU, DIVUW, INC, DEC, ROR, ROL, RORC, ROLC, ROR4, ROL4, PUSH, POP, DBNZ

2nd Operand / 1st Operand	#byte	A	r[Note]	sfr	saddr	!addr16	PSW	[DE]	[HL]	[HL + byte] [HL + B] [HL + C]	$addr16	1	None
A	ADD ADDC SUB SUBC AND OR XOR CMP		MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV	MOV XCH	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP	MOV XCH ADD ADDC SUB SUBC AND OR XOR CMP		ROR ROL RORC ROLC	
r	MOV	MOV ADD ADDC SUB SUBC AND OR XOR CMP											INC DEC
B, C											DBNZ		
sfr	MOV	MOV											
saddr	MOV ADD ADDC SUB SUBC AND OR XOR CMP	MOV									DBNZ		INC DEC
!addr16		MOV											
PSW	MOV	MOV											PUSH POP
[DE]		MOV											
[HL]		MOV											ROR4 ROL4
[HL + byte] [HL + B] [HL + C]		MOV											
X													MULU
C													DIVUW

Note Except for r = A

(2) 16-bit instructions

MOVW, XCHW, ADDW, SUBW, CMPW, PUSH, POP, INCW, DECW

1st Operand \ 2nd Operand	#word	AX	rp[Note]	sfrp	saddrp	!addr16	SP	None
AX	ADDW SUBW CMPW		MOVW XCHW	MOVW	MOVW	MOVW	MOVW	
rp	MOVW	MOVW[Note]						INCW DECW PUSH POP
sfrp	MOVW	MOVW						
saddrp	MOVW	MOVW						
!addr16		MOVW						
SP	MOVW	MOVW						

Note Only when rp = BC, DE, HL

(3) Bit manipulation instructions

MOV1, AND1, OR1, XOR1, SET1, CLR1, NOT1, BT, BF, BTCLR

1st Operand \ 2nd Operand	A.bit	sfr.bit	saddr.bit	PSW.bit	[HL].bit	CY	$addr16	None
A.bit						MOV1	BT BF BTCLR	SET1 CLR1
sfr.bit						MOV1	BT BF BTCLR	SET1 CLR1
saddr.bit						MOV1	BT BF BTCLR	SET1 CLR1
PSW.bit						MOV1	BT BF BTCLR	SET1 CLR1
[HL].bit						MOV1	BT BF BTCLR	SET1 CLR1
CY	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1	MOV1 AND1 OR1 XOR1			SET1 CLR1 NOT1

(4) Call/branch instructions

CALL, CALLF, CALLT, BR, BC, BNC, BZ, BNZ, BT, BF, BTCLR, DBNZ

1st Operand \ 2nd Operand	AX	!addr16	!addr11	[addr5]	$addr16
Basic instruction	BR	CALL BR	CALLF	CALLT	BR BC BNC BZ BNZ
Compound instruction					BT BF BTCLR DBNZ

(5) Other instructions

ADJBA, ADJBS, BRK, RET, RETI, RETB, SEL, NOP, EI, DI, HALT, STOP

[MEMO]

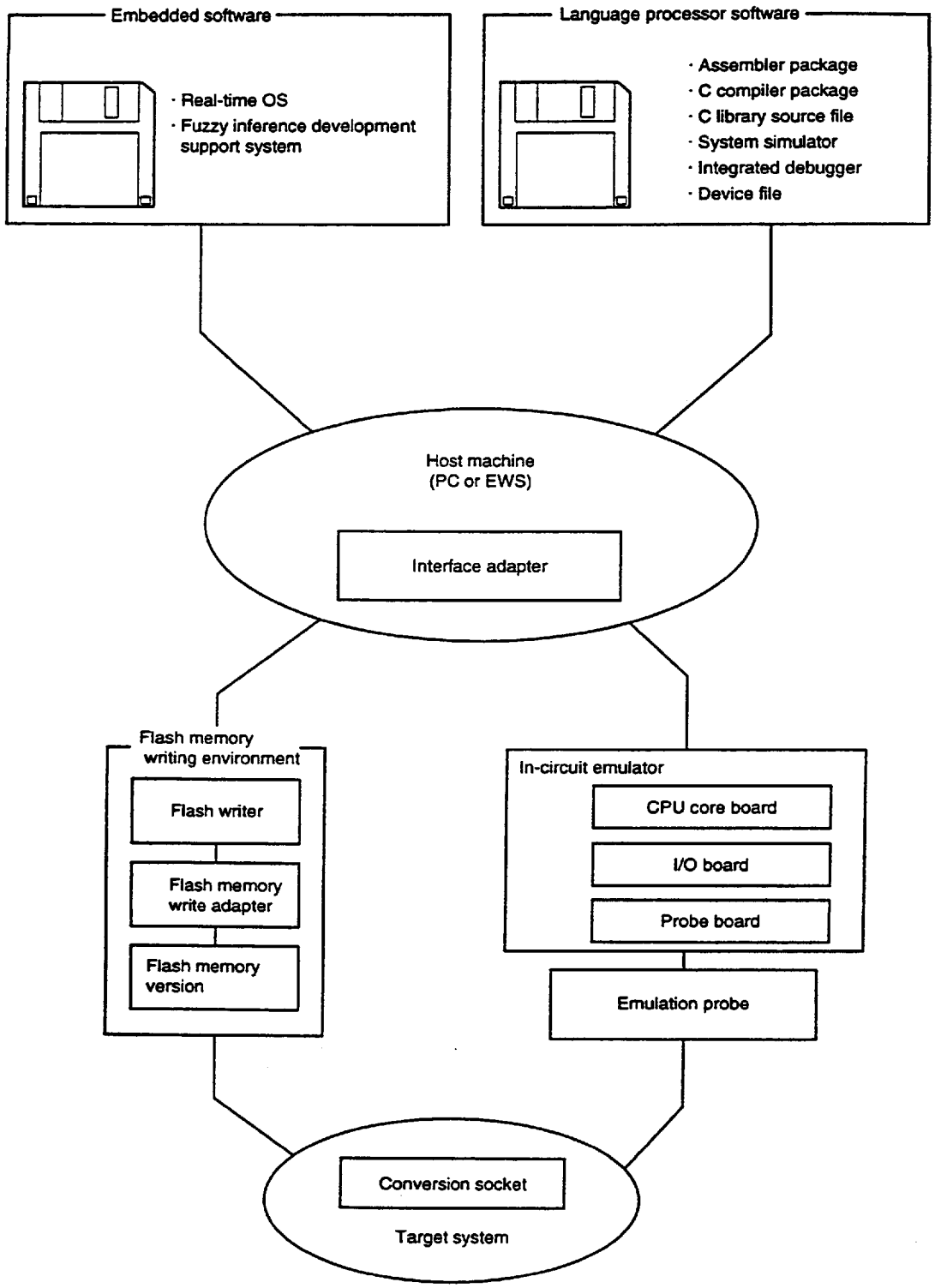
PAGE(S) INTENTIONALLY BLANK

APPENDIX A DEVELOPMENT TOOLS

The following development tools are available for development of systems using the μPD780924 and 780964 subseries.

Figure A-1 shows development tools.

Figure A-1. Development Tool Configuration



A.1 Language Processing Software

RA78K/0 Assembler package	Program that converts program written in mnemonic into object codes that can be executed by microcontroller. In addition, automatic functions to generate symbol table and optimize branch instructions are also provided. Used in combination with optional device file (DF780964).
	Part number: μSxxxxRA78K0
CC78K/0 C compiler package	Program that converts program written in C language into object codes that can be executed by microcontroller. Used in combination with optional assembler package (RA78K/0) and device file (DF780964).
	Part number: μSxxxxCC78K0
DF780964Note 1, 2 Device file	File containing information peculiar to the device. Used in combination with optional RA78K/0, CC78K/0, SM78K0, or ID78K0.
	Part number: μSxxxxDF780964
CC78K/0-L C library source file	Source program of functions for generating object library included in C compiler package (CC78K/0). Necessary for changing object library included in CC78K/0 according to customer's specifications.
	Part number: μSxxxxCC78K0-L

Notes 1. DF780964 is a common file that can be used with RA78K/0, CC78K/0, SM78K0, and ID78K0.
2. Under development

Remark xxxx in part number differs depending on the host machine and OS used.

μSxxxxRA78K0
μSxxxxCC78K0
μSxxxxDF780964
μSxxxxCC78K0-L

xxxx	Host Machine	OS	Supply Media
5A13	PC-9800 Series	MS-DOS™ (Ver.3.30 to Ver.6.2Note)	3.5" 2HD
5A10			5" 2HD
7B13	IBM PC/AT™ and compatible machines	Refer to A.4	3.5" 2HC
7B10			5" 2HC
3H15	HP9000 Series 300™	HP-UX™ (rel.7.05B)	Cartridge tape (QIC-24)
3P16	HP9000 Series 700™	HP-UX (rel.9.01)	Digital audio tape (DAT)
3K15	SPARCstation™	SunOS™ (rel.4.1.1)	Cartridge tape (QIC-24)
3M15	EWS4800 Series (RISC)	EWS-UX/V (rel.4.0)	

Note Although MS-DOS Ver. 5.0 and above have a task swap function, this function cannot be used with this software.

A.2 Flash Memory Writing Tools

Flashpro Flash writer	Flash writer dedicated to microcontroller with flash memory. This is a product of Naitou Densei Machidaseisakusho Co., Ltd.
FA-64CW[Note] Flash memory writing adapter	Flash memory writing adapter for the μPD780924 and 780964 subseries and is connected to the Flashpro. This adapter is for a 64-pin plastic shrink DIP (CW type). This is a product of Naitou Densei Machidaseisakusho Co., Ltd.
FA-64GC[Note] Flash memory writing adapter	Flash memory writing adapter for the μPD780924 and 780964 subseries and is connected to the Flashpro. This adapter is for a 64-pin plastic QFP (GC-AB8 type). This is a product of Naitou Densei Machidaseisakusho Co., Ltd.

Note Under development

A.3 Debugging Tools

A.3.1 Hardware

IE-780000-SL[Note] in-circuit emulator	This in-circuit emulator is used to debug hardware and software when an application system using the 78K/0 series is developed. It supports the integrated debugger (ID78K0). This emulator is used in combination with an emulation probe and an interface adapter that connects the emulator with the host machine. For connection with EWS, 10Base-T and 10Base-5 are supported as Ethernet™.
IE-70000-98-IF-B interface adapter	Adapter necessary when using the PC-9800 series (except the notebook type) as the host machine of the IE-780000-SL.
IE-70000-98N-IF interface adapter	Adapter and cable necessary when using the notebook type PC-9800 series as the host machine of the IE-780000-SL. This adapter cannot be connected to a notebook type computer unless the connector of the expansion bus is of 110-pin type.
IE-70000-PC-IF-B interface adapter	Adapter necessary when using IBM PC/AT and compatible machines as the host machine of the IE-780000-SL.
IE-78K0-SL-EM[Note] CPU core board	This board is used in combination with an in-circuit emulator, I/O board, and probe board, to emulate the CPU of the 78K/0 series.
IE-780964-SL-EM1[Note] probe board	This board is used to set mask options and convert pin connection.
EP-64CW-SL[Note] emulation probe	This is a probe to connect an in-circuit emulator and target system. It is for a 64-pin plastic shrink DIP (CW type).
EP-64GC-SL[Note] emulation probe	This is a probe to connect an in-circuit emulator and target system. It is for 64-pin plastic QFP (GC-AB8 type). One 64-pin conversion socket, TQPACK064SA, which facilitates development of the target system is supplied as an accessory.
TQPACK064SA conversion socket	This conversion socket connects the board of the target system created for mounting 64-pin plastic QFP (GC-AB8 type) and the EP-64GC-SL. The TQPACK064SA is made by TOKYO ELETECH CORPORATION (Tokyo (03)5295-1661). Contact an authorized NEC distributor to purchase this product.

Note Under development

A.3.2 Software (1/2)

SM78K0 system simulator	This simulator simulates the operation of the target system on the host machine and is used to debug the target system at C source level or assembler level. The SM78K0 operates in Windows™. By using the SM78K0, the logic and performance of the application can be verified independently of hardware development even if an in-circuit emulator is not used, so that the development efficiency can be enhanced and software quality can be improved. This simulator is used in combination with an optional device file (DF780964).
	Part number: μS××××SM78K0

Remark ×××× in the part number differs depending on the host machine and OS used.

μS××××SM78K0

××××	Host Machine	OS	Supply Media
AA13	PC-9800 series	MS-DOS (Ver.3.30 to Ver.6.2[Note]) + Windows (Ver.3.0 to Ver.3.1)	3.5" 2HD
AA10			5" 2HD
AB13	IBM PC/AT and compatible machines (Japanese Windows)	Refer to **A.4.**	3.5" 2HC
AB10			5" 2HC
BB13	IBM PC/AT and compatible machines (English Windows)		3.5" 2HC
BB10			5" 2HC

Note Although MS-DOS Ver.5.0 and above have a task swap function, this function cannot be used with this software.

A.3.2 Software (2/2)

ID78K0 integrated debugger	This is a control program that is used to debug the 78K/0 series. It uses Windows on a personal computer and OSF/Motif™ on EWS as a graphical user interface, and has the appearance and operability conforming to these interfaces. Moreover, debugging functions supporting C language are reinforced, and the trace result can be displayed in C language level by using a window integrating function that associates the source program, disassemble display, and memory display with the trace result. In addition, it can enhance the debugging efficiency of a program using a real-time OS by incorporating function expansion modules such as a task debugger and system performance analyzer. This debugger is used in combination with an optional device file (DF780964).
	Part number: μSxxxxID78K0
DF780964Note 1, 2 device file	File storing information peculiar to the device. Used in combination with optional RA78K/0, CC78K/0, SM78K0, or ID78K0.
	Part number: μSxxxxDF780964

Notes 1. The DF780964 is a common file that can be used with RA78K/0, CC78K/0, SM78K0, and ID78K0.
2. Under development

Remark xxxx in the part number differs depending on the host machine and OS used.

μSxxxxID78K0
μSxxxxDF780964

xxxx	Host Machine	OS	Supply Media
AA13	PC-9800 series	MS-DOS (Ver.3.30 to Ver.6.2Note) + Windows (Ver.3.1)	3.5" 2HD
AA10			5" 2HD
AB13	IBM PC/AT and compatible machines (Japanese Windows)	Refer to **A.4.**	3.5" 2HC
AB10			5" 2HC
BB13	IBM PC/AT and compatible machines (English Windows)		3.5" 2HC
BB10			5" 2HC
3P16	HP9000 Series 700	HP-UX (rel.9.01)	Digital audio tape (DAT)
3K15	SPARCstation	SunOS (rel.4.1.1)	Cartridge tape (QIC-24)
3K13			3.5" 2HC
3R16	NEWS™ (RISC)	NEWS-OS™ (6.1x)	1/4" CGMT
3R13			3.5" 2HC
3M15	EWS4800 series (RISC)	EWS-UX/V (rel.4.0)	Cartridge tape (QIC-24)

Note Although MS-DOS Ver.5.0 and above have a task swap function, this function cannot be used with this software.

A.4 OS for IBM PC

The following OSs for the IBM PC are supported.

To operate SM78K0, ID78K0, and FE9200 (refer to **B.2 Fuzzy Inference Development Support System**), Windows (Ver. 3.0 to Ver. 3.1) is necessary.

OS	Version
PC DOS™	Ver. 5.02 to Ver. 6.3
	J6.1/V[Note] to J6.3/V[Note]
IBM DOS™	J5.02/V[Note]
MS-DOS	Ver. 5.0 to Ver. 6.22
	5.0/V[Note] to 6.2/V[Note]

Note Only English mode is supported.

Caution Although Ver. 5.0 and above have a task swap function, this function cannot be used with this software.

A.5 Changing Other In-Circuit Emulators to In-Circuit Emulator for 78K/0 Series

If you have an in-circuit emulator for the 78K series or 75X/XL series, your in-circuit emulator can be upgraded to be equivalent to the in-circuit emulator IE-780000-SL for the 78K/0 series by exchanging the break board with the IE-78000-R-BK and connecting interface board IE-78000-R-IF. CPU core board IE-78K0-SL-EM is not necessary.

Table A-1. Changing Other In-Circuit Emulators to IE-780000-SL

Series Name	Your In-Circuit Emulator	Necessary Board
78K/0 series	IE-78000-R, IE-78000-R-A	IE-78000-R-BK IE-78000-R-IF
75X/XL series	IE-75000-R[Note], IE-75001-R	
78K/I series	IE-78130-R, IE-78140-R	
78K/II series	IE-78230-R[Note], IE-78230-R-A, IE-78240-R[Note], IE-78240-R-A	
78K/III series	IE-78320-R[Note], IE-78327-R, IE-78330-R, IE-78350-R	
78K/IV series	IE-784000-R	

Note Maintenance parts

[MEMO]

PAGE(S) INTENTIONALLY BLANK

APPENDIX B EMBEDDED SOFTWARE

For efficient program development and maintenance of μPD780924 and 780964 subseries, the following embedded softwares are available.

B.1 Real-Time OS (1/2)

RX78K/0 Real-time OS	RX78K/0 is real-time OS conforming to μITRON specifications. Tool (configurator) that creates nucleus of RX78K/0 and plural information tables is supplied. Used in combination with an optional assembler package (RA78K/0) and device file (DF780964).
	Part number: μS××××RX78013-ΔΔΔΔ

Caution When purchasing the RX78K/0, fill an application form and conclude the contract for use permission in advance.

Remark ×××× and ΔΔΔΔ in the part number differ depending on the host machine and OS used.

μS××××RX78013-ΔΔΔΔ

ΔΔΔΔ	Product outline	Upper limit of quantity for mass production
001	Evaluation object	Do not use for mass-produced product.
100K	Object for mass-produced product	0.1 million units
001M		1 million units
010M		10 million units
S01	Source program	Source program for mass-produced object

××××	Host machine	OS	Supply media
5A13	PC-9800 series	MS-DOS (Ver.3.30 to Ver.6.2 Note)	3.5" 2HD
5A10			5" 2HD
7B13	IBM PC/AT and compatible machines	Refer to **A.4**	3.5" 2HC
7B10			5" 2HC
3H15	HP9000 series 300	HP-UX (rel.7.05B)	Cartridge tape (QIC-24)
3P16	HP9000 series 700	HP-UX (rel.9.01)	Digital audio tape (DAT)
3K15	SPARCstation	SunOS (rel.4.1.1)	Cartridge tape (QIC-24)
3M15	EWS4800 series (RISC)	EWS-UX/V (rel.4.0)	

Note Although MS-DOS Ver. 5.0 and above have a task swap function, this function cannot be used with this software.

B.1 Real-Time OS (2/2)

MX78K0 OS	μITRON-specification subset OS. Nucleus of MX78K0 is supplied. This OS performs task management, event management, and time management. It controls the task execution sequence for task management and selects the task to be executed next.
	Part number: μS××××MX78K0-ΔΔΔ

Remark ×××× and ΔΔΔ in the part number differ depending on the host machine and OS used.

μS××××MX78K0-ΔΔΔ

ΔΔΔ	Product outline	Note
001	Evaluation object	Use for trial product
××	Object for mass-produced product	Use for mass-produced product
S01	Source program	Can be purchased only when object for mass-produced product is purchased

××××	Host machine	OS	Supply media
5A13	PC-9800 series	MS-DOS (Ver.3.30 to Ver.6.2 Note)	3.5" 2HD
5A10			5" 2HD
7B13	IBM PC/AT and compatible machines	Refer to **A.4**	3.5" 2HC
7B10			5" 2HC
3H15	HP9000 series 300	HP-UX (rel.7.05B)	Cartridge tape (QIC-24)
3P16	HP9000 series 700	HP-UX (rel.9.01)	Digital audio tape (DAT)
3K15	SPARCstation	SunOS (rel.4.1.1)	Cartridge tape (QIC-24)
3M15	EWS4800 series (RISC)	EWS-UX/V (rel.4.0)	

Note Although MS-DOS Ver. 5.0 and above have a task swap function, this function cannot be used with this software.

B.2 Fuzzy Inference Development Support System

FE9000/FE9200 Fuzzy knowledge data creation tool	Program that supports input, edit, and evaluation (simulation) of fuzzy knowledge data (fuzzy rule and membership function). FE9200 works on Windows.
	Part number: μSxxxxFE9000 (PC-9800 series) μSxxxxFE9200 (IBM PC/AT and compatible machines)
FT9080/FT9085 Translator	Program that translates fuzzy knowledge data obtained by using fuzzy knowledge data creation tool into assembler source program for RA78K/0.
	Part number: μSxxxxFT9080 (PC-9800 series) μSxxxxFT9085 (IBM PC/AT and compatible machines)
FI78K0 Fuzzy inference module	Program that executes fuzzy inference. Executes fuzzy inference when linked with fuzzy knowledge data translated by translator.
	Part number: μSxxxxFI78K0 (PC-9800 series, IBM PC/AT and compatible machines)
FD78K0 Fuzzy inference debugger	Support software for evaluation and adjustment of fuzzy knowledge data by using in-circuit emulator and at hardware level.
	Part number: μSxxxxFD78K0 (PC-9800 series, IBM PC/AT and compatible machines)

Remark xxxx in the part number differs depending on the host machine and OS used.

μSxxxxFE9000
μSxxxxFT9080
μSxxxxFI78K0
μSxxxxFD78K0

xxxx	Host machine	OS	Supply media
5A13	PC-9800 series	MS-DOS (Ver. 3.30 to Ver. 6.2[Note])	3.5" 2HD
5A10			5" 2HD

Note Although MS-DOS Ver. 5.0 and above have a task swap function, this function cannot be used with this software.

μSxxxxFE9200
μSxxxxFT9085
μSxxxxFI78K0
μSxxxxFD78K0

xxxx	Host machine	OS	Supply media
7B13	IBM PC/AT and compatible machines	Refer to A.4	3.5" 2HC
7B10			5" 2HC

APPENDIX C REGISTER INDEX

C.1 Register Index (In Alphabetical Order with Respect to the Register Name)

C.2 Register Index (In Alphabetical Order with Respect to the Register Symbol)