NEC

MOS INTEGRATED CIRCUIT

μPD78330, 78334

16/8 BIT SINGLE-CHIP MICROCOMPUTER

The μPD78334 is a product of the 78K/III series. It contains a 16-bit high-performance CPU.

The μPD78334 has a more powerful function for the real-time pulse unit and contains more extended internal memory than the μPD78322. This is the most suitable product for real-time control such as DC servomotor control.

The μPD78334 contains a 32K-byte mask ROM and a 1024-byte RAM.

The μPD78330 is a ROM-less version of the μPD78334. The μPD78P334 is a PROM version of the μPD78334.

The following user's manual describes the details of functions. Be sure to read it before design.

μPD78334 User's Manual: IEU-1315

FEATURES

- 16-bit internal architecture, 8-bit external data bus
- High-speed data processing using the pipeline control system and instruction prefetching
  - Minimum instruction execution time: 250 ns
    (internal clock frequency at 8 MHz, external clock frequency at 16 MHz)
- An instruction set suited for control applications (μPD78312 upward compatible)
  - 16-bit arithmetic/logical instruction
  - Multiply/divide instruction (16 bits × 16 bits, 32 bits ÷ 16 bits)
  - Signed multiply instruction
  - Bit manipulation instruction and so on
- More powerful real-time pulse unit
- 10-bit high-resolution A/D (analog-to-digital) converter: 16 channels
- Discrete serial interfaces: 2 channels
  - UART
  - Synchronous serial interface/SBI
- Built-in advanced function interrupt controller
  - A 3-level priority can be specified by software.
  - One interrupt processing mode can be selected out of three types: vectored interrupt function, macro service function, and context switching function.
- High-procession PWM signal output function: 2 channels
- Watchdog timer function for detecting a program crash

APPLICATIONS

Factory automation (FA) field

Unless other wise specified, the explanation of the μPD78334 also applies to the μPD78330.

Document No. U10185EJ5V0DS00
(5th edition)
Date Published September 1995 P
Printed in Japan

Major changes in this revision are indicated by stars (★) in the margins.

ORDERING INFORMATION

Part number	Package	Internal ROM
μPD78330GJ-5BG	94-pin plastic QFP (20 × 20 mm)	Not provided
μPD78330LQ	84-pin plastic QFJ (square-type 1150 mil)	Not provided
μPD78334GJ-xxx-5BG	94-pin plastic QFP (20 × 20 mm)	Mask ROM
μPD78334LQ-xxx	84-pin plastic QFJ (square-type 1150 mil)	Mask ROM

Remark xxx indicates the ROM code number

PIN CONFIGURATION (TOP VIEW)

- **84-pin plastic QFJ (square-type 1150 mil)**
  μPD78330LQ
  μPD78330LQ-xxx

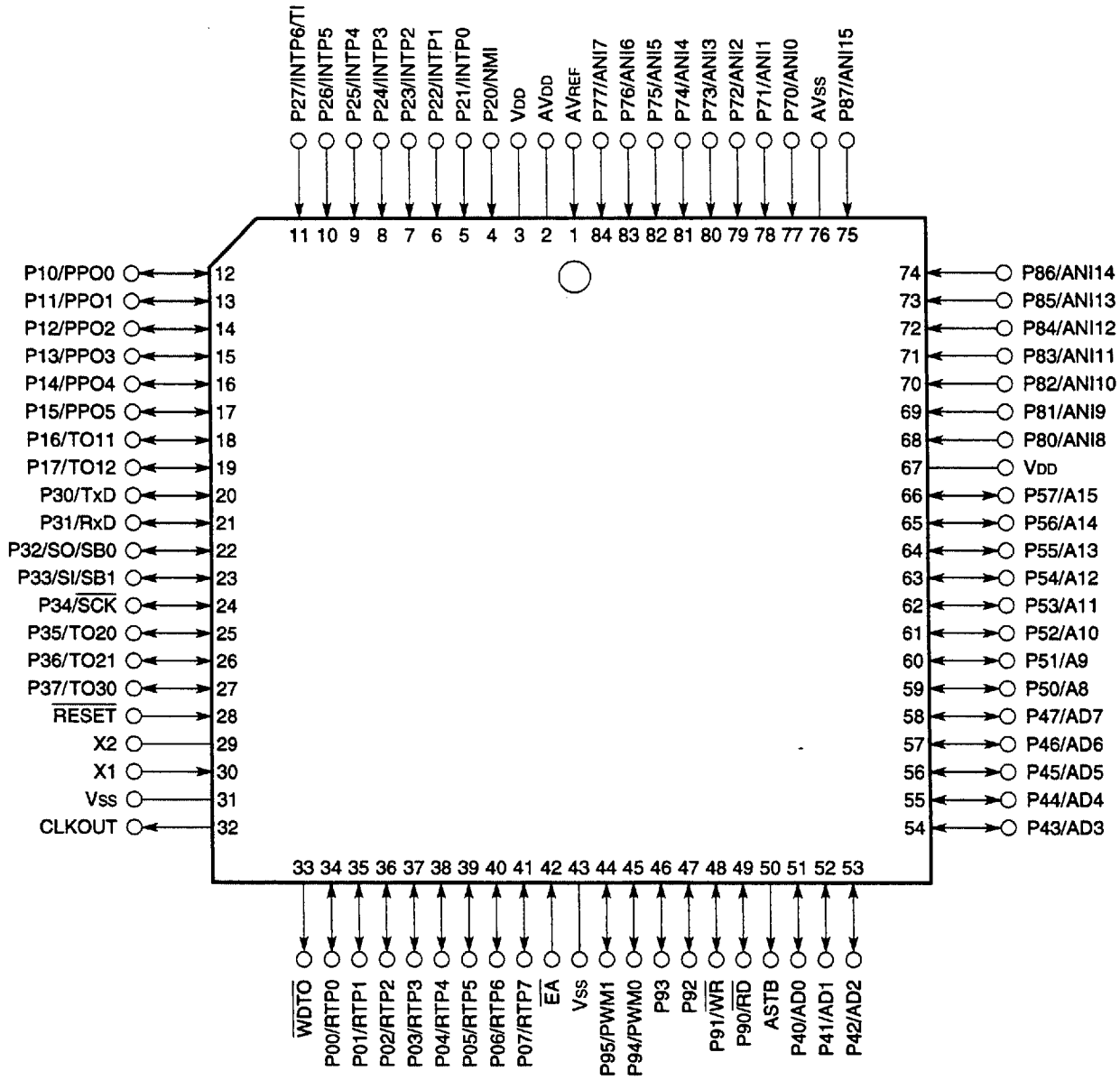
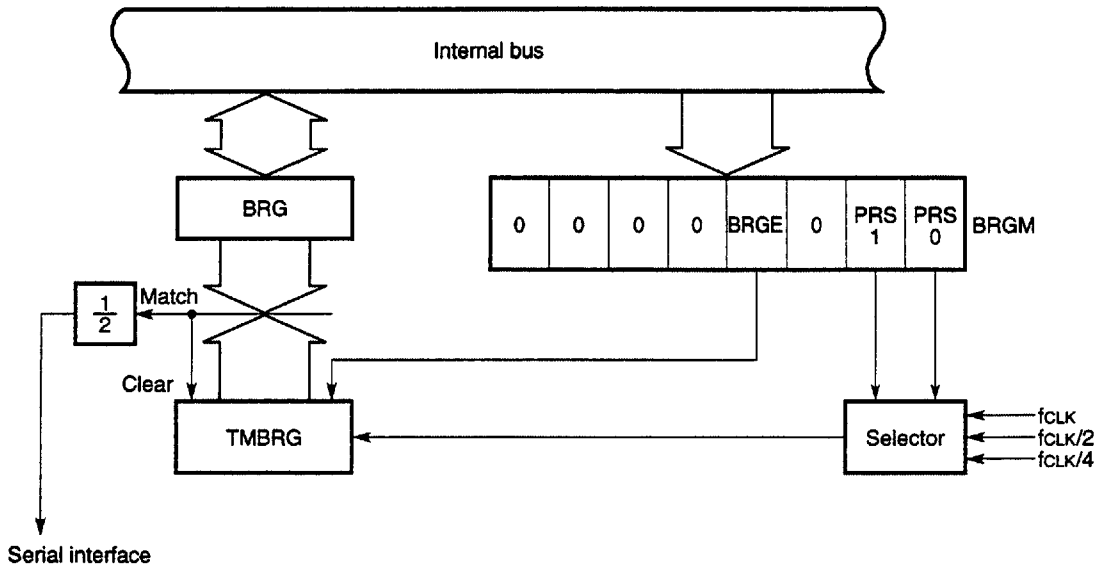


Fig. 4-23 Configuration of the Baud Rate Generator



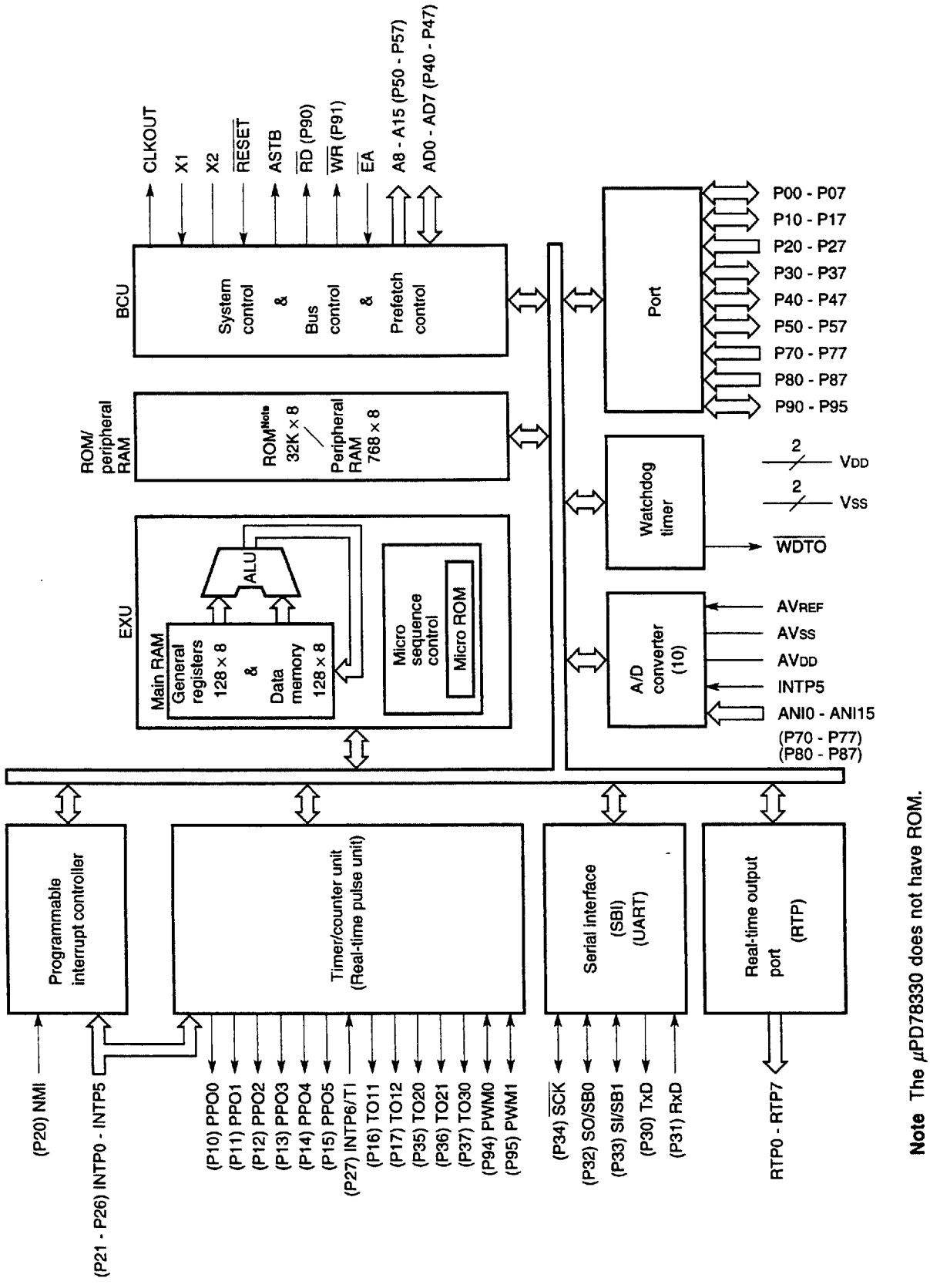
P00 - P07	:	Port 0
P10 - P17	:	Port 1
P20 - P27	:	Port 2
P30 - P37	:	Port 3
P40 - P47	:	Port 4
P50 - P57	:	Port 5
P70 - P77	:	Port 7
P80 - P87	:	Port 8
P90 - P95	:	Port 9
AD0 - AD7	:	Address/data bus
A8 - A15	:	Address bus
ANI0 - ANI15	:	Analog input
TI	:	Timer input
TO11, TO12	:	Timer output
TO20, TO21	:	Timer output
TO30	:	Timer output
CLKOUT	:	Clock output
PWM0, PWM1	:	Pulse width modulation output
INTP0 - INTP6	:	Interrupt from peripherals
PTP0 - RTP7	:	Real-time port
PPO0 - PPO5	:	Programmable pulse output

NMI	:	Nonmaskable interrupt
TxD	:	Transmit data
RxD	:	Receive data
SI	:	Serial input
SO	:	Serial output
SB0, SB1	:	Serial bus
$\overline{RD}$	:	Read strobe
$\overline{WR}$	:	Write strobe
ASTB	:	Address strobe
$\overline{EA}$	:	External access
$\overline{RESET}$	:	Reset
$\overline{SCK}$	:	Serial clock
X1, X2	:	Crystal
$\overline{WDTO}$	:	Watchdog timer output
AV_{DD}	:	Analog V_{DD}
AV_{REF}	:	Analog reference voltage
AV_{SS}	:	Analog V_{SS}
V_{DD}	:	Power supply
V_{SS}	:	Ground
NC	:	Open

FUNCTION OVERVIEW

Item	Description
Number of basic instructions	111
Minimum instruction execution time	250 ns (internal clock frequency at 8 MHz, external clock frequency at 16 MHz)
Internal memory	• ROM: 32K bytes (μPD78334) Not provided (μPD78330) • RAM: 1K bytes
Memory space	64K bytes (can externally be extended)
General register	8 bits × 16 × 8 banks
Instruction set	• 16-bit transfer or arithmetic/logical instruction • Multiply/divide instruction (16 bits × 16 bits, 32 bits ÷ 16 bits) • Bit manipulation instruction (transfer, Boolean operation, set, reset, and test) • Context switching instruction • String instruction
Real-time pulse unit	• 18/16-bit timer/counter: 1 channel 18/16-bit compare registers: 5 18/16-bit capture registers: 3 18/16-bit capture/compare registers: 2 Pulse outputs: 6 • 16-bit timers/counters: 3 channels 16-bit compare registers: 5 16-bit capture register: 1 Timer outputs: 5
Real-time output port	Pulse output synchronized with RPU: 8
A/D converter	10-bit resolution: 16 channels
Interrupt function	• External: 8, internal: 13 (also external: 2) • A 3-level priority can be specified by software. • One interrupt processing mode can be selected out of three types: vectored interrupt function, macro service function, and context switching function.
Test source	Internal: 1
I/O line	• Input ports: 24 (16 ports for analog input) • I/O ports: 46 (μPD78334) 28 (μPD78330)
Serial interface	Serial interface with a dedicated baud rate generator UART: 1 channel Clock synchronous serial interface/SBI: 1 channel
Package	• 84-pin plastic QFJ (square-type 1150 mil) • 94-pin plastic QFP (20 × 20 mm)
Others	• Watchdog timer function • Standby function (HALT/STOP)

BLOCK DIAGRAM



Note The μPD78330 does not have ROM.

CONTENTS

1. PIN FUNCTIONS 10
   1.1 PORT PINS 10
   1.2 NON-PORT PINS 12
   1.3 INPUT/OUTPUT CIRCUITS OF EACH PIN 14
   1.4 HANDLING UNUSED PINS 16

2. CPU ARCHITECTURE 17
   2.1 MEMORY SPACE 17
   2.2 DATA MEMORY ADDRESSING 18
   2.3 PROCESSOR REGISTER 19
      2.3.1 Control Register 19
      2.3.2 General Register 20
      2.3.3 Special Function Register (SFR) 20

3. BLOCK FUNCTION 26
   3.1 BUS CONTROL UNIT (BCU) 26
   3.2 EXECUTION UNIT (EXU) 26
   3.3 ROM/RAM 26
   3.4 INTERRUPT CONTROLLER 26
   3.5 REAL-TIME PULSE UNIT (RPU) 26
   3.6 SERIAL INTERFACE 27
   3.7 A/D CONVERTER 27
   3.8 REAL-TIME OUTPUT PORT (RTP) 27
   3.9 WATCHDOG TIMER (WDT) 27
   3.10 PORT 27

4. PERIPHERAL HARDWARE FUNCTIONS 28
   4.1 PORT FUNCTIONS 28
      4.1.1 Hardware Configuration 28
      4.1.2 Functions of the Digital I/O Ports 30
      4.1.3 Port Output Check Function 30
   4.2 CLOCK GENERATOR 31
   4.3 REAL-TIME PULSE UNIT (RPU) 33
      4.3.1 RPU Configuration 33
      4.3.2 Function 35
   4.4 REAL-TIME OUTPUT PORT (RTP) 43
      4.4.1 RTP Configuration 43
      4.4.2 RTP Operation 43
   4.5 A/D CONVERTER 44
      4.5.1 Configuration 44

4.5.2 Operation 45
4.6 SERIAL INTERFACE 49
4.6.1 Asynchronous Serial Interface (UART) 49
4.6.2 Synchronous Serial Interface 51
4.6.3 Baud Rate Generator (BRG) 56
4.7 WATCHDOG TIMER (WDT) 58
4.7.1 WDT Configuration 58
4.7.2 WDT Operation 59
4.8 PWM SIGNAL OUTPUT FUNCTION 60

5. INTERRUPT FUNCTION 61
5.1 TYPES OF INTERRUPT REQUESTS 61
5.2 INTERRUPT HANDLING MODES 63
5.3 MACRO SERVICE 65
5.4 CONTEXT SWITCHING 69
5.4.1 Context Switching Function Based on an Interrupt Request 69
5.4.2 Context Switching Function Based on the BRKCS Instruction 70
5.4.3 Return from Context Switching 70

6. EXTERNAL DEVICE EXPANSION FUNCTION 71

7. STANDBY FUNCTION 72

8. RESET FUNCTION 73

9. INSTRUCTION SET 76

10. ELECTRICAL CHARACTERISTICS 93

11. CHARACTERISTICS CURVE (REFERENCE) 104

12. PACKAGE DIMENSIONS 107

13. RECOMMENDED SOLDERING CONDITIONS 109

APPENDIX A DIFFERENCES BETWEEN THE μPD78334, μPD78330 AND THE μPD78322, μPD78320 110

APPENDIX B TOOLS 111
B.1 DEVELOPMENT TOOLS 111
B.2 EVALUATION TOOLS 115
B.3 EMBEDDED SOFTWARE ..... 115 ★

1. PIN FUNCTIONS

1.1 PORT PINS (1/2)

Pin name	I/O	Function	Dual-function pin
P00 - P07	I/O	Port 0 8-bit I/O port Can be specified as input and output bit by bit. These pins also function as a real-time output port.	RTP0 - RTP7
P10 - P15	I/O	Port 1 8-bit I/O port Can be specified as input and output bit by bit. These pins also function as timer output pins of the real-time pulse unit.	PPO0 - PPO5
P16			TO11
P17			TO12
P20	I	Port 2 8-bit input dedicated port	NMI
P21			INTP0
P22			INTP1
P23			INTP2
P24			INTP3
P25			INTP4
P26			INTP5
P27			INTP6/T1
P30	I/O	Port 3 8-bit I/O port Can be specified as input and output bit by bit. These pins also function as serial interface pins or timer output pins of the real-time pulse unit.	TxD
P31			RxD
P32			SO/SB0
P33			SI/SB1
P34			$\overline{SCK}$
P35			TO20
P36			TO21
P37			TO30
P40 - P47	I/O	Port 4 8-bit I/O port Can be specified as input and output in 8-bit units.	AD0 - AD7
P50 - P57	I/O	Port 5 8-bit I/O port Can be specified as input and output bit by bit.	A8 - A15
P70 - P77	I	Port 7 8-bit input dedicated port These pins also function as analog input pins.	ANI0 - ANI7
P80 - P87	I	Port 8 8-bit input dedicated port These pins also function as analog input pins.	ANI8 - ANI15

1.1 PORT PINS (2/2)

Pin name	I/O	Function	Dual-function pin
P90	I/O	Port 9 6-bit I/O port Can be specified as input and output bit by bit.	$\overline{RD}$
P91			$\overline{WR}$
P92			—
P93			—
P94			PWM0
P95			PWM1

1.2 NON-PORT PINS (1/2)

Pin name	I/O	Function	Dual-function pin
RTP0 - RTP7	O	Synchronized with a trigger signal sent from the real-time pulse unit (RPU) and output a pulse in real time.	P00 - P07
PPO0 - PPO5	O	Programmable pulse output from the RPU	P10 - P15
TO11	O	Timer output from the RPU	P16
TO12			P17
NMI	I	Nonmaskable interrupt request input	P20
INTP0 - INTP5	I	External interrupt input	P21 - P26
INTP6			P27/TI
TI	I	External count input to timer 1 (TM1)	P27/INTP6
TxD	O	Serial data output of the asynchronous serial interface (UART)	P30
RxD	I	Serial data input of the UART	P31
SO	O	Serial data output in the 3-wire mode of the clock synchronous serial interface	P32/SB0
SI	I	Serial data input in the 3-wire mode of the clock synchronous serial interface	P33/SB1
SB0	I/O	Serial data I/O in the serial bus mode of the clock synchronous serial interface	P32/SO
SB1			P33/SI
$\overline{SCK}$	I/O	Serial clock I/O of the clock synchronous serial interface	P34
TO20	O	Timer output from the RPU	P35
TO21			P36
TO30			P37
AD0 - AD7	I/O	Multiplexed address/data bus when an external memory is expanded	P40 - P47
A8 - A15	O	Address bus when an external memory is expanded	P50 - P57
ANI0 - ANI7	I	Analog input to the A/D converter	P70 - P77
ANI8 - ANI15			P80 - P87
$\overline{RD}$	O	Read strobe signal output to the external device	P90
$\overline{WR}$		Write strobe signal output to the external device	P91
PWM0	O	PWM signal output	P94
PWM1			P95

1.2 NON-PORT PINS (2/2)

Pin name	I/O	Function	Dual-function pin
CLKOUT	O	System clock output	–
$\overline{\text{WDTO}}$	O	Output of the signal indicating that a watchdog timer interrupt occurred	–
ASTB	O	Address strobe signal output	–
$\overline{\text{EA}}$	I	Input of the control signal to select external memory access: The $\overline{\text{EA}}$ pin is normally connected to the VDD pin for the μPD78334. Connecting the $\overline{\text{EA}}$ pin to the Vss pin enters the ROM-less mode, enabling access to external memory. The $\overline{\text{EA}}$ pin is connected to the Vss pin for the μPD78330. The level of the $\overline{\text{EA}}$ pin cannot be changed during operations.	–
AVREF	I	A/D converter reference voltage input	–
AVDD	–	Analog power supply of the A/D converter	–
AVSS	–	Ground of the A/D converter	–
$\overline{\text{RESET}}$	I	System reset input	–
X1	I	Crystal input pin for system clock oscillation: A clock signal provided externally is input to the X1 pin. The external clock signal is inverted and input to the X2 pin. (The X2 pin can be left open.)	–
X2	–		–
VDD	–	Positive power supply	–
Vss	–	Ground	–
NC	–	Not internally connected. Connect the NC pin to the Vss pin; or, open.	–

1.3 INPUT/OUTPUT CIRCUITS OF EACH PIN

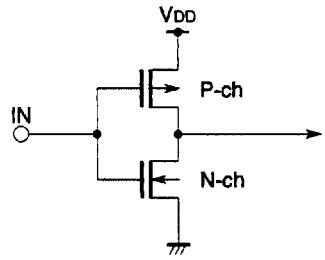
Table 1-1 and Fig. 1-1 show the input and output circuits of each pin in a simplified format.

Table 1-1 Input/Output Circuits of Each Pin

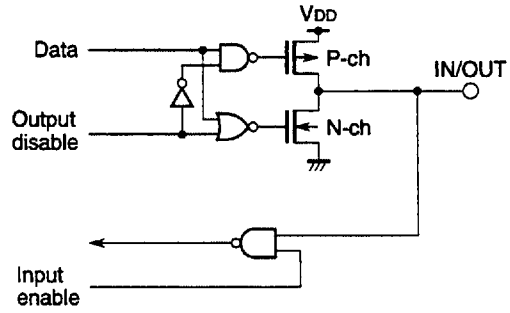
Pin	I/O circuit type	Pin	I/O circuit type
P00/RTP0 - P07/RTP7	5	P35/TO20, P36/TO21, P37/TO30	5
P10/PPO0 - P15/PPO5		P40/AD0 - P47/AD7	
P16/TO11, P17/TO12		P50/A8 - P57/A15	
P20/NMI	2	P70/ANI0 - P77/ANI7	9
P21/INTP0		P80/ANI8 - P87/ANI15	
P22/INTP1		P90/$\overline{RD}$	5
P23/INTP2		P91/$\overline{WR}$	
P24/INTP3		P92	
P25/INTP4		P93	
P26/INTP5		P94/PWM0, P95/PWM1	
P27/INTP6/TI		$\overline{WDTO}$	3
P30/TxD	5	ASTB	4
P31/RxD		CLKOUT	3
P32/SO/SB0	8	$\overline{EA}$	1
P33/SI/SB1		$\overline{RESET}$	2
P34/$\overline{SCK}$			

Fig. 1-1 Input/Output Circuits of Each Pin

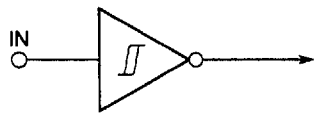
Type 1



Type 5

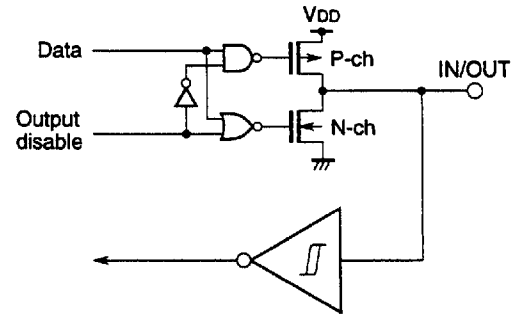


Type 2

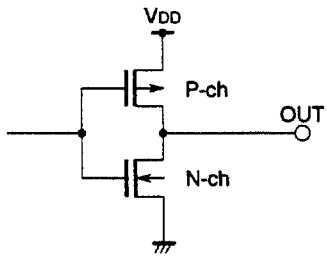


Schmitt trigger input with hysteresis characteristics

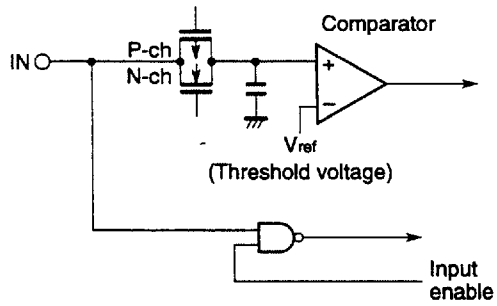
Type 8



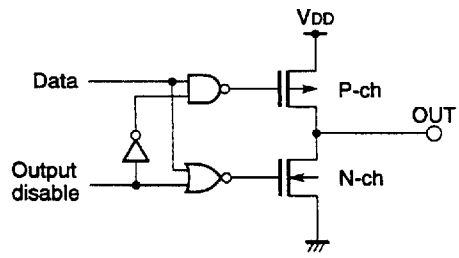
Type 3



Type 9



Type 4



Push-pull output which can output high impedance
(Both the positive and negative channels are off.)

1.4 HANDLING UNUSED PINS

Table 1-2 Handling Unused Pins

Pin	Recommended connection method
P00 - P07	Input status: Each pin is connected to the Vss or VDD pin via a pull-up resister. Output status: Open
P10 - P17	
P20 - P27	Connected to the Vss pin.
P30 - P37	Input status: Each pin is connected to the Vss or VDD pin via a pull-up resister. Output status: Open
P40 - P47	
P50 - P57	
P70 - P77	Connected to the Vss pin.
P80 - P87	
P90 - P95	Input status: Each pin is connected to the Vss or VDD pin via a pull-up resistor. Output status: Open
CLKOUT	Open
$\overline{\text{WDTO}}$	
ASTB	
AVDD	Connected to the VDD pin.
AVREF	Connected to the Vss pin.
AVss	
NC	Connected to the Vss pin; or, open.

2. CPU ARCHITECTURE

2.1 MEMORY SPACE

The μPD78334 can access memory of up to 64K bytes. Fig. 2-1 shows the memory map.

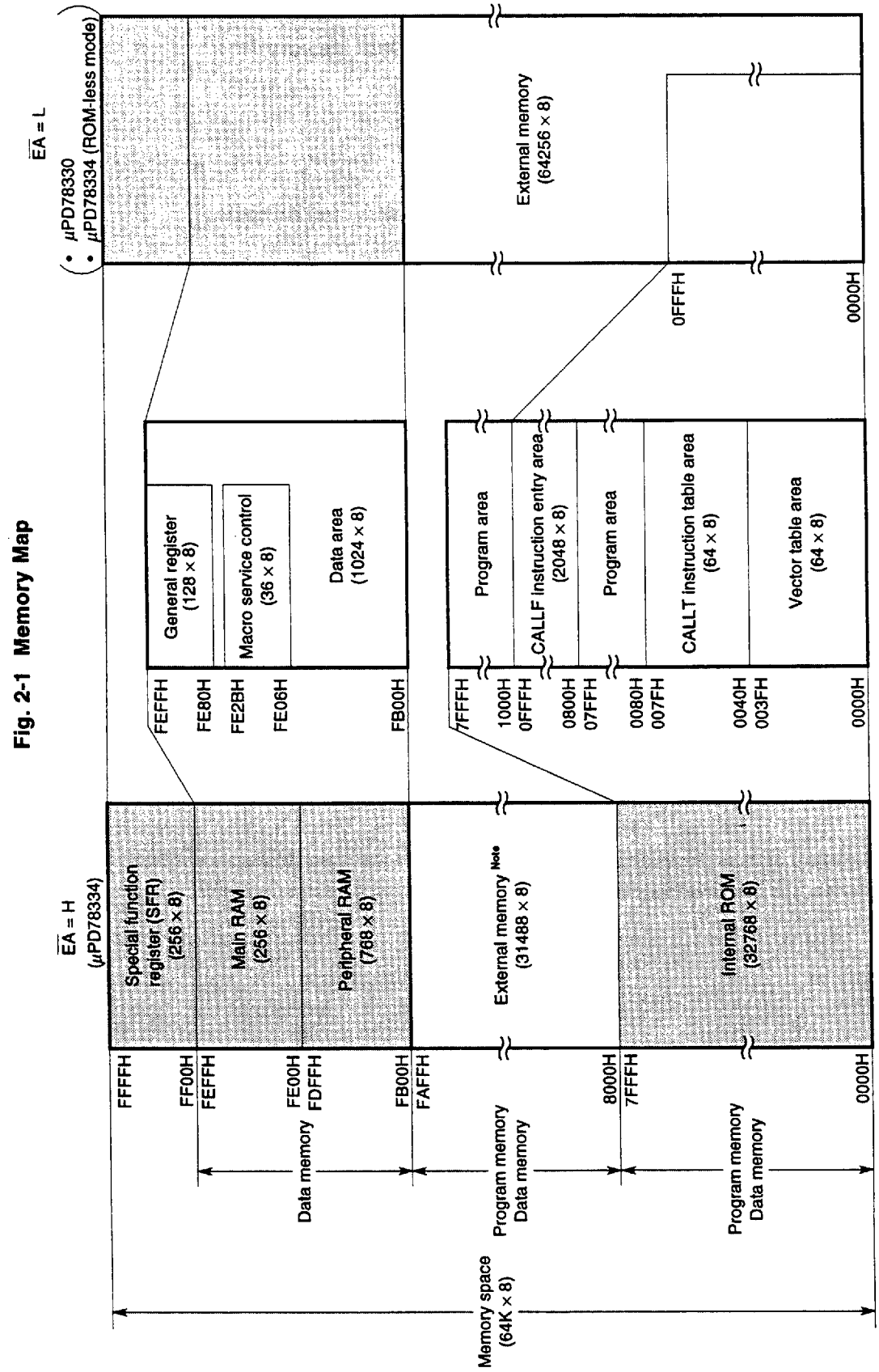


Fig. 2-1 Memory Map

Note To be accessed in the external memory expansion mode.

Remark Shaded portions indicate internal memory.

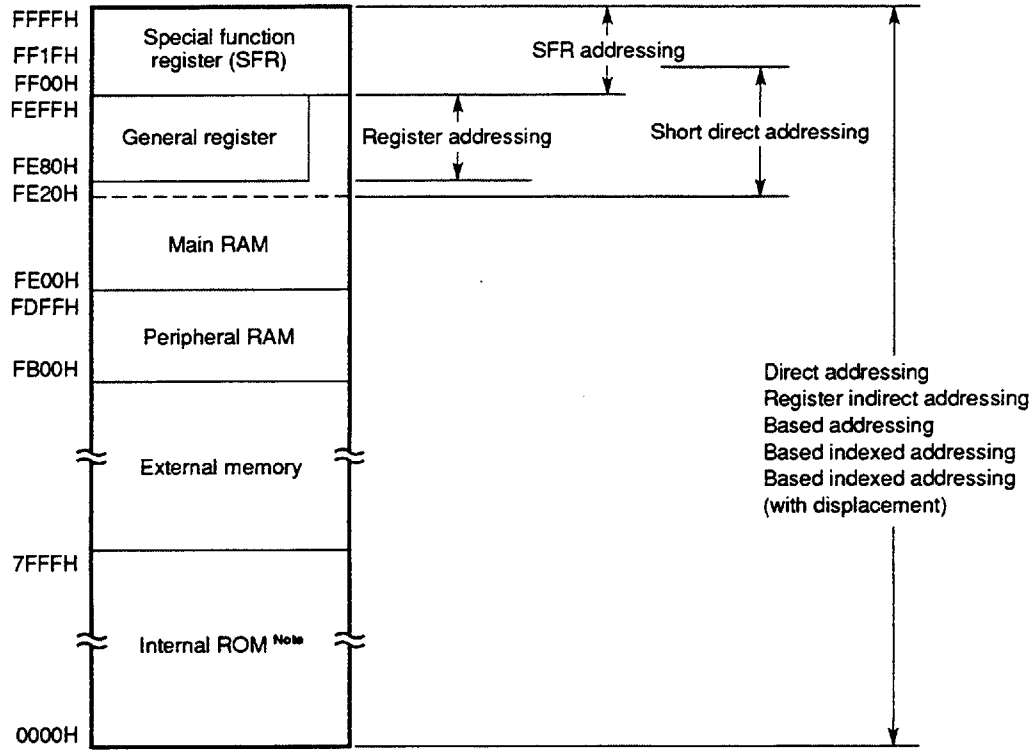
★ **Caution When word access (including the stack operation) to the main RAM space (FE00H to FEFFH) is executed, the addresses specified in the operand must be even numbers.**

2.2 DATA MEMORY ADDRESSING

Various addressing modes are provided for the μPD78334 to improve memory operability or to enable the use of a high-level language. Special addressing is applicable, in particular, to the space of data memory from FB00H to FFFFH according to each function of the special function register (SFR) group and general register group.

Fig. 2-2 shows the addressing space of data memory.

Fig. 2-2 Addressing Space of Data Memory



Note This space is used as external memory when $\overline{EA}$ = L or for the μPD78330.

★ **Caution When word access (including the stack operation) to the main RAM space (FE00H to FEFFH) is executed, the addresses specified in the operand must be even numbers.**

2.3 PROCESSOR REGISTER

The μPD78334 contains three processor register groups.

2.3.1 Control Register

(1) Program counter (PC)

The program counter is a 16-bit register which contains the address of the next instruction to be executed.

(2) Program status word (PSW)

The program status word is a 16-bit register which contains the status of the CPU according to the instruction execution result.

(3) Stack pointer (SP)

The stack pointer is a register which contains the first address of the stack area (LIFO type) in memory.

(4) CPU control word (CCW)

The CPU control word is an 8-bit register which is related to CPU control.

Fig. 2-3 Control Register Configuration

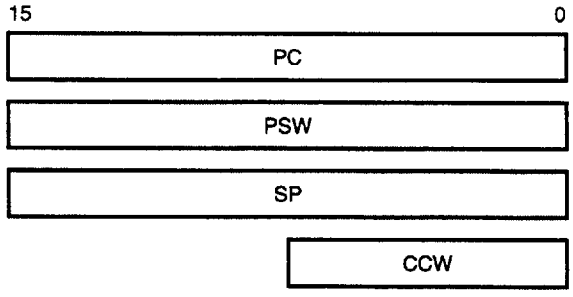
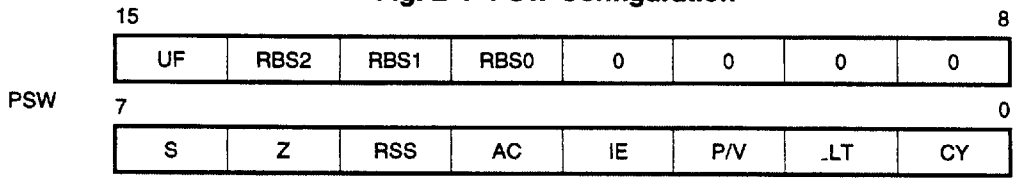
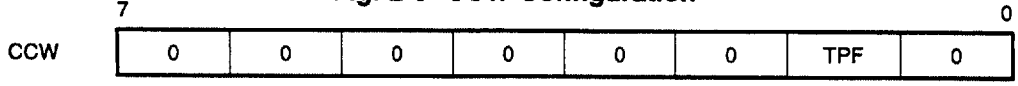


Fig. 2-4 PSW Configuration



- UF : User flag
- RBS0 - RBS2: Register bank selection flag
- S : Sign flag (MSB after arithmetic/logical operation)
- Z : Zero flag
- RSS : Register set selection flag
- AC : Auxiliary carry flag
- IE : Interrupt request enable flag
- P/V : Parity/overflow flag
- LT : Interrupt priority level transition flag
- CY : Carry flag

Fig. 2-5 CCW Configuration



TPF: Table position flag

2.3.2 General Register

The general register group consists of eight banks (one bank: 8 words x 16 bits). Fig. 2-6 shows general register configuration. The general register group is mapped into addresses from FFE0H to FFEFH, and functions as a 16-bit register as well as an 8-bit register (see **Fig. 2-7**). The use of these various registers enables easy control of complicated multitask processing.

Fig. 2-6 General Register Configuration

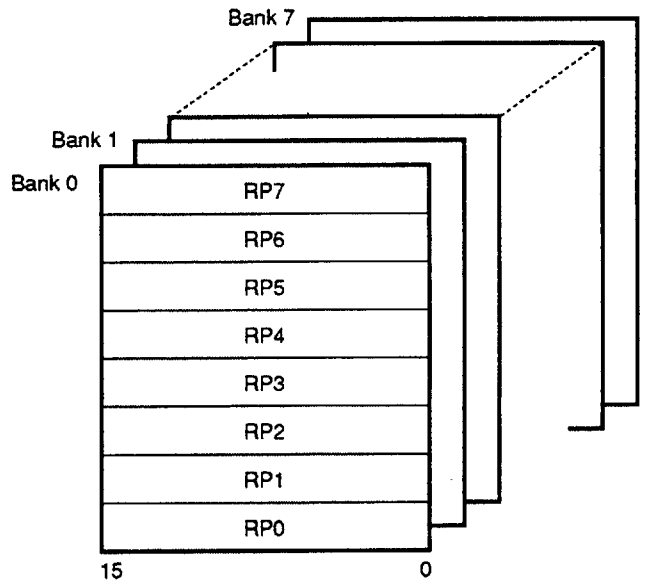
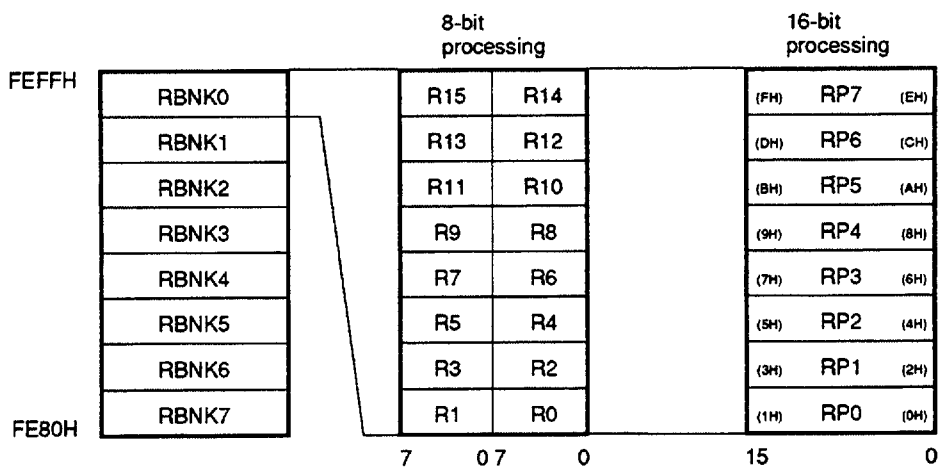


Fig. 2-7 Bit Processing for General Register



2.3.3 Special Function Register (SFR)

The special function register group consists of the registers for control of the peripheral hardware the μPD78334 contains. This register group is mapped into addresses from FF00H to FFFFH. The operation of these registers enables control of ports, an A/D converter, and timer.

Table 2-1 Special Function Registers (1/5)

Address	Special function register (SFR) name	Abbreviation	R/W	Manipulation bit unit			At resetting
				1	8	16	
FF00H	Port 0	P0	R/W	○	○	–	Undefined
FF01H	Port 1	P1		○	○	–	
FF02H	Port 2	P2	R	–	○	–	
FF03H	Port 3	P3	R/W	○	○	–	
FF04H	Port 4	P4		○	○	–	
FF05H	Port 5	P5		○	○	–	
FF07H	Port 7	P7	R	–	○	–	
FF08H	Port 8	P8		–	○	–	
FF09H	Port 9	P9	R/W	○	○	–	
FF0AH	Timer low access register	TLA	R/W	○	○	–	
FF0CH FF0DH	Timer register 2	TM2	R	–	–	○	0000H
FF10H FF11H	Capture register 00	CT00	R	–	–	○	Undefined
FF12H FF13H	Capture register 01	CT01		–	–	○	
FF14H FF15H	Capture register 02	CT02		–	–	○	
FF1CH FF1DH	Capture register 10	CT10		–	–	○	
FF20H	Port 0 mode register	PM0	W	–	○	–	FFH
FF21H	Port 1 mode register	PM1		–	○	–	
FF23H	Port 3 mode register	PM3		–	○	–	
FF25H	Port 5 mode register	PM5		–	○	–	
FF29H	Port 9 mode register	PM9		–	○	–	xx11 1111B
FF2AH FF2BH	Timer register 0	TM0	R	–	–	○	0000H
FF2CH FF2DH	Timer register 1	TM1		–	–	○	
FF2EH FF2FH	Timer register 3	TM3		–	–	○	
FF40H	Port 0 mode control register	PMC0	W	–	○	–	00H
FF41H	Port 1 mode control register	PMC1		–	○	–	
FF43H	Port 3 mode control register	PMC3		–	○	–	

Table 2-1 Special Function Registers (2/5)

Address	Special function register (SFR) name	Abbreviation	R/W	Manipulation bit unit			At resetting
				1	8	16	
FF4CH FF4DH	Baud rate generator compare register	BRG	R/W	–	–	○	0000H
FF60H	Real-time output port register	RTP		○	○	–	Undefined
FF61H	Real-time output port reset register	RTPR		○	○	–	00H
FF62H	Port read control register	PRDC		–	○	–	00H
FF63H	Real-time output port set register	RTPS		○	○	–	00H
FF64H	PWM control register	PWMC		○	○	–	
FF66H	PWM buffer register 0	PWM0		–	○	–	Undefined
FF68H	A/D converter mode register	ADM		○	○	–	00H
FF6EH	PWM buffer register 1	PWM1		–	○	–	Undefined
FF70H FF71H	Compare register 11	CM11		–	–	○	Undefined
FF72H FF73H	Compare register 12	CM12		–	–	○	
FF74H FF75H	Compare register 20	CM20		–	–	○	
FF76H FF77H	Compare register 21	CM21		–	–	○	
FF78H FF79H	Compare register 30	CM30		–	–	○	
FF80H	Clock synchronous serial interface mode register	CSIM		○	○	–	00H
FF82H	Serial bus interface control register	SBIC		○	○	–	
FF86H	Serial I/O shift register	SIO		○	○	–	Undefined
FF88H	Asynchronous serial interface mode register	ASIM		○	○	–	80H
FF8AH	Asynchronous serial interface status register	ASIS	R	○	○	–	00H
FF8CH	Serial reception register (UART)	RXB	R	–	○	–	Undefined
FF8EH	Serial transmission shift register (UART)	TXS	W	–	○	–	

Table 2-1 Special Function Registers (3/5)

Address	Special function register (SFR) name	Abbreviation	R/W	Manipulation bit unit			At resetting
				1	8	16	
FF90H FF91H	Compare register X0	CMX0	R/W	–	–	○	Undefined
FF92H FF93H	Compare register 01	CM01R		–	–	○	
FF94H FF95H	Compare register 02	CM02R		–	–	○	
FF96H FF97H	Compare register 03	CM03R		–	–	○	
FF98H FF99H	Compare register 04	CM04R		–	–	○	
FF9AH FF9BH	Capture/compare register 00	CC00R		–	–	○	
FF9CH FF9DH	Capture/compare register 01	CC01R		–	–	○	
FFA0H	A/D conversion result register 0 (for 16-bit access)	ADCR0	R	–	–	○	
FFA1H	A/D conversion result register 0 (for high-order 8-bit access)	ADCR0H		–	○	–	
FFA2H	A/D conversion result register 1 (for 16-bit access)	ADCR1		–	–	○	
FFA3H	A/D conversion result register 1 (for high-order 8-bit access)	ADCR1H		–	○	–	
FFA4H	A/D conversion result register 2 (for 16-bit access)	ADCR2		–	–	○	
FFA5H	A/D conversion result register 2 (for high-order 8-bit access)	ADCR2H		–	○	–	
FFA6H	A/D conversion result register 3 (for 16-bit access)	ADCR3		–	–	○	
FFA7H	A/D conversion result register 3 (for high-order 8-bit access)	ADCR3H		–	○	–	
FFA8H	A/D conversion result register 4 (for 16-bit access)	ADCR4		–	–	○	
FFA9H	A/D conversion result register 4 (for high-order 8-bit access)	ADCR4H		–	○	–	
FFAAH	A/D conversion result register 5 (for 16-bit access)	ADCR5		–	–	○	
FFABH	A/D conversion result register 5 (for high-order 8-bit access)	ADCR5H		–	○	–	

Table 2-1 Special Function Registers (4/5)

Address	Special function register (SFR) name	Abbreviation	R/W	Manipulation bit unit			At resetting
				1	8	16	
FFACH	A/D conversion result register 6 (for 16-bit access)	ADCR6	R	–	–	○	Undefined
FFADH	A/D conversion result register 6 (for high-order 8-bit access)	ADCR6H		–	○	–	
FFAEH	A/D conversion result register 7 (for 16-bit access)	ADCR7		–	–	○	
FFAFH	A/D conversion result register 7 (for high-order 8-bit access)	ADCR7H		–	○	–	
FFB0H	Timer control register 0	TMC0	R/W	○	○	–	00H
FFB1H	Baud rate generator mode register	BRGM		○	○	–	
FFB2H	Timer control register 1	TMC1		○	○	–	40H
FFB4H	Timer unit mode register 0	TUM0		○	○	–	00H
FFB5H	Timer unit mode register 1	TUM1		○	○	–	
FFB8H	Timer output control register 0	TOC0		○	○	–	
FFB9H	Timer output control register 1	TOC1		○	○	–	
FFBCH	Programmable pulse output set register	PPOS		○	○	–	
FFC0H	Standby control register	STBC		○	○	–	0000 ×000B
FFC1H	CPU control word	CCW		○	○	–	00H
FFC2H	Watchdog timer mode register	WDM		○	○	–	
FFC4H	Memory expansion mode register	MM		○	○	–	
FFC6H	Programmable wait control register	PWC		○	○	–	22H
FFC9H	Fetch cycle control register	FCC		○	○	–	00H
FFD0H | FFDFH	External SFR area	———		○	○	○	Undefined

Table 2-1 Special Function Registers (5/5)

Address	Special function register (SFR) name	Abbreviation		R/W	Manipulation bit unit			At resetting
					1	8	16	
FFE0H	Interrupt request flag register 0L	IF0L	IF0	R/W	○	○	○	00H
FFE1H	Interrupt request flag register 0H	IF0H			○	○		
FFE2H	Interrupt request flag register 1L	IF1L	IF1		○	○	–	
FFE4H	Interrupt mask flag register 0L	MK0L	MK0		○	○	○	FFH
FFE5H	Interrupt mask flag register 0H	MK0H			○	○		
FFE6H	Interrupt mask flag register 1L	MK1L	MK1		○	○	–	xxx x111B
FFE8H	Priority specification buffer register 0L	PB0L	PB0		○	○	○	00H
FFE9H	Priority specification buffer register 0H	PB0H			○	○		
FFEAH	Priority specification buffer register 1L	PB1L	PB1		○	○	–	
FFECH	Interrupt processing mode specification register 0L	ISM0L	ISM0		○	○	○	
FFEDH	Interrupt processing mode specification register 0H	ISM0H			○	○		
FFEEH	Interrupt processing mode specification register 1L	ISM1L	ISM1		○	○	–	
FFF0H	Context switching enable register 0L	CSE0L	CSE0		○	○	○	
FFF1H	Context switching enable register 0H	CSE0H			○	○		
FFF2H	Context switching enable register 1L	CSE1L	CSE1		○	○	–	
FFF4H	External interrupt mode register 0	INTM0			○	○	–	
FFF5H	External interrupt mode register 1	INTM1			○	○	–	
FFF8H	In-service priority register	ISPR		R	–	○	–	
FFF9H	Priority specification register	PRSL		R/W	○	○	–	

3. BLOCK FUNCTION

3.1 BUS CONTROL UNIT (BCU)

The bus control unit (BCU) activates a required bus cycle according to the physical address obtained from the execution unit (EXU). When the EXU does not issue a bus cycle activation request, the BCU generates an address required for prefetching an instruction. The prefetched instruction code is fetched into the instruction queue.

3.2 EXECUTION UNIT (EXU)

The execution unit (EXU) controls address calculation, arithmetic/logical operations, and data transfer by a microprogram. The EXU contains 256-byte main RAM.

The 256-byte main RAM in the EXU can be accessed at higher speed with an instruction than the 768-byte peripheral RAM.

3.3 ROM/RAM

This area consists of a 32K-byte ROM area and 768-byte peripheral RAM area. The μPD78330 does not contain ROM, however.

Access to this ROM area can be inhibited by using the $\overline{EA}$ pin.

3.4 INTERRUPT CONTROLLER

The interrupt controller processes various interrupt requests (NMI, INTP0 to INTP6) issued from peripheral hardware and external device with the context switching, vectored interrupt, or macro service function.

The interrupt controller also specifies the 3-level interrupt priority.

3.5 REAL-TIME PULSE UNIT (RPU)

The real-time pulse unit (RPU) consists of the following hardware.

- 18/16-bit timer/counter: 1 channel
  - 18/16-bit compare registers : 5
  - 18/16-bit capture registers : 3
  - 18/16-bit capture/compare registers : 2
  - Pulse outputs : 6
- 16-bit timers/counters: 3 channels
  - 16-bit compare registers : 5
  - 16-bit capture register : 1
  - Timer outputs : 5

The RPU can measure a pulse width and frequency and output a programmable pulse. It also can control the set/reset timing of the port output of the real-time output port (P0).

3.6 SERIAL INTERFACE

There are two independent serial interfaces: the asynchronous serial interface (UART) and clock synchronous serial interface.

The asynchronous serial interface transfers data using the two pins, TxD and RxD.

The clock synchronous serial interface operates in the following two modes.

- Serial bus interface (SBI) mode
  Transfers data with two pins, the serial clock pin and serial data bus pin.
- Three-wire serial I/O mode
  Transfers data with three pins, the serial clock pin, serial input pin, and serial output pin.

3.7 A/D CONVERTER

The A/D converter uses the successive approximation system. This converter is a 10-bit high-speed high-accuracy converter having 16 analog input pins.

3.8 REAL-TIME OUTPUT PORT (RTP)

The real-time output port (RTP) can control the output timing of the port by the signal from the RPU as a trigger pulse. The port can be set or reset bit by bit. This port is also used as port 0 and can output 8 real-time pulses.

3.9 WATCHDOG TIMER (WDT)

The 8-bit watchdog timer is built into the CPU to detect a program crash and system error.

This microcomputer has the $\overline{\text{WDTO}}$ pin to notify the external device that a watchdog timer interrupt occurs.

3.10 PORT

The following ports having the general port function and control pin function are provided.

Table 3-1 Pin Function

Port	I/O	Function	
P0	8-bit I/O	General port	Real-time output port
P1	8-bit I/O		RPU programmable pulse output
P2	8-bit input		External interrupt, RPU capture trigger
P3	8-bit I/O		Serial interface, RPU timer output
P4	8-bit I/O		Address/data bus
P5	8-bit I/O		Address bus
P7	8-bit input		A/D converter input
P8	8-bit input		
P9	6-bit I/O		External access control signal output, PWM signal output

Remark RPU: Real-time pulse unit

4. PERIPHERAL HARDWARE FUNCTIONS

4.1 PORT FUNCTIONS

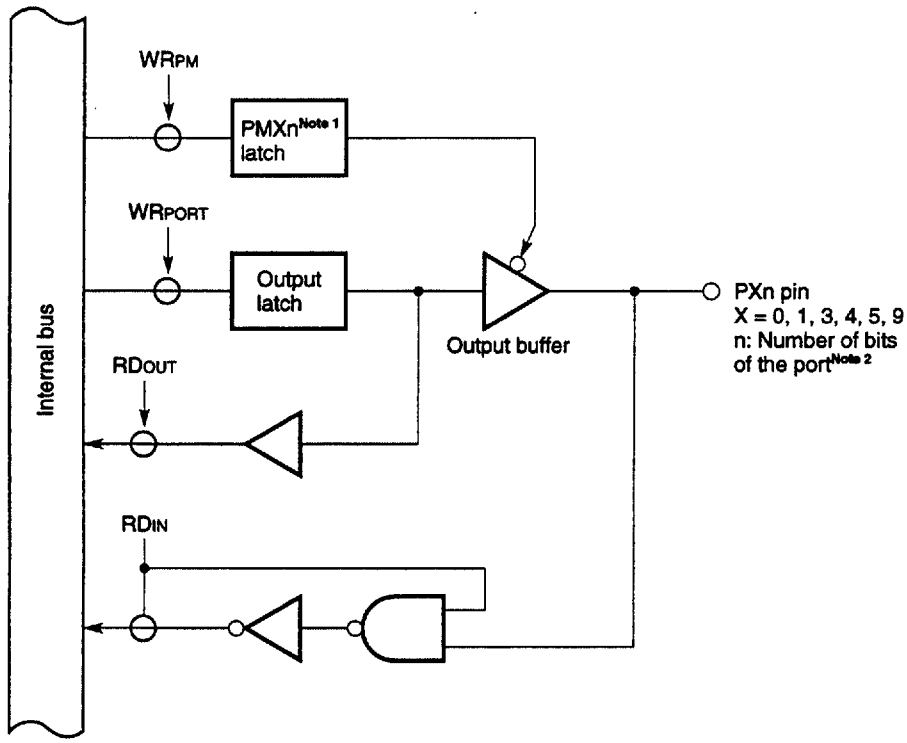
4.1.1 Hardware Configuration

As shown in Fig. 4-1, three-state bidirectional ports are basically used for the ports of the μPD78334.

A $\overline{\text{RESET}}$ input signal sets each bit of a port mode register to 1, specifying the port as an input port. All port pins go into the high-impedance state. A $\overline{\text{RESET}}$ input signal makes the contents of the output latch undefined.

Fig. 4-2 shows the port configuration.

Fig. 4-1 Basic I/O Port Configuration



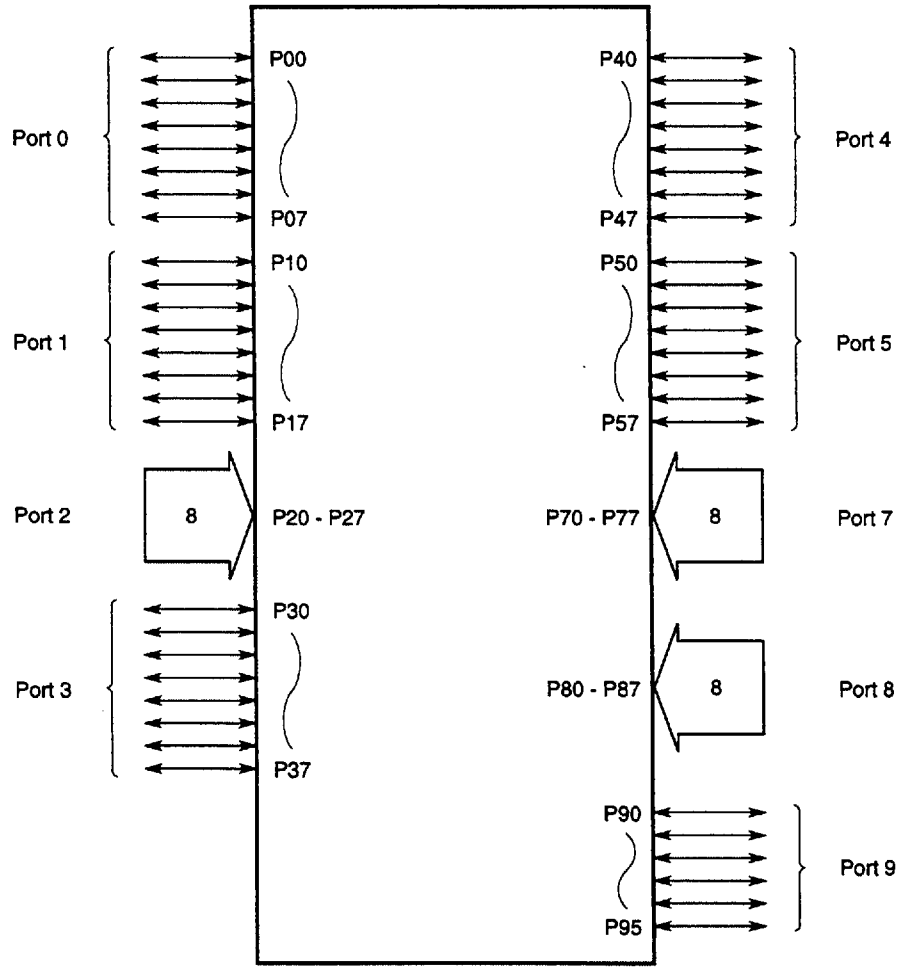
Notes 1. PMXn latch: Bit n of port mode register PMX (X = 0, 1, 3, 4, 5, 9)

2. When X = 0, 1, 3, 4, 5, n = 0 to 7.
   When X = 9, n = 0 to 5.

Remarks 1. Ports 2, 7, and 8 are used only for input.

2. Port 4 can be specified as an input port or output port by using the memory expansion mode register (MM).

Fig. 4-2 Port Configuration



4.1.2 Functions of the Digital I/O Ports

Table 4-1 lists the ports of the μPD78334.

Each port allows bit manipulations as well as 8-bit data manipulations, thus enabling a wide variety of control. Each port functions as a digital port and also functions as I/O pins for internal hardware.

Table 4-1 Port Functions and Additional Functions of the Ports

Port name	Port function	Additional function
Port 0	8-bit I/O port. Specifiable as input or output bit by bit.	Real-time output port (RTP) in control mode
Port 1	8-bit I/O port. Specifiable as input or output bit by bit.	Output for real-time pulse unit (RPU) output in control mode
Port 2	Port used only for 8-bit input	Capture trigger input and count pulse input for real-time pulse unit (RPU), and external interrupt input in control mode
Port 3	8-bit I/O port. Specifiable as input or output bit by bit.	Output for real-time pulse unit (RPU), and I/O for serial interface (UART, CSI) in control mode
Port 4	8-bit I/O port. Specifiable as input or output bit by bit.	Address/data bus for memory expansion (AD0 - AD7)
Port 5	8-bit I/O port. Specifiable for input or output bit by bit.	Address bus for memory expansion (A8 - A15)
Port 7	Port used only for 8-bit input	Analog input for A/D converter in control mode
Port 8	Port used only for 8-bit input	Analog input for A/D converter in control mode
Port 9	6-bit I/O port. Specifiable as input or output bit by bit.	Control signal output for memory expansion, PWM signal output in control mode

4.1.3 Port Output Check Function

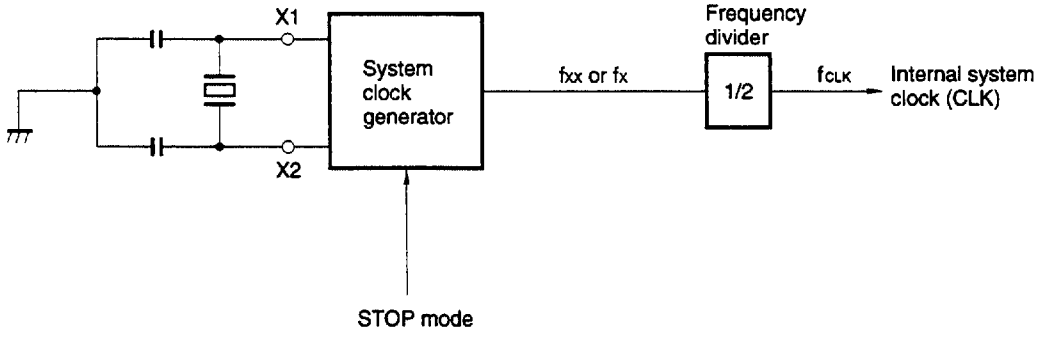
The μPD78334 has a function of reading pin state (pin access mode) to improve system application reliability in port output mode. With this function, output data (output latch data) and actual pin state can be checked as required. For frequent port state checking, special instructions (CHKL, CHKLA) are available.

4.2 CLOCK GENERATOR

The clock generator generates and controls an internal system clock (CLK) supplied to the CPU.

The clock generator is configured as shown in Fig. 4-3.

Fig. 4-3 Block Diagram of the Clock Generator



Remarks 1. fxx: Crystal oscillator frequency
2. fx: External clock frequency
3. fCLK: Internal system clock frequency

The system clock generator generates a clock signal with a crystal resonator connected to the X1 and X2 pins. The system clock generator stops oscillation when the standby mode (STOP) is set.

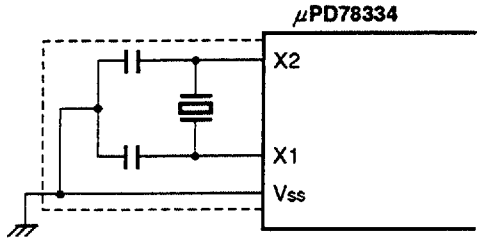
An external clock can be applied. In this case, a clock signal is to be applied to the X1 pin, with the inverted signal to be applied to the X2 pin. The X2 pin can be left open.

Caution When using an external clock, do not set the STOP mode.

The frequency divider divides system clock generator output (fxx for the crystal oscillator or fx for an external clock) by two to produce an internal system clock (fCLK).

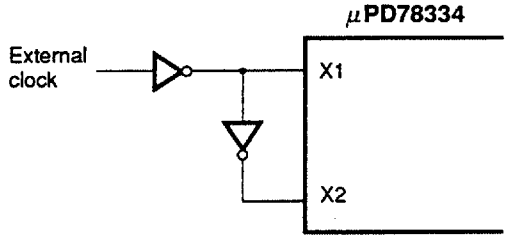
Fig. 4-4 External Circuitry of the System Clock Generator

(a) Crystal oscillator

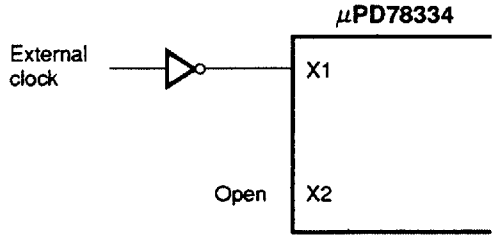


(b) External clock

(i) When an external clock signal input to the X1 pin is inverted and input to the X2 pin



(ii) When the X2 pin is open



Cautions 1. When using the system clock generator, run wires enclosed in broken lines ( ) in Fig. 4-4 according to the following rules to avoid effects such as stray capacitance:

- Minimize the wiring.
- Never cause the wires to cross other signal lines or run near a line carrying a large varying current.
- Cause the grounding point of the capacitor of the oscillator circuit to have the same potential as Vss. Never connect the capacitor to a ground pattern carrying a large current.
- Never extract a signal from the oscillator.

2. Keep loads such as stray capacitance around wiring away form the X2 pin, when an external clock signal is input to the X1 pin and the X2 pin is left open.

4.3 REAL-TIME PULSE UNIT (RPU)

The RPU can output programmable pulses and can also measure pulse intervals and frequencies.

The RPU mainly consists of four timers and 16 registers. The user can select a function and configuration of each register by programming so that the user can match a wide variety of applications in a flexible manner.

The major feature of the RPU is its multifunction timer pulse output capability. Each timer allows six types of pulse output such as toggle output and set/reset output to be controlled independently. In addition, the output timing of a real-time output port can be controlled.

4.3.1 RPU Configuration

The RPU consists of the hardware components listed in Table 4-2. Fig. 4-5 shows the configuration of the RPU.

Table 4-2 Components of the Real-time Pulse Unit (RPU)

Timer	Count clock	Register	Compare register match interrupt	Capture trigger	Timer output
18-bit timer (TM0)	$f_{CLK}/4$, $f_{CLK}/8$	18/16-bit compare register (CMX0)[Note 1]	INTCMX0	–	6
		18/16-bit compare register (CM01R)	–	–	
		18/16-bit compare register (CM02R)	–	–	
		18/16-bit compare register (CM03R)	–	–	
		18/16-bit compare register (CM04R)	–	–	
		18/16-bit capture/compare register (CC00R)	INTCC00R	INTP3	Used also for port 1
		18/16-bit capture/compare register (CC01R)	INTCC01R	INTP4	
		18/16-bit capture register (CT00)	–	INTCMX0/INTP1	
		18/16-bit capture register (CT01)	–	INTP0	
		18/16-bit capture register (CT02)	–	INTP2	
16-bit timer (TM1)	TI pin input or $f_{CLK}/8$	16-bit compare register (CM11)	INTCM11[Note 2]	–	2
		16-bit compare register (CM12)	INTCM12[Note 2]	–	
		16-bit capture register (CT10)	–	INTP0/INTP5	
16-bit timer (TM2)	$f_{CLK}/16$	16-bit compare register (CM20)	INTCM20	–	2
		16-bit compare register (CM21)	INTCM21	–	
16-bit timer (TM3)	$f_{CLK}/16$	16-bit compare register (CM30)	INTCM30	–	1

Notes 1. Can be used as a timer 1 (TM1) compare register by software setting.

2. Used also as an output trigger signal for the real-time output port (RTP).

Remarks 1. f_{CLK}: Internal system clock

2. INTPn (n = 0 to 6): External interrupt

3. Timer 0 has an overflow interrupt function.

4. Pulse output is set to 1 by a timer 0 overflow or INTCMX0 signal.

5. Timer 1 can be cleared by $\overline{\text{INTP0}}$/INTCM11.

6. Timer 2 can be cleared by INTCM20.

7. Timer 3 can be cleared by INTCM30.

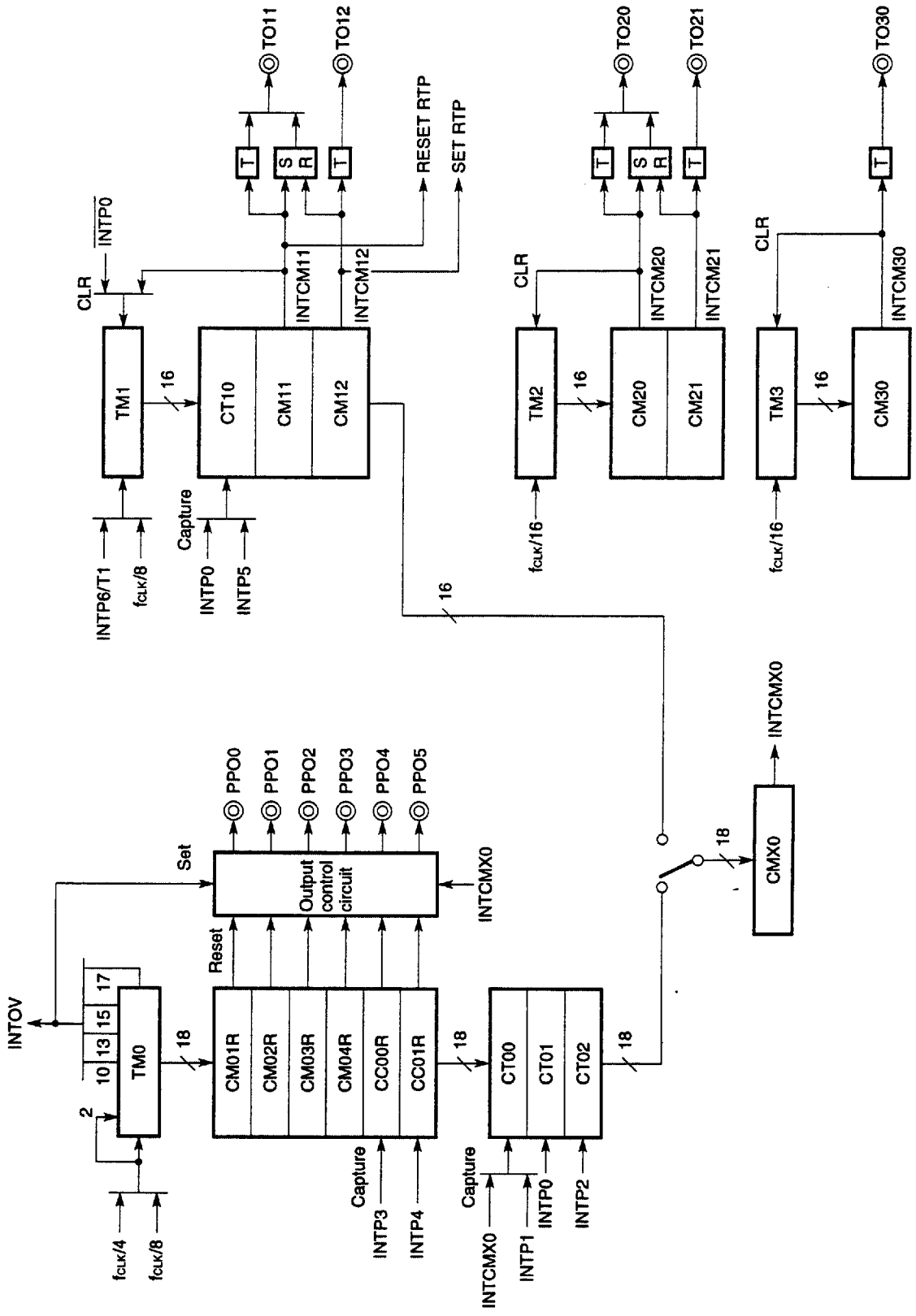


Fig. 4-5 Configuration of the Real-time Pulse Unit

4.3.2 Function

(1) Timer 0 (TM0)

Timer 0 is an 18-bit or 16-bit free-running timer.

Timer 0 counts an internal clock, and generates an overflow interrupt (INTOV) when a timer overflow occurs. Timer overflow bits can be selected using a control register.

TM0 and the timer low access register (TLA)[Note] (special function registers (SFRs)) are used to read the contents of timer 0 used as a 18-bit timer. The instruction for reading TM0 is first executed, then the instruction for reading the TLA register is executed. The contents of the high-order 16 bits in the 18-bit timer can be read with TM0, and the contents of the low-order two bits can be read with the TLA register.

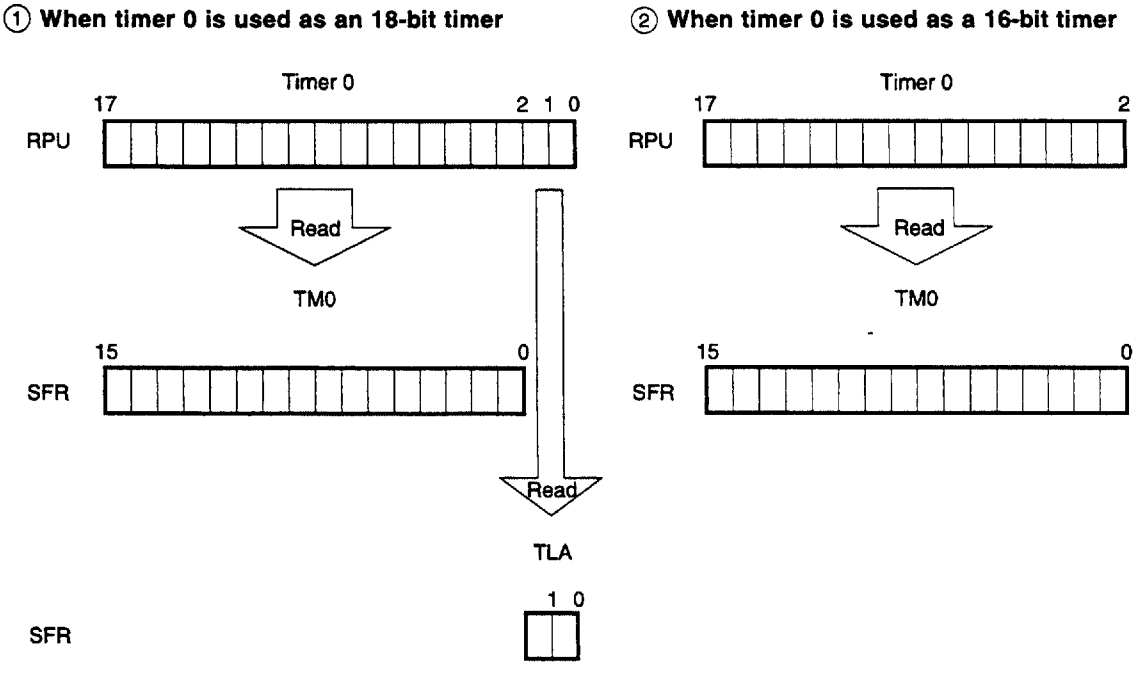
When timer 0 is used as a 16-bit timer, the high-order 16 bits of timer 0 used as an 18-bit timer are used. The contents of timer 0 are read with the TM0 SFR only.

The input of the $\overline{\text{RESET}}$ signal clears all the bits of TM0 to 0.

Note TLA register: SFR used to access the low-order two bits in an 18-bit timer, compare register, capture register, or compare/capture register

Caution To read the contents of timer 0 used as an 18-bit timer, be sure to execute the instructions for reading TM0 and the TLA register continuously. If the instruction for reading the TLA register is not executed, the contents of TM0 cannot be read correctly.

Fig. 4-6 Configuration of Timer 0



(2) Timer 1 (TM1)

Timer 1 is a 16-bit timer/event counter. Timer 1 can count an internal clock or external event applied to the TI pin. When a timer overflow occurs as the result of counting, the overflow bit (OVF1) is set to 1.

Timer 1 can be cleared by external interrupt input (inverted INTP0 edge input) or by a match interrupt (INTCM11) from the compare register CM11.

(3) Timer 2 (TM2)

Timer 2 is a 16-bit free-running timer. Timer 2 can count an internal clock. When a timer overflow occurs, the overflow bit (OVF2) is set to 1.

Timer 2 can be cleared by a match interrupt (INTCM20) from the compare register CM20. With this feature, it can also be used as an interval timer.

(4) Timer 3 (TM3)

Timer 3 is a 16-bit interval timer. Timer 3 counts an internal clock. Timer 3 is cleared when a match interrupt (INTCM30) from the compare register CM30 occurs.

(5) Compare registers

A compare register compares the contents of each timer with the data held in the compare register at all times, and generates a match signal when a match is found. See Table 4-2 for detailed information about the configuration of the timers and compare registers, and the correspondence between the compare registers and interrupt sources.

(a) 18/16-bit compare registers (CMX0[Note 1], CM0nR: n = 1 to 4)

An 18/16-bit compare register is connected to timer 0 (TM0), and has a timer output function. It sets a timer output to 1 when a match is detected by the CMX0 register[Note 2], or resets a timer output to 0 when a match is detected by the CM0nR register.

Notes 1. The CMX0 register can also be connected to timer 1 (TM1) by software setting.

2. Three of the six timer outputs can also be set to 1 by software setting. This is done according to the timer 0 (TM0) overflow signal.

(b) 16-bit compare registers (CM11, CM12, CM20, CM21, CM30)

The 16-bit compare registers are connected to timer 1 to timer 3 as indicated in Table 4-2, and have timer output functions. Each compare register inverts the corresponding timer output when a match interrupt request is found (toggle output operation). It can be also used to set or reset an output using CM11 and CM12, or CM20 and CM21 (set/reset output operation).

(6) Capture/compare registers (CC00R, CC01R)

(a) Function

The capture/compare registers are 18/16-bit registers connected to timer 0 (TM0). A capture/compare register can function either as a capture register or compare register. This selection can be made using the respective control register.

A bit length (18 bits or 16 bits) is automatically set to match the setting of timer 0.

When a capture/compare register functions as a capture register, an external interrupt (INTP3, INTP4) can be used as a capture trigger. When a capture/compare register functions as a compare register, it resets timer output to 0 in phase with the generation of an interrupt based on a match between the timer and register values.

The interrupt source depends on whether the capture register function or compare register function is selected. When the capture register function is selected, the corresponding external interrupt capture trigger is selected. When the compare register function is selected, the corresponding match interrupt is selected (**Table 4-2**).

Caution When the compare register function is selected, interrupt generation based on the corresponding external interrupt (capture trigger signal) is impossible.

(b) Operation

(i) Data read/write operation

When the contents of an 18-bit capture/compare register are read or data is written in the 18-bit capture compare register, the 16-bit capture/compare register (CC00R or CC01R) and TLA register (special function registers (SFRs)) are used.

① **To read the contents of the 18-bit capture/compare register**

The instruction for reading the CC00R (or CC01R) register is first executed, then the instruction for reading the TLA register is executed. The contents of the high-order 16 bits in the 18-bit capture/compare register can be read with the CC00R (or CC01R) register, and the contents of the low-order two bits can be read with the TLA register.

② **To write data in the 18-bit capture/compare register**

The data corresponding to the contents of the low-order two bits in the 18-bit capture/compare register is first written in the TLA register. The data corresponding to the contents of the high-order 16 bits in the 18-bit capture/compare register is then written in the CC00R (or CC01R) register.

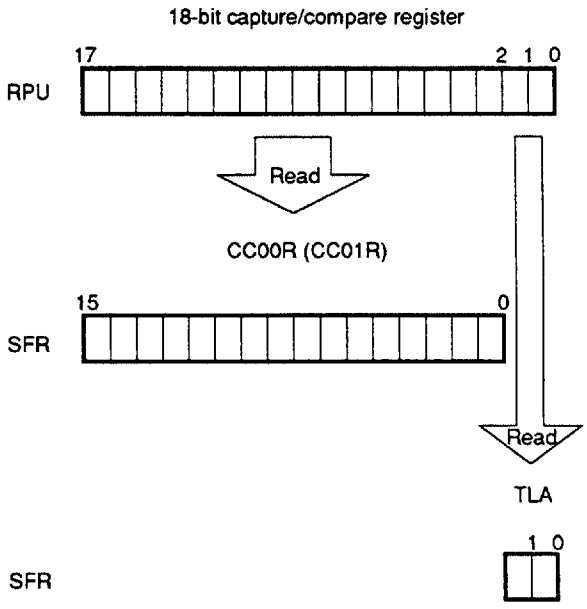
When a capture/compare register is used as a 16-bit capture/compare register, the high-order 16 bits (CC00R or CC01R) are used.

Cautions 1. To read the contents of the 18-bit capture/compare register, be sure to execute the instructions for reading the CC00R (or CC01R) register and TLA register continuously. If the instruction for reading the TLA register is not executed, the contents of the 18-bit capture/compare register cannot be read correctly.

2. To write data in the 18-bit capture/compare register, be sure to execute the instructions for writing to the TLA register and the CC00R (or CC01R) register continuously. If the instruction for writing to the CC00R (or CC01R) register is not executed, data cannot be written in the 18-bit capture/compare register correctly.

Fig. 4-7 Configuration of a Capture/Compare Register (for Reading Data)

① When the capture/compare register is used as an 18-bit register



② When the capture/compare register is used as a 16-bit register

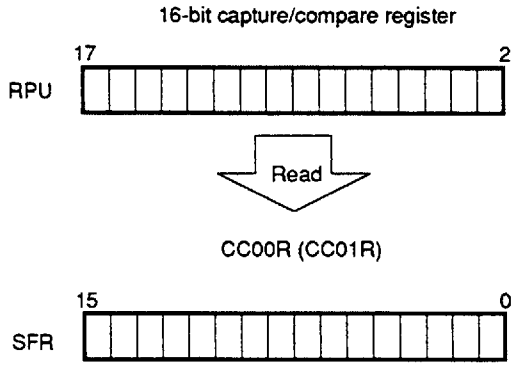
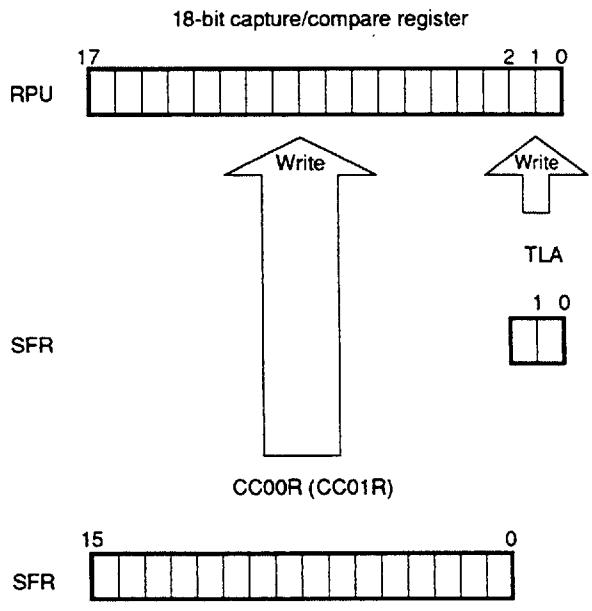
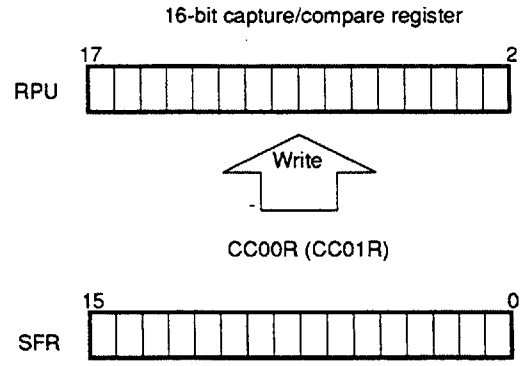


Fig. 4-8 Configuration of a Capture/Compare Register (for Writing Data)

① When the capture/compare register is used as an 18-bit register



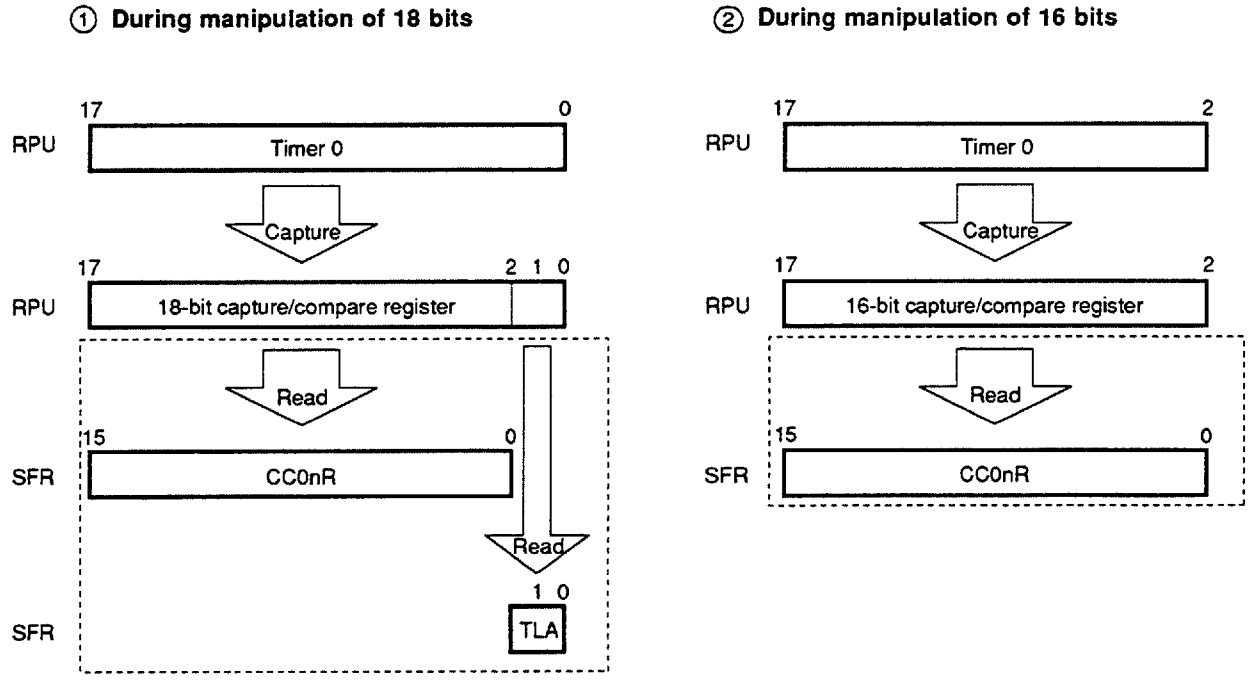
② When the capture/compare register is used as a 16-bit register



(ii) Capture operation

When an external capture trigger signal is applied, the contents of timer 0 are captured. When timer 0 is used as an 18-bit timer, the contents of the TLA register are used as the low-order two bits. When timer 0 is used as a 16-bit timer, the high-order 16-bit data (CC0nR: n = 0, 1) is used, with the low-order 2-bit data being undefined.

Fig. 4-9 Capture Operation



Remarks 1. The operation of the SFRs are enclosed by the square ( ).

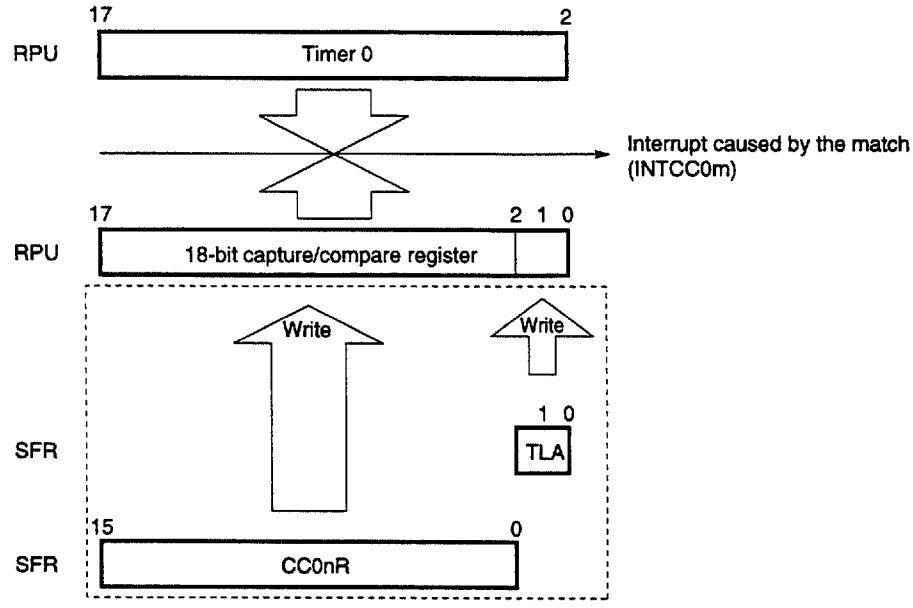
2. n = 0, 1

(iii) Compare operation

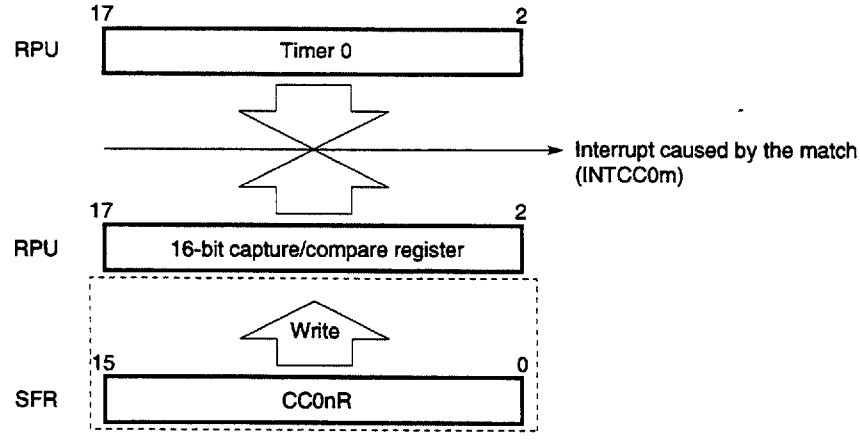
A comparison is made with the contents of timer 0 at all times. When a match is found, an interrupt signal is generated.

Fig. 4-10 Compare Operation

① **During manipulation of 18 bits**



② **During manipulation of 16 bits**



Remarks 1. The operations of the SFRs are enclosed by the square (⋯⋯).

2. n = 0, 1

3. m = 0 to 3

(7) Capture registers

A capture register takes in (captures) the contents of each timer when a capture trigger signal occurs. As a capture trigger, an external interrupt (INTP0 to INTP5) or match detection interrupt (INTCMX0) from the compare register CMX0 can be used. See Table 4-2 for the correspondence between the registers and capture triggers. The occurrence of a capture trigger also means the occurrence of an interrupt. By using a capture register, the pulse width and period of an externally applied pulses can be easily measured.

(a) 18/16-bit capture registers (CT00 to CT02)

When a capture trigger occurs, an 18/16-bit capture register captures the contents of timer 0 (TM0).

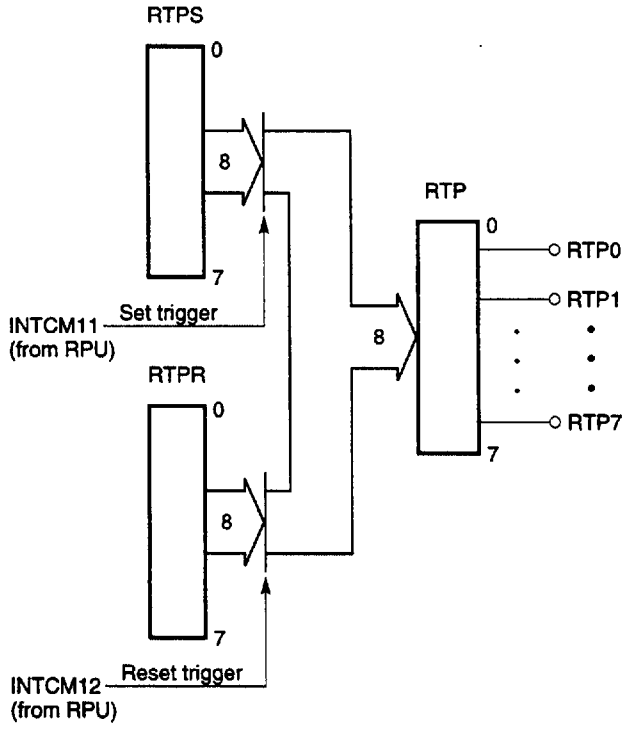
(b) 16-bit capture register (CT10)

When a capture trigger occurs, the 16-bit capture register captures the contents of timer 1.

4.4 REAL-TIME OUTPUT PORT (RTP)

The real-time output port is an 8-bit port whose output can be set or reset on a bit-by-bit basis in phase with a trigger signal from the real-time pulse unit (RPU). This port allows multi-channel synchronous pulse output to be controlled easily. Fig. 4-11 outlines the real-time output port.

Fig. 4-11 Outline of the Real-time Output Port



4.4.1 RTP Configuration

The real-time output port consists of the following hardware components:

- Real-time output port register (RTP)
- Real-time output port set register (RTPS)
- Real-time output port reset register (RTPR)

The pins of the real-time output port also function as the port 0 pins.

4.4.2 RTP Operation

The real-time output port (RTP) is set/reset by a trigger signal from the real-time pulse unit (RPU). The real-time output port set/reset register (RTPS/RTPR) register is used to specify the bit(s) to be set/reset. When 1 is set in RTPS, the real-time output latch (RTP) is automatically set by hardware on a set trigger (INTCM12) output from the RPU.

Similarly, when 1 is set in RTPR, the real-time output latch (RTP) is automatically reset by hardware on a reset trigger (INTCM11) output from the RPU.

In addition, the contents of the RTP can be directly rewritten by software. In this case, if a set/reset trigger signal occurs, the rewriting of the RTP by software has higher priority.

Caution If a set trigger and reset trigger occur simultaneously, reset operation has higher priority.

4.5 A/D CONVERTER

The μPD78334 contains a fast, high-precision 10-bit analog/digital (A/D) converter. The A/D converter has 16 analog inputs (ANI0 to ANI15) and can operate in various operation modes such as the select mode, scan mode, and mix mode to match many different types of applications.

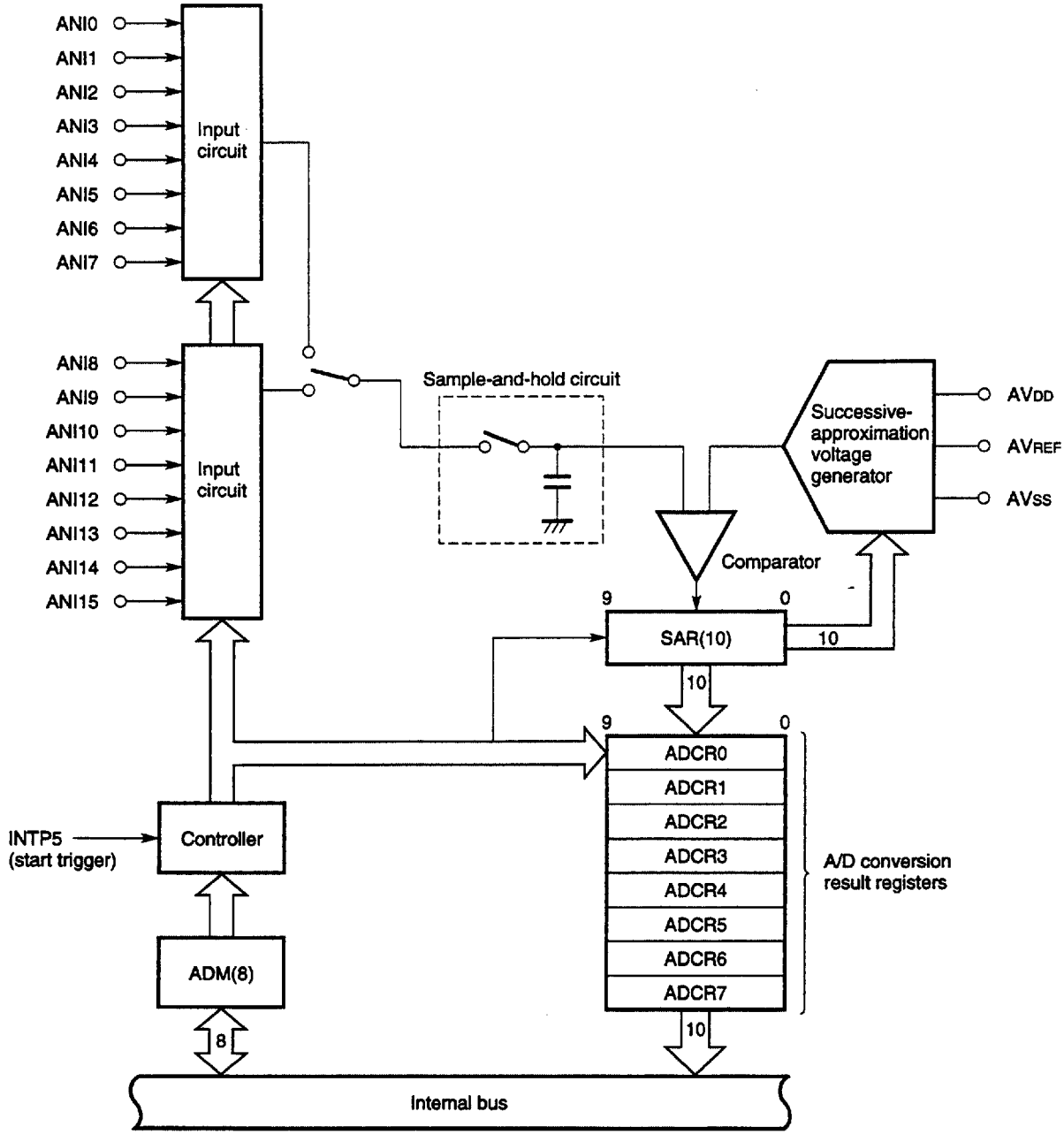
An internal interrupt (INTAD) is generated upon completion of A/D conversion. This interrupt can start a macro service that performs operations such as automatic data transfer by hardware.

4.5.1 Configuration

Fig. 4-12 shows the configuration of the A/D converter of the μPD78334.

Software selection is used to switch between the higher eight channels (ANI8 to ANI15) and lower eight channels (ANI0 to ANI7) of the analog inputs for the A/D converter.

Fig. 4-12 Configuration of the A/D Converter



4.5.2 Operation

The higher eight channels (ANI8 to ANI15) and lower eight channels (ANI0 to ANI7) of the analog inputs for the A/D converter have exactly the same function. So unless otherwise specified, the explanation of the lower eight channels (ANI0 to ANI7) below also applies to the higher eight channels.

Three modes are available for selection to operate the A/D converter. An operation mode can be selected with the A/D converter mode register (ADM). A/D conversion is started by writing data to the ADM. This means that synchronization of A/D conversion by software is possible.

(1) Select mode

The select mode performs an A/D conversion of one analog input (ANIn: n = 0 to 7) selected with the ADM register. The result of conversion is loaded into the A/D conversion result register (ADCRn: n = 0 to 7) corresponding to the analog input.

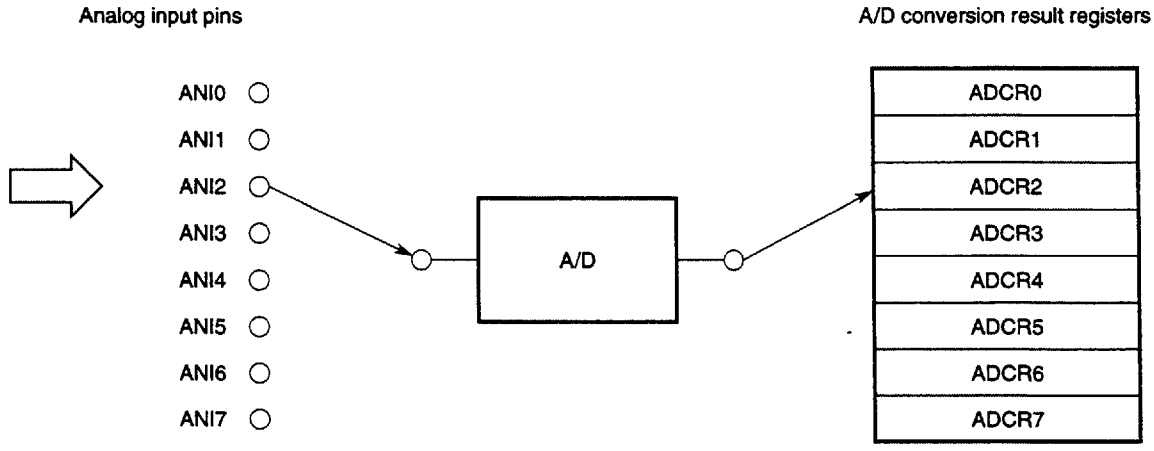
The select mode has two modes in which the result of A/D conversion is loaded into registers.

(a) One-buffer mode

An A/D conversion is performed once for one analog input, then the result is loaded into one A/D conversion result register. In this case, an analog input corresponds to an A/D conversion result register on a one-to-one basis.

The completion of A/D conversion can be detected using the interrupt INTAD, which is generated each time an A/D conversion operation is performed. This mode repeats A/D conversion operation.

Fig. 4-13 A/D Conversion Operation in the Select Mode (One-Buffer Mode)



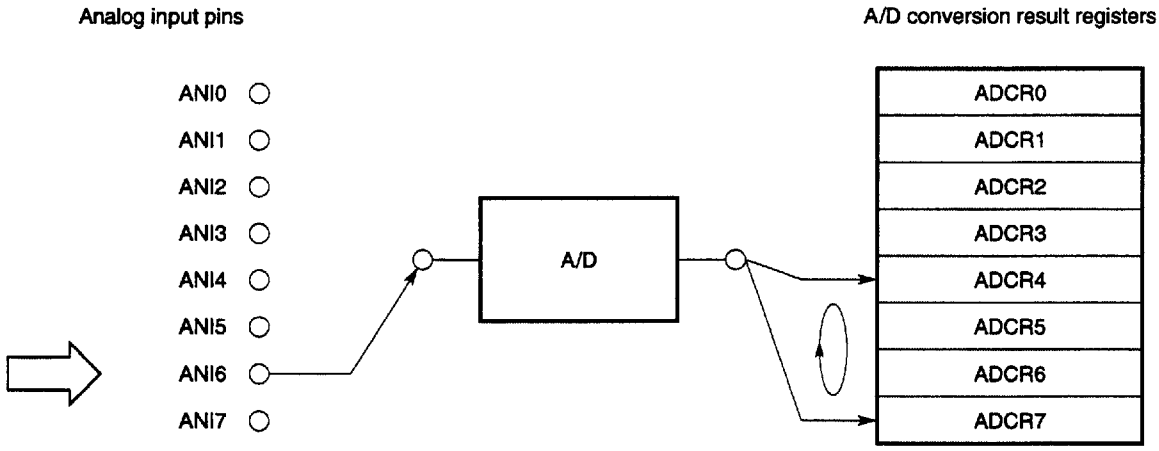
(b) Four-buffer mode

For one analog input, four A/D conversion operations are performed, then the results are loaded into four A/D conversion result registers. When one of the analog inputs ANI0 to ANI3 is selected, the results of A/D conversions are loaded into the A/D conversion result registers ADCR0 to ADCR3. Similarly, when one of the analog inputs ANI4 to ANI7 is selected, the results of A/D conversions are loaded into the A/D conversion result registers ADCR4 to ADCR7.

INTAD is generated upon completion of four A/D conversion operations. This mode repeats A/D conversion operation four times.

When the average of A/D conversions is to be found, this mode is most suitable.

Fig. 4-14 A/D Conversion Operation in the Select Mode (Four-Buffer Mode)



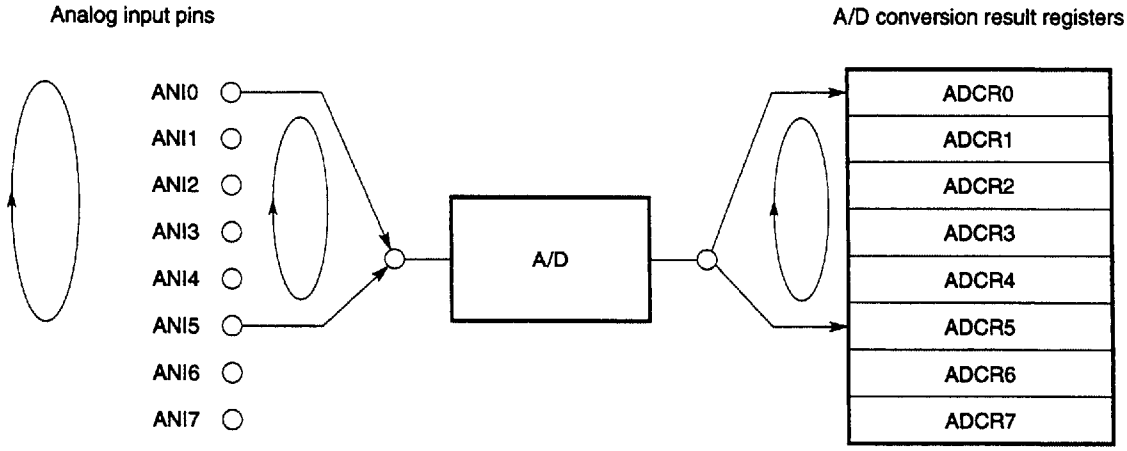
(2) Scan mode

To perform A/D conversion, the scan mode sequentially selects those analog inputs that are specified with the ADM register. The results of A/D conversions are loaded into the A/D conversion result registers that correspond to the analog inputs on a one-to-one basis. INTAD is generated when an A/D conversion operation for the specified analog inputs has been completed. This mode repeats A/D conversion operation based on scan operation.

For example, if ANI0 to ANI5 are specified for the scan mode, ANI0 to ANI5 are converted to digital form in the stated order. When A/D conversion for ANI5 is completed, INTAD is generated to repeat A/D conversion, starting at ANI0.

This mode is most suitable for monitoring multiple analog inputs at all times.

Fig. 4-15 A/D Conversion Operation in the Scan Mode



(3) Mix mode

The mix mode is a mixture of the select mode and scan mode. This mode performs A/D conversion operation (scan processing) in the scan mode after performing A/D conversion operation (select processing) in the select mode on an external trigger or software trigger. In this mode, analog inputs subject to scan processing are uniquely determined by analog inputs and buffer mode selected for select processing.

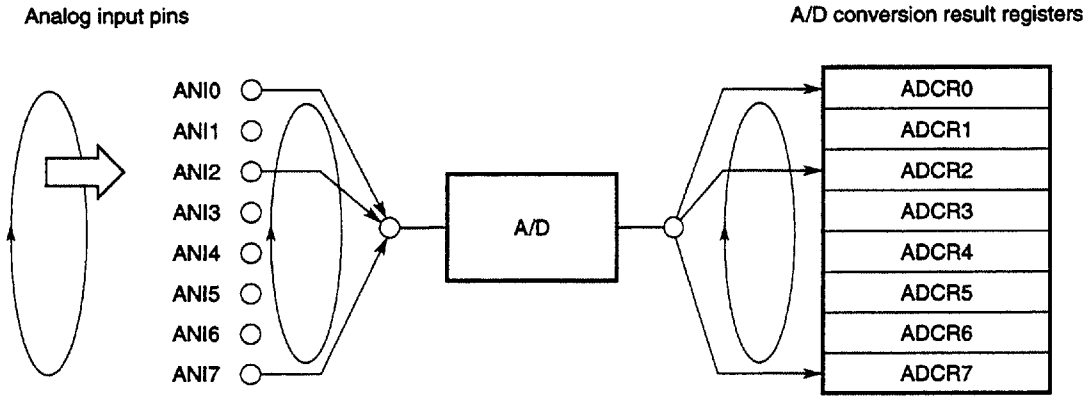
INTAD is generated upon completion of select processing. A/D conversion operation repeats scan processing until a new trigger is applied.

Table 4-3 Analog Input in the Mix Mode

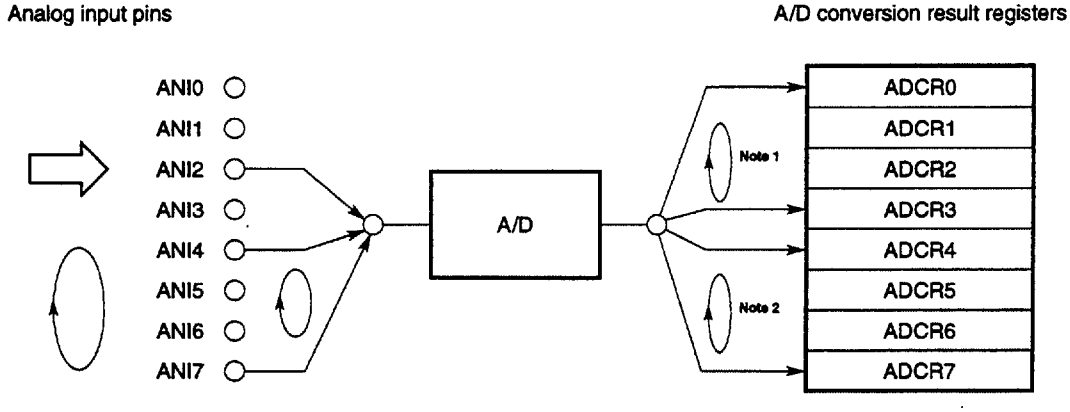
Select processing		Scan processing
Buffer mode	Analog input selection	
One-buffer mode	Selects any of ANI0 to ANI7.	Scans ANI0 to ANI7.
Four-buffer mode	Selects any of ANI0 to ANI3.	Scans ANI4 to ANI7.
	Selects any of ANI4 to ANI7.	Scans ANI0 to ANI3.

Fig. 4-16 A/D Conversion Operation in the Mix Mode

(a) Example of select processing in one-buffer mode



(b) Example of select processing in four-buffer mode



Notes 1. Select processing (four-buffer mode)

2. Scan processing

4.6 SERIAL INTERFACE

The μPD78334 is provided with the following two separate channels for the serial interface function:

- Asynchronous serial interface
- Synchronous serial interface

In addition, the μPD78334 contains a baud rate generator, enabling serial transfer to be performed at a desired rate independently of the operation frequency. The baud rate generator works with both of the two serial interface channels.

4.6.1 Asynchronous Serial Interface (UART)

The asynchronous serial interface is configured as shown in Fig. 4-17.

The asynchronous serial interface uses start and stop bits to indicate the beginning and end of bits or characters in transmission operation.

The asynchronous serial interface of the μPD78334 has a "zero parity" function, which is a parity selection option for a special data transmission format.

When the zero parity is selected, a 0 is unconditionally added as a parity bit to serial data being sent. When serial data is received, no parity error occurs regardless of any parity bit added to the data.

The zero parity function is useful for performing serial communication when a serial data transmission format is yet to be determined immediately after the system is powered.

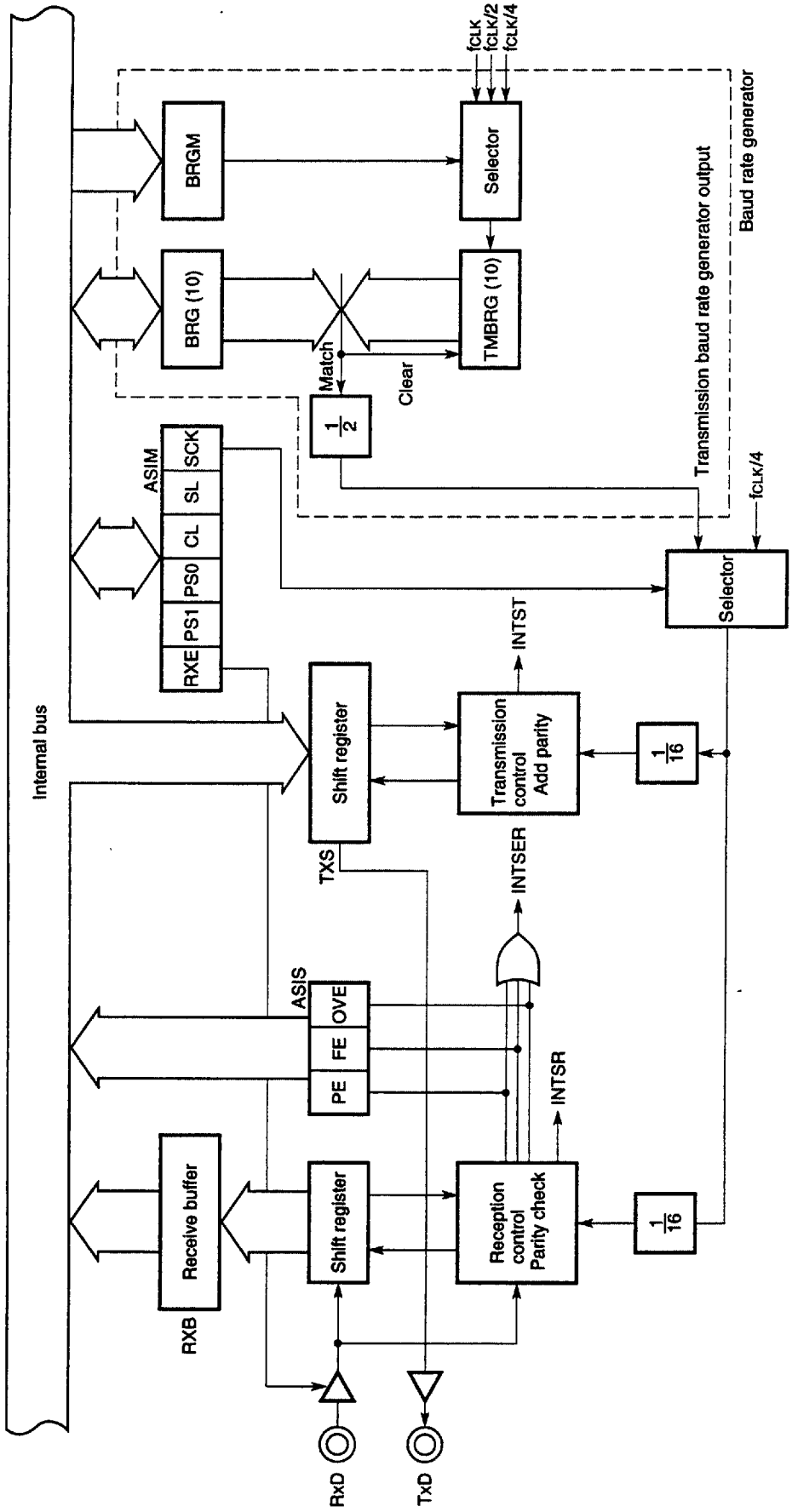
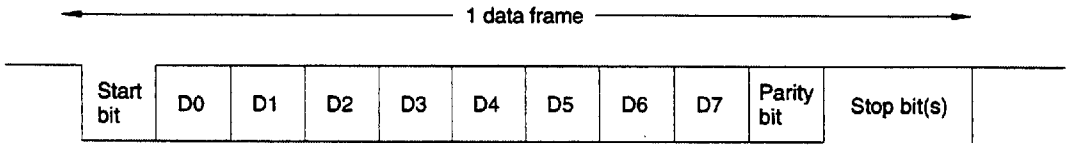


Fig. 4-17 Block Diagram of the Asynchronous Serial Interface

As shown in Fig. 4-18, the format of transmission is such that one data frame consists of a start bit, character bits, a parity bit, and stop bit(s).

The number of character bits in a data frame, the type of parity, and the number of stop bits are specified using the asynchronous serial data mode register.

Fig. 4-18 Data Transmission Format for the Asynchronous Serial Interface



- Start bit : 1 bit
- Character bit : 7 or 8 bits
- Parity bit : Even parity, odd parity, zero parity, or without parity
- Stop bit : 1 or 2 bits

A serial transmission rate is selected in the 75 bps to 287 kbps range according to the contents of the asynchronous serial interface mode register and the setting of the baud rate generator.

If a serial data receive error occurs, the error can be identified by reading the contents of the asynchronous serial interface status register.

4.6.2 Synchronous Serial Interface

The synchronous serial interface of the μPD78334 is configured as shown in Fig. 4-19.

The synchronous serial interface of the μPD78334 operates in one of the two modes described below.

- **Three-wire serial I/O mode**

  Eight-bit data is transferred via three lines: the serial clock ($\overline{SCK}$), serial input (SI), and serial output (SO). This mode is useful when a peripheral I/O device or display controller containing a conventional synchronous serial interface is to be connected.

- **Serial bus interface (SBI) mode**

  This mode enables communication with more than one device by using two lines: the serial clock ($\overline{SCK}$) and serial data bus (SB0 or SB1).

  The SBI mode conforms to the NEC serial bus format.

  In the SBI mode, an address for selecting a target device for serial communication, commands directed to the device, and data to be transmitted can be output onto the serial data bus. This means that a line for handshaking, required when multiple devices are connected via a conventional synchronous serial interface, can be eliminated, thus enabling I/O ports to be used more effectively.

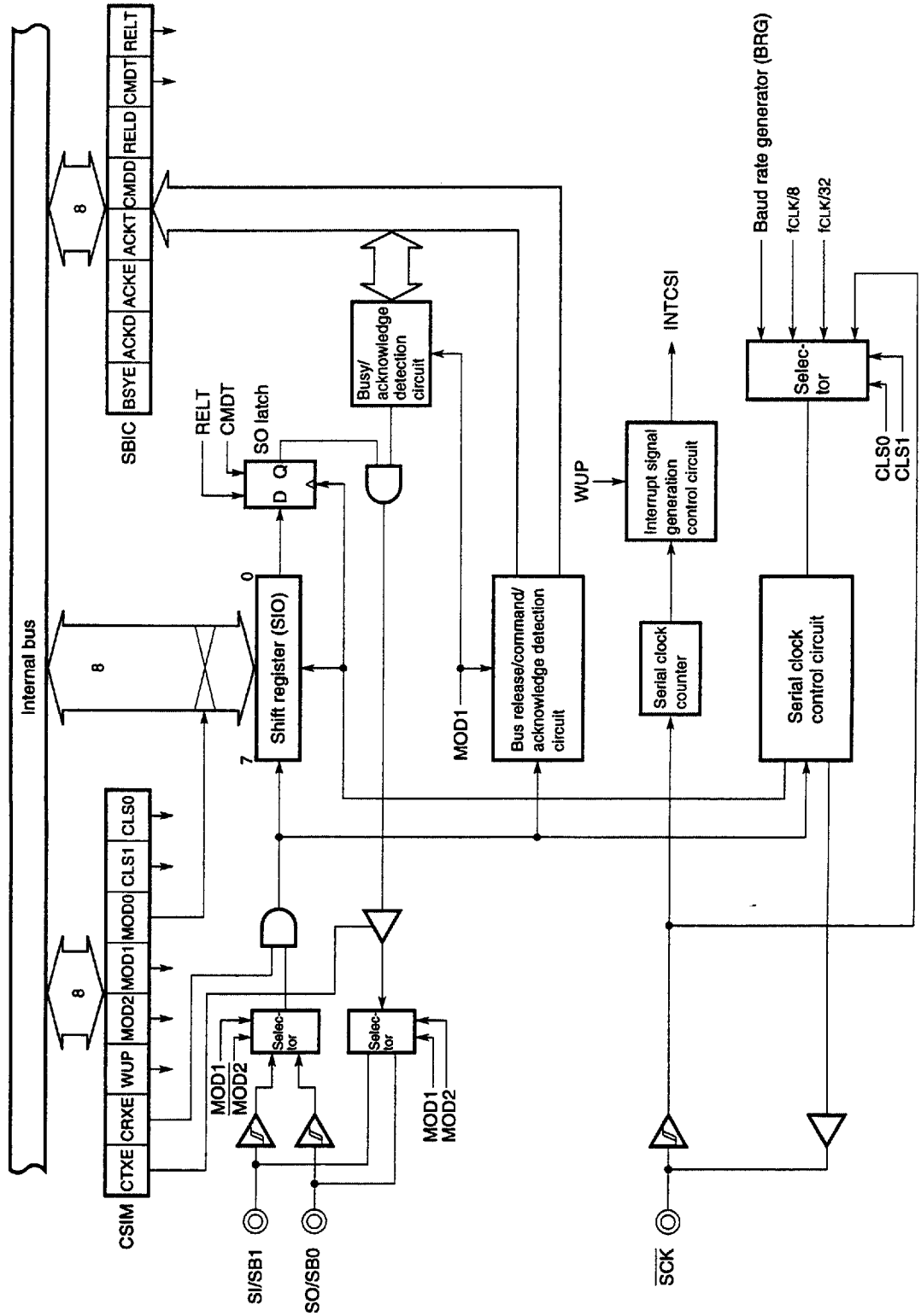


Fig. 4-19 Block Diagram of the Synchronous Serial Interface

(1) Three-wire serial I/O mode

The three-wire serial I/O mode allows communication with a device having a conventional synchronous serial interface.

Basically, communication is performed using three lines: the serial clock ($\overline{SCK}$), serial data output (SO), and serial data input (SI). To connect more than one device, another line is required for handshaking.

Fig. 4-20 Example of System Configuration in the Three-Wire Serial I/O Mode

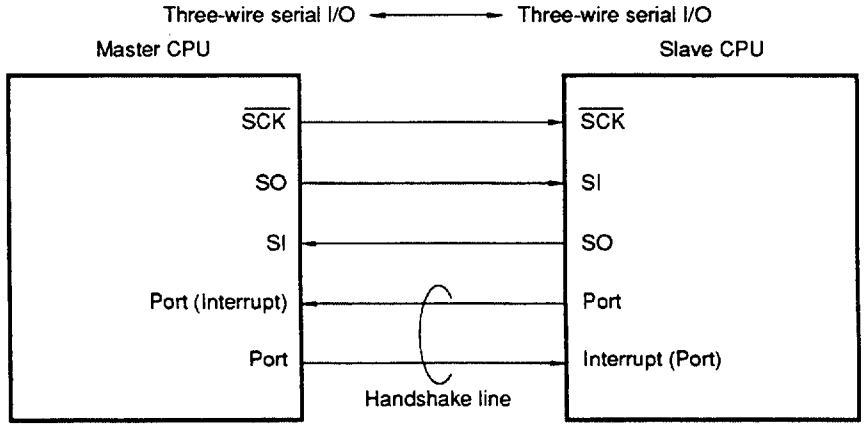


Fig. 4-21 shows the timing of the three-wire serial I/O mode.

Transmission is initiated by setting the transmission permission bit (CTXE) of the clock synchronous serial interface mode register (CSIM) to 1.

When data to be transmitted is written to the shift register (SIO) in this condition, the value of the SIO register is shifted on a rising edge of the serial clock ($\overline{SCK}$). The shifted data is held in the SO latch, then the data is output on the SO pin on a falling edge of $\overline{SCK}$, starting with the MSB. In the three-wire serial I/O mode, the SO pin is configured by hardware as a CMOS output.

Reception is initiated by changing the CRXE bit from 0 to 1 when the CTXE bit of the CSIM register is 0 or by ★ reading the received data from the shift register (SIO) with the CRXE bit set to 1.

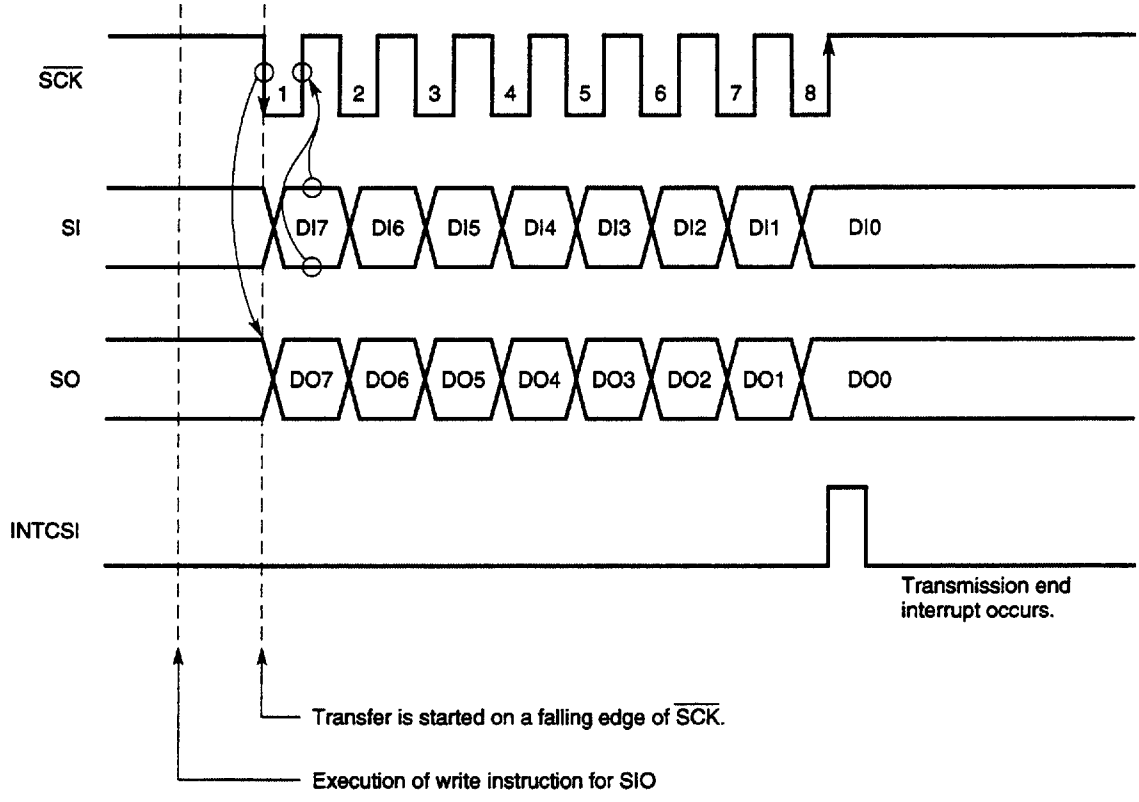
When 1 is rewritten to the CRXE bit of the CSIM register with the bit already set to 1, reception is not initiated.

When the CTXE bit is 1 and the CRXE bit is changed from 0 to 1, reception is not also initiated.

A rising edge of the serial clock ($\overline{SCK}$) latches data received via the SI pin in the SIO register.

$\overline{SCK}$ is stopped on the eighth rising edge of $\overline{SCK}$, and an interrupt request signal (INTCSI) is generated.

Fig. 4-21 Timing of the Three-Wire Serial I/O Mode



(2) SBI mode

The SBI mode allows communication with more than one device via two lines: the serial clock ($\overline{SCK}$) and serial bus (SB0).

A slave device with which communication is to be performed can be selected from multiple devices by outputting its address onto the serial bus (SB0).

After a target device is selected, commands and data are transmitted between the master device and slave device, thus performing serial communication (wake-up function).

In the SBI mode, serial transmission is initiated by loading information to be transmitted into the shift register (SIO), with the transmission permission bit (CTXE) of the clock synchronous serial interface mode register (CSIM) set to 1. When CRXE = 0, a zero is written to the shift register (SIO) after transmission.

The shift register is shifted on a falling edge of the serial clock ($\overline{SCK}$), and the send data is held in the SO latch. The data held in the SO latch is output on the SB0 pin, starting with the MSB.

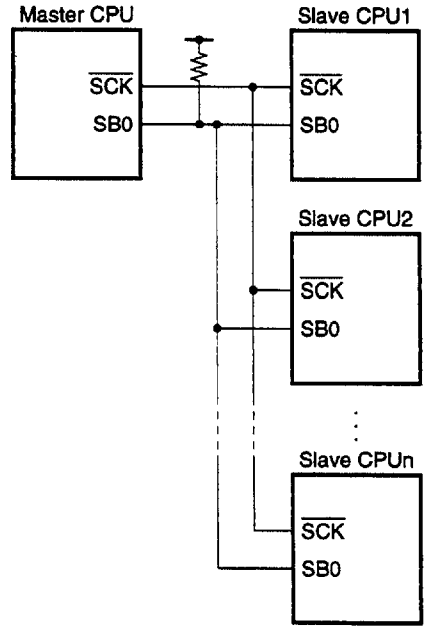
Serial reception is initiated by reading received data from the shift register (SIO) when the transmission permission bit (CTXE) of the clock synchronous serial interface mode register (CSIM) is set to 0 and the reception permission bit (CRXE) is set to 1.

When 1 is rewritten to the CRXE bit of the CSIM register with the bit already set to 1, reception is not initiated. ★

When the CTXE bit is 1 and the CRXE bit is changed from 0 to 1, reception is not also initiated.

On a rising edge of the serial clock ($\overline{SCK}$), data applied to the SB0 pin is latched in the shift register, starting with the MSB. When the SB0 pin is specified as the serial data bus, it functions as an N-ch open-drain I/O, so an external pull-up resistor is required.

Fig. 4-22 Example of System Configuration in the Serial Bus Interface (SBI) Mode



4.6.3 Baud Rate Generator (BRG)

The baud rate generator generates a transmission shift clock for the serial interface function, and consists of the following hardware components:

- Mode register : BRGM
- 10-bit timer : TMBRG
- 10-bit compare register : BRG
- Selector

Fig. 4-23 shows the configuration of the baud rate generator.

With this baud rate generator, a clock of 75 to 287200 (bps) can be generated.

A baud rate to be set can be found from the expression below. The required baud rate is set by setting the value of BRG and the count clock value of TMBRG in the baud rate generator mode register (BRGM).

Expression for baud rate:

$$\text{Baud rate (bps)} = \frac{f_{CLK}}{2 \times 2^n \times (m+1)} \times \frac{1}{16}$$

$$= \frac{f_{CLK}}{2^{n+5} \times (m+1)}$$

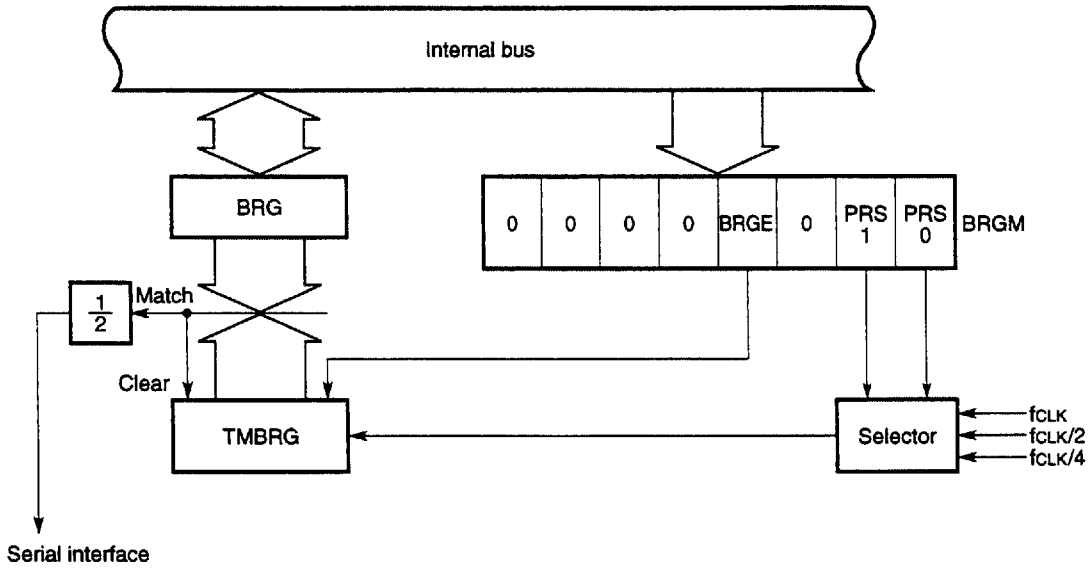
where f_{CLK} : Internal system clock (external oscillator frequency fosc/2)

m : Value set in BRG

n : Value corresponding to the values set in BRGM (PRS1, PRS0)

PRS1	PRS0	Count clock	n
0	0	f_{CLK}	0
0	1	$f_{CLK}/2$	1
1	0	$f_{CLK}/4$	2
1	1	Not to be set	Not to be set

Fig. 4-23 Configuration of the Baud Rate Generator



4.7 WATCHDOG TIMER (WDT)

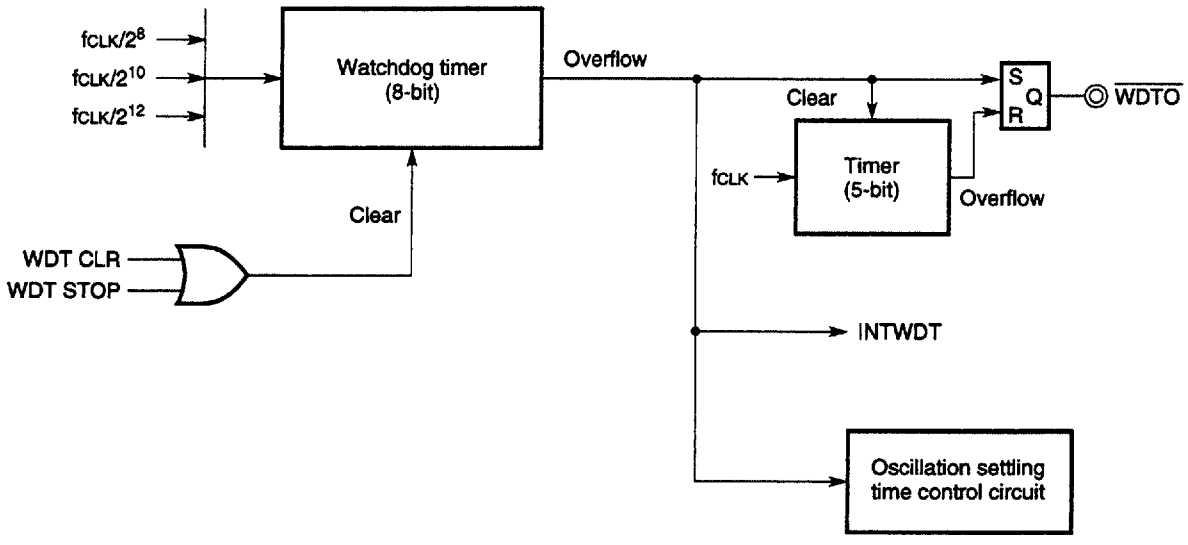
The watchdog timer is a free-running counter with a nonmaskable interrupt function designed to prevent crashes or deadlocks. A program error can be detected when a watchdog timer overflow interrupt (INTWDT) is generated or when the watchdog timer output pin ($\overline{WDTO}$) goes low. By connecting this output to the $\overline{RESET}$ pin, abnormal application system operation caused by a program error can be prevented.

The watchdog timer detects any program error by hardware. So it ensures the detection of crashes and deadlocks for restarting the program. The watchdog timer can also be used to guarantee a time required for the oscillator to perform stable operation when the stop mode is released.

4.7.1 WDT Configuration

Fig. 4-24 shows the configuration of the watchdog timer.

Fig. 4-24 Configuration of the Watchdog Timer



4.7.2 WDT Operation

The watchdog timer generates an interrupt at specified time intervals to detect a program error. So a program should be divided into modules so that the processing of each module can be completed within the WDT interval. Each module should contain an instruction to clear and restart the watchdog timer. For this control, the watchdog timer mode register (WDM) is used.

Once the watchdog timer is started after $\overline{\text{RESET}}$ signal input, it cannot be stopped with an instruction. This is intended to prevent a program error from stopping the watchdog timer. Only a $\overline{\text{RESET}}$ input signal can stop the watchdog timer. As another means to prevent an error, a special instruction is used to write data into WDM.

When a WDT overflow occurs, the watchdog timer output pin ($\overline{\text{WDTO}}$) allows the low level to be output for the period of 32 f_{CLK}. This pin is externally connected with the $\overline{\text{RESET}}$ pin, and is used to reset the system automatically when a program error occurs.

Cautions 1. $\overline{\text{WDTO}}$ is designed to output the low level for the period of 32 f_{CLK} even after $\overline{\text{RESET}}$ input by considering its direct connection to the $\overline{\text{RESET}}$ pin.

2. $\overline{\text{WDTO}}$ may go low for a maximum of 32 f_{CLK} immediately after power-on.

Remark f_{CLK}: Internal system clock (oscillator frequency/2)

4.8 PWM SIGNAL OUTPUT FUNCTION

The μPD78334 has two 8-bit PWM signal outputs. By externally connecting a low-pass filter, a PWM output can be used as a digital-to-analog conversion output. The PWM outputs are most suitable, for example, for a control signal for the actuator of a motor.

One of four PWM signal output repetition frequencies listed in Table 4-4 can be selected by software setting.

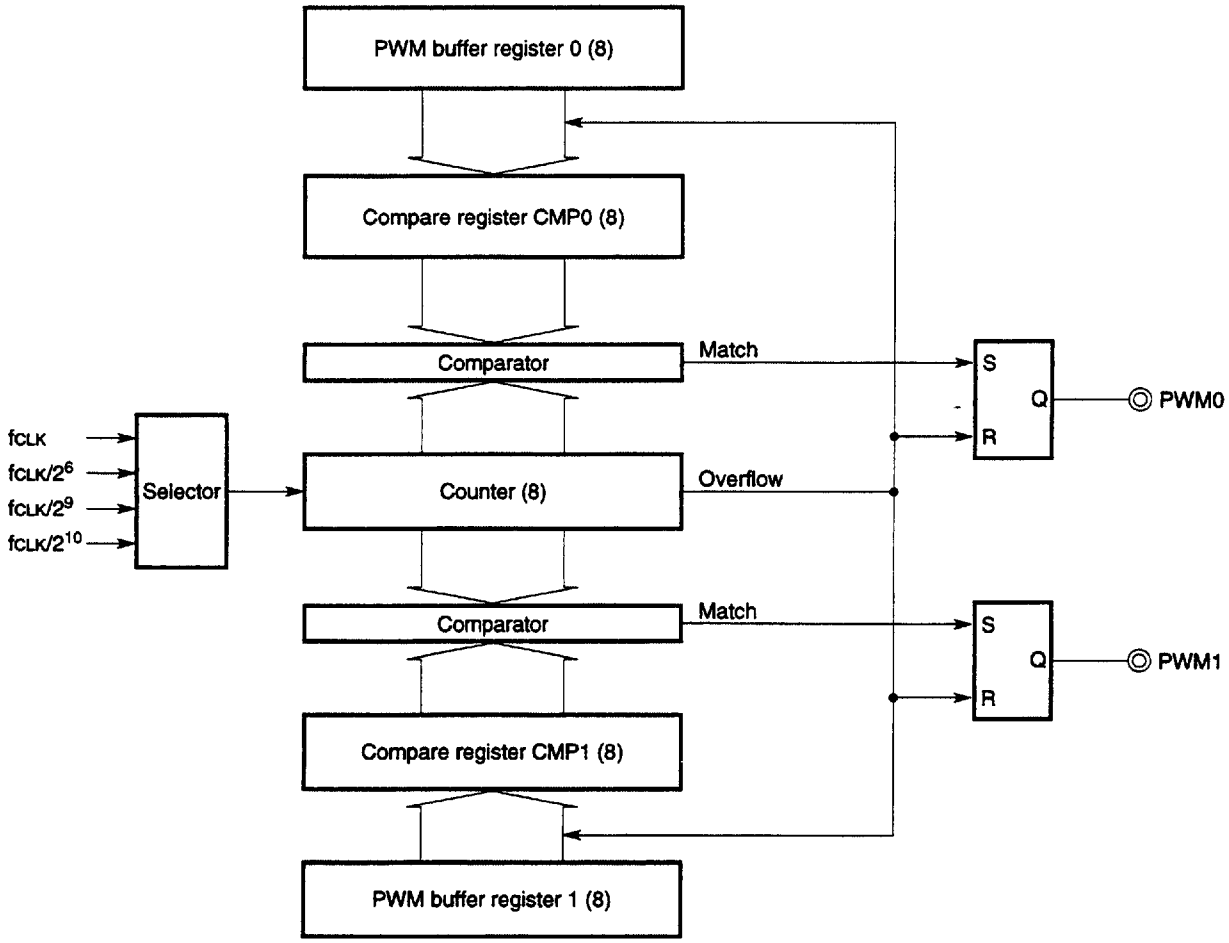
Fig. 4-25 shows the configuration of the PWM output function.

Table 4-4 PWM Signal Repetition Frequencies

Resolution per bit	Repetition frequency
1/fCLK (125 ns)	fCLK/2^8 (31.25 kHz)
2^6/fCLK (8 μs)	fCLK/2^{14} (488 Hz)
2^9/fCLK (64 μs)	fCLK/2^{17} (61 Hz)
2^{10}/fCLK (128 μs)	fCLK/2^{18} (30.5 Hz)

Remark The values in parentheses are for fCLK = 8 MHz.

Fig. 4-25 Configuration of the PWM Output Function



5. INTERRUPT FUNCTION

The μPD78334 has a powerful interrupt function that can handle up to 22 interrupt requests. Three interrupt handling modes are available:

- Vectored interrupt handling
- Macro service
- Context switching

With this interrupt function, complex multitask processing can be efficiently performed at high speed.

Table 5-1 Types of Interrupt Requests and Handling Modes

Interrupt request \ Handling mode	Vectored interrupt handling	Macro service	Context switching
Nonmaskable interrupt	○	–	–
Maskable interrupt	○	○	○
Software interrupt	○	–	○
Exception trap	○	–	–

5.1 TYPES OF INTERRUPT REQUESTS

With the μPD78334, four types of interrupt requests are used:

- Nonmaskable interrupt
- Maskable interrupt
- Software interrupt
- Exception trap

Each type of interrupt request is explained below.

(1) Nonmaskable interrupt

The nonmaskable interrupt is a type of interrupt whose acceptance cannot be disabled with an instruction. A nonmaskable interrupt can be accepted at all times. Nonmaskable interrupt requests are classified into the following two types:

- NMI pin input (NMI)
- Watchdog timer output (WDT)

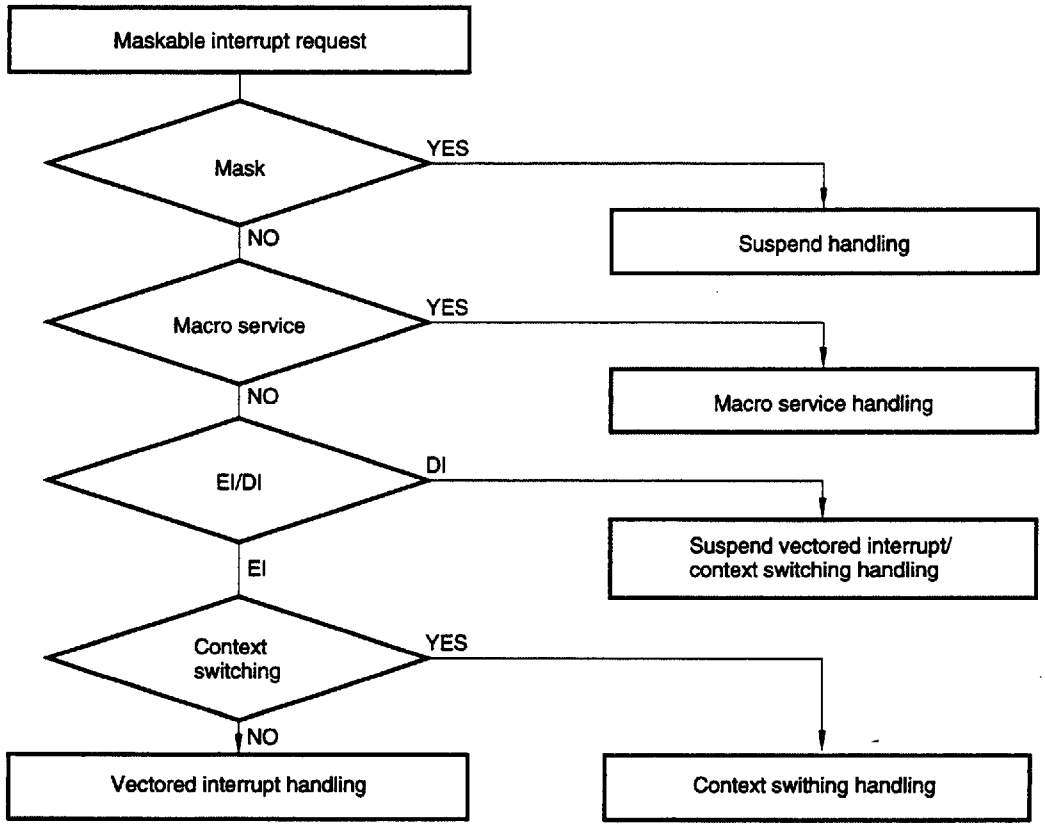
For a nonmaskable interrupt, vectored interrupt handling can be performed.

(2) Maskable interrupt

The maskable interrupt is a type of interrupt whose acceptance can be masked with a control register. Eighteen interrupt sources are available. For a maskable interrupt, one of the following three handling modes can be selected:

- Vectored interrupt handling
- Macro service
- Context switching

Fig. 5-1 Maskable Interrupt Handling



If multiple maskable interrupts occur at the same time, their priorities are determined according to the default priorities. Besides the default priorities, three priority levels can be set by software.

(3) Software interrupt

The software interrupt request is an interrupt request made by executing a CPU break instruction, and can be accepted at all times. For a software interrupt, vectored interrupt handling is performed. The following two instructions can generate a software interrupt:

- BRK : Causes a branch to the address indicated by the contents of memory addresses 003EH and 003FH.
- BRKCS: Causes a branch by context switching processing for switching to the register bank specified in the instruction.

(4) Exception trap

For an exception trap, vectored interrupt handling can be performed. An exception trap occurs in the following case:

- Invalid op code (TRAP):
  Occurs when an instruction for writing to the standby control register and watchdog timer mode register is not executed normally.

5.2 INTERRUPT HANDLING MODES

With the μPD78334, three interrupt handling modes are available:

- Vectored interrupt handling
- Macro service
- Context switching

(1) Vectored interrupt handling

When an interrupt is accepted, the contents of PC and PSW are saved automatically. Then a branch is made to the address indicated by the data contained in the vector address table to execute the interrupt service routine.

(2) Macro service

When an interrupt is accepted, CPU execution is terminated temporarily to execute the service set by firmware. The macro service is performed without CPU involvement, so that the CPU statuses such as PC and PSW need not be saved or restored. Thus the macro service much increases CPU service time.

(3) Context switching

When an interrupt is accepted, a specified register bank is selected by hardware. Then a branch is made to the already selected vector address in the register bank, and the current contents of PC and PSW are saved in the register bank at the same time.

Remark The context means CPU registers that can be accessed from a program being executed. The registers include general registers, PC, PSW, and SP.

Table 5-2 lists the interrupt sources.

Table 5-2 Interrupt Source List

Type	Note 1	Interrupt source		Unit requesting interrupt	Vector table address	Macro service	Context switch
		Name	Trigger				
Non-maskable	–	NMI	NMI pin input	External	0002H	No	No
	–	INTWDT	Watchdog timer	WDT	0004H		
Maskable	0	INTOV	Timer 0 overflow	RPU	0006H	Yes	Yes
	1	INTP0	INTP0 pin input	External	0008H		
	2	INTP1	INTP1 pin input	External	000AH		
	3	INTP2	INTP2 pin input	External	000CH		
	4	INTP3/INTCC00R	INTP3 pin input/CC00R match signal	External/RPU	000EH		
	5	INTP4/INTCC01R	INTP4 pin input/CC01R match signal	External/RPU	0010H		
	6	INTP5	INTP5 pin input	External	0012H		
	7	INTP6/TI	INTP6 pin input/TI input	External	0014H		
	8	INTCMX0	CMX0 match signal	RPU	0016H		
	9	INTCM11	CM11 match signal	RPU	0018H		
	10	INTCM12	CM12 match signal	RPU	001AH		
	11	INTCM20	CM20 match signal	RPU	001CH		
	12	INTCM21	CM21 match signal	RPU	001EH		
	13	INTCM30	CM30 match signal	RPU	0020H		
	14	INTSR	UART reception complete	UART	0024H		
	15	INTST	UART transmission complete	UART	0026H		
	16	INTCSI	CSI transmission complete	CSI	0028H		
	17	INTAD	A/D conversion complete	A/D	002AH		
–	–	INTSER Note 2	UART receive error	UART	–Note 2	No	No
Software	–	BRK	BRK instruction	–	003EH		
	–	BRKCS	BRKCS instruction	–	–		Yes
Exception	–	TRAP	Invalid op code trap	–	003CH		No
Reset	–	RESET	RESET input	–	0000H		

Notes 1. Default priority: Priority used when multiple maskable interrupts occur at the same time, with 0 for the highest priority and 17 for the lowest priority.

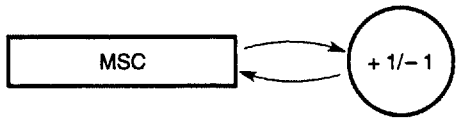
2. Test source: No vectored interrupt occurs.

5.3 MACRO SERVICE

The μPD78334 has nine types of macro services. Each macro service is explained below.

(1) Counter mode: EVTCNT

- **Operation**
  - (a) This mode increments or decrements the 8-bit macro service counter (MSC).
  - (b) When the MSC reaches 0, a vectored interrupt request occurs.

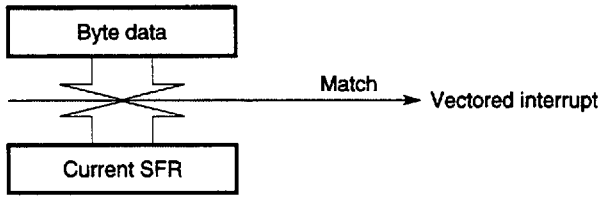


- **Sample application**

  This mode can be used to measure the event counter and capture count.

(2) Data compare mode: DTACMP

- **Operation**
  - (a) This mode compares data set beforehand with the contents of the current SFR determined for each interrupt source.
  - (b) When a match is found, a vectored interrupt occurs. If no match is found, processing just continues.

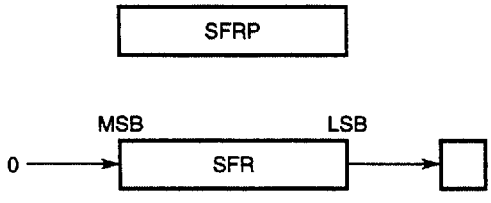


- **Sample application**

  This mode can be used to detect a slave address match in the SBI mode.

(3) Data shift mode: BITSHT

- **Operation**
  - (a) This mode shifts the contents of the SFR pointed to by the SFR pointer (SFRP) one bit right.
  - (b) A vectored interrupt occurs when data 1 is shifted out from the bit 0 position.

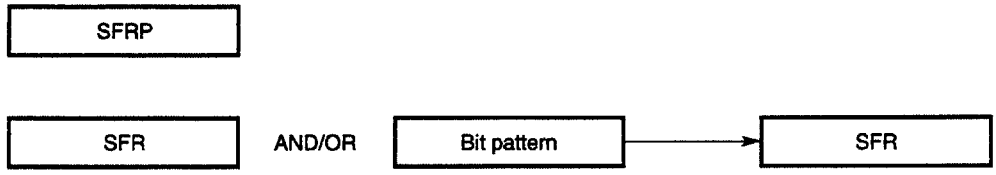


- **Sample application**

  This mode can be used to shift the values of the real-time output port set register (RTPS) and real-time output port reset register (RTPR) for controlling the real-time output port (RTP).

(4) Bit pattern operation mode: BITLOG

- **Operation**
  - (a) This mode performs an AND/OR operation between data (bit pattern) set beforehand and the contents of the SFR pointed to by the SFRP.
  - (b) The result of operation is reloaded into the SFR, and a vectored interrupt occurs.

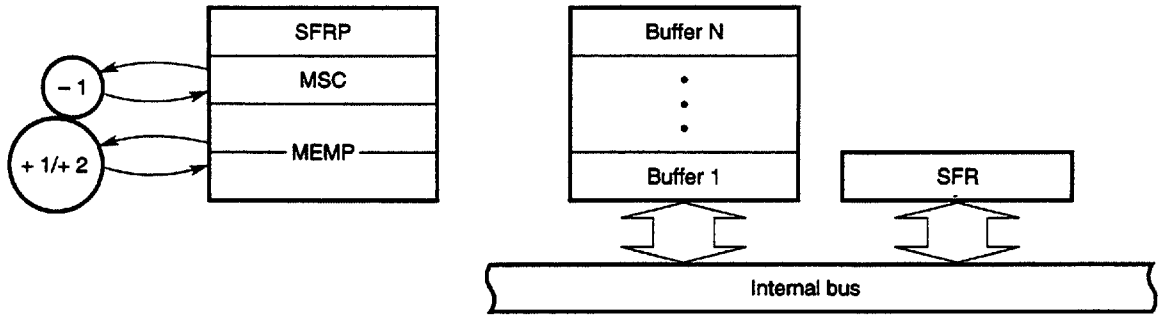


- **Sample application**

  This mode can be used to perform logical operations at an output port.

(5) Block transfer mode: BLKTRS

- **Operation**
  - (a) This mode transfers a data block between the buffer pointed to by the memory pointer (MEMP) and the SFR pointed to by the SFRP.
  - (b) Either an SFR or buffer area can be specified as a transfer source or transfer destination. In addition, either the byte or word can be selected as the length of transfer data.
  - (c) The MSC is used to specify the number of data transfers (block size).
  - (d) Each time the macro service is executed, the MSC is automatically decremented by one, and the MEMP is automatically incremented by one for a byte-data transfer or by two for a word-data transfer.
  - (e) When the MSC reaches 0, vectored interrupt handling is activated.



- **Sample application**

  This mode can be used for data transfer using a serial interface.

(6) Data difference mode: DTADIF

- **Operation**

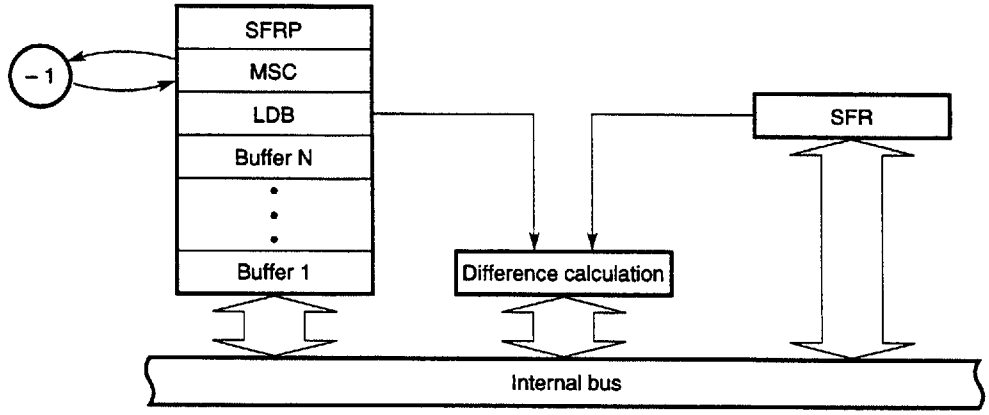
(a) This mode calculates the difference between the contents (current value) of the SFR pointed to by the SFRP and the contents of the SFR already held in the last data buffer (LDB).

(b) The result of calculation is stored in a buffer area specified beforehand.

(c) The current value of the SFR is loaded into the LDB.

(d) The MSC is used to specify the number of data transfers (block size). Each time the macro service is executed, the MSC is automatically decremented by one.

(e) When the MSC reaches 0, vectored interrupt handling is activated.



Remark To match the data length of an SFR subject to difference calculation, either the byte or word can be selected as the data length. However, even if the SFR is an 8-bit SFR, data stored in the LDB is handled as word data.

- **Sample application**

This mode can be used to measure periods or pulse widths by a capture register of the real-time pulse unit (RPU).

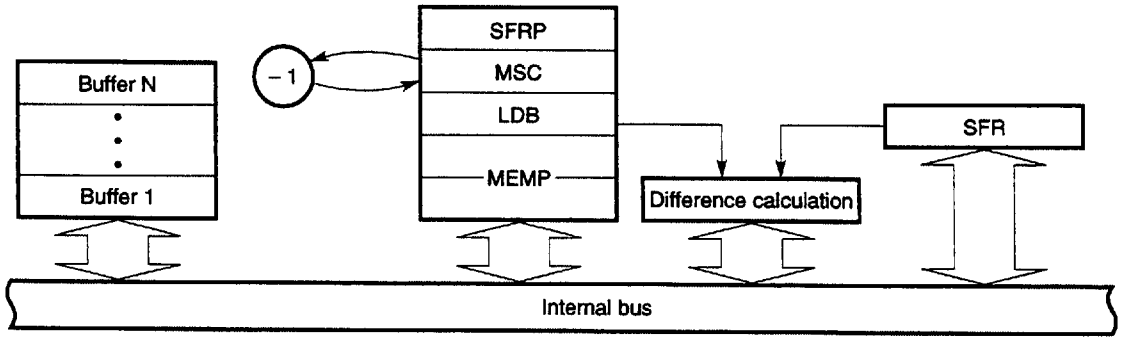
(7) Data difference mode (with memory pointer): DTADIF-P

- **Operation**

This mode is the data difference mode with a memory pointer (MEMP) added. With this MEMP addition, a buffer area for storing difference data can be freely set in memory space.

Remark A buffer is specified by the result of operation on the MEMP and MSC[Note]. The MEMP is not updated after data transfer.

Note • For an 8-bit MSC: MEMP – MSC
• For a 16-bit MSC: MEMP – (MSC × 2)



- **Sample application**

Same as (6) above

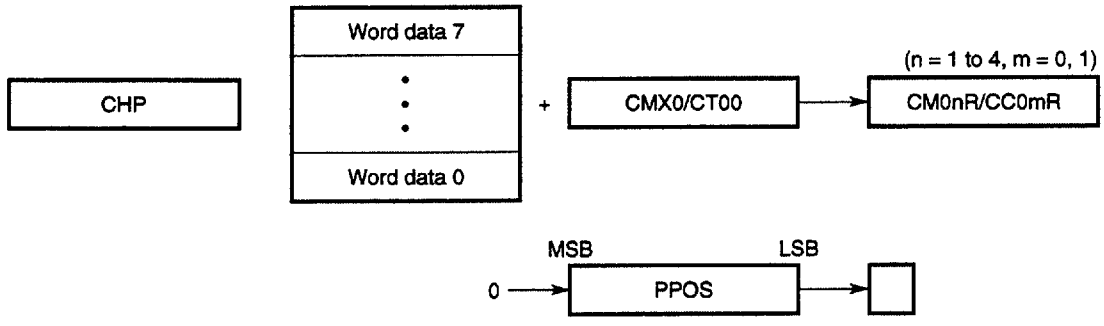
(8) Continuous pulse output control mode 1: PPOSEQ

- **Operation**

(a) This mode adds word data n specified by the channel pointer (CHP) to the value of CMX0 or CT00, and stores the result into the compare register (CM0nR: n = 1 to 4) or capture/compare register (CC0mR: m = 0, 1) of the real-time pulse unit (RPU).
CM0nR or CC0mR: Compare register that corresponds to the bits with PPOS set to 1

(b) Then the contents of PPOS are shifted one bit right.

(c) A vectored interrupt occurs when data 1 is shifted out from the bit 0 position of PPOS.



- **Sample application**

In an engine control application, for example, this mode can be used for full-sequential fuel injection pulse output using programmable pulse output from the RPU.

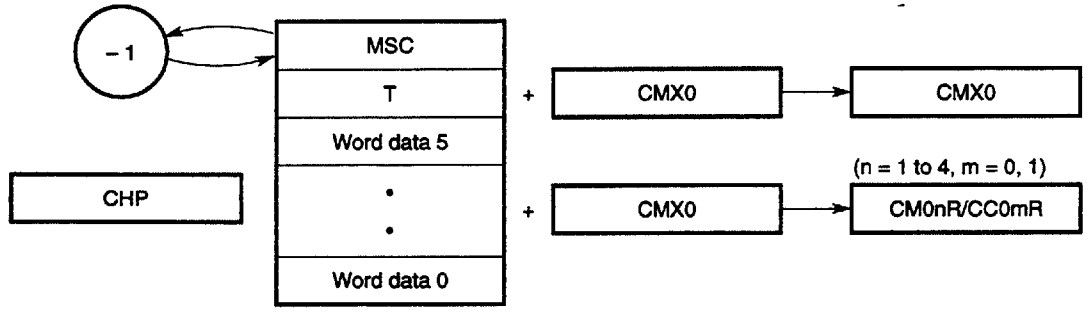
(9) Continuous pulse output control mode 2: PPOPRL

- **Operation**

(a) This mode adds word data n specified by the channel pointer (CHP) to the value of CMX0, and stores the result into the compare register (CM0nR: n = 1 to 4) or capture/compare register (CC0mR: m = 0, 1) of the real-time pulse unit (RPU).
CM0nR or CC0mR: Compare register that corresponds to the bits with PPOS set to 1

(b) Then word data is added to CMX0, and the MSC is decremented.

(c) A vectored interrupt occurs when MSC reaches 0.



- **Sample application**

See the sample application in (8).

5.4 CONTEXT SWITCHING

The context switching is a function that selects a specified register bank by hardware when an interrupt occurs or a BRKCS instruction is executed, then causes a branch to the vector address set beforehand in the register bank and saves the current contents of PC and PSW in the register bank at the same time.

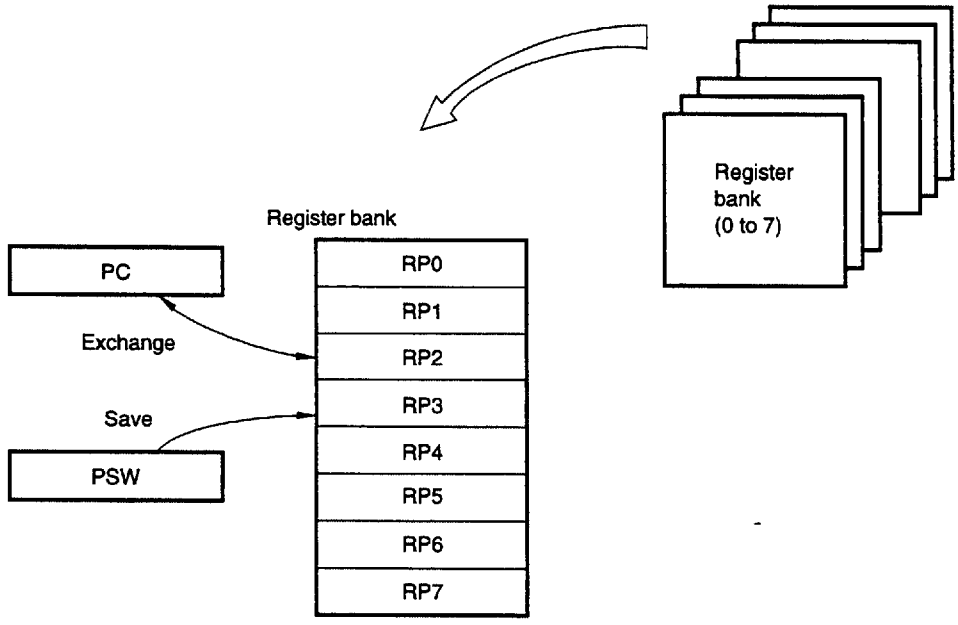
5.4.1 Context Switching Function Based on an Interrupt Request

The context switching function can be activated when the context switching enable register corresponding to each maskable interrupt request is set to 1 in the EI (interrupt enable) state.

Context switching operation based on an interrupt request is performed as described below.

(1) When an interrupt request occurs, a register bank subject to context switching is specified from the contents of the lower three bits of the row address (even address) of the corresponding vector table.
(2) The vector address set beforehand in the register bank subject to context switching is transferred to PC, and the contents of PC and PSW present immediately before switching operation are saved in the register bank.
(3) A branch is made to the address pointed to by the newly set contents of PC.

Fig. 5-2 Context Switching Operation



5.4.2 Context Switching Function Based on the BRKCS Instruction

The context switching function can be activated with the BRKCS instruction.

Context switching operation based on an interrupt request is performed as described below.

(1) An 8-bit register is specified in an operand of the BRKCS instruction. The contents of the register determine a register bank subject to context switching. (Only the low-order three bits of the eight bits are used.)
(2) The vector address set beforehand in the register bank subject to context switching is transferred to PC, and the contents of PC and PSW present immediately before switching operation are saved in the register bank at the same time.
(3) A branch is made to the address pointed to by the newly set contents of PC.

5.4.3 Return from Context Switching

To return from context switching, one of the following two instructions is used. The source of context switching activation determines which instruction to use.

Table 5-3 Return from Context Switching

Return instruction	Context switching activation source
RETCS	Activation based on interrupt occurrence
RETCSB	Activation based on BRKCS instruction

6. EXTERNAL DEVICE EXPANSION FUNCTION

Besides its internal ROM and RAM areas, the μPD78334 can have external devices (data memory, program memory, and peripheral devices) expanded. When an external device is connected, port 4, 5, or 9 is used for address/data, $\overline{RD}$, and $\overline{WR}$ control.

Table 6-1 Pin Functions Assigned when External Devices are Connected

Pin	Pin function when external device is connected	
	Function	Name
P40 - P47	Multiplexed address/data bus	AD0 - AD7
P50 - P57	Address bus	A8 - A15
P90	Read strobe	$\overline{RD}$
P91	Write strobe	$\overline{WR}$
CLKOUT	System clock output	CLKOUT
ASTB	Address strobe	ASTB

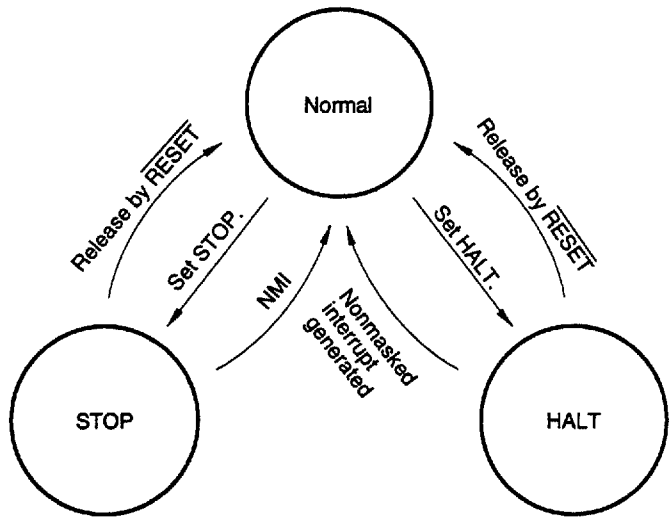
7. STANDBY FUNCTION

The μPD78334 has a standby function to reduce power consumption of the system. With the standby function, two modes are available:

- HALT mode : In this mode, the CPU operation clock is stopped. Intermittent operation, when combined with the normal operation mode, can reduce overall system power consumption.
- STOP mode : In this mode, the oscillator is stopped to stop the entire system.
  Since only leakage currents may flow in this mode, system power consumption can be minimized.

Each mode is set by software. Fig. 7-1 is the transition diagram of the standby modes (STOP/HALT modes).

Fig. 7-1 Transition Diagram of the Standby Modes



8. RESET FUNCTION

When the signal applied to the $\overline{\text{RESET}}$ input pin is low, the system is reset, and each hardware component is placed in the status indicated in Table 8-1. When the signal applied to the $\overline{\text{RESET}}$ input port goes high, the reset status is released, and program execution starts. The contents of registers must be initialized in the program as required.

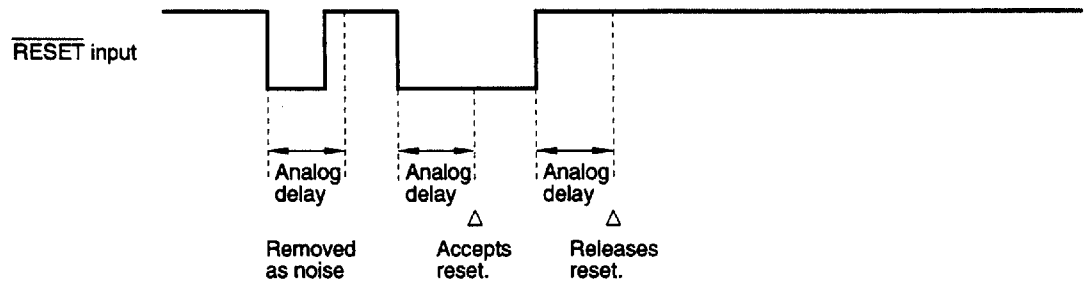
In particular, the number of cycles specified in the programmable wait control register and fetch cycle control register must be changed as required.

The $\overline{\text{RESET}}$ input pin contains a noise eliminator based on analog delays to prevent abnormal operation due to noise.

Cautions 1. When $\overline{\text{RESET}}$ is active (low level), all pins except $\overline{\text{WDTO}}$, CLKOUT, AV_{REF}, AV_{DD}, AV_{SS}, V_{DD}, Vss, X1, and X2 go into the high-impedance state.

2. When RAM is expanded externally, attach a pull-up resistor to the P90/$\overline{\text{RD}}$ pin and P91/$\overline{\text{WR}}$ pin. Otherwise, these pins may go into the high-impedance state, and the contents of the external RAM may be destroyed. Signals may contend for the address/data bus, resulting in damage to the input/output circuits.

Fig. 8-1 Acceptance of the $\overline{\text{RESET}}$ Signal



In reset operation at power-on, a time for stabilized operation between power-on to reset acceptance is required as shown in Fig. 8-2.

Fig. 8-2 Reset Operation at Power-on

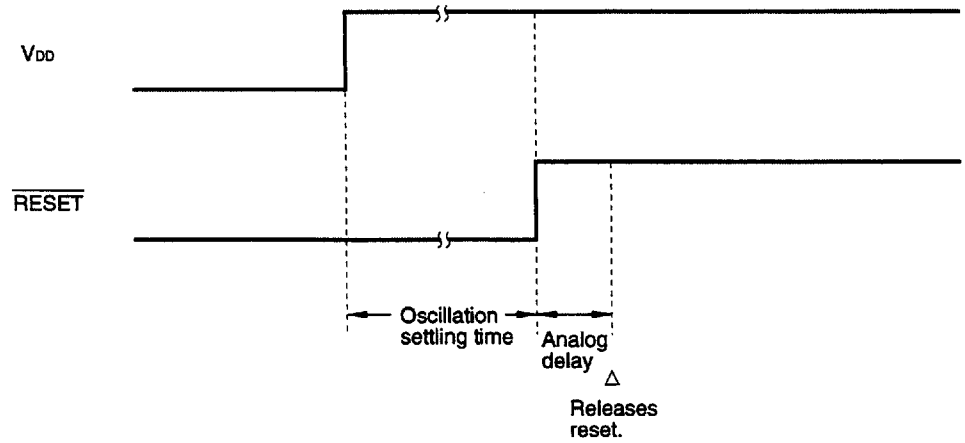


Table 8-1 Hardware Statuses after Reset (1/2)

Hardware			Status after reset
Control registers	Program counter (PC)		The contents of a reset vector table (0000H, 0001H) are set.
	Stack pointer (SP)		Undefined[Note]
	Program status word (PSW)		0000H
	CPU control word (CCW)		00H
Internal RAM	Data memory		Undefined[Note]
	General registers (R0 - R15)		
Ports	Output latches (P0, P1, P3 - P5, P9)		Undefined
	Mode registers	(PM0, PM1, PM3, PM5)	FFH
		(PM9)	××11 1111B
	Mode control registers (PMC0, PMC1, PMC3)		00H
	Port read control register (PRDC)		
Real-time output ports (RTP)	Real-time output port register (RTP)		Undefined
	Real-time output port set register (RTPS)		00H
	Real-time output port reset register (RTPR)		
Real-time pulse unit (RPU)	Timers (TM0, TM1, TM2, TM3)		0000H
	Timer low access register (TLA)		Undefined
	Timer unit mode registers (TUM0, TUM1)		00H
	Timer control register 0 (TMC0)		
	Timer control register 1 (TMC1)		40H
	Timer output control registers (TOC0, TOC1)		00H
	Programmable pulse output set register (PPOS)		
	Compare registers (CMX0, CM01R, CM02R, CM03R, CM04R) (CM11, CM12, CM20, CM21, CM30)		Undefined
	Capture/compare registers (CC00R, CC01R)		
	Capture registers (CT00, CT01, CT02, CT10)		
PWM output function	PWM control register (PWMC)		00H
	PWM buffer registers (PWM0, PWM1)		Undefined
A/D converter	A/D converter mode register (ADM)		00H
	A/D conversion result registers (ADCR0 - ADCR7)		Undefined

Note When the input of the $\overline{\text{RESET}}$ signal releases the STOP mode, the unit of hardware holds the value stored immediately before the STOP mode is set.

Table 8-1 Hardware Statuses after Reset (2/2)

Hardware			Status after reset
Serial interface	Asynchronous serial interface mode register (ASIM)		80H
	Asynchronous serial interface status register (ASIS)		00H
	Clock synchronous serial interface mode register (CSIM)		00H
	Serial bus interface control register (SBIC)		
	Serial shift register (SIO)		Undefined
	Serial receive buffer (RXB)		
	Serial transmit shift register (TXS)		
	Baud rate generator compare register (BRG)		0000H
	Baud rate generator mode register (BRGM)		00H
External expansion function	Memory expansion mode register (MM)		00H
	Programmable wait control register (PWC)		22H
	Fetch cycle control register (FCC)		00H
Watchdog timer	Watchdog timer mode register (WDM)		00H
Interrupt function	External interrupt mode registers (INTM0, INTM1)		00H
	Interrupt request flag registers	(IF0L, IF0H, IF1L)	
	Interrupt mask registers	(MK0L, MK0H)	FFH
		(MK1L)	××××× ×111B
	Interrupt handling mode specification registers	(ISM0L, ISM0H, ISM1L)	00H
	Context switching enable registers	(CSE0L, CSE0H, CSE1L)	
	Priority specification buffer registers	(PB0L, PB0H, PB1L)	
	Priority specification register (PRSL)		
	In-service priority register (ISPR)		
CPU control	Standby control register (STBC)		0000 ×000B

9. INSTRUCTION SET

This chapter explains only the operations of the instructions.

Refer to the *μPD78334 User's Manual* (IEU-1315) for the number of clocks and instruction codes for executing each instruction.

(1) Operand notation and coding format

Operands are coded in the operand field of each instruction as listed in the coding column of Table 9-1. For details of the operand format, refer to the relevant assembler specifications. When several coding forms are presented, any one of them is selected. Uppercase letters and the symbols, +, –, #, $, !, and [ ], are keywords and must be written as they are.

For immediate data, an appropriate numeric or label must be written. The symbols #, $, !, and [ ] must not be omitted when describing labels.

Table 9-1 Operand Notation and Coding Format

Notation	Coding
r	R0, R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11, R12, R13, R14, R15
r1	R0, R1, R2, R3, R4, R5, R6, R7
r2	C, B
rp	RP0, RP1, RP2, RP3, RP4, RP5, RP6, RP7
rp1	RP0, RP1, RP2, RP3, RP4, RP5, RP6, RP7
rp2	DE, HL, VP, UP
sfr	Special function register abbreviation (See **Table 2-1**.)
sfrp	Special function register abbreviation (16-bit manipulation register. See **Table 2-1**.)
post	RP0, RP1, RP2, RP3, RP4, RP5/PSW, RP6, RP7 (Can be coded more than once. However, RP5 can only be used in a PUSH or POP instruction and PSW can only be used in a PUSHU or POPU instruction.)
mem	[DE], [HL], [DE+], [HL+], [DE−], [HL−], [VP], [UP] : Register indirect mode [DE+A], [HL+A], [DE+B], [HL+B], [VP+DE], [VP+HL] : Based indexed mode [DE+byte], [HL+byte], [VP+byte], [UP+byte], [SP+byte] : Based mode word[A], word[B], word[DE], word[HL] : Indexed mode
saddr	FE20H - FF1FH Immediate data or label
saddrp	FE20H - FF1EH Immediate data (bit 0 = 0, however) or label (for 16-bit manipulation)
$ addr16	0000H - FDFFH Immediate data or label: Relative addressing
! addr16	0000H - FDFFH Immediate data or label: Immediate addressing (Data up to FFFFH can be coded in an MOV instruction.)
addr11	800H - FFFH Immediate data or label
addr5	40H - 7EH Immediate data (bit 0 = 0, however)[Note] or label
word	16-bit immediate data or label
byte	8-bit immediate data or label
bit	3-bit immediate data or label
n	3-bit immediate data (0 to 7)

Note Do not attempt to access word data at an odd-numbered address (bit 0 = 1).

Remarks 1. The same register name can be specified in rp and rp1, but different codes are generated. Refer to the *User's Manual* (IEU-1315) for details.

2. Functional names (X, A, C, B, E, D, L, H, AX, BC, DE, HL, VP, and UP) can be specified in r, r1, rp, rp1, and post, as well as absolute names (R0 to R15 and RP0 to RP7).
Refer to the *User's Manual* (IEU-1315) for details.

3. Immediate addressing is effective for entire address spaces. Relative addressing is effective for the locations within a displacement range of −128 to +127 from the starting address of the next instruction.

(2) Legend

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
R0 - R15	:	Register 0 to register 15 (absolute name)
AX	:	Register pair (AX); 16-bit accumulator
BC	:	Register pair (BC)
DE	:	Register pair (DE)
HL	:	Register pair (HL)
RP0 - RP7	:	Register pair 0 to register pair 7 (absolute name)
PC	:	Program counter
SP	:	Stack pointer
UP	:	User stack pointer
PSW	:	Program status word
CY	:	Carry flag
AC	:	Auxiliary carry flag
Z	:	Zero flag
P/V	:	Parity/overflow flag
S	:	Sign flag
TPF	:	Table position flag
RBS	:	Register bank selection flag
RSS	:	Register set selection flag
IE	:	Interrupt request enable flag
STBC	:	Standby control register
WDM	:	Watchdog timer mode register
jdisp8	:	Signed 8-bit data (displacement value: −128 to +127)
( )	:	Contents at an address enclosed in parentheses or at an address indicated in a register indicated in parentheses. (+) and (−) indicate that an address or the contents of a register indicated in parentheses are incremented and decremented by one after execution of the instruction, respectively.
(( ))	:	Contents at an address indicated by the contents at an address indicated in parentheses (( )).
xxH	:	Hexadecimal number
xH, xL	:	High-order 8 bits and low-order 8 bits of 16-bit register

(3) Notational symbols in flag operation field

Table 9-2 Notational Symbols in Flag Operation Field

Symbol	Explanation
(Blank)	No change
0	Cleared to zero.
1	Set to 1.
×	Set or reset according to the result.
P	P/V flag operates as a parity flag.
V	P/V flag operates as an overflow flag.
R	Saved values are restored.

Instruction set	Mnemonic	Operand	Byte	Operation	Flag S Z AC P/V CY
8-bit data transfer	MOV	r1, #byte	2	r1 ← byte	
		saddr, #byte	3	(saddr) ← byte	
		sfr[Note], #byte	3	sfr ← byte	
		r, r1	2	r ← r1	
		A, r1	1	A ← r1	
		A, saddr	2	A ← (saddr)	
		saddr, A	2	(saddr) ← A	
		saddr, saddr	3	(saddr) ← (saddr)	
		A, sfr	2	A ← sfr	
		sfr, A	2	sfr ← A	
		A, mem	1 - 4	A ← (mem)	
		mem, A	1 - 4	(mem) ← A	
		A, [saddrp]	2	A ← ((saddrp))	
		[saddrp], A	2	((saddrp)) ← A	
		A, !addr16	4	A ← (addr16)	
		!addr16, A	4	(addr16) ← A	
		PSWL, #byte	3	PSWL ← byte	× × × × ×
		PSWH, #byte	3	PSWH ← byte	
		PSWL, A	2	PSWL ← A	× × × × ×
		PSWH, A	2	PSWH ← A	
		A, PSWL	2	A ← PSWL	
		A, PSWH	2	A ← PSWH	
	XCH	A, r1	1	A ↔ r1	
		r, r1	2	r ↔ r1	
		A, mem	2 - 4	A ↔ (mem)	
		A, saddr	2	A ↔ (saddr)	
		A, sfr	3	A ↔ sfr	
		A, [saddrp]	2	A ↔ ((saddrp))	
		saddr, saddr	3	(saddr) ↔ (saddr)	

Note If STBC or WDM is coded in sfr, a different instruction having the different byte count is generated.

Instruction set	Mnemonic	Operand	Byte	Operation	Flag S	Z	AC	P/V	CY
16-bit data transfer	MOVW	rp1, #word	3	rp1 ← word					
		saddrp, #word	4	(saddrp) ← word					
		sfrp, #word	4	sfrp ← word					
		rp, rp1	2	rp ← rp1					
		AX, saddrp	2	AX ← (saddrp)					
		saddrp, AX	2	(saddrp) ← AX					
		saddrp, saddrp	3	(saddrp) ← (saddrp)					
		AX, sfrp	2	AX ← sfrp					
		sfrp, AX	2	sfrp ← AX					
		rp1, !addr16	4	rp1 ← (addr16)					
		!addr16, rp1	4	(addr16) ← rp1					
		AX, mem	2 - 4	AX ← (mem)					
		mem, AX	2 - 4	(mem) ← AX					
	XCHW	AX, saddrp	2	AX ↔ (saddrp)					
		AX, sfrp	3	AX ↔ sfrp					
		saddrp, saddrp	3	(saddrp) ↔ (saddrp)					
		rp, rp1	2	rp ↔ rp1					
		AX, mem	2 - 4	AX ↔ (mem)					
8-bit arithmetic/ logical	ADD	A, #byte	2	A, CY ← A + byte	×	×	×	V	×
		saddr, #byte	3	(saddr), CY ← (saddr) + byte	×	×	×	V	×
		sfr, #byte	4	sfr, CY ← sfr + byte	×	×	×	V	×
		r, r1	2	r, CY ← r + r1	×	×	×	V	×
		A, saddr	2	A, CY ← A + (saddr)	×	×	×	V	×
		A, sfr	3	A, CY ← A + sfr	×	×	×	V	×
		saddr, saddr	3	(saddr), CY ← (saddr) + (saddr)	×	×	×	V	×
		A, mem	2 - 4	A, CY ← A + (mem)	×	×	×	V	×
		mem, A	2 - 4	(mem), CY ← (mem) + A	×	×	×	V	×
	ADDC	A, #byte	2	A, CY ← A + byte + CY	×	×	×	V	×
		saddr, #byte	3	(saddr), CY ← (saddr) + byte + CY	×	×	×	V	×
		sfr, #byte	4	sfr, CY ← sfr + byte + CY	×	×	×	V	×
		r, r1	2	r, CY ← r + r1 + CY	×	×	×	V	×
		A, saddr	2	A, CY ← A + (saddr) + CY	×	×	×	V	×
		A, sfr	3	A, CY ← A + sfr + CY	×	×	×	V	×
		saddr, saddr	3	(saddr), CY ← (saddr) + (saddr) + CY	×	×	×	V	×
		A, mem	2 - 4	A, CY ← A + (mem) + CY	×	×	×	V	×
		mem, A	2 - 4	(mem), CY ← (mem) + A + CY	×	×	×	V	×

Instruction set	Mnemonic	Operand	Byte	Operation	Flag				
					S	Z	AC	P/V	CY
8-bit arithmetic/ logical	SUB	A, #byte	2	A, CY ← A – byte	×	×	×	V	×
		saddr, #byte	3	(saddr), CY ← (saddr) – byte	×	×	×	V	×
		sfr, #byte	4	sfr, CY ← sfr – byte	×	×	×	V	×
		r, r1	2	r, CY ← r – r1	×	×	×	V	×
		A, saddr	2	A, CY ← A – (saddr)	×	×	×	V	×
		A, sfr	3	A, CY ← A – sfr	×	×	×	V	×
		saddr, saddr	3	(saddr), CY ← (saddr) – (saddr)	×	×	×	V	×
		A, mem	2 - 4	A, CY ← A – (mem)	×	×	×	V	×
		mem, A	2 - 4	(mem), CY ← (mem) – A	×	×	×	V	×
	SUBC	A, #byte	2	A, CY ← A – byte – CY	×	×	×	V	×
		saddr, #byte	3	(saddr), CY ← (saddr) – byte – CY	×	×	×	V	×
		sfr, #byte	4	sfr, CY ← sfr – byte – CY	×	×	×	V	×
		r, r1	2	r, CY ← r – r1 – CY	×	×	×	V	×
		A, saddr	2	A, CY ← A – (saddr) – CY	×	×	×	V	×
		A, sfr	3	A, CY ← A – sfr – CY	×	×	×	V	×
		saddr, saddr	3	(saddr), CY ← (saddr) – (saddr) – CY	×	×	×	V	×
		A, mem	2 - 4	A, CY ← A – (mem) – CY	×	×	×	V	×
		mem, A	2 - 4	(mem), CY ← (mem) – A – CY	×	×	×	V	×
	AND	A, #byte	2	A ← A ∧ byte	×	×		P	
		saddr, #byte	3	(saddr) ← (saddr) ∧ byte	×	×		P	
		sfr, #byte	4	sfr ← sfr ∧ byte	×	×		P	
		r, r1	2	r ← r ∧ r1	×	×		P	
		A, saddr	2	A ← A ∧ (saddr)	×	×		P	
		A, sfr	3	A ← A ∧ sfr	×	×		P	
		saddr, saddr	3	(saddr) ← (saddr) ∧ (saddr)	×	×		P	
		A, mem	2 - 4	A ← A ∧ (mem)	×	×		P	
		mem, A	2 - 4	(mem) ← (mem) ∧ A	×	×		P	

Instruction set	Mnemonic	Operand	Byte	Operation	Flag S	Z	AC	P/V	CY
8-bit arithmetic/ logical	OR	A, #byte	2	A ← A ∨ byte	×	×		P	
		saddr, #byte	3	(saddr) ← (saddr) ∨ byte	×	×		P	
		sfr, #byte	4	sfr ← sfr ∨ byte	×	×		P	
		r, r1	2	r ← r ∨ r1	×	×		P	
		A, saddr	2	A ← A ∨ (saddr)	×	×		P	
		A, sfr	3	A ← A ∨ sfr	×	×		P	
		saddr, saddr	3	(saddr) ← (saddr) ∨ (saddr)	×	×		P	
		A, mem	2 - 4	A ← A ∨ (mem)	×	×		P	
		mem, A	2 - 4	(mem) ← (mem) ∨ A	×	×		P	
	XOR	A, #byte	2	A ← A ⊻ byte	×	×		P	
		saddr, #byte	3	(saddr) ← (saddr) ⊻ byte	×	×		P	
		sfr, #byte	4	sfr ← sfr ⊻ byte	×	×		P	
		r, r1	2	r ← r ⊻ r1	×	×		P	
		A, saddr	2	A ← A ⊻ (saddr)	×	×		P	
		A, sfr	3	A ← A ⊻ sfr	×	×		P	
		saddr, saddr	3	(saddr) ← (saddr) ⊻ (saddr)	×	×		P	
		A, mem	2 - 4	A ← A ⊻ (mem)	×	×		P	
		mem, A	2 - 4	(mem) ← (mem) ⊻ A	×	×		P	
	CMP	A, #byte	2	A – byte	×	×	×	V	×
		saddr, #byte	3	(saddr) – byte	×	×	×	V	×
		sfr, #byte	4	sfr – byte	×	×	×	V	×
		r, r1	2	r – r1	×	×	×	V	×
		A, saddr	2	A – (saddr)	×	×	×	V	×
		A, sfr	3	A – sfr	×	×	×	V	×
		saddr, saddr	3	(saddr) – (saddr)	×	×	×	V	×
		A, mem	2 - 4	A – (mem)	×	×	×	V	×
		mem, A	2 - 4	(mem) – A	×	×	×	V	×

Instruction set	Mnemonic	Operand	Byte	Operation	Flag				
					S	Z	AC	P/V	CY
16-bit arithmetic/ logical	**ADDW**	AX, #word	3	AX, CY ← AX + word	×	×	×	V	×
		saddrp, #word	4	(saddrp), CY ← (saddrp) + word	×	×	×	V	×
		sfrp, #word	5	sfrp, CY ← sfrp + word	×	×	×	V	×
		rp, rp1	2	rp, CY ← rp + rp1	×	×	×	V	×
		AX, saddrp	2	AX, CY ← AX + (saddrp)	×	×	×	V	×
		AX, sfrp	3	AX, CY ← AX + sfrp	×	×	×	V	×
		saddrp, saddrp	3	(saddrp), CY ← (saddrp) + (saddrp)	×	×	×	V	×
	SUBW	AX, #word	3	AX, CY ← AX – word	×	×	×	V	×
		saddrp, #word	4	(saddrp), CY ← (saddrp) – word	×	×	×	V	×
		sfrp, #word	5	sfrp, CY ← sfrp – word	×	×	×	V	×
		rp, rp1	2	rp, CY ← rp – rp1	×	×	×	V	×
		AX, saddrp	2	AX, CY ← AX – (saddrp)	×	×	×	V	×
		AX, sfrp	3	AX, CY ← AX – sfrp	×	×	×	V	×
		saddrp, saddrp	3	(saddrp), CY ← (saddrp) – (saddrp)	×	×	×	V	×
	CMPW	AX, #word	3	AX – word	×	×	×	V	×
		saddrp, #word	4	(saddrp) – word	×	×	×	V	×
		sfrp, #word	5	sfrp – word	×	×	×	V	×
		rp, rp1	2	rp – rp1	×	×	×	V	×
		AX, saddrp	2	AX – (saddrp)	×	×	×	V	×
		AX, sfrp	3	AX – sfrp	×	×	×	V	×
		saddrp, saddrp	3	(saddrp) – (saddrp)	×	×	×	V	×
Multiply/divide	**MULU**	r1	2	AX ← A × r1					
	DIVUW	r1	2	AX (quotient), r1 (remainder) ← AX ÷ r1					
	MULUW	rp1	2	AX (high-order 16 bits), rp1 (low-order 16 bits) ← AX × rp1					
	DIVUX	rp1	2	AXDE (quotient), rp1 (remainder) ← AXDE ÷ rp1					
Signed multiply	**MULW**	rp1	2	AX (high-order 16 bits), rp1 (low-order 16 bits) ← AX × rp1					

Instruction set	Mnemonic	Operand	Byte	Operation	Flag				
					S	Z	AC	P/V	CY
Increment/ decrement	**INC**	r1	1	r1 ← r1 + 1	×	×	×	V	
		saddr	2	(saddr) ← (saddr) + 1	×	×	×	V	
	DEC	r1	1	r1 ← r1 − 1	×	×	×	V	
		saddr	2	(saddr) ← (saddr) − 1	×	×	×	V	
	INCW	rp2	1	rp2 ← rp2 + 1					
		saddrp	3	(saddrp) ← (saddrp) + 1					
	DECW	rp2	1	rp2 ← rp2 − 1					
		saddrp	3	(saddrp) ← (saddrp) − 1					
Shift/rotate	**ROR**	r1, n	2	(CY, r17 ← r10, r1m−1 ← r1m) × n times				P	×
	ROL	r1, n	2	(CY, r10 ← r17, r1m+1 ← r1m) × n times				P	×
	RORC	r1, n	2	(CY ← r10, r17 ← CY, r1m−1 ← r1m) × n times				P	×
	ROLC	r1, n	2	(CY ← r17, r10 ← CY, r1m+1 ← r1m) × n times				P	×
	SHR	r1, n	2	(CY ← r10, r17 ← 0, r1m−1 ← r1m) × n times	×	×	0	P	×
	SHL	r1, n	2	(CY ← r17, r10 ← 0, r1m+1 ← r1m) × n times	×	×	0	P	×
	SHRW	rp1, n	2	(CY ← rp10, rp115 ← 0, rp1m−1 ← rp1m) × n times	×	×	0	P	×
	SHLW	rp1, n	2	(CY ← rp1115, rp10 ← 0, rp1m+1 ← rp1m) × n times	×	×	0	P	×
	ROR4	[rp1]	2	A3–0 ← (rp1)3–0, (rp1)7–4 ← A3–0, (rp1)3–0 ← (rp1)7–4					
	ROL4	[rp1]	2	A3–0 ← (rp1)7–4, (rp1)3–0 ← A3–0, (rp1)7–4 ← (rp1)3–0					
BCD adjust	**ADJBA**		2	Decimal adjust accumulator	×	×	0	P	×
	ADJBS								
Data conversion	**CVTBW**		1	When A7 = 0, X ← A, A ← 00H When A7 = 1, X ← A, A ← FFH					

Remark n in the shift/rotate instructions indicates the number of shifts or rotations.

Instruction set	Mnemonic	Operand	Byte	Operation	Flag S Z AC P/V CY
Bit manipulation	MOV1	CY, saddr.bit	3	CY ← (saddr.bit)	×
		CY, sfr.bit	3	CY ← sfr.bit	×
		CY, A.bit	2	CY ← A.bit	×
		CY, X.bit	2	CY ← X.bit	×
		CY, PSWH.bit	2	CY ← PSWH.bit	×
		CY, PSWL.bit	2	CY ← PSWL.bit	×
		saddr.bit, CY	3	(saddr.bit) ← CY	
		sfr.bit, CY	3	sfr.bit ← CY	
		A.bit, CY	2	A.bit ← CY	
		X.bit, CY	2	X.bit ← CY	
		PSWH.bit, CY	2	PSWH.bit ← CY	
		PSWL.bit, CY	2	PSWL.bit ← CY	
	AND1	CY, saddr.bit	3	CY ← CY ∧ (saddr.bit)	×
		CY, /saddr.bit	3	CY ← CY ∧ $\overline{(saddr.bit)}$	×
		CY, sfr.bit	3	CY ← CY ∧ sfr.bit	×
		CY, /sfr.bit	3	CY ← CY ∧ $\overline{sfr.bit}$	×
		CY, A.bit	2	CY ← CY ∧ A.bit	×
		CY, /A.bit	2	CY ← CY ∧ $\overline{A.bit}$	×
		CY, X.bit	2	CY ← CY ∧ X.bit	×
		CY, /X.bit	2	CY ← CY ∧ $\overline{X.bit}$	×
		CY, PSWH.bit	2	CY ← CY ∧ PSWH.bit	×
		CY, /PSWH.bit	2	CY ← CY ∧ $\overline{PSWH.bit}$	×
		CY, PSWL.bit	2	CY ← CY ∧ PSWL.bit	×
		CY, /PSWL.bit	2	CY ← CY ∧ $\overline{PSWL.bit}$	×
	OR1	CY, saddr.bit	3	CY ← CY ∨ (saddr.bit)	×
		CY, /saddr.bit	3	CY ← CY ∨ $\overline{(saddr.bit)}$	×
		CY, sfr.bit	3	CY ← CY ∨ sfr.bit	×
		CY, /sfr.bit	3	CY ← CY ∨ $\overline{sfr.bit}$	×
		CY, A.bit	2	CY ← CY ∨ A.bit	×
		CY, /A.bit	2	CY ← CY ∨ $\overline{A.bit}$	×
		CY, X.bit	2	CY ← CY ∨ X.bit	×
		CY, /X.bit	2	CY ← CY ∨ $\overline{X.bit}$	×
		CY, PSWH.bit	2	CY ← CY ∨ PSWH.bit	×
		CY, /PSWH.bit	2	CY ← CY ∨ $\overline{PSWH.bit}$	×
		CY, PSWL.bit	2	CY ← CY ∨ PSWL.bit	×
		CY, /PSWL.bit	2	CY ← CY ∨ $\overline{PSWL.bit}$	×

Instruction set	Mnemonic	Operand	Byte	Operation	Flag S	Z	AC	P/V	CY
Bit manipulation	XOR1	CY, saddr.bit	3	CY ← CY ∀ (saddr.bit)					×
		CY, sfr.bit	3	CY ← CY ∀ sfr.bit					×
		CY, A.bit	2	CY ← CY ∀ A.bit					×
		CY, X.bit	2	CY ← CY ∀ X.bit					×
		CY, PSWH.bit	2	CY ← CY ∀ PSW$_H$.bit					×
		CY, PSWL.bit	2	CY ← CY ∀ PSW$_L$.bit					×
	SET1	saddr.bit	2	(saddr.bit) ← 1					
		sfr.bit	3	sfr.bit ← 1					
		A.bit	2	A.bit ← 1					
		X.bit	2	X.bit ← 1					
		PSWH.bit	2	PSW$_H$.bit ← 1					
		PSWL.bit	2	PSW$_L$.bit ← 1	×	×	×	×	×
	CLR1	saddr.bit	2	(saddr.bit) ← 0					
		sfr.bit	3	sfr.bit ← 0					
		A.bit	2	A.bit ← 0					
		X.bit	2	X.bit ← 0					
		PSWH.bit	2	PSW$_H$.bit ← 0					
		PSWL.bit	2	PSW$_L$.bit ← 0	×	×	×	×	×
	NOT1	saddr.bit	3	(saddr.bit) ← $\overline{\text{(saddr.bit)}}$					
		sfr.bit	3	sfr.bit ← $\overline{\text{sfr.bit}}$					
		A.bit	2	A.bit ← $\overline{\text{A.bit}}$					
		X.bit	2	X.bit ← $\overline{\text{X.bit}}$					
		PSWH.bit	2	PSW$_H$.bit ← $\overline{\text{PSW}_H\text{.bit}}$					
		PSWL.bit	2	PSW$_L$.bit ← $\overline{\text{PSW}_L\text{.bit}}$	×	×	×	×	×
	SET1	CY	1	CY ← 1					1
	CLR1	CY	1	CY ← 0					0
	NOT1	CY	1	CY ← $\overline{\text{CY}}$					×

Instruction set	Mnemonic	Operand	Byte	Operation	Flag S Z AC P/V CY
Call/return	**CALL**	!addr16	3	(SP–1)←(PC+3)H, (SP–2)←(PC+3)L, PC←addr16, SP←SP–2	
	CALLF	!addr11	2	(SP–1)←(PC+2)H, (SP–2)←(PC+2)L, PC15-11←00001, PC10-0←addr11, SP←SP–2	
	CALLT	[addr5]	1	(SP–1)←(PC+1)H, (SP–2)←(PC+1)L, PCH←(TPF, 00000000, addr5+1), PCL←(TPF, 00000000, addr5), SP←SP–2	
	CALL	rp1	2	(SP–1)←(PC+2)H, (SP–2)←(PC+2)L, PCH←rp1H, PCL←rp1L, SP←SP–2	
		[rp1]	2	(SP–1)←(PC+2)H, (SP–2)←(PC+2)L, PCH←(rp1+1), PCL←(rp1), SP←SP–2	
	BRK		1	(SP–1)←PSWH, (SP–2)←PSWL (SP–3)←(PC+1)H, (SP–4)←(PC+1)L, PCL←(003EH), PCH←(003FH), SP←SP–4, IE←0	
	RET		1	PCL←(SP), PCH←(SP+1), SP←SP+2	
	RETB		1	PCL←(SP), PCH←(SP+1) PSWL←(SP+2), PSWH←(SP+3) SP←SP+4	R R R R R
	RETI		1	PCL←(SP), PCH←(SP+1) PSWL←(SP+2), PSWH←(SP+3) SP←SP+4	R R R R R
Stack manipulation	**PUSH**	sfrp	3	(SP–1)←sfrH, (SP–2)←sfrL, SP←SP–2	
		post	2	{(SP–1)←postH, (SP–2)←postL, SP←SP–2} × n times	
		PSW	1	(SP–1)←PSWH, (SP–2)←PSWL, SP←SP–2	
	PUSHU	post	2	{(UP–1)←postH, (UP–2) postL, UP←UP–2} × n times	
	POP	sfrp	3	sfrL←(SP), sfrH←(SP+1), SP←SP+2	
		post	2	{postL←(SP), postH←(SP+1), SP←SP+2} × n times	
		PSW	1	PSWL←(SP), PSWH←(SP+1), SP←SP+2	R R R R R
	POPU	post	2	{postL←(UP), postH←(UP+1), UP←UP+2} × n times	
	MOVW	SP, #word	4	SP ← word	
		SP, AX	2	SP ← AX	
		AX, SP	2	AX ← SP	
	INCW	SP	2	SP ← SP + 1	
	DECW	SP	2	SP ← SP – 1	

Remark n in the stack manipulation instructions indicates the number of registers specified in post.

Instruction set	Mnemonic	Operand	Byte	Operation	Flag				
					S	Z	AC	P/V	CY
Special	**CHKL**	sfr	3	(Pin level) ⊻ (Signal level before output buffer)	×	×		P	
	CHKLA	sfr	3	A ← (Pin level) ⊻ (Signal level before output buffer)	×	×		P	
Unconditional branch	**BR**	!addr16	3	PC ← addr16					
		rp1	2	PC$_H$ ← rp1$_H$, PC$_L$ ← rp1$_L$					
		[rp1]	2	PC$_H$ ← (rp1 + 1), PC$_L$ ← (rp1)					
		$addr16	2	PC ← PC + 2 + jdisp8					
Conditional branch	**BC**	$addr16	2	PC ← PC + 2 + jdisp8 if CY = 1					
	BL								
	BNC	$addr16	2	PC ← PC + 2 + jdisp8 if CY = 0					
	BNL								
	BZ	$addr16	2	PC ← PC + 2 + jdisp8 if Z = 1					
	BE								
	BNZ	$addr16	2	PC ← PC + 2 + jdisp8 if Z = 0					
	BNE								
	BV	$addr16	2	PC ← PC + 2 + jdisp8 if P/V = 1					
	BPE								
	BNV	$addr16	2	PC ← PC + 2 + jdisp8 if P/V = 0					
	BPO								
	BN	$addr16	2	PC ← PC + 2 + jdisp8 if S = 1					
	BP	$addr16	2	PC ← PC + 2 + jdisp8 if S = 0					
	BGT	$addr16	3	PC ← PC + 3 + jdisp8 if (P/V ⊻ S) ∨ Z = 0					
	BGE	$addr16	3	PC ← PC + 3 + jdisp8 if P/V ⊻ S = 0					
	BLT	$addr16	3	PC ← PC + 3 + jdisp8 if P/V ⊻ S = 1					
	BLE	$addr16	3	PC ← PC + 3 + jdisp8 if (P/V ⊻ S) ∨ Z = 1					
	BH	$addr16	3	PC ← PC + 3 + jdisp8 if Z ∨ CY = 0					
	BNH	$addr16	3	PC ← PC + 3 + jdisp8 if Z ∨ CY = 1					
	BT	saddr.bit, $addr16	3	PC ← PC + 3 + jdisp8 if (saddr.bit) = 1					
		sfr.bit, $addr16	4	PC ← PC + 4 + jdisp8 if sfr.bit = 1					
		A.bit, $addr16	3	PC ← PC + 3 + jdisp8 if A.bit = 1					
		X.bit, $addr16	3	PC ← PC + 3 + jdisp8 if X.bit = 1					
		PSWH.bit, $addr16	3	PC ← PC + 3 + jdisp8 if PSW$_H$.bit = 1					
		PSWL.bit, $addr16	3	PC ← PC + 3 + jdisp8 if PSW$_L$.bit = 1					
	BF	saddr.bit, $addr16	4	PC ← PC + 4 + jdisp8 if (saddr.bit) = 0					
		sfr.bit, $addr16	4	PC ← PC + 4 + jdisp8 if sfr.bit = 0					
		A.bit, $addr16	3	PC ← PC + 3 + jdisp8 if A.bit = 0					
		X.bit, $addr16	3	PC ← PC + 3 + jdisp8 if X.bit = 0					
		PSWH.bit, $addr16	3	PC ← PC + 3 + jdisp8 if PSW$_H$.bit = 0					
		PSWL.bit, $addr16	3	PC ← PC + 3 + jdisp8 if PSW$_L$.bit = 0					

Instruction set	Mnemonic	Operand	Byte	Operation	Flag S	Z	AC	P/V	CY
Conditional branch	**BTCLR**	saddr.bit, $addr16	4	PC ← PC + 4 + jdisp8 if (saddr.bit) = 1 then reset (saddr.bit)					
		sfr.bit, $addr16	4	PC ← PC + 4 + jdisp8 if sfr.bit = 1 then reset sfr.bit					
		A.bit, $addr16	3	PC ← PC + 3 + jdisp8 if A.bit = 1 then reset A.bit					
		X.bit, $addr16	3	PC ← PC + 3 + jdisp8 if X.bit = 1 then reset X.bit					
		PSWH.bit, $addr16	3	PC ← PC + 3 + jdisp8 if PSWH.bit = 1 then reset PSWH.bit					
		PSWL.bit, $addr16	3	PC ← PC + 3 + jdisp8 if PSWL.bit = 1 then reset PSWL.bit	×	×	×	×	×
	BFSET	saddr.bit, $addr16	4	PC ← PC + 4 + jdisp8 if (saddr.bit) = 0 then set (saddr.bit)					
		sfr.bit, $addr16	4	PC ← PC + 4 + jdisp8 if sfr.bit = 0 then set sfr.bit					
		A.bit, $addr16	3	PC ← PC + 3 + jdisp8 if A.bit = 0 then set A.bit					
		X.bit, $addr16	3	PC ← PC + 3 + jdisp8 if X.bit = 0 then set X.bit					
		PSWH.bit, $addr16	3	PC ← PC + 3 + jdisp8 if PSWH.bit = 0 then set PSWH.bit					
		PSWL.bit, $addr16	3	PC ← PC + 3 + jdisp8 if PSWL.bit = 0 then set PSWL.bit	×	×	×	×	×
	DBNZ	r2, $addr16	2	r2 ← r2 − 1, then PC ← PC + 2 + jdisp8 if r2 ≠ 0					
		saddr, $addr16	3	(saddr) ← (saddr) − 1, then PC ← PC + 3 + jdisp8 if (saddr) ≠ 0					
Context switching	**BRKCS**	RBn	2	PCH ↔ R5, PCL ↔ R4, R7 ← PSWH, R6 ← PSWL, RBS2 − 0 ← n, RSS ← 0, IE ← 0					
	RETCS	!addr16	3	PCH ← R5, PCL ← R4, R5, R4 ← addr16, PSWH ← R7, PSWL ← R6	R	R	R	R	R
	RETCSB	!addr16	4	PCH ← R5, PCL ← R4, R5, R4 ← addr16, PSWH ← R7, PSWL ← R6	R	R	R	R	R

Instruction set	Mnemonic	Operand	Byte	Operation	Flag S	Z	AC	P/V	CY
String	**MOVM**	[DE+], A	2	(DE+) ← A, C ← C – 1 End if C = 0					
		[DE–], A	2	(DE–) ← A, C ← C – 1 End if C = 0					
	MOVBK	[DE+], [HL+]	2	(DE+) ← (HL+), C ← C – 1 End if C = 0					
		[DE–], [HL–]	2	(DE–) ← (HL–), C ← C – 1 End if C = 0					
	XCHM	[DE+], A	2	(DE+) ↔ A, C ← C – 1 End if C = 0					
		[DE–], A	2	(DE–) ↔ A, C ← C – 1 End if C = 0					
	XCHBK	[DE+], [HL+]	2	(DE+) ↔ (HL+), C ← C – 1 End if C = 0					
		[DE–], [HL–]	2	(DE–) ↔ (HL–), C ← C – 1 End if C = 0					
	CMPME	[DE+], A	2	(DE+) – A, C ← C – 1 End if C = 0 or Z = 0	×	×	×	V	×
		[DE–], A	2	(DE–) – A, C ← C – 1 End if C = 0 or Z = 0	×	×	×	V	×
	CMPBKE	[DE+], [HL+]	2	(DE+) – (HL+), C ← C – 1 End if C = 0 or Z = 0	×	×	×	V	×
		[DE–], [HL–]	2	(DE–) – (HL–), C ← C – 1 End if C = 0 or Z = 0	×	×	×	V	×
	CMPMNE	[DE+], A	2	(DE+) – A, C ← C – 1 End if C = 0 or Z = 1	×	×	×	V	×
		[DE–], A	2	(DE–) – A, C ← C – 1 End if C = 0 or Z = 1	×	×	×	V	×
	CMPBKNE	[DE+], [HL+]	2	(DE+) – (HL+), C ← C – 1 End if C = 0 or Z = 1	×	×	×	V	×
		[DE–], [HL–]	2	(DE–) – (HL–), C ← C – 1 End if C = 0 or Z = 1	×	×	×	V	×
	CMPMC	[DE+], A	2	(DE+) – A, C ← C – 1 End if C = 0 or CY = 0	×	×	×	V	×
		[DE–], A	2	(DE–) – A, C ← C – 1 End if C = 0 or CY = 0	×	×	×	V	×
	CMPBKC	[DE+], [HL+]	2	(DE+) – (HL+), C ← C – 1 End if C = 0 or CY = 0	×	×	×	V	×
		[DE–], [HL–]	2	(DE–) – (HL–), C ← C – 1 End if C = 0 or CY = 0	×	×	×	V	×

Instruction set	Mnemonic	Operand	Byte	Operation	Flag				
					S	Z	AC	P/V	CY
String	**CMPMNC**	[DE+], A	2	(DE+) – A, C ← C – 1 End if C = 0 or CY = 1	×	×	×	V	×
		[DE–], A	2	(DE–) – A, C ← C – 1 End if C = 0 or CY =1	×	×	×	V	×
	CMPBKNC	[DE+], [HL+]	2	(DE+) – (HL+), C ← C – 1 End if C = 0 or CY = 1	×	×	×	V	×
		[DE–], [HL–]	2	(DE–) – (HL–), C ← C – 1 End if C = 0 or CY = 1	×	×	×	V	×
CPU control	**MOV**	STBC, #byte	4	STBC ← byte[Note]					
		WDM, #byte	4	WDM ← byte[Note]					
	SWRS		1	RSS ← $\overline{RSS}$					
	SEL	RBn	2	RBS2-0 ← n, RSS ← 0					
		RBn, ALT	2	RBS2-0 ← n, RSS ← 1					
	NOP		1	No operation					
	EI		1	IE ← 1 (Enable interrupt)					
	DI		1	IE ← 0 (Disable interrupt)					

Note An op-code trap interrupt occurs if an invalid op-code is specified in an STBC or WDM register manipulation instruction.

Trap operation: (SP – 1) ← PSW_H, (SP – 2) ← PSW_L,
(SP – 3) ← $(PC - 4)_H$, (SP – 4) ← $(PC - 4)_L$,
PC_L ← (003CH), PC_H ← (003DH),
SP ← SP – 4, IE ← 0

10. ELECTRICAL CHARACTERISTICS

ABSOLUTE MAXIMUM RATINGS (T_A = 25 °C)

Parameter	Symbol	Conditions		Rating	Unit	
Supply voltage	V_{DD}			−0.5 to +7.0	V	
	AV_{DD}			−0.5 to V_{DD}+0.5	V	
	AV_{SS}			−0.5 to +0.5	V	
Input voltage	V_I	Note 1		−0.5 to V_{DD}+0.5	V	
Output voltage	V_O			−0.5 to V_{DD}+0.5	V	
Low-level output current	I_{OL}	Each pin		4.0	mA	
		Total of all output pins		90	mA	
High-level output current	I_{OH}	Each pin		−1.0	mA	
		Total of all output pins		−20	mA	
Analog input voltage	V_{IAN}	Note 2	AV_{DD} > V_{DD}	−0.5 to V_{DD}+0.5	V	
			$V_{DD} \geqq AV_{DD}$	−0.5 to AV_{DD}+0.5		
A/D converter reference input voltage	AV_{REF}		AV_{DD} > V_{DD}	−0.5 to V_{DD}+0.5	V	★
			$V_{DD} \geqq AV_{DD}$	−0.5 to AV_{DD}+0.5		★
Operating ambient temperature	T_A			−10 to +70	°C	
Storage temperature	T_{stg}			−65 to +150	°C	

Notes 1. Pins other than those listed below

2. P70/ANI0 - P77/ANI7 and P80/ANI8 - P87/ANI15

Caution **Absolute maximum ratings are rated values beyond which some physical damages may be caused to the product; if any of the parameters in the table above exceeds its rated value even for a moment, the quality of the product may deteriorate. Be sure to use the product within the rated values.** ★

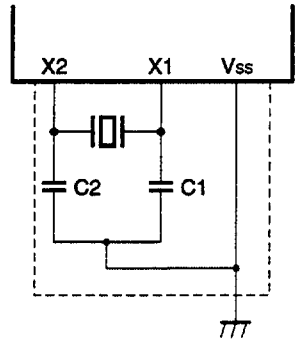
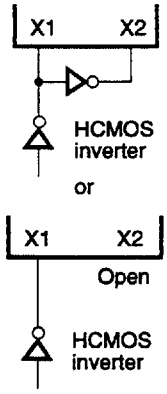
RECOMMENDED OPERATING CONDITIONS

Oscillator frequency	T_A	V_{DD}
8 MHz ≦ fxx ≦ 16 MHz	−10 to +70 °C	+5.0 V ±10 %

CAPACITANCE (T_A = 25 °C, V_{SS} = V_{DD} = 0 V)

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Input capacitance	C_I	f = 1 MHz 0 V on pins other than measured pins			20	pF
Output capacitance	C_O				20	pF
I/O capacitance	C_{IO}				20	pF

OSCILLATOR CHARACTERISTICS (T_A = −10 to +70 °C, V_{DD} = +5 V ±10 %, V_{SS} = 0 V)

Oscillator	Recommended circuit	Parameter	Min.	Max.	Unit
Ceramic or crystal oscillator		Oscillator frequency (f_{XX})	8	16	MHz
External clock		X1 input frequency (f_X)	8	16	MHz
		X1 input rise and fall times (t_{XR}, t_{XF})	0	20	ns
		X1 input high- and low-level widths (t_{WXH}, t_{WXL})	25	80	ns

★ **Caution When using the system clock generator, run wires enclosed in broken lines ( ) according to the following rules to avoid effects such as stray capacitance:**

- **Minimize the wiring.**
- **Never cause the wires to cross other signal lines or run near a line carrying a large varying current.**
- **Cause the grounding point of the capacitor of the oscillator circuit to have the same potential as Vss. Never connect the capacitor to a ground pattern carrying a large current.**
- **Never extract a signal from the oscillator.**

DC CHARACTERISTICS (T_A = −10 to +70 °C, V_{DD} = +5 V ±10 %, V_{SS} = 0 V)

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Low-level input voltage	V_{IL}		0		0.8	V
High-level input voltage	V_{IH1}	Note 1	2.2			V
	V_{IH2}	Note 2	0.8V_{DD}			
Low-level output voltage	V_{OL}	I_{OL} = 2.0 mA			0.45	V
High-level output voltage	V_{OH}	I_{OH} = −400 μA	V_{DD} − 1.0			V
Input leakage current	I_{LI}	0 V ≦ V_I ≦ V_{DD}			±10	μA
Analog pin input leakage current	I_{LIAN}	Note 3, 0 V ≦ V_I ≦ AV_{REF}			±1	μA ★
Output leakage current	I_{LO}	0 V ≦ V_O ≦ V_{DD}			±10	μA
V_{DD} supply current	I_{DD1}	Operation mode		55	80	mA
	I_{DD2}	HALT mode		25	45	mA
Data retention voltage	V_{DDDR}	STOP mode	2.5			V
Data retention current	I_{DDDR}	STOP mode, V_{DDDR} = 2.5 V		2	15	μA
		STOP mode, V_{DDDR} = 5.0 V ±10 %		10	50	μA
Inverted SBF voltage	V_{DDFR}		0.9	1.4	2.4	V

Notes 1. Pins other than those listed below

2. $\overline{\text{RESET}}$, X1, X2, P20/NMI, P21/INTP0, P22/INTP1, P23/INTP2, P24/INTP3, P25/INTP4, P26/INTP5, P27/INTP6/TI, P32/SO/SB0, P33/SI/SB1, and P34/$\overline{\text{SCK}}$

3. P70 - P77, P80 - P87 (pins used as analog inputs only when operating during non-sampling)

AC CHARACTERISTICS (T_A = −10 to +70 °C, V_{DD} = +5 V ±10 %, V_{SS} = 0 V, C_L = 100 pF)

Discontinuous Read/Write Operation (When the General Memory is Connected)

Parameter	Symbol	Conditions	Min.	Max.	Unit
System clock cycle time	t_{CYK}		125	250	ns
Address setup time (referred to ASTB ↓)	t_{SAST}		22		ns
Address hold time (referred to ASTB ↓)	t_{HSTA}		32		ns
Delay from address to $\overline{RD}$ ↓	t_{DAR}		85		ns
Address float time (referred to $\overline{RD}$ ↓)	t_{FRA}			10	ns
Delay from address to data input	t_{DAID}			222	ns
Delay from $\overline{RD}$ ↓ to data input	t_{DRID}			112	ns
Delay from ASTB ↓ to $\overline{RD}$ ↓	t_{DSTR}		42		ns
Data hold time (referred to $\overline{RD}$ ↑)	t_{HRID}		0		ns
Delay from $\overline{RD}$ ↑ to address active	t_{DRA}		50		ns
$\overline{RD}$ low-level width	t_{WRL}		157		ns
ASTB high-level width	t_{WSTH}		37		ns
Delay from address to $\overline{WR}$ ↓	t_{DAW}		85		ns
Delay from ASTB ↓ to data output	t_{DSTOD}			102	ns
Delay from $\overline{WR}$ ↓ to data output	t_{DWOD}			40	ns
Delay from ASTB ↓ to $\overline{WR}$ ↓	t_{DSTW}		42		ns
Data setup time (referred to $\overline{WR}$ ↑)	t_{SODW}		137		ns
Data hold time (referred to $\overline{WR}$ ↑)	t_{HWOD}		22		ns
Delay from $\overline{WR}$ ↑ to ASTB ↑	t_{DWST}		42		ns
$\overline{WR}$ low-level width	t_{WWL}		147		ns
Higher order byte address hold time (referred to $\overline{RD}$ ↑)	t_{HRHA}		27		ns
Higher order byte address hold time (referred to $\overline{WR}$ ↑)	t_{HWHA}		27		ns

Clock

Parameter	Symbol	Conditions	Min.	Max.	Unit
CLKOUT low-level width	t_{WKL}		42		ns
CLKOUT high-level width	t_{WKH}		42		ns
CLKOUT rise time	t_{RK}			15	ns
CLKOUT fall time	t_{FK}			15	ns

t_{CYK}-Dependent Bus Timing Definition

Symbol	Formula	Min./Max.	Unit
t_{SAST}	0.5T – 40	Min.	ns
t_{HSTA}	0.5T – 30	Min.	ns
t_{DAR}	T – 40	Min.	ns
t_{DAID}	(2.5 + n) T – 90	Max.	ns
t_{DRID}	(1.5 + n) T – 75	Max.	ns
t_{DSTR}	0.5T – 20	Min.	ns
t_{DRA}	0.5T – 12	Min.	ns
t_{WRL}	(1.5 + n) T – 30	Min.	ns
t_{WSTH}	0.5T – 25	Min.	ns
t_{DAW}	T – 40	Min.	ns
t_{DSTOD}	0.5T + 40	Max.	ns
t_{DSTW}	0.5T – 20	Min.	ns
t_{SODW}	1.5T – 50	Min.	ns
t_{HWOD}	0.5T – 40	Min.	ns
t_{DWST}	0.5T – 20	Min.	ns
t_{WWL}	(1.5 + n) T – 40	Min.	ns
t_{HRHA}	0.5T – 35	Min.	ns
t_{HWHA}	0.5T – 35	Min.	ns
t_{WKL}	0.5T – 20	Min.	ns
t_{WKH}	0.5T – 20	Min.	ns

Remarks 1. T = t_{CYK} = 1/f_{CLK} (f_{CLK} is the internal system clock frequency.)

2. n represents the number of wait cycles specified by the user software.

3. The items listed above are dependent on t_{CYK}.

Serial Operation (T_A = −10 to +70 °C, V_{DD} = +5 V ±10 %, V_{SS} = 0 V)

Parameter	Symbol	Conditions		Min.	Max.	Unit
Serial clock cycle time	t_{CYSK}	$\overline{SCK}$ output	Internal, divided by 8	1		μs
		$\overline{SCK}$ input	External clock	1		μs
Serial clock low-level width	t_{WSKL}	$\overline{SCK}$ output	Internal, divided by 8	420		ns
		$\overline{SCK}$ input	External clock	420		ns
Serial clock high-level width	t_{WSKH}	$\overline{SCK}$ output	Internal, divided by 8	420		ns
		$\overline{SCK}$ input	External clock	420		ns
SI setup time (to $\overline{SCK}$ ↑)	t_{SRXSK}			80		ns
SI hold time (to $\overline{SCK}$ ↑)	t_{HSKRX}			80		ns
$\overline{SCK}$ ↓ → SO delay time	t_{DSKTX}	R = 1 kΩ, C = 100 pF			210	ns

t_{CYK}-Dependent Serial Operations

Symbol	Conditions		Formula	Min./Max.	Unit
t_{CYSK}	$\overline{SCK}$ output	Internal, divided by 8	8T	Min.	ns
	$\overline{SCK}$ input	External clock	8T	Min.	ns
t_{WSKL}	$\overline{SCK}$ output	Internal, divided by 8	4T − 80	Min.	ns
	$\overline{SCK}$ input	External clock	4T − 80	Min.	ns
t_{WSKH}	$\overline{SCK}$ output	Internal, divided by 8	4T − 80	Min.	ns
	$\overline{SCK}$ input	External clock	4T − 80	Min.	ns

Remarks 1. T = t_{CYK} = 1/f_{CLK} (f_{CLK} is the internal system clock frequency.)

2. The items listed above are dependent on t_{CYK}.

Other Operations (T_A = −10 to +70 °C, V_{DD} = +5 V ±10 %, V_{SS} = 0 V)

Parameter		Symbol	Conditions	Min.	Max.	Unit	
NMI high/low level width		t_{WNIH}, t_{WNIL}	Excluding analog noise	4		μs	
NMI rise/fall time		t_{RNI}, t_{FNI}		0	400	ns	★
INTP0 high/low level width		t_{WI0H}, t_{WI0L}		1		μs	
INTP1 high/low level width		t_{WI1H}, t_{WI1L}		1		μs	
INTP2 high/low level width		t_{WI2H}, t_{WI2L}		1		μs	
INTP3 high/low level width		t_{WI3H}, t_{WI3L}		1		μs	
INTP4 high/low level width		t_{WI4H}, t_{WI4L}		1		μs	
INTP5 high/low level width		t_{WI5H}, t_{WI5L}		1		μs	
INTP6 high/low level width		t_{WI6H}, t_{WI6L}		1		μs	
$\overline{\text{RESET}}$ high/low level width		t_{WRSH}, t_{WRSL}	Excluding analog noise	3.5		μs	★
TI high/low level width		t_{WTIH}, t_{WTIL}		1		μs	
NMI analog noise width	Eliminated	t_{WNIP}			70	ns	
	Passed			4		μs	
$\overline{\text{RESET}}$ analog noise width	Eliminated	t_{WRSP}			70	ns	
	Passed			3.5		μs	★
V_{DD} rise/fall time		t_{RVD}, t_{FVD}		200		μs	

Other t_{CYK}-Dependent Operations

Symbol	Formula	Min./Max.	Unit
t_{WI0H}	8T	Min.	ns
t_{WI0L}	8T	Min.	ns
t_{WI1H}	8T	Min.	ns
t_{WI1L}	8T	Min.	ns
t_{WI2H}	8T	Min.	ns
t_{WI2L}	8T	Min.	ns
t_{WI3H}	8T	Min.	ns
t_{WI3L}	8T	Min.	ns
t_{WI4H}	8T	Min.	ns
t_{WI4L}	8T	Min.	ns
t_{WI5H}	8T	Min.	ns
t_{WI5L}	8T	Min.	ns
t_{WI6H}	8T	Min.	ns
t_{WI6L}	8T	Min.	ns
t_{WTIH}	8T	Min.	ns
t_{WTIL}	8T	Min.	ns

Remarks 1. T = t_{CYK} = 1/f_{CLK} (f_{CLK} is the internal system clock frequency.)

2. The items listed above are dependent on t_{CYK}.

A/D CONVERTER CHARACTERISTICS

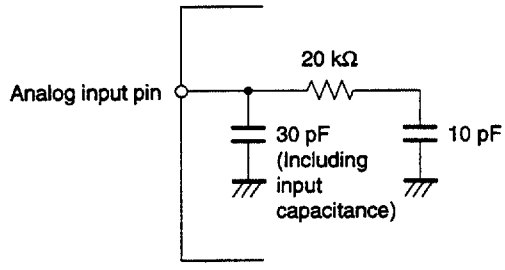
(T_A = −10 to +70 °C, V_{DD} = +5 V ±10 %, V_{SS} = AV_{SS} = 0 V, V_{DD} − 0.5 V ≦ AV_{DD} ≦ V_{DD})

	Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
	Resolution			10			bit
	Total error[Note 1]		4.5 V ≦ AV_{REF} ≦ AV_{DD}			±0.4	%FSR
			3.4 V ≦ AV_{REF} ≦ AV_{DD}			±0.7	%FSR
	Quantization error					±1/2	LSB
	Conversion time	t_{CONV}		144			t_{CYK}
	Sampling time	t_{SAMP}		24			t_{CYK}
	Zero-scale calibration[Note 1]		4.5 V ≦ AV_{REF} ≦ AV_{DD}		±1.5	±2.5	LSB
			3.4 V ≦ AV_{REF} ≦ AV_{DD}		±1.5	±4.5	LSB
	Full scale calibration[Note 1]		4.5 V ≦ AV_{REF} ≦ AV_{DD}		±1.5	±2.5	LSB
			3.4 V ≦ AV_{REF} ≦ AV_{DD}		±1.5	±4.5	LSB
	Nonlinearity calibration[Note 1]		4.5 V ≦ AV_{REF} ≦ AV_{DD}		±1.5	±2.5	LSB
			3.4 V ≦ AV_{REF} ≦ AV_{DD}		±1.5	±4.5	LSB
★	Analog input voltage[Note 2]	V_{IAN}		−0.3		AV_{DD}	V
	Analog input impedance	R_{AN}	Nonsampling		10		MΩ
			Sampling		Note 3		
	Reference voltage	AV_{REF}		3.4		AV_{DD}	V
	AV_{REF} current	AI_{REF}			1.0	3.0	mA
	AV_{DD} supply current	AI_{DD}	Operation mode		2.0	6.0	mA
	A/D converter data retention current	AI_{DDDR}	STOP mode: AV_{DDDR} = 2.5 V		2.0	15	μA
			STOP mode: AV_{DDDR} = 5 V ±10 %		10	50	μA

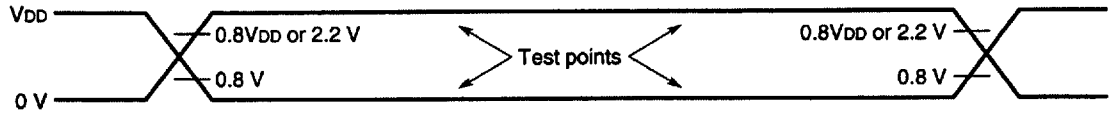
Notes 1. Quantization error is excluded.

2. When −0.3 V ≦ V_{IAN} ≦ 0 V, the conversion result is 00H.
When 0 V ≤ V_{IAN} ≤ AV_{REF}, the voltage is converted with a 10-bit resolution.
When AV_{REF} ≦ V_{IAN} ≦ AV_{DD}, the conversion result is 3FFH.

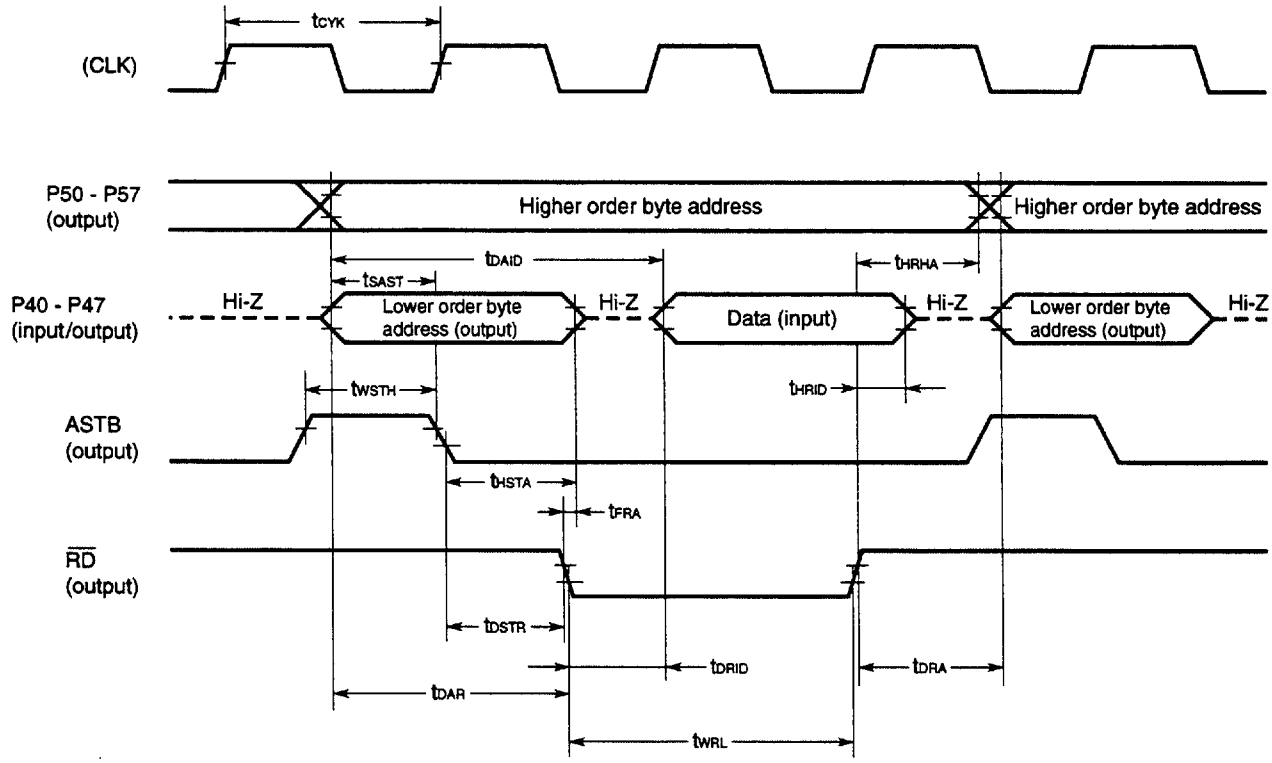
3. During sampling, the analog input impedance is equal to that of the following equivalent circuit.
(The Fig. shows typical values. These values may not be correct for your application.)



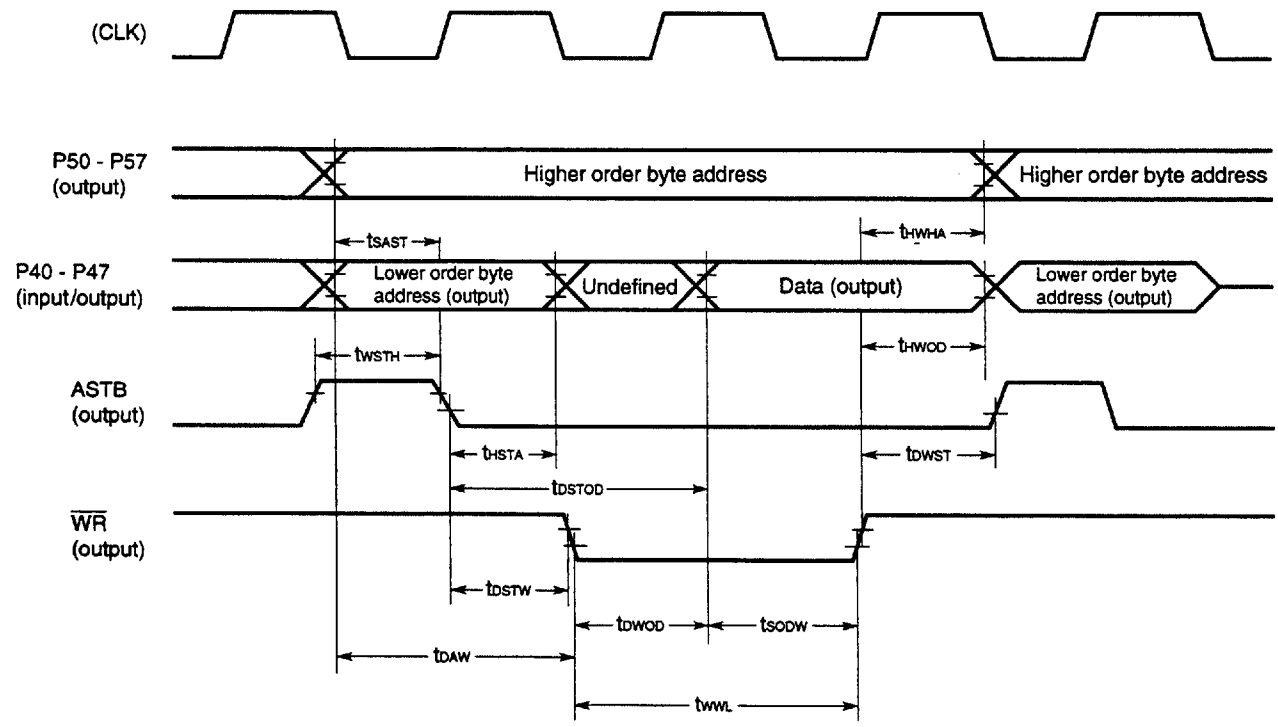
AC Timing Test Points



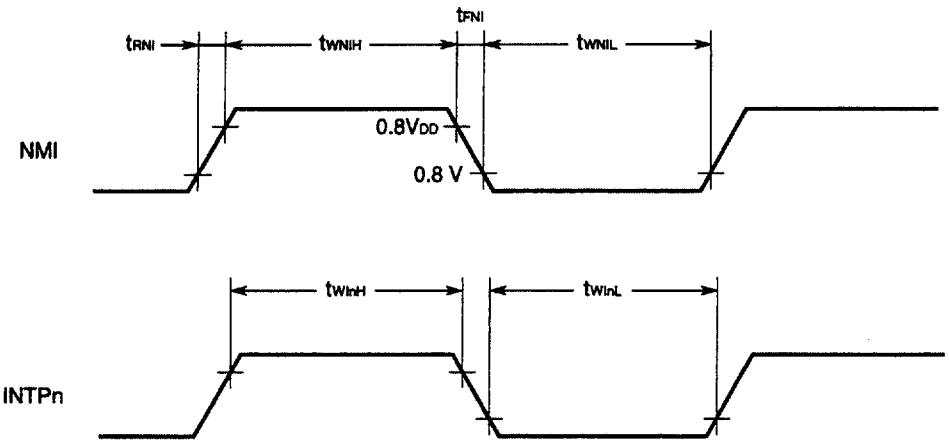
Discontinuous Read Operation



Discontinuous Write Operation

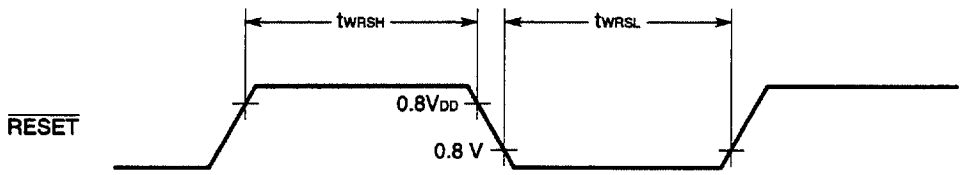


Interrupt Input Timing

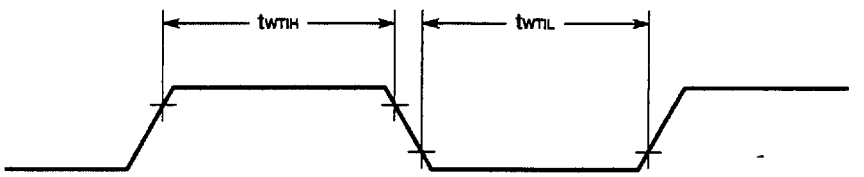


Remark n = 0 to 6

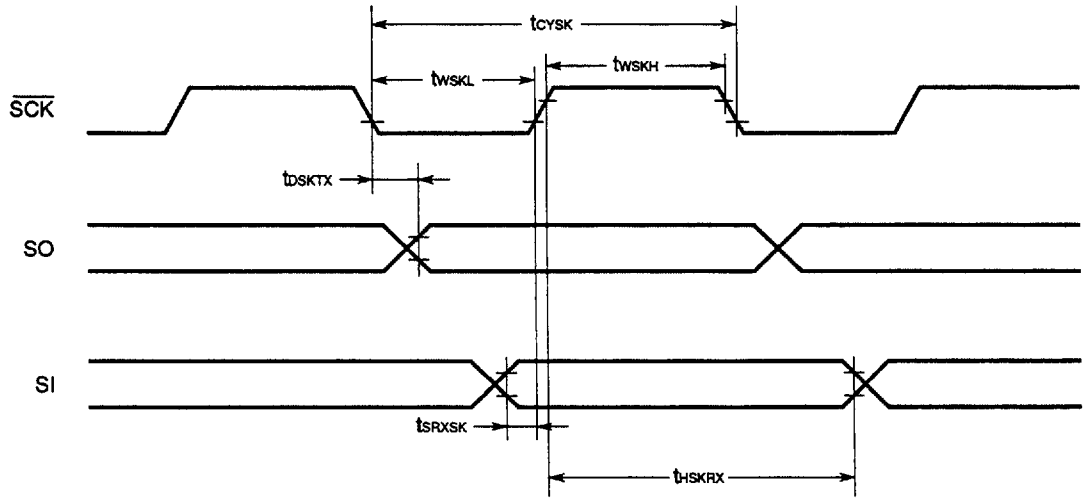
Reset Input Timing



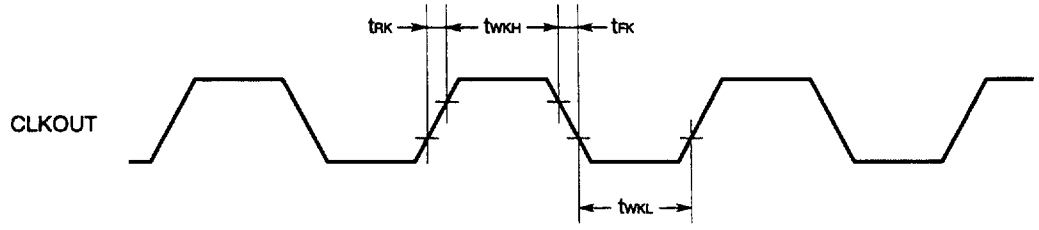
TI Pin Input Timing



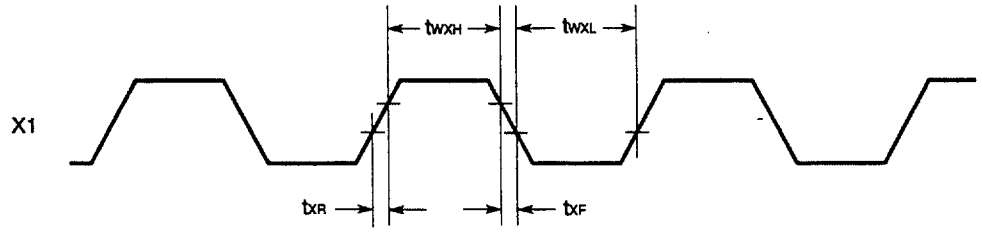
Serial Operation



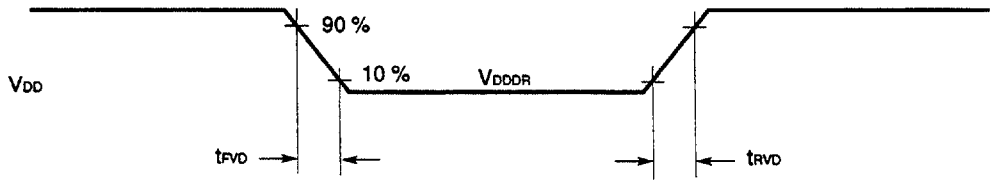
Clock



External clock ★

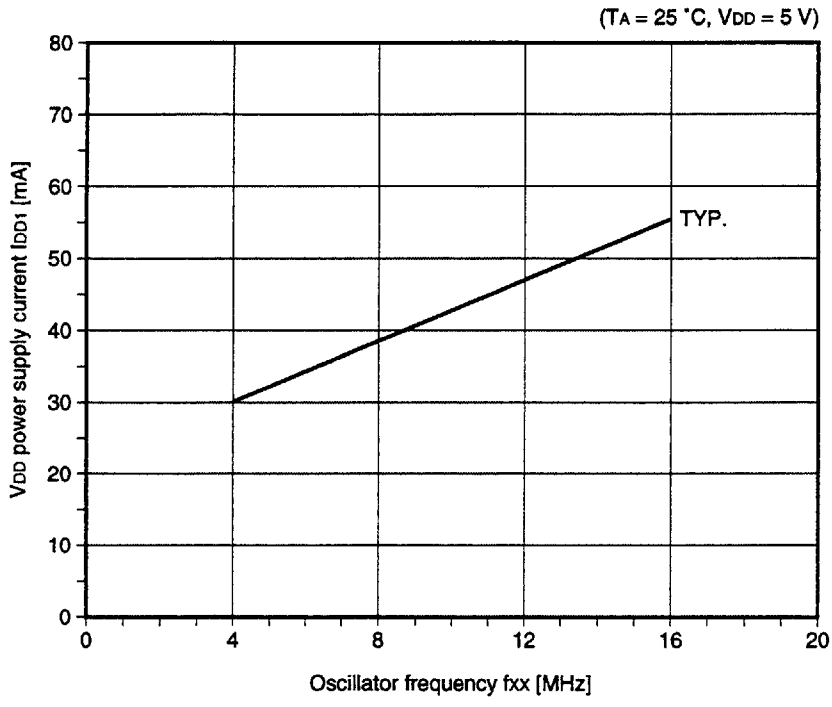


Data hold timing ★

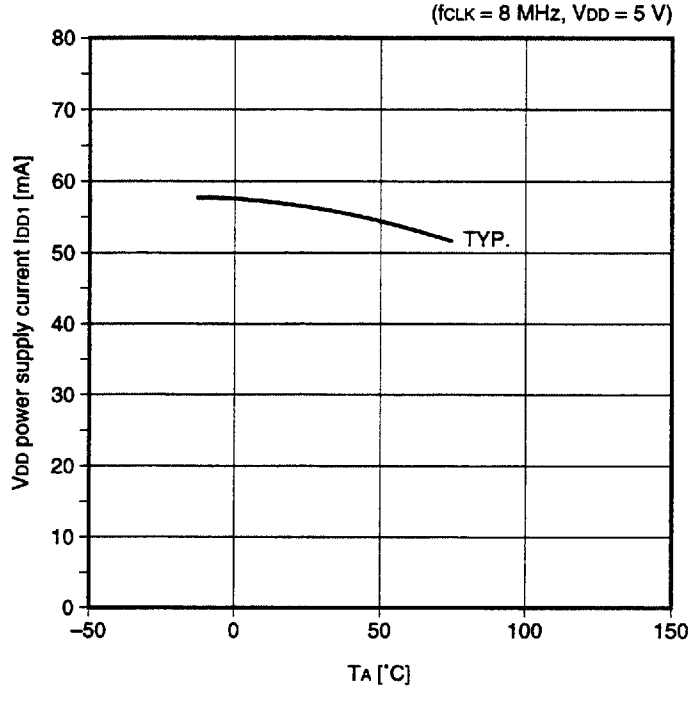


11. CHARACTERISTICS CURVE (REFERENCE)

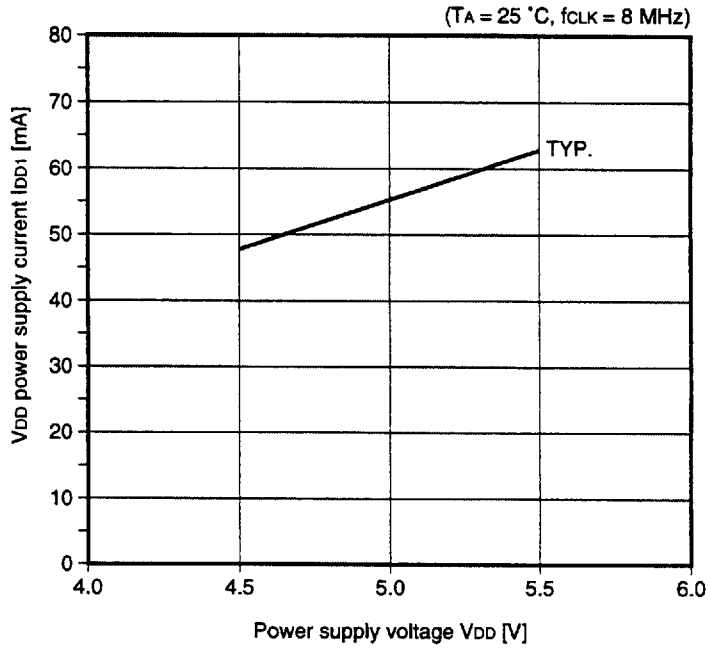
I_{DD1} vs fxx



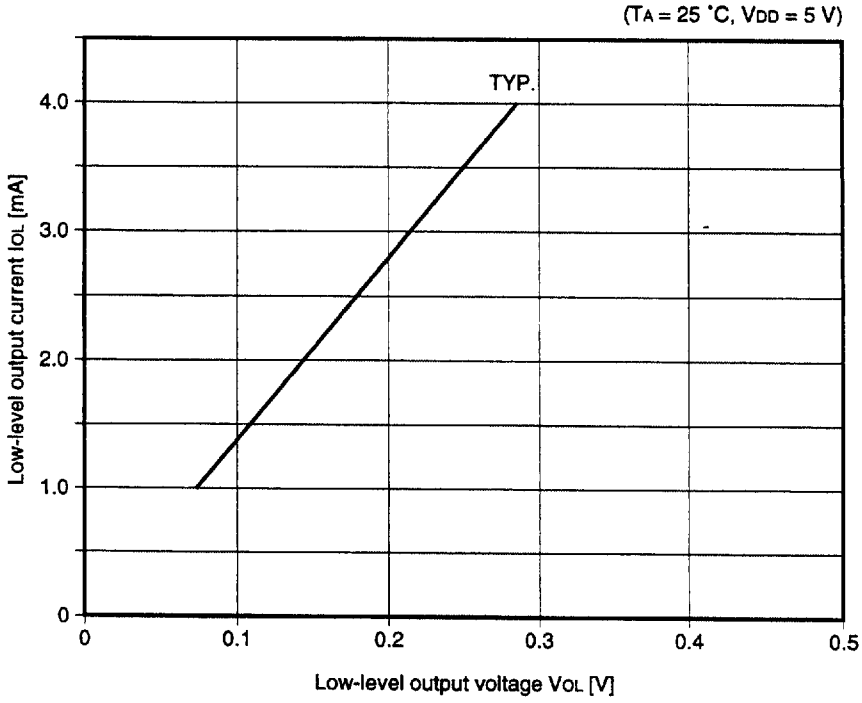
I_{DD1} vs T_A



I_{DD1} vs V_{DD}

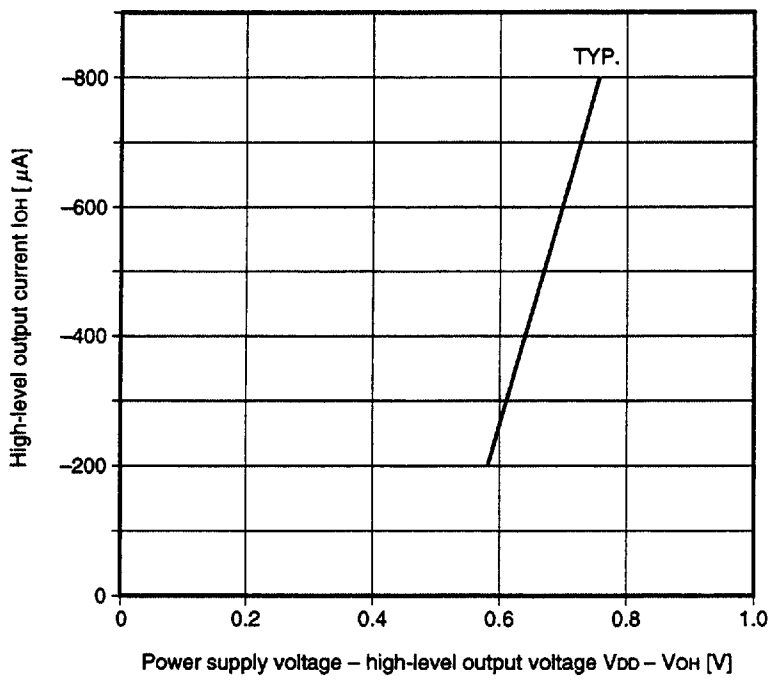


I_{OL} vs V_{OL}



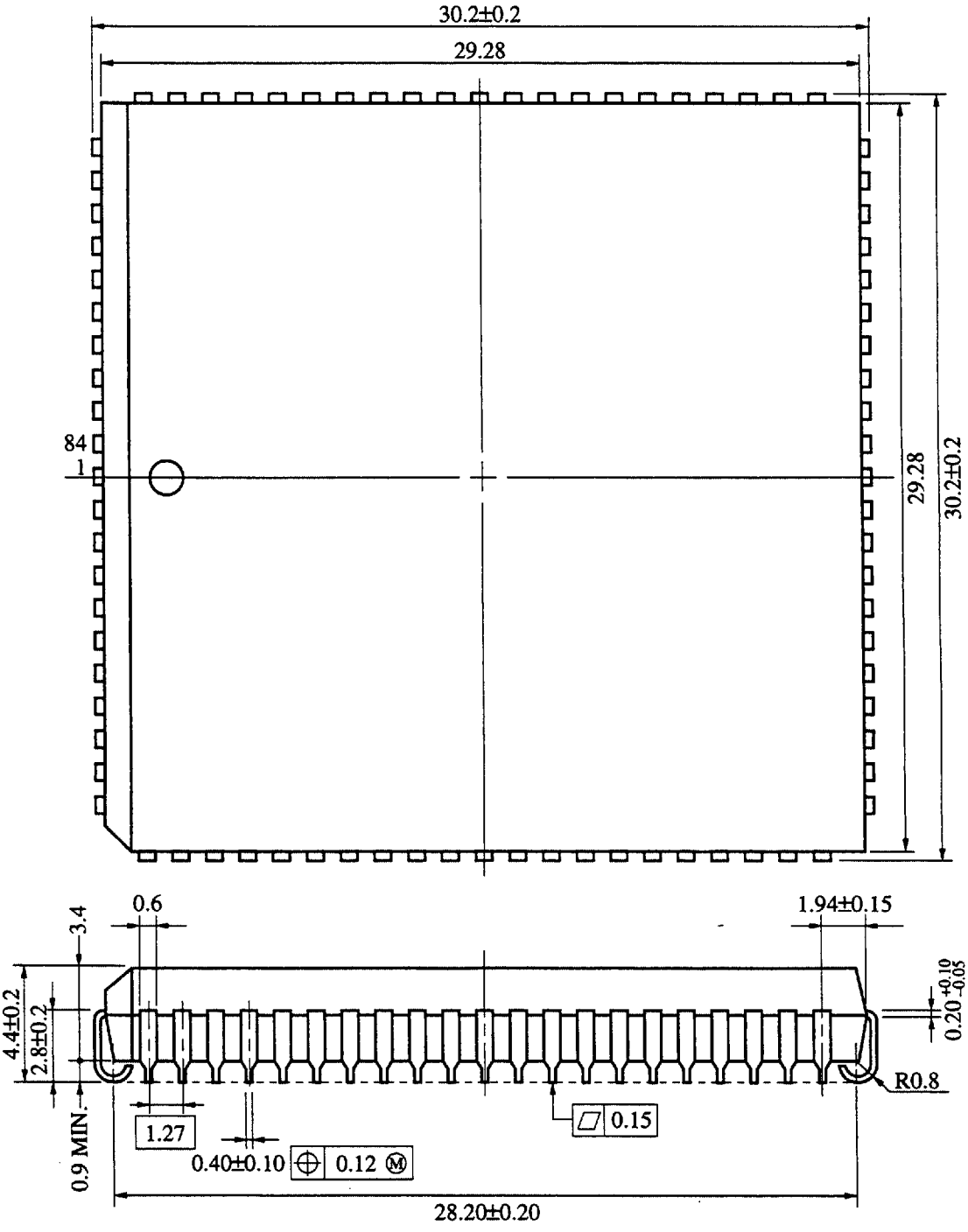
I_{OH} vs (V_{DD} - V_{OH})

(T_A = 25 °C, V_{DD} = 5 V)



12. PACKAGE DIMENSIONS

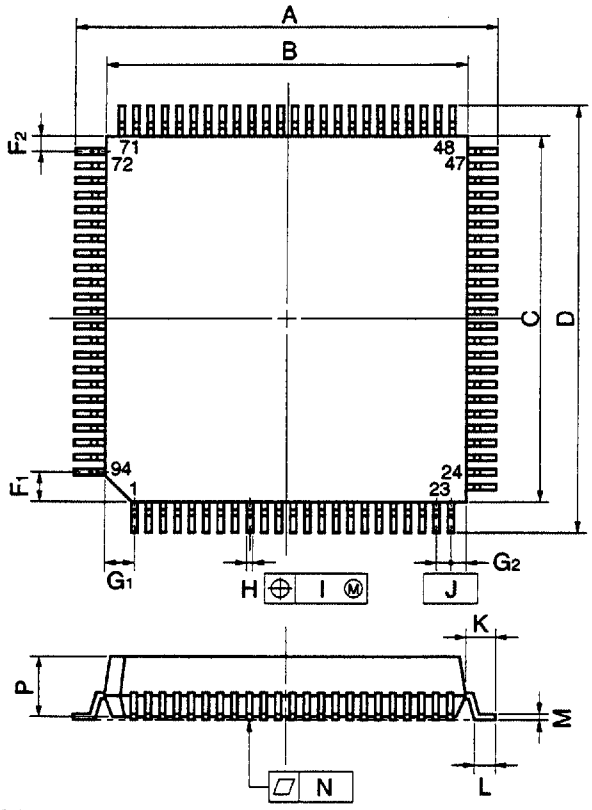
84-pin QFJ (1150 x 1150) (units: mm)



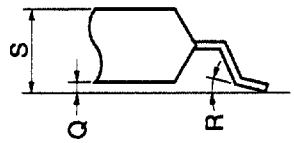
P84L-50A3-2

94-PIN PLASTIC QFP (SQUARE-TYPE 20) (UNITS: mm)

94 PIN PLASTIC QFP (□20)



detail of lead end



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	23.2±0.4	$0.913^{+0.017}_{-0.016}$
B	20.0±0.2	$0.787^{+0.009}_{-0.008}$
C	20.0±0.2	$0.787^{+0.009}_{-0.008}$
D	23.2±0.4	$0.913^{+0.017}_{-0.016}$
F1	1.6	0.063
F2	0.8	0.031
G1	1.6	0.063
G2	0.8	0.031
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	1.6±0.2	0.063±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	3.7	0.146
Q	0.1±0.1	0.004±0.004
R	5°±5°	5°±5°
S	4.0 MAX.	0.158 MAX.

S94GJ-80-5BG-3

13. RECOMMENDED SOLDERING CONDITIONS ★

The following conditions (see table below) must be met when soldering this product.

For the details of the recommended soldering conditions refer to our document *SMT MANUAL* (IE-1207).

Please consult with our sales offices in case other soldering process is used, or in case soldering is done under different conditions.

Table 13-1 Soldering Conditions for Surface-Mount Devices

μPD78330GJ-5BG : 94-pin plastic QFP (20 × 20 mm)
μPD78334GJ-xxx-5BG : 94-pin plastic QFP (20 × 20 mm)
μPD78330LQ : 84-pin plastic QFJ (square-type 1150 mil)
μPD78334LQ-xxx : 84-pin plastic QFJ (square-type 1150 mil)

Soldering process	Soldering conditions	Symbol
Infrared ray reflow	Peak package's surface temperature: 230 °C Reflow time: 30 seconds or less (at 210 °C or more) Number of reflow processes: 1 Exposure limit[Note]: 7 days (10 hours of pre-baking is required at 125 °C afterward.)	IR30-107-1
VPS	Peak package's surface temperature: 215 °C Reflow time: 40 seconds or less (at 200 °C or more) Number of reflow processes: 1 Exposure limit[Note]: 7 days (10 hours of pre-baking is required at 125 °C afterward.)	VP15-107-1
Partial heating method	Terminal temperature: 300 °C or less Heat time: 3 seconds or less (one side per device)	–

Note Exposure limit before soldering after dry-pack package is opened.
Storage conditions: Temperature of 25 °C and maximum relative humidity at 65 % or less

Caution Do not apply more than a single process at once, except for "Partial heating method."

APPENDIX A DIFFERENCES BETWEEN THE μPD78334, μPD78330 AND THE μPD78322, μPD78320

Item \ Product		μPD78334	μPD78330	μPD78322	μPD78320
Number of basic instructions		111			
Minimum instruction execution time		250 ns (when internal clock operates at 8 MHz or when external clock operates at 16 MHz)			
Internal memory	ROM	32768 × 8 bits	–	16384 × 8 bits	–
	RAM	1024 × 8 bits		640 × 8 bits	
Memory space		64K bytes			
I/O line	Input	24 (analog input: 16)		16 (analog input: 8)	
	Output	–			
	I/O	46	28	39	21
Real-time pulse unit		• 18/16-bit timer/counter : 1 18/16-bit compare registers : 5 18/16-bit capture registers : 3 18/16-bit capture/compare registers: 2 • 16-bit timers/counters : 3 16-bit compare registers : 5 16-bit capture register : 1 • Timer outputs : 5 • Programmable pulse outputs : 6		• 18/16-bit free running timer : 1 • 16-bit timer/event counter : 1 • 16-bit compare registers : 6 • 18-bit capture registers : 4 • 18-bit capture/compare registers: 2 • Real-time output ports : 8	
Serial communication interface		• Dedicated baud rate generator : Provided • UART : 1 channel • SBI, • Three-wire serial I/O } : 1 channel			
A/D converter		10-bit resolution, 16 inputs		10-bit resolution, 8 inputs	
Interrupt		• External: 8, internal: 14 (also external: 2) • 3-level programmable priority • Three processing modes: Vectored interrupt function, context switching function, and macro service function			
Test source		Internal: 1			
Instruction set		Much more instructions are added as compared with the μPD78312 and μPD78310.			
Other specifications		• Watchdog timer: Provided • Standby function (STOP/HALT)			
Package		• 84-pin plastic QFJ (square-type 1150 mil) • 94-pin plastic QFP (20 × 20 mm)		• 68-pin plastic QFJ (square-type 950 mil) • 74-pin plastic QFP (20 × 20 mm) • 80-pin plastic QFP (14 × 20 mm)	

APPENDIX B TOOLS ★

B.1 DEVELOPMENT TOOLS

The following tools are provided for developing a system that uses the μPD78334:

Language processor

<table>
<tr><td>78K/III series relocatable assembler (RA78K/III)</td><td>This relocatable program can be used for all 78K/III series emulators. With its macro functions, it allows the user to improve program development efficiency. A structured-programming assembler is also provided, which enables explicit description of program control structures. This assembler could improve productivity in program production and maintenance.
<table>
<tr><th rowspan="2">Host machine</th><th colspan="2">OS</th><th rowspan="2">Distribution media</th><th rowspan="2">Part number</th></tr>
<tr><td>PC-9800 series</td><td>MS-DOS™</td><td>3.5-inch 2HD</td><td>μS5A13RA78K3</td></tr>
<tr><td></td><td></td><td>5.25-inch 2HD</td><td>μS5A10RA78K3</td></tr>
<tr><td>IBM PC/AT™ or compatibles</td><td>PC DOS™</td><td>3.5-inch 2HC</td><td>μS7B13RA78K3</td></tr>
<tr><td></td><td></td><td>5.25-inch 2HC</td><td>μS7B10RA78K3</td></tr>
<tr><td>HP9000 series 700™</td><td>HP-UX™</td><td>DAT</td><td>μS3P16RA78K3</td></tr>
<tr><td>SPARCstation™</td><td>SunOS™</td><td rowspan="2">Cartridge tape (QIC-24)</td><td>μS3K15RA78K3</td></tr>
<tr><td>NEWS™</td><td>NEWS-OS™</td><td>μS3R15RA78K3</td></tr>
</table></td></tr>
<tr><td>78K/III series C compiler (CC78K/III)</td><td>This C compiler can be used for all 78K/III series emulators. The compiler converts programs written in C language into object codes executable on the microcomputer. When the compiler is used, the 78K/III series relocatable assembler package (RA78K/III) is needed.
<table>
<tr><th rowspan="2">Host machine</th><th colspan="2">OS</th><th rowspan="2">Distribution media</th><th rowspan="2">Part number</th></tr>
<tr><td>PC-9800 series</td><td>MS-DOS</td><td>3.5-inch 2HD</td><td>μS5A13CC78K3</td></tr>
<tr><td></td><td></td><td>5.25-inch 2HD</td><td>μS5A10CC78K3</td></tr>
<tr><td>IBM PC/AT or compatibles</td><td>PC DOS</td><td>3.5-inch 2HC</td><td>μS7B13CC78K3</td></tr>
<tr><td></td><td></td><td>5.25-inch 2HC</td><td>μS7B10CC78K3</td></tr>
<tr><td>HP9000 series 700</td><td>HP-UX</td><td>DAT</td><td>μS3P16CC78K3</td></tr>
<tr><td>SPARCstation</td><td>SunOS</td><td rowspan="2">Cartridge tape (QIC-24)</td><td>μS3K15CC78K3</td></tr>
<tr><td>NEWS</td><td>NEWS-OS</td><td>μS3R15CC78K3</td></tr>
</table></td></tr>
</table>

Remark It is guaranteed that the relocatable assembler and C compiler run only under the OSs on the corresponding host machines described above.

PROM programming tools

<table>
<tr><td rowspan="11">Hardware</td><td>PG-1500</td><td colspan="3">The PG-1500 PROM programmer is used together with an accessory board and optional program adapter. It allows the user to program a single chip microcomputer containing PROM independently or from a host machine. The PG-1500 can be used to program typical 256K-bit to 4M-bit PROMs.</td></tr>
<tr><td>UNISITE[Note 1]</td><td colspan="3" rowspan="3">PROM programmer manufactured by Data IO Japan.</td></tr>
<tr><td>2900[Note 1]</td></tr>
<tr><td>3900[Note 2]</td></tr>
<tr><td>R4945[Note 1]</td><td colspan="3" rowspan="2">PROM programmer manufactured by Advantest Corporation. Use the PROM programmer and an optional socket adapter in combination.
For R4945 socket adapter: R49451A
For R4952 socket adapter: R49512B</td></tr>
<tr><td>R4952[Note 3]</td></tr>
<tr><td>PKW-1100+RX-1[Note 4]</td><td colspan="3" rowspan="2">PROM programmer manufactured by AVAL Data Corporation</td></tr>
<tr><td>RKW-3100+ADAPTER B mkII [Note 4]</td></tr>
<tr><td>PA-78P334GJ
PA-78P334LQ
PA-78P334KM
PA-78P334KW[Note 5]</td><td colspan="3">Programmer adapter for writing programs to the μPD78P334. Used with a PROM programmer such as the PG-1500.
PA-78P334GJ: For the 94-pin plastic QFP
PA-78P334LQ: For the 84-pin plastic QFJ
PA-78P334KM: For the 94-pin ceramic WQFN
PA-78P334KW: For the 84-pin ceramic WQFN</td></tr>
<tr><td rowspan="5">Software</td><td rowspan="5">PG-1500 controller</td><td colspan="3">This program enables the host machine to control the PG-1500 through the serial and parallel interfaces.</td></tr>
<tr><td rowspan="2">Host machine</td><td rowspan="2">OS</td><td rowspan="2">Distribution media</td><td rowspan="2">Part number</td></tr>
<tr></tr>
<tr><td>PC-9800 series</td><td>MS-DOS</td><td>3.5-inch 2HD</td><td>μS5A13PG1500</td></tr>
<tr><td rowspan="2">IBM PC/AT or compatibles</td><td rowspan="2">PC DOS</td><td>5.25-inch 2HD</td><td>μS5A10PG1500</td></tr>
<tr><td>3.5-inch 2HD</td><td>μS7B13PG1500</td></tr>
<tr><td></td><td></td><td>5.25-inch 2HC</td><td>μS7B10PG1500</td></tr>
</table>

Notes 1. μPD78P334KM-S programming is under evaluation.
2. Under evaluation.
3. μPD78P334GJ-5BG, μPD78P334KM-S and μPD78P334KW programming is under evaluation.
4. μPD78P334GJ-5BG, μPD78P334LQ and μPD78P334KM-S programming is under evaluation.
5. The former product, PA-78P334KE, can also be used.

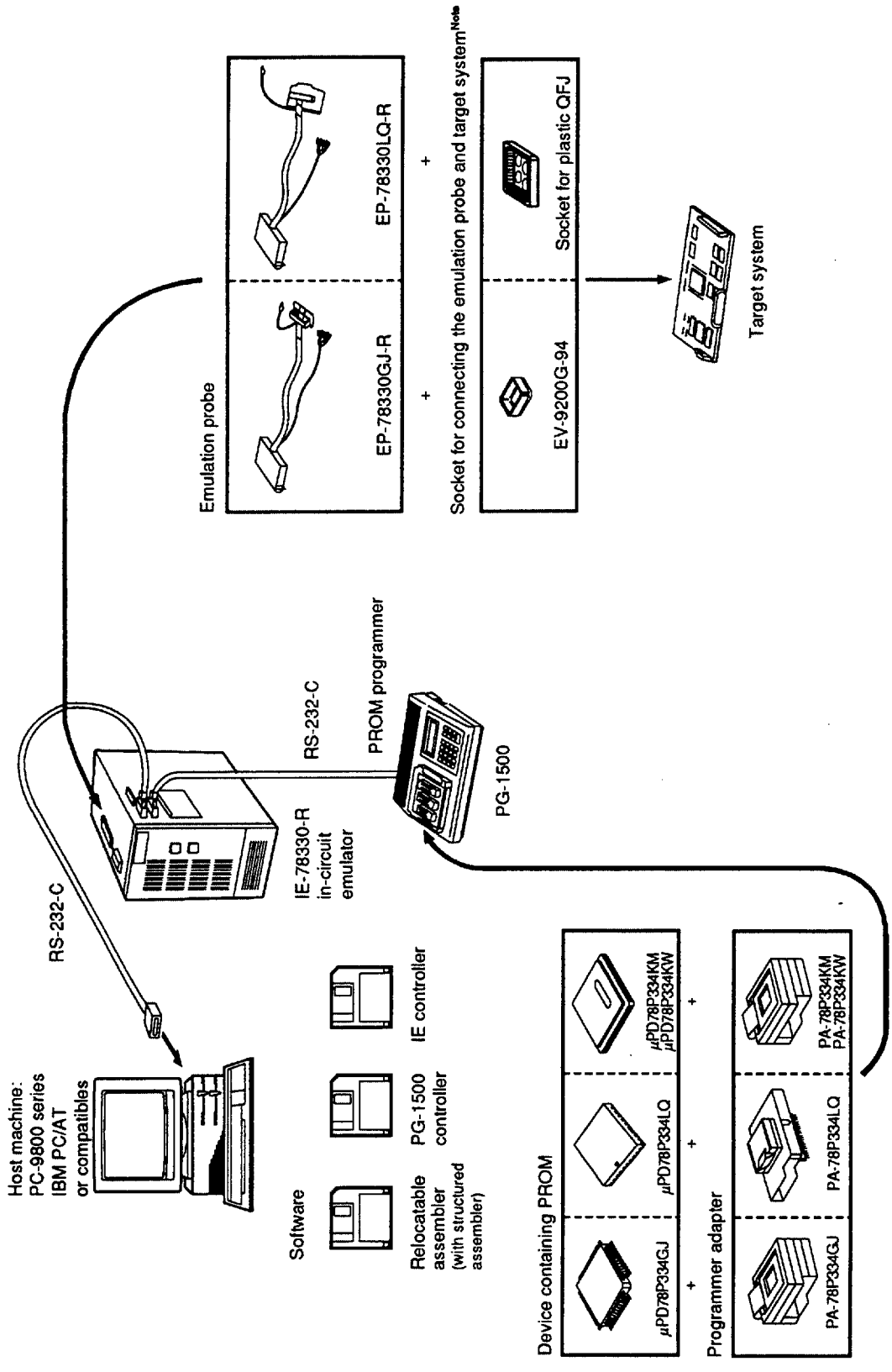
Remark It is only guaranteed that the software operates under the OSs on the corresponding host machines described above.

Debugging tools

Hardware	IE-78330-R	In-circuit emulator for developing and debugging an application system. For debugging, connect the emulator to the host machine.		
	EP-78330GJ-R EV-9200G-94	Emulation probe for connecting the IE-78330-R to the target system. Used for the 94-pin plastic QFP. One EV-9200G-94 conversion socket is provided for connection to the target system.		
	EP-78330LQ-R	Emulation probe for connecting the IE-78330-R to the target system. Used for the 84-pin plastic QFJ.		
Software	IE-78330-R control program (IE-controller)	This control program allows the user to control the IE-78330-R from the host machine. Its automatic command execution function ensures more efficient debugging.		
		Host machine		Part number
			OS	Distribution media
		PC-9800 series	MS-DOS	3.5-inch 2HD	μS5A13IE78330
				5.25-inch 2HD	μS5A10IE78330
		IBM PC/AT or compatibles	PC DOS	3.5-inch 2HC	μS7B13IE78330
				5.25-inch 2HC	μS7B10IE78330

Remark It is only guaranteed that the software operates under the OSs on the corresponding host machines described above.

Configuration of development tools



Note The socket is supplied with the emulation probe.

Remarks 1. The PG-1500 can be directly connected to the host machine via the RS-232-C interface.

2. In this figure, the distribution media of software is represented by the 3.5-inch floppy disk.

B.2 EVALUATION TOOLS

The following evaluation tools are provided for evaluating the function of the μPD78334:

Part number	Host machine	Function
EB-78330-98	PC-9800 series	When the evaluation tool is connected to the host machine, the functions of the μPD78334 can easily be evaluated. As the command system of the EB-78330-98/PC conforms to that of the IE-78330-R, the migration can easily be made to the development of the application system with the IE-78330-R.
EB-78330-PC	IBM PC/AT or compatibles	

Cautions 1. These products are not development tools for the application system that uses the μPD78334.

2. These products do not have an emulation function used during the execution of the program in internal ROM for the μPD78334.

B.3 EMBEDDED SOFTWARE

To improve the efficiency of program development and simplify the maintenance of systems incorporating this microcontroller, the following embedded software is provided.

Real-time OS

Real-time OS (RX78K/III)	This operating system was designed to provide a multitasking environment for control applications that require real-time processing. System performance is improved by using the idling CPU for other processing. RX78K/III provides system calls that conform to μITRON specifications. The RX78K/III package provides the RX78K/III nucleus and a tool (Configurator) that is used for creating multiple information tables.			
	Host machine	OS	Distribution media	Part number
	PC-9800 series	MS-DOS	3.5-inch 2HD	μS5A13RX78320
			5.25-inch 2HD	μS5A10RX78320
	IBM PC/AT or compatibles	PC DOS	3.5-inch 2HC	μS7B13RX78320
			5.25-inch 2HC	μS7B10RX78320

Caution Before purchasing this software, complete the purchase application sheet and sign the software license agreement.

Remark To use the RX78K/III real-time operating system, the optional RA78K/III assembler package is required.

Fuzzy inference development support system

Tool for creating fuzzy knowledge data (FE9000, FE9200)	This program supports the input/editing and simulation of fuzzy knowledge data (fuzzy rules and membership functions).				
	Host machine	OS		Distribution media	Part number
	PC-9800 series	MS-DOS		3.5-inch 2HD	μS5A13FE9000
				5.25-inch 2HD	μS5A10FE9000
	IBM PC/AT or compatibles	PC DOS	Windows™	3.5-inch 2HC	μS7B13FE9200
				5.25-inch 2HC	μS7B10FE9200
Translator (FT78K3)Note	This program converts fuzzy knowledge data, obtained using the tool for creating fuzzy knowledge data, into an assembler source program for RA78K/III.				
	Host machine	OS		Distribution media	Part number
	PC-9800 series	MS-DOS		3.5-inch 2HD	μS5A13FT78K3
				5.25-inch 2HD	μS5A10FT78K3
	IBM PC/AT or compatibles	PC DOS		3.5-inch 2HC	μS7B13FT78K3
				5.25-inch 2HC	μS7B10FT78K3
Fuzzy inference module (FI78K/III)Note	This program performs fuzzy inference by linking the fuzzy knowledge data converted by Translator.				
	Host machine	OS		Distribution media	Part number
	PC-9800 series	MS-DOS		3.5-inch 2HD	μS5A13FI78K3
				5.25-inch 2HD	μS5A10FI78K3
	IBM PC/AT or compatibles	PC DOS		3.5-inch 2HC	μS7B13FI78K3
				5.25-inch 2HC	μS7B10FI78K3
Fuzzy inference debugger (FD78K/III)	This software supports the evaluation and adjustment of fuzzy knowledge data at the hardware level, by using an in-circuit emulator.				
	Host machine	OS		Distribution media	Part number
	PC-9800 series	MS-DOS		3.5-inch 2HD	μS5A13FD78K3
				5.25-inch 2HD	μS5A10FD78K3
	IBM PC/AT or compatibles	PC DOS		3.5-inch 2HC	μS7B13FD78K3
				5.25-inch 2HC	μS7B10FD78K3

Note Under development

Cautions on CMOS Devices

① **Countermeasures against static electricity for all MOSs**

Caution When handling MOS devices, take care so that they are not electrostatically charged.

Strong static electricity may cause dielectric breakdown in gates. When transporting or storing MOS devices, use conductive trays, magazine cases, shock absorbers, or metal cases that NEC uses for packaging and shipping. Be sure to ground MOS devices during assembling. Do not allow MOS devices to stand on plastic plates or do not touch pins.

Also handle boards on which MOS devices are mounted in the same way.

② **CMOS-specific handling of unused input pins**

Caution Hold CMOS devices at a fixed input level.

Unlike bipolar or NMOS devices, if a CMOS device is operated with no input, an intermediate-level input may be caused by noise. This allows current to flow in the CMOS device, resulting in a malfunction. Use a pull-up or pull-down resistor to hold a fixed input level. Since unused pins may function as output pins at unexpected times, each unused pin should be separately connected to the V_{DD} or GND pin through a resistor.

If handling of unused pins is documented, follow the instructions in the document.

③ **Statuses of all MOS devices at initialization**

Caution The initial status of a MOS device is unpredictable when power is turned on.

Since characteristics of a MOS device are determined by the amount of ions implanted in molecules, the initial status cannot be determined in the manufacture process. NEC has no responsibility for the output statuses of pins, input and output settings, and the contents of registers at power on. However, NEC assures operation after reset and items for mode setting if they are defined.

When you turn on a device having a reset function, be sure to reset the device first.

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.

NEC
NEC Electronics Inc.
CORPORATE HEADQUARTERS*
2880 Scott Boulevard
P.O. Box 58062
Santa Clara, CA 95052-8062
TEL 408-588-6000
* As of September 3, 1996

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**

U10185EJ5V0DS01