DATA SHEET

NEC

MOS INTEGRATED CIRCUIT

μPD78134A, 78136, 78138

8 BIT SINGLE-CHIP MICROCOMPUTER

The μPD78138 is an 8-bit single-chip microcomputer. It contains a high-speed, high-performance 8-bit CPU.

With on-chip peripheral hardware, the μPD78138 can be used in VCRs and other devices that require digital servo control via the software. The μPD78138 can support system control on one chip in addition to servo control, thus further miniaturizing the application set.

The μPD78P138 with the PROM is also provided, which is suited for evaluation and trial manufacture during system development, early stage start-up of applications, and short-run and multiple-device production.

FEATURES

- High-speed instruction execution via internal multiplexed bus: 333 ns (at 12 MHz)
- Built-in super timer unit that best suits VCR servo control
  - Speed and phase control of drums, capstans, and motors
  - Head switch signal output of two channels including audio and video
  - Vertical synchronizing signal detection function
  - Input pulse duty ratio determining function (applicable to VCR index searching system etc.)
  - Built-in two-channel PWM output circuit that can specify active levels
- Additional functions that improve responsibility of servo control
  - Signal multiply instruction
  - Variable PWM output carrier frequency (23.4/46.9 kHz)
- Built-in real-time output port best suited for outputting the VCR head switching signal and controlling a step motor
- Built-in powerful interrupt functions providing two service modes
  - Vector interrupt function
  - Macro service function (Facilitates automatic data transfer and AMSS function on VCRs.)
- 44 built-in pull-up resistor eliminating the need for external resistors
- Various ROM sizes (16K/24K/32K bytes)

APPLICATIONS

The μPD78134A applies to servo controlling of VCRs (normal type and camcorder type) and DATs.

Some references made to the μPD78138 in this document pertain to functions common to the μPD78134A, μPD78136, and μPD78138.

The information in this document is subject to change without notice.

Document No. IC-3065A
(O. D. No. IC-8447A)
Date Published January 1993 P
Printed in Japan

The mark * shows major revised points.

ORDERING INFORMATION

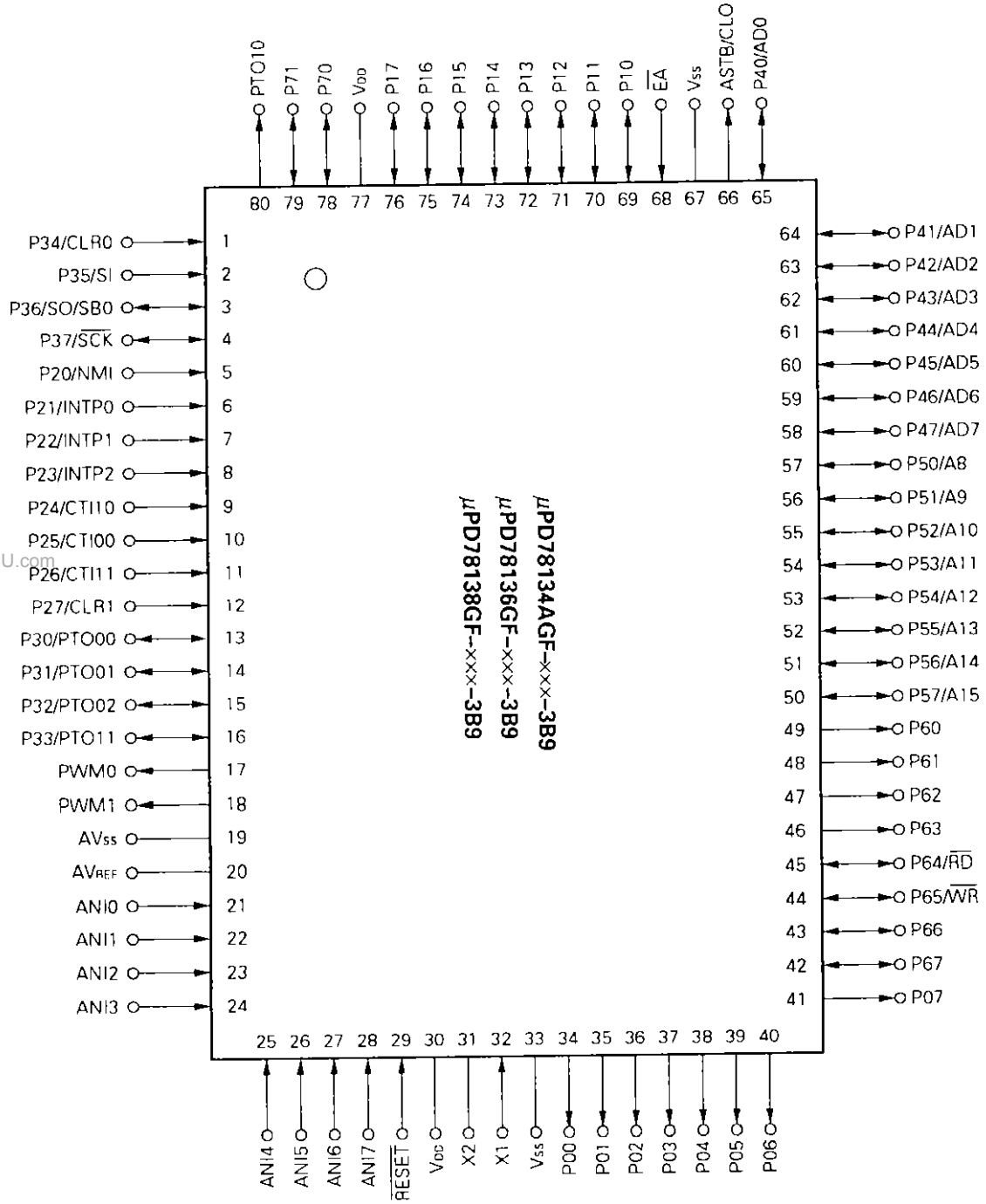
Part number	Package	Quality grade
μPD78134AGF-xxx-3B9	80-pin plastic QFP (14 × 20 mm)	Standard
μPD78136GF-xxx-3B9	80-pin plastic QFP (14 × 20 mm)	Standard
μPD78138GF-xxx-3B9	80-pin plastic QFP (14 × 20 mm)	Standard

Remark xxx represents the code of the ROM chip you want to purchase

Please refer to "Quality grade on NEC Semiconductor Devices" (Document number IEI-1209) published by NEC Corporation to know the specification of quality grade on the devices and its recommended applications.

PIN CONFIGURATION (TOP VIEW)

80-PIN PLASTIC QFP (14 × 20 mm excluding the dimensions of the pins)



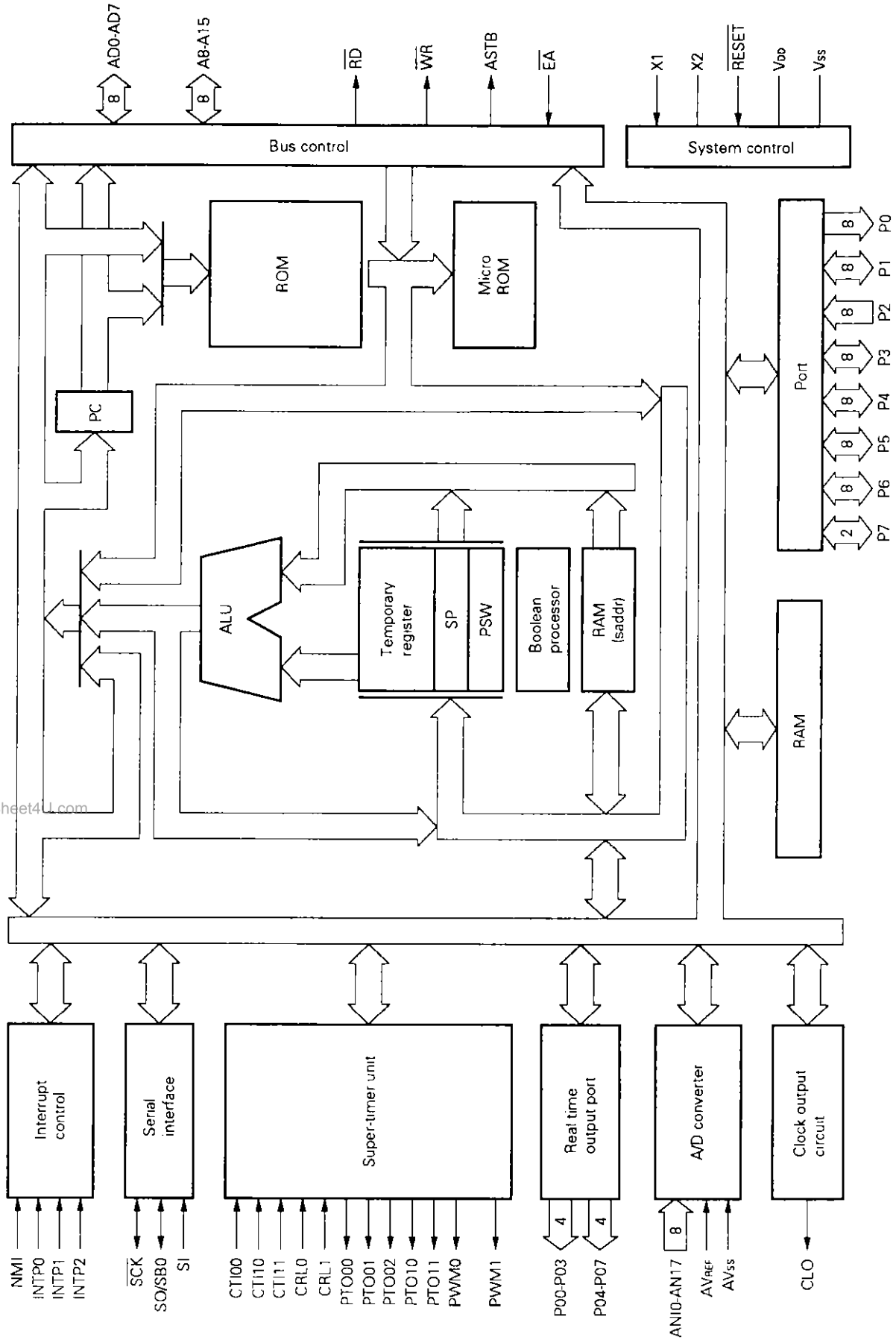
P00-P07	:	Port 0
P10-P17	:	Port 1
P20-P27	:	Port 2
P30-P37	:	Port 3
P40-P47	:	Port 4
P50-P57	:	Port 5
P60-P67	:	Port 6
P70, P71	:	Port 7
PWM0, PWM1	:	Pulse width modulation output
CLO	:	Clock output
ANI0-ANI7	:	Analog input
AV_{REF}	:	Reference voltage
AVss	:	Analog Vss
X1, X2	:	Crystal
$\overline{RESET}$	:	Reset

CTI00, CTI10, CTI11	:	Capture trigger input
CLR0, CLR1	:	Timer clear input
PTO00-PTO02, PTO10, PTO11	:	Programmable timer input
NMI	:	Nonmaskable interrupt
INTP0-INTP2	:	Interrupt from peripherals
SI	:	Serial input
SO	:	Serial output
SB0	:	Serial bus
$\overline{SCK}$	:	Serial clock
AD0-AD7	:	Address data
A8-A15	:	Address
$\overline{RD}$	:	Read
$\overline{WR}$	:	Write
ASTB	:	Address strobe
$\overline{EA}$	:	External access

FUNCTIONAL OVERVIEW

Item \ Product		μPD78134A	μPD78136	μPD78138
Internal memory	ROM	16K bytes (16384 × 8 bits)	24K bytes (24576 × 8 bits)	32K bytes (32768 × 8 bits)
	RAM	384 bytes	640 bytes	
Number of basic instructions		64		
Minimum instruction execution time		333 ns (at 12 MHz)		
Memory expansion		Externally expandable up to 64K bytes		
General register		8 bits × 8 × 4 banks (memory mapping)		
Instruction set		• 16-bit addition, subtraction, comparison • Signed multiplication (signed 16 bits × unsigned 8 bits) • Unsigned multiplication/division (16 bits × 8 bits, 16 bits ÷ 8 bits) • Bit manipulation (transfer, Boolean operation, set, reset, test) • BCD correction		
I/O line		• 66 total Input port : 10 Output port : 12 I/O port : 36 Analog input : 8		
Super timer unit		• Timer : 16 bits × 3 7 bits × 1 • Counter : 18 bits × 1 • Capture register : 18 bits × 1 16 bits × 4 7 bits × 1 • Compare register : 16 bits × 6 7 bits × 1 • PWM output : 12 bits × 2 (variable active level, variable carrier frequency (23.4/46.9 kHz)		
Real-time output port		• Timer connected port output function • 4 bits × 2 or 8 bits × 1		
Serial interface		• Either NEC format serial bus interface (SBI) or 3-wire serial interface can be selected.		
A/D converter		• 8-bit resolution × 8 inputs • Conversion time: 30 μs/1 analog input (at 12 MHz) • Either select or scan mode can be selected.		
Interrupt		• Interrupt source: 17 (5 external and 12 internal) • One of the two service modes can be selected (macro service/vector interrupt). • Variable 2-level interrupt priority		
Standby		STOP mode		
Pull-up resistor		44, built-in (enable/disable built-in can be specified via software)		
Package		80-pin plastic QFP (14 × 20 mm excluding the dimensions of the pins)		

BLOCK DIAGRAM



Caution The capacity of internal ROM or RAM depends on each product.

CONTENTS

1. PIN FUNCTIONS

1.1 PORT PINS

Pin name	I/O	Dual-function pin	Function
P00-P07	O	—	Port 0 (P0) : Can be specified to output or high impedance 8 bits by 8 bits. Also function as an 8 bits × 1 or 4 bits × 2 real-time output port.
P10-P17	I/O	—	Port 1 (P1) : Can be specified to input or output bit by bit. Can directly drive LED. Software pull-up resistor (P10-P17) can be built in.
P20	I	NMI	Port 2 (P2) : Software pull-up resistor (P22-P27) can be built in.
P21		INTP0	
P22		INTP1	
P23		INTP2	
P24		CTI10	
P25		CTI00	
P26		CTI11	
P27		CLR1	
P30	I/O	PTO00	Port 3 (P3) : P30-P33, P36, P37: I/O port (Can be specified to input or output bit by bit.) P34, P35: Input port Software pull-up resistor (P30-P37) can be built in.
P31		PTO01	
P32		PTO02	
P33		PTO11	
P34	I	CLR0	
P35	I	SI	
P36	I/O	SO/SB0	
P37	I/O	$\overline{SCK}$	
P40-P47	I/O	AD0-AD7	Port 4 (P4) : Can be specified to input or output bit by bit. Software pull-up resistor (P40-P47) can be built in.
P50-P57	I/O	A8-A15	Port 5 (P5) : Can be specified to input or output bit by bit. Software pull-up resistor (P50-P57) can be built in.
P60-P63	O	—	Port 6 (P6) : P60-P63: Output port P64-P67: I/O port (Can be specified to input or output bit by bit.) Software pull-up resistor (P64-P67) can be built in.
P64	I/O	$\overline{RD}$	
P65	I/O	$\overline{WR}$	
P66, P67	I/O	—	
P70, P71	I/O	—	Port 7 (P7) : Can be specified to input or output 2 bits by 2 bits. Software pull-up resistor (P70, P71) can be built in.

1.2 NON-PORT PINS

Pin name	I/O	Dual-function pin	Function
PWM0, PWM1	O	—	Super timer unit PWM output
ANI0-ANI7	I	—	Analog voltage input to A/D converter
AVREF		—	Reference voltage input to A/D converter
AVss		—	Ground potential of A/D converter
NMI	I	P20	Non-maskable interrupt request input. Either rising edge or falling edge can be selected via mode register (INTM0)
INTP0	I	P21	External interrupt request input . Rising edge, falling edge, or rising and falling edges can be selected via mode register (INTM0).
INTP1	I	P22	External interrupt request input. Can select rising edge, falling edge, or rising and falling edges by mode register (INTM0)
INTP2		P23	
SI	I	P35	Serial data input (3-wire serial I/O mode)
SO	I/O	P36/SB0	Serial data output (3-wire serial I/O mode)
SB0	I/O	P36/SO	Serial data input (SBI mode)
$\overline{SCK}$	I/O	P37	Serial clock input/output
CTI00	I	P25	Super timer unit capture trigger input
CTI10		P24	
CTI11		P26	
CLR0	I	P34	Super timer unit timer clear signal input
CLR1		P27	
PTO00	I/O	P30	Super timer unit timer output
PTO01		P31	
PTO02		P32	
PTO10	O	—	
PTO11	I/O	P33	
AD0-AD7	I/O	P40-P47	Time multiplexing address/data bus for when external memory is connected
A8-A15	O	P50-P57	Address output port for when external memory is connected
$\overline{RD}$	O	P64	Strobe signal output for reading external memory
$\overline{WR}$	O	P65	Strobe signal output for writing external memory
ASTB	O	CLO	Timing signal output that externally latches address data for accessing external memory
CLO	O	ASTB	Clock output
$\overline{EA}$	I	—	External expansion function control input
X1	I	—	Crystal connection for system clock signal oscillation. Input the externally supplied clock signal to X1 and input its inverted phase to X2.
X2	—		
$\overline{RESET}$	I	—	System reset input. Contains an analog delay noise reduction circuit.
VDD		—	Positive power supply
Vss		—	GND potential

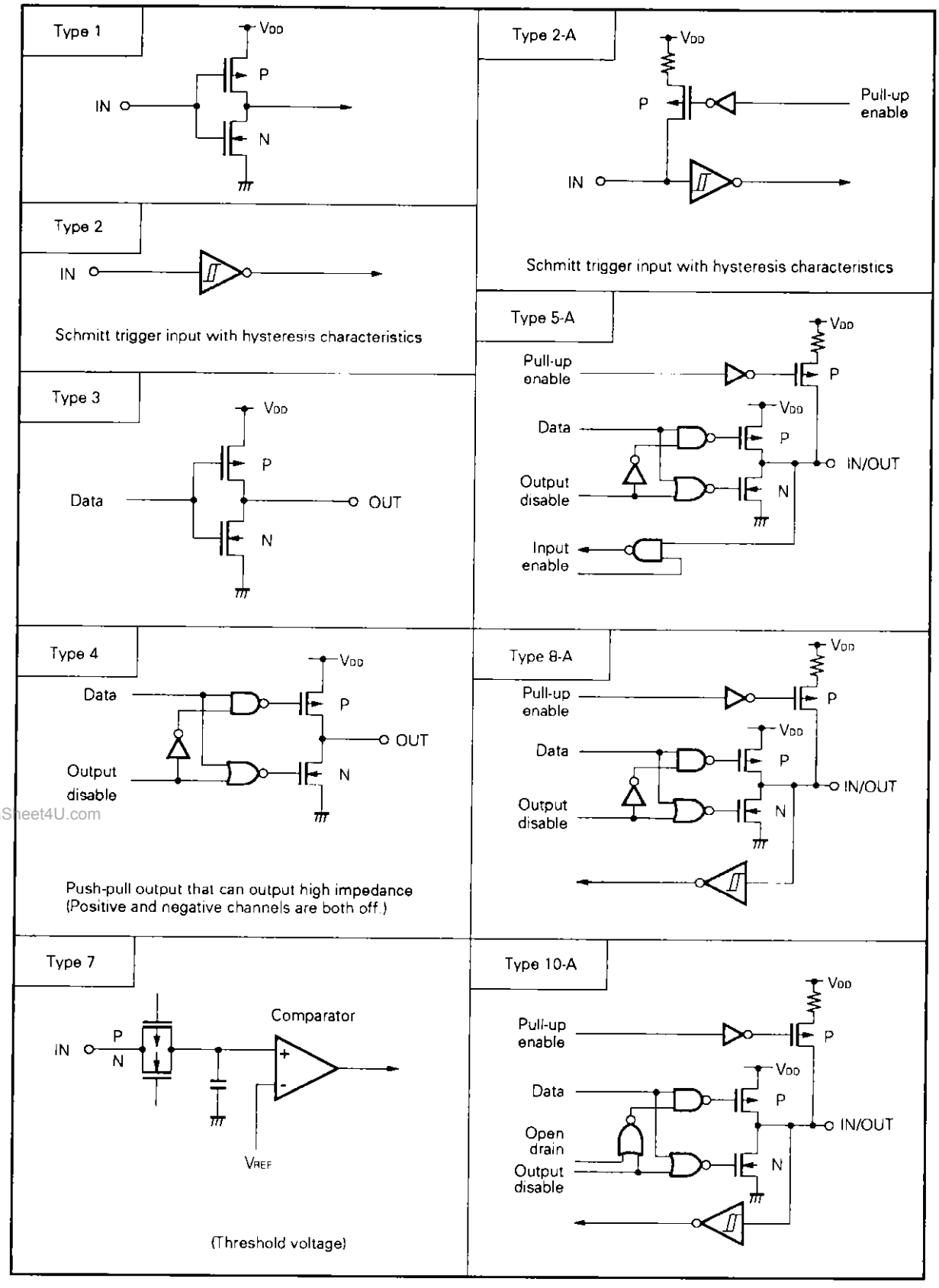
1.3 INPUT/OUTPUT CIRCUITS AND CONNECTION OF UNUSED PINS

Table 1-1 and Fig. 1-1 show simplified pin input/output circuits.

Table 1-1 I/O Circuit Type of Each Pin and Recommended Connection of Unused Pins

Pin name	I/O circuit type	Recommended connection of unused pins
P00-P07	4	Open
P10-P17	5-A	Input : Connected to VDD via pull-up resistor Output : Open
P20/NMI	2	Connected to VDD
P21/INTP0		
P22/INTP1	2-A	
P23/INTP2		
P24/CTI10		
P25/CTI00		
P26/CTI11		
P27/CLR1		
P30/PTO00	5-A	Input : Connected to VDD via pull-up resistor Output : Open
P31/PTO01		
P32/PTO02		
P33/PTO11		
P34/CLR0	2-A	Connected to VDD
P35/SI		
P36/SO/SB0	10-A	Input : Connected to VDD via pull-up resistor Output : Open
P37/$\overline{SCK}$	8-A	
P40-P47/AD0-AD7	5-A	Input : Connected to VDD via pull-up resistor Output : Open
P50-P57/A8-A15		
P60-P63	3	Open
P64/$\overline{RD}$	5-A	Input : Connected to VDD via pull-up resistor Output : Open
P65/$\overline{WR}$		
P66, P67		
P70, P71		
PWM0, PWM1	3	Open
PTO10		
ANI0-ANI7	7	Connected to Vss
$\overline{EA}$	1	—
ASTB/CLO	3	Open
$\overline{RESET}$	2	—
AVREF	–	Connected to Vss
AVss		

Fig. 1-1 Pin Input/Output Circuits



2. CPU ARCHITECTURE

2.1 MEMORY SPACE

The μPD78138 allows access to a memory space of up to 64K bytes. Fig. 2-1, 2-2, and 2-3 show the memory space of each product. Program memory is mapped differently according to the state of the external access pin ($\overline{EA}$).

(1) $\overline{EA}$ = high

Program memory is mapped in internal ROM and external memory. (See Table 2-1.) External memory is accessed in the external memory expansion mode. The external memory area can also be used as data memory.

Data memory is mapped in the internal RAM. (See Table 2-1.)

Table 2-1 Memory Mapping (When $\overline{EA}$ = High)

Item \ Product	μPD78134A	μPD78136	μPD78138
Internal ROM	16K bytes (0000H-3FFFH)	24K bytes (0000H-5FFFH)	32K bytes (0000H-7FFFH)
External memory	48512 bytes (4000H-FD7FH)	40064 bytes (6000H-FC7FH)	31872 bytes (8000H-FC7FH)
Internal RAM	384 bytes (FD80H-FEFFH)	640 bytes (FC80H-FEFFH)	

(2) $\overline{EA}$ = low

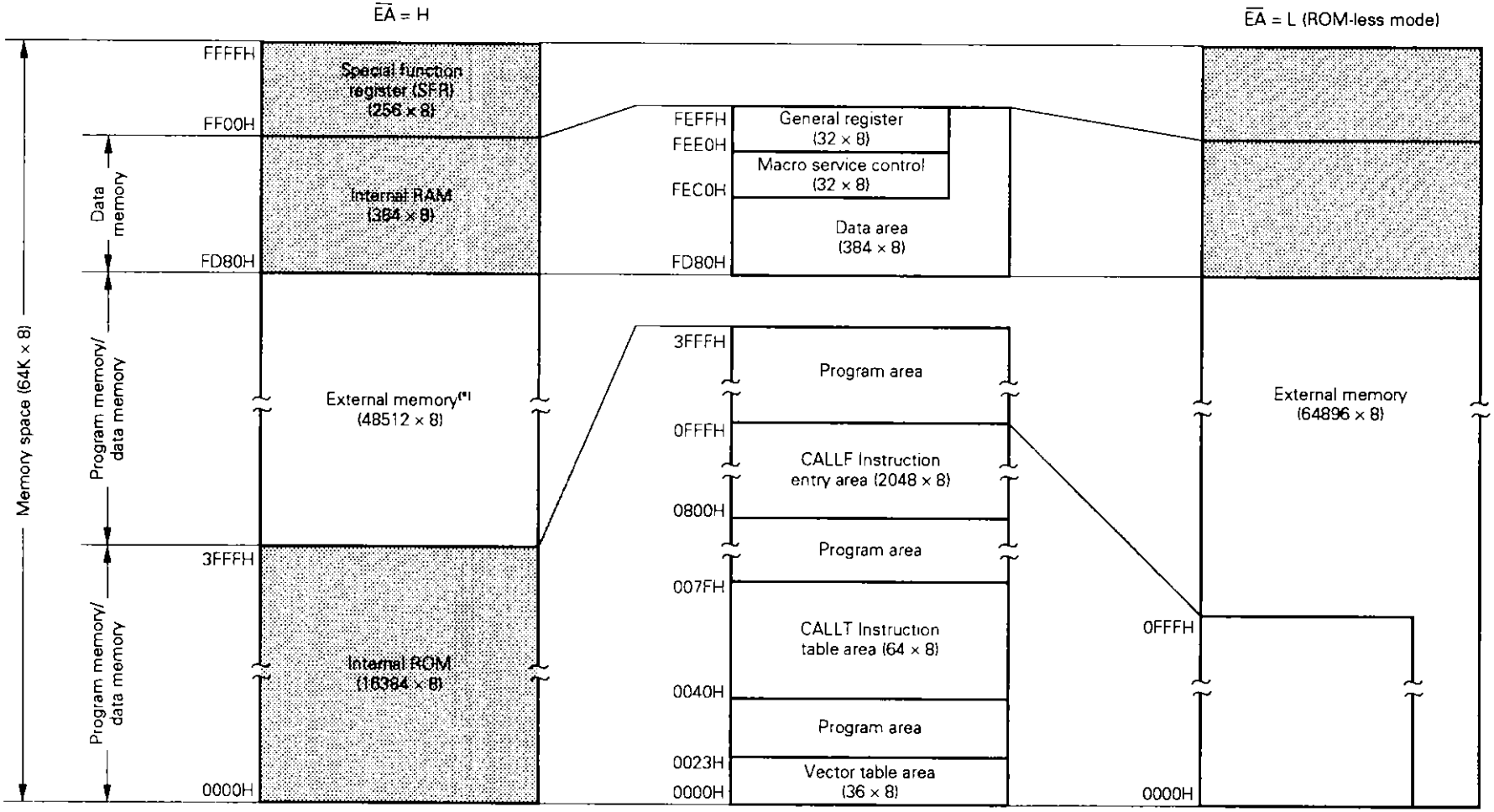
All program memory is mapped in external memory in the ROM-less mode. (See Table 2-2.) This area can also be used as data memory.

Data memory is mapped in the internal RAM. (See Table 2-2.)

Table 2-2 Memory Mapping (When $\overline{EA}$ = Low)

Item \ Product	μPD78134A	μPD78136	μPD78138
External memory	64896 bytes (0000H-FD7FH)	64640 bytes (0000H-FC7FH)	
Internal RAM	384 bytes (FD80H-FEFFH)	640 bytes (FC80H-FEFFH)	

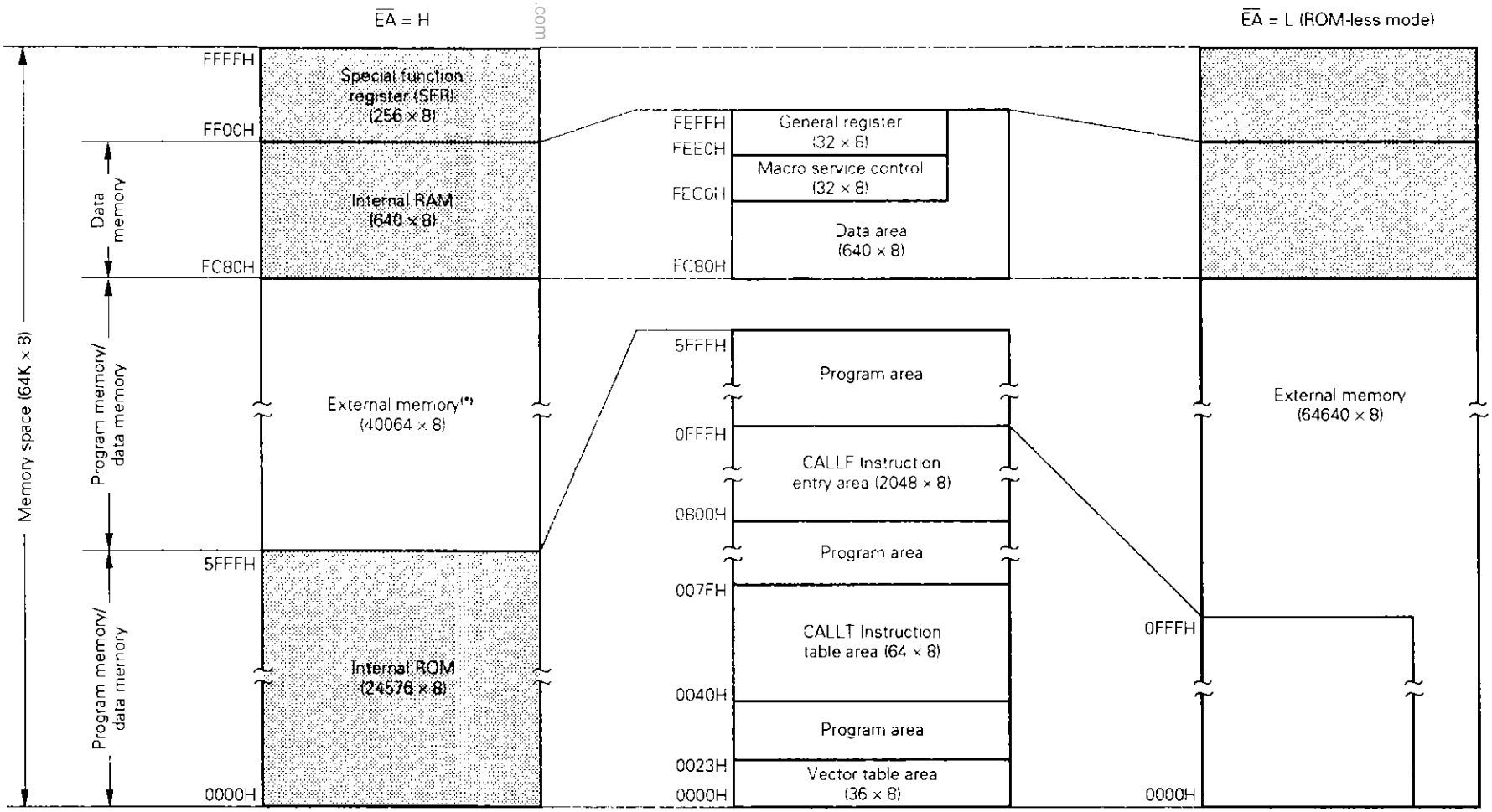
Fig. 2-1 Memory Map (μPD78134A)



* Accessed in the external memory expansion mode

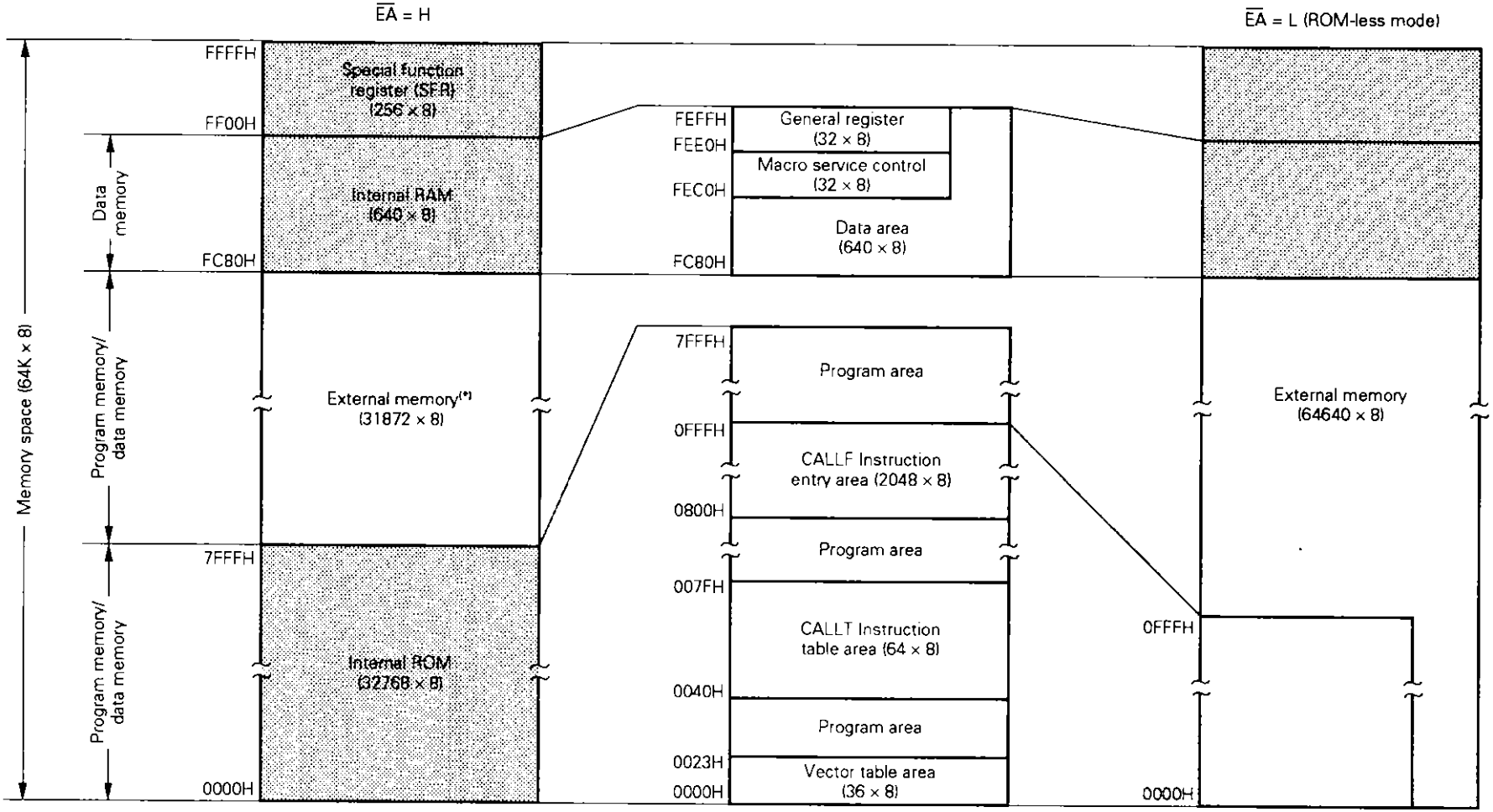
Shaded portions indicate internal memory.

Fig. 2-2 Memory Map (μPD78136)



* Accessed in the external memory expansion mode
Shaded portions indicate internal memory.

Fig. 2-3 Memory Map (μPD78138)



* Accessed in the external memory expansion mode
Shaded portions indicate internal memory.

2.2 PROCESSOR REGISTER

Registers of the μPD78138 are classified into three groups according to their functions:

- Control register
- General register
- Special function register (SFR)

2.2.1 Control Register

The control register group controls the program sequence, status, and stack memory. There are three control register:

- Program counter (PC): 16-bit register
- Program status word (PSW): 16-bit register
- Stack pointer (SP): 16-bit register

Fig. 2-4, 2-5, and 2-6 show the formats of each register.

Fig. 2-4 Format of Program Counter (PC)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
PC	PC15	PC14	PC13	PC12	PC11	PC10	PC9	PC8	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0

Fig. 2-5 Format of Program Status Word (PSW)

	7	6	5	4	3	2	1	0
PSW	IE	Z	RBS1	AC	RBS0	0	ISP	CY

- CY ..... Carry flag
- ISP ..... Interrupt priority status flag
- RBS0, RBS1 ..... Register bank selection flag
- AC ..... Auxiliary carry flag
- Z ..... Zero flag
- IE ..... Interrupt request enable flag

Fig. 2-6 Format of Stack Pointer (SP)

	15	14	13	12	11	10	9	8	7	6	5	4	3	2	1	0
SP	SP15	SP14	SP13	SP12	SP11	SP10	SP9	SP8	SP7	SP6	SP5	SP4	SP3	SP2	SP1	SP0

2.2.2 General Register

The general register group of the μPD78138 consists of four banks of general registers. Each bank consists of eight 8-bit registers, so there are 32 registers in total. A pair of 8-bit registers can function as a 16-bit register pair.

The general register group is mapped into addresses from FEE0H to FEFFH of the internal RAM space. Fig. 2-7 shows the configuration of the general register and Table 2-3 shows the correspondence between function names and absolute names.

Fig. 2-7 Format of General Register

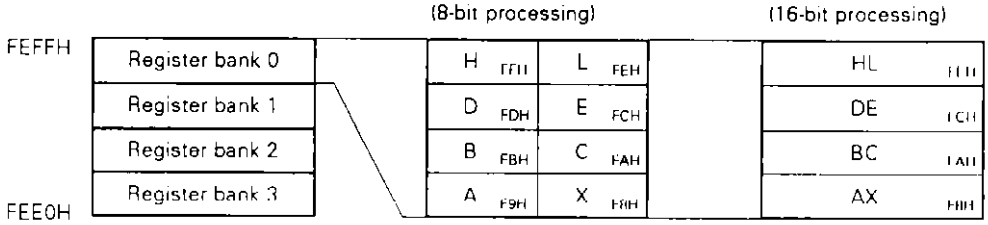


Table 2-3 Correspondence between Function Names and Absolute Names

Function name	Absolute name
X	R0
A	R1
C	R2
B	R3
E	R4
D	R5
L	R6
H	R7

Function name	Absolute name
AX	RP0
BC	RP1
DE	RP2
HL	RP3

2.2.3 Special Function Register (SFR)

The special function register group consists of the registers to which special functions are given such as mode register of the peripheral hardware.

This register group is assigned to the 256-byte space from FF00H to FFFFH. Short direct memory addressing can be applied to the 32-byte area from FF00H to FF1FH, allowing short-word data processing.

Bit manipulation, arithmetic, and move instructions can be executed in all the areas.

Table 2-4 lists the special function registers (SFRs). The items in Table 2-4 mean:

- Abbreviation ..... A symbol indicating the address of a built-in special function register
  This can be specified in the operand field of an instruction.
- R/W ..... Indicates whether data can be read from the special function register and/or data can be written into the register.
  R/W : Can be read and written.
  R : Can be read. (The bits of the register can be tested.)
  W : Can be written.
- Manipulation bit unit
  ..... Indicates the unit of bits that can be manipulated at one time.
  The SFR which can be manipulated in units of 16 bits can be specified in the sfrp operand. An even address is specified for the address specification.
  The SFR which can be manipulated bit by bit can be specified by a bit manipulation instruction.
- At resetting ..... Indicates the status of each register for the input to the $\overline{\text{RESET}}$ pin.

Cautions 1. The addresses to which a special function register is not assigned in the area from FF00H to FFFFH can not be accessed.

2. Do not write data into the register which is only used for data reading. If an attempt is made to write data into such registers, the internal circuit may not operate normally.

Table 2-4 Special Function Registers (SFRs) (1/3)

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		○	○		
FF04H	Port 4	P4		○	○	–	
FF05H	Port 5	P5		○	○		
FF06H	Port 6	P6		○	○	–	××××0000
FF07H	Port 7	P7		○	○		
FF08H	16-bit timer 0 compare register 0	CR00	R/W	–	–	○	Undefined
FF09H				–	–		
FF0AH	16-bit timer 0 compare register 1	CR01		–	–	○	
FF0BH				–	–		
FF0CH	16-bit timer 0 compare register 2	CR02		–	–	○	
FF0DH				–	–		
FF0EH	16-bit timer 1 compare register 0	CR10		–	–	○	
FF0FH				–	–		
FF10H	16-bit timer 1 compare register 1	CR11		–	–	○	
FF11H				–	–		
FF12H	16-bit timer 1 compare register 2	CR12		–	–	○	
FF13H				–	–		
FF14H	16-bit FRC capture register 0	CPT0	R	–	–	○	
FF15H				–	–		
FF16H	16-bit FRC capture register 1	CPT1		–	–	○	
FF17H				–	–		
FF18H	16-bit FRC capture register 2	CPT2H		–	–	○	
FF19H				–	–		
FF1AH	16-bit FRC capture register 3	CPT3		–	–	○	
FF1BH				–	–		
FF1CH	16-bit FRC capture register 2	CPT2L		–	○	–	××000000
FF1DH	Prescaler mode register 3	PRM3	R/W	○	○	–	0××××000
FF1EH	16-bit timer 2 compare register	CR20		–	–	○	Undefined
FF1FH				–	–		

○: Allowed
–: Not allowed

Table 2-4 Special Function Registers (SFRs) (2/3)

Address	Special function register (SFR) name	Abbreviation	R/W	Manipulation bit unit			At resetting
				1	8	16	
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		–	○	–	
FF26H	Port 6 mode register	PM6		–	○	–	F0H
FF27H	Port 7 mode register	PM7		–	○	–	FFH
FF30H FF31H	16-bit timer register 0	TM0	R	– –	– –	○	Undefined for up to 16 clock pulses Cleared to 0 after the 17th clock pulse
FF32H FF33H	16-bit timer register 1	TM1		– –	– –	○	
FF34H FF35H	16-bit free running counter	FRC		– –	– –	○	
FF36H FF37H	16-bit timer register 2	TM2		– –	– –	○	
FF38H	Timer control register 0	TMC0	W	–	○	–	0××00000
FF39H	Timer control register 1	TMC1	R/W	–	○		00H
FF3AH	Capture mode register	CPTM	W	–	○	–	×××××0×0
FF3DH	7-bit timer register 3	TM3	R	–	○	–	00H
FF3EH	7-bit timer 3 compare register	CR30	R/W	–	○	–	×1111111
FF3FH	7-bit timer 3 capture register	CPT30	R	–	○	–	Undefined
FF40H	Register for optional pull-up resistor	PUO	R/W	○	○	–	00H
FF43H	Port 3 mode control register	PMC3		○	○	–	30H
FF4AH	Port 0 buffer register	P0L		○	○	–	Undefined
FF4BH	Port 0 buffer register	P0H		○	○	–	
FF4CH	Real time output port control register	RTPC		○	○	–	00H
FF50H	Input control register	ICR	W	–	○	–	0×0×0×××
FF53H	Event divider control register	EDVC		–	○	–	Undefined
FF54H	Event counter compare register 1	ECC1		–	○	–	××111111
FF55H	Event counter compare register 0	ECC0		–	○	–	××111111
FF56H	Event counter	EC	R	–	○	–	××000000
FF58H	Timer 0 output mode register	TOM0	W	–	○	–	××000000
FF59H	Timer 0 output control register	TOC0		–	○	–	××000000
FF5AH	Timer 1 output mode register	TOM1		–	○	–	××××0000
FF5BH	Timer 1 output control register	TOC1	R/W*	–	○	–	××××0000

* Only bit 0 of TOC1 can be read.

○: Allowed
–: Not allowed

Table 2-4 Special Function Registers (SFRs) (3/3)

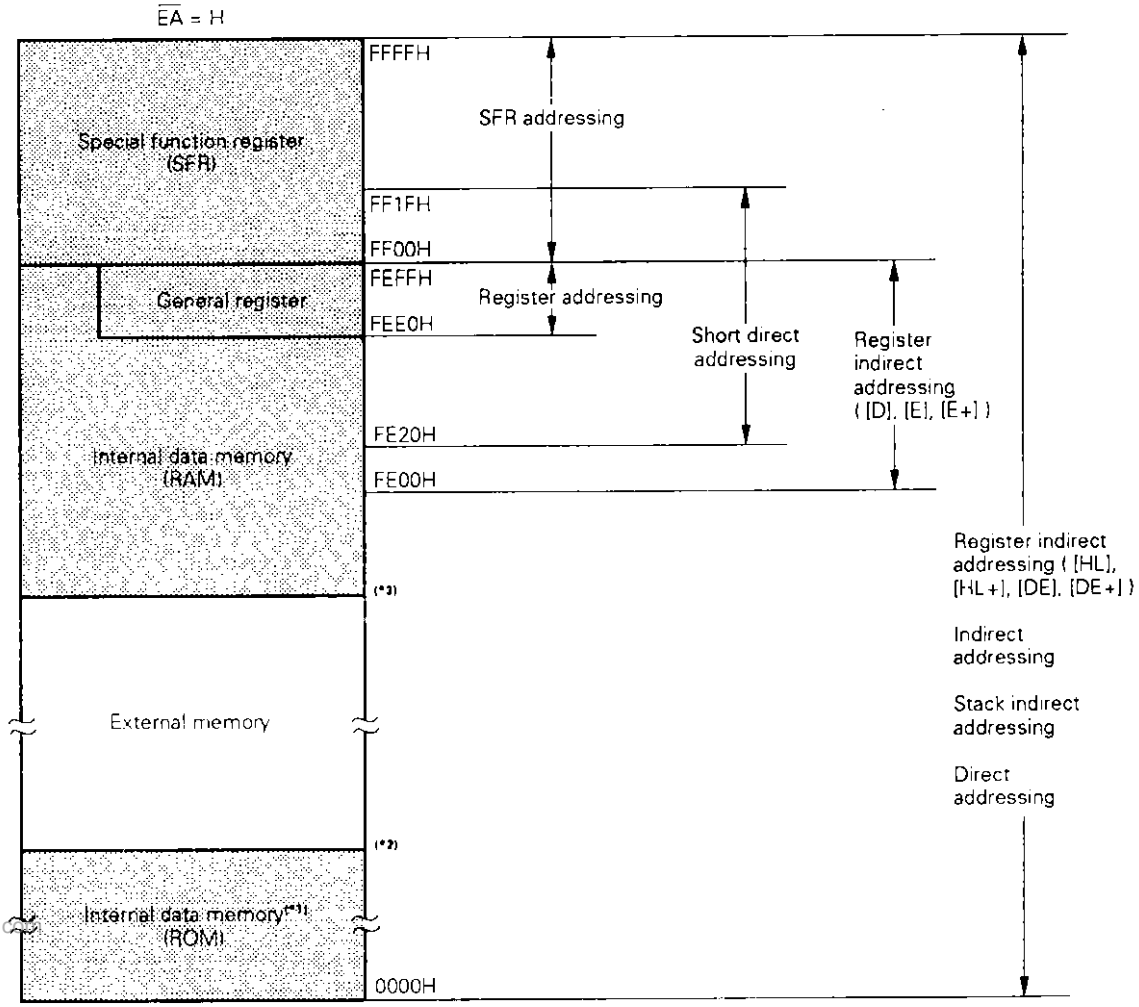
Address	Special function register (SFR) name	Abbreviation		R/W	Manipulation bit unit			At resetting
					1	8	16	
FF68H	A/D conversion mode register	ADM		R/W	○	○	–	00H
FF6AH	A/D conversion result register	ADCR		R	–	○	–	Undefined
FF70H	PWM control register	PWMC		R/W	○	○	–	05H
FF72H	PWM0 modulo register	PWM0		W	–	–	○	Undefined
FF73H					–	–		
FF74H	PWM1 modulo register	PWM1			–	–	○	
FF75H					–	–		
FF7FH	Clock output mode register	CLOM		R/W	○	○	–	00H
FF80H	Serial interface mode register	CSIM		R/W	○	○	–	00H
FF82H	Serial bus interface control register	SBIC		R/W	○	○	–	00H
FF86H	Serial shift register	SIO			–	○	–	Undefined
FFC0H	Standby control register	STBC			–	○	–	00H
FFC4H	Memory mapping register	MM		W	–	○	–	20H
FFE0H	Interrupt request flag register	IF0L	IF0	R/W	○	○	○	00H
FFE1H		IF0H			○	○		00H
FFE4H	Interrupt mask register	MK0L	MK0		○	○	○	FFH
FFE5H		MK0H			○	○		FFH
FFE8H	Priority specification flag register	PR0L	PR0		○	○	○	FFH
FFE9H		PR0H			○	○		FFH
FFECH	Interrupt service mode register	ISM0L	ISM0		○	○	○	00H
FFEDH		ISM0H			○	○		00H
FFF4H	External interrupt mode register	INTM0			○	○	–	50H
FFF5H	External capture input mode register	INTM1			○	○		0000××01

○: Allowed
–: Not allowed

2.3 DATA MEMORY ADDRESSING

Fig. 2-8 shows the data memory map of the μPD78138 and the applied addressing scheme. With these various addressing, μPD78138 programs can be coded efficiently.

Fig. 2-8 Memory Map and Addressing of Data Memory



*1 If $\overline{EA}$ is low, external memory is mapped.

*2 See the following Table

*3 See the following Table.

	μPD78134A	μPD78136	μPD78138
*2	3FFFH	5FFFH	7FFFH
*3	FD80H	FC80H	

Caution Do not place the stack pointer in the SFR area or ROM area.

Remark Shaded portions indicate internal memory.

3. PERIPHERAL HARDWARE FUNCTIONS

3.1 PORT FUNCTIONS

The μPD78138 is provided with the digital I/O ports shown in Table 3-1.

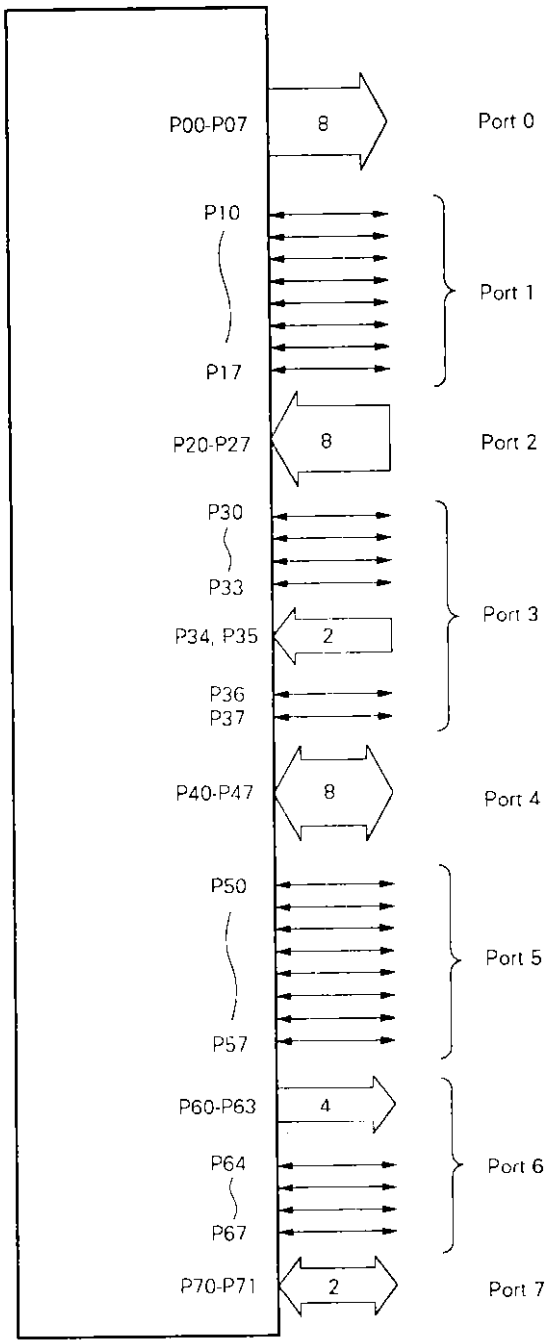
Every port enables 1-bit manipulation on I/O data as well as 8-bit manipulation, providing various ways of control. The use of a built-in pull-up resistor can be specified for ports 1 to 7 by software. Fig. 3-1 shows the configuration of the ports.

Table 3-1 Port Functions ★

Port	Pin	Function	Specification of the use of a pull-up resistor by software
Port 0	P00-P07	Output or high impedance can be specified in units of 8 bits. Also functions as a real-time output port of one 8-bit channel or two 4-bit channels.	—
Port 1	P10-P17	Input or output can be specified bit by bit. Can directly drive an LED.	For the pins in the input mode at one time
Port 2	P20-P27	8-bit input port	In units of 6 bits at one time (P22-P27)
Port 3	P30-P37	8-bit I/O port Input or output can be specified bit by bit. P30-P33, P36, P37: I/O can be specified bit by bit. P34, P35: Port for input only	For the pins in the input mode at one time
Port 4	P40-P47	8-bit I/O port Input or output can be specified in units of 8 bits.	In units of 8 bits at one time
Port 5	P50-P57	8-bit I/O port Input or output can be specified bit by bit.	For the pins in the input mode at one time
Port 6	P60-P67	8-bit I/O port P60-P63: Port for output only P64-P67: Input or output can be specified bit by bit.	For the pins in the input mode at one time (P64-P67)
Port 7	P70-P71	2-bit I/O port Input or output can be specified in units of 2 bits.	In units of 2 bits at one time

Caution In the ports which provide bit by bit specification of the pull-up resistors, the pull-up resistors can be used only for port pins in the input mode.

Fig. 3-1 Port Configuration

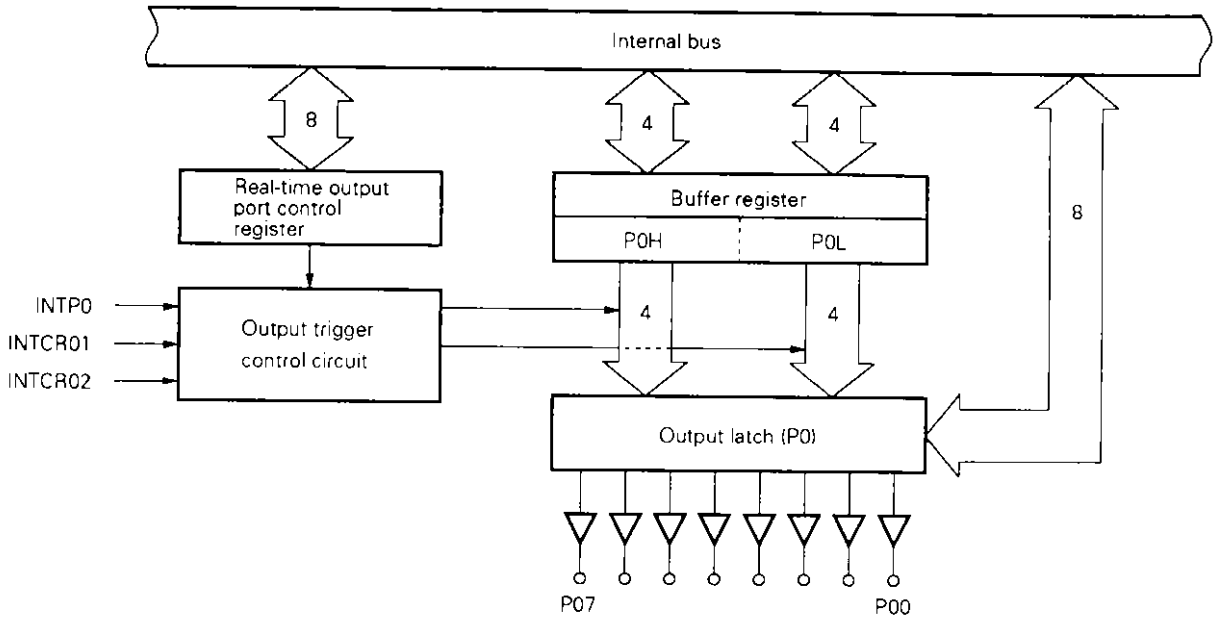


3.2 REAL-TIME OUTPUT PORT

As shown in Fig. 3-2, the real-time output port consists of port 0, buffer registers, and other hardware.

Upon generation of a timer match interrupt or external interrupt, data stored in the buffer registers is transferred to the output latch on a hardware basis for output. This enables obtaining pulse output without jitters. So, this port is suitable for the application of arbitrary pattern output at any intervals such as controlling the open loop of a stepping motor.

Fig. 3-2 Configuration of Real-Time Output Port



Remark INTP0: Valid edge input to the P21/INTP0 pin

INTCR01, INTCR02: Signal issued when the contents of 16-bit timer (TM0) and compare register (CR01, CR02) match

3.3 SUPER TIMER UNIT

The μPD78138 contains a super timer unit which consists of the following timer units.

Table 3-2 Components of the Super Timer Unit

Unit name	Timer/counter	Register	Remarks
Timer 0	16-bit timer × 1 (TM0)	16-bit compare register × 3	Contains an auxiliary 6-bit counter
Free running counter	18-bit counter × 1 (FRC)	16-bit capture register × 3 18-bit capture register × 1	Contains a digital noise eliminator.
Timer 1	16-bit timer × 1 (TM1)	16-bit compare register × 2 16-bit capture register × 1	• Contains an auxiliary 6-bit counter • Pulse width detection function
	7-bit count register × 1 (TM3)	7-bit compare register × 1 7-bit capture register × 1	
Timer 2	16-bit timer × 1 (TM2)	16-bit compare register × 1	
PWM output	(8+4)-bit counter × 2 (PWM0, PWM1)	16-bit modulo register × 2 (with 12 bits used)	Variable active level of output.

Fig. 3-3 shows the configuration of the super timer unit.

The most significant feature of the super timer unit of the μPD78138 includes timer 0 (TM0), the free running counter (FRC), and timer 1 (TM1).

Fig. 3-4 shows the configuration of these timer units. The timer units facilitate VCR index search, DC motor control, and servo control using software.

Fig. 3-3 Configuration of the Super Timer Unit

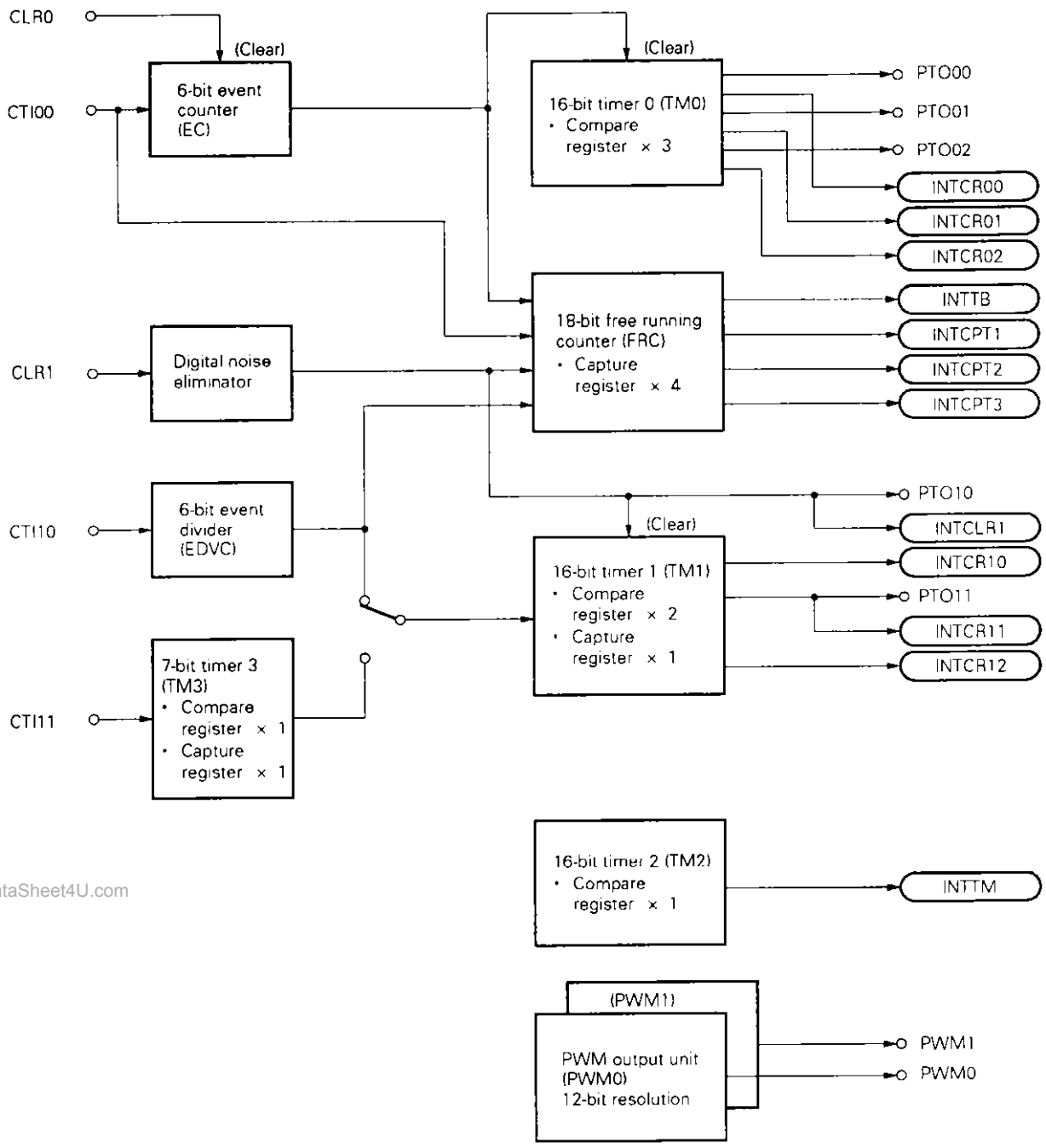
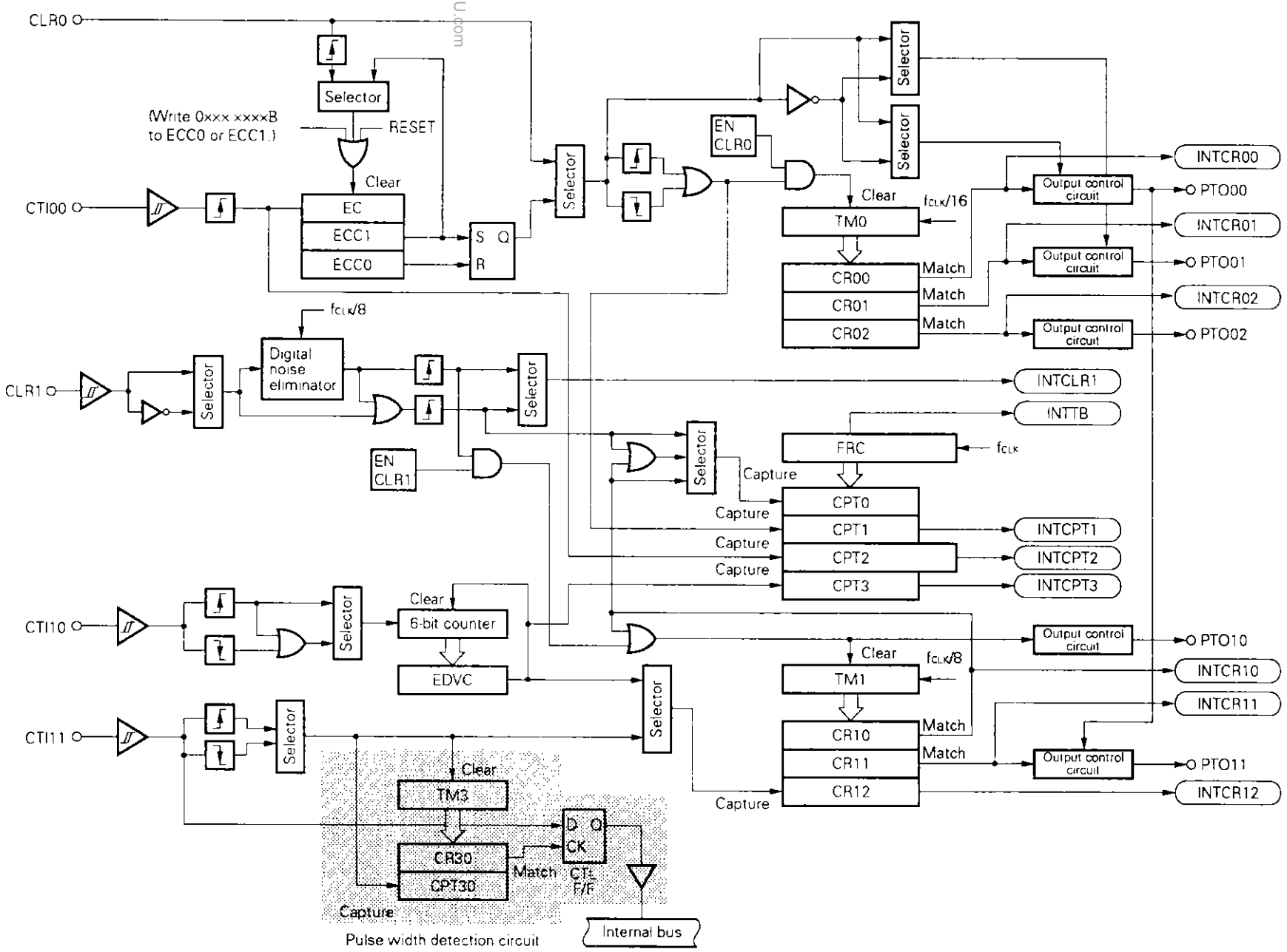


Fig. 3-4 Configuration of Timer 0, Timer 1, and the Free Running Counter



3.3.1 Overview of the Super Timer Unit

(1) Timer 0 (TM0): 16-bit timer

Timer 0 is a timer unit suitable for pulse output timing control. By using an external input signal, TM0 enables the timing of pulse output to be delayed by programming. Three channels of pulse output are available, and can be used, for example, for VCR sound and video head switching signals.

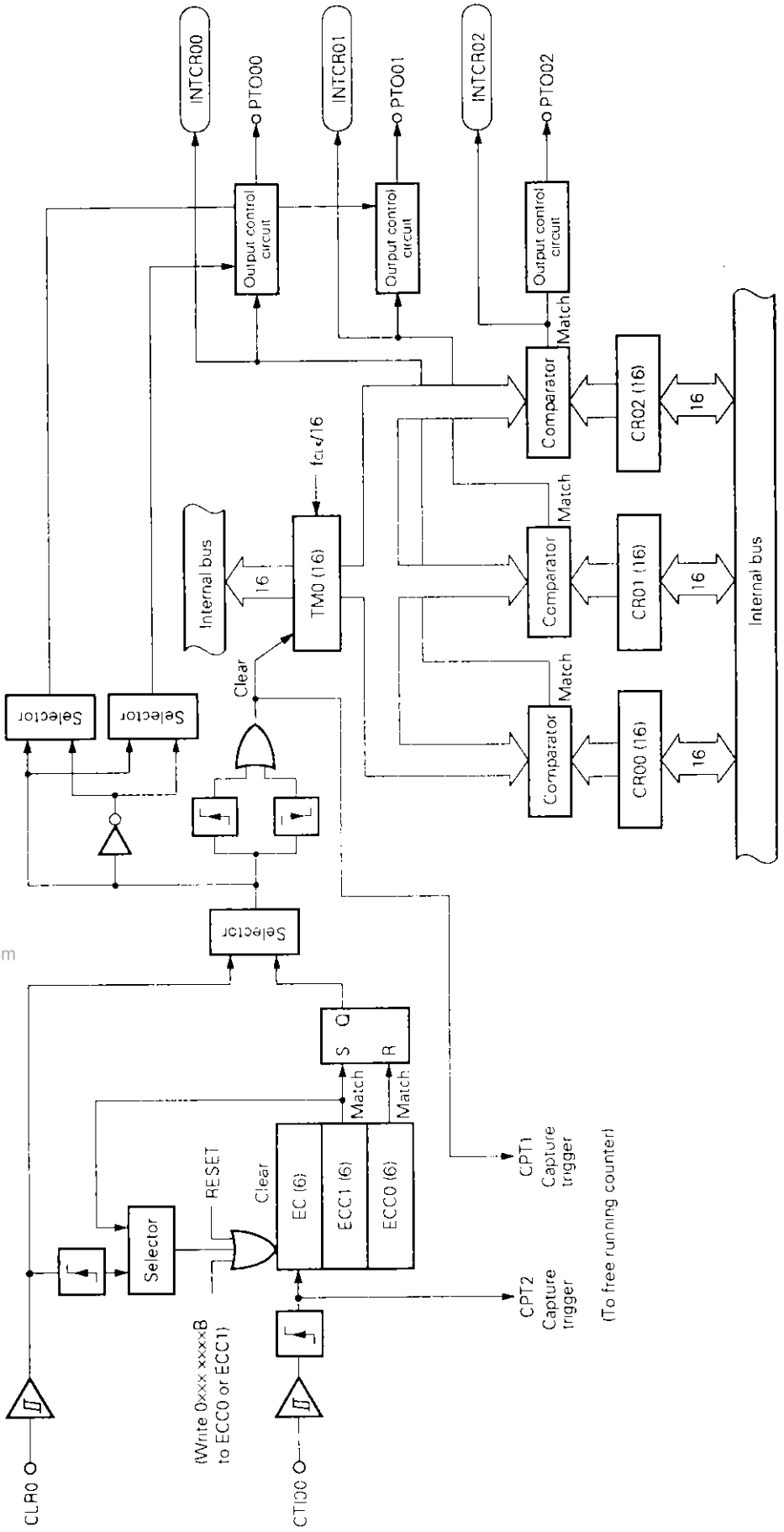
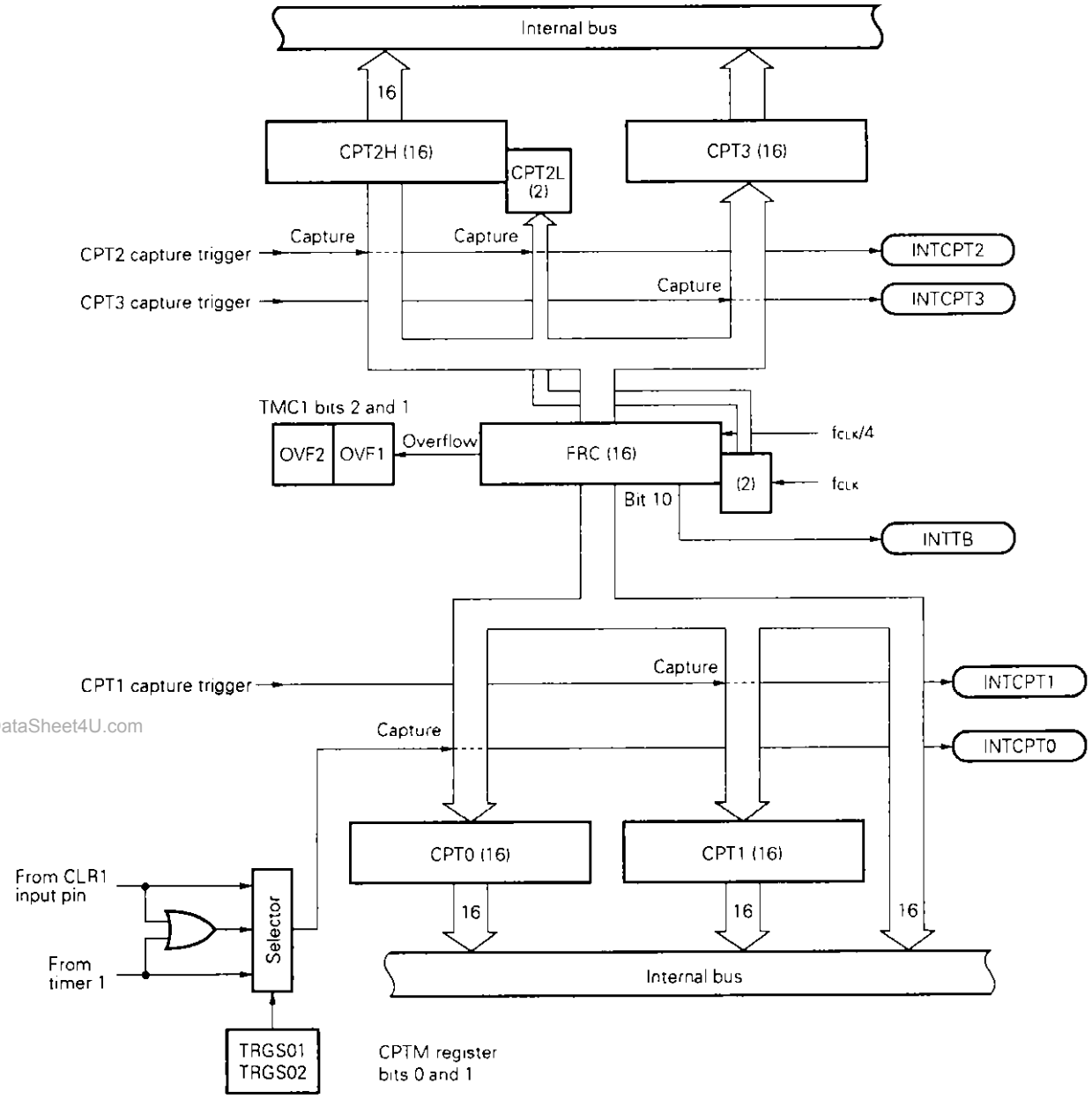


Fig. 3-5 Configuration of Timer 0

(2) Free running counter (FRC): 18-bit counter

The free running counter can be used for external pulse period measurement. This unit contains four capture registers, so that period measurements can be performed for four triggers in parallel. Since an 18-bit counter is used for this unit, a high-precision phase and speed detection is possible for a VCR drum rotating at high speed.

Fig. 3-6 Configuration of the Free Running Counter



(3) **Timer 1 (TM1): 16-bit timer**

Timer 1 is a timer unit for generating a reference signal for internal processing. Timer 1 can be used for various applications such as pulse output and reference signal generation using external trigger input. Timer 1 also allows programmable delay pulse output as with timer 0. Timer 1 contains timer 3 (TM3), which is a 7-bit timer. Timer 3 can be used for external pulse width detection and period measurement.

Fig. 3-7 Configuration of Timer 1

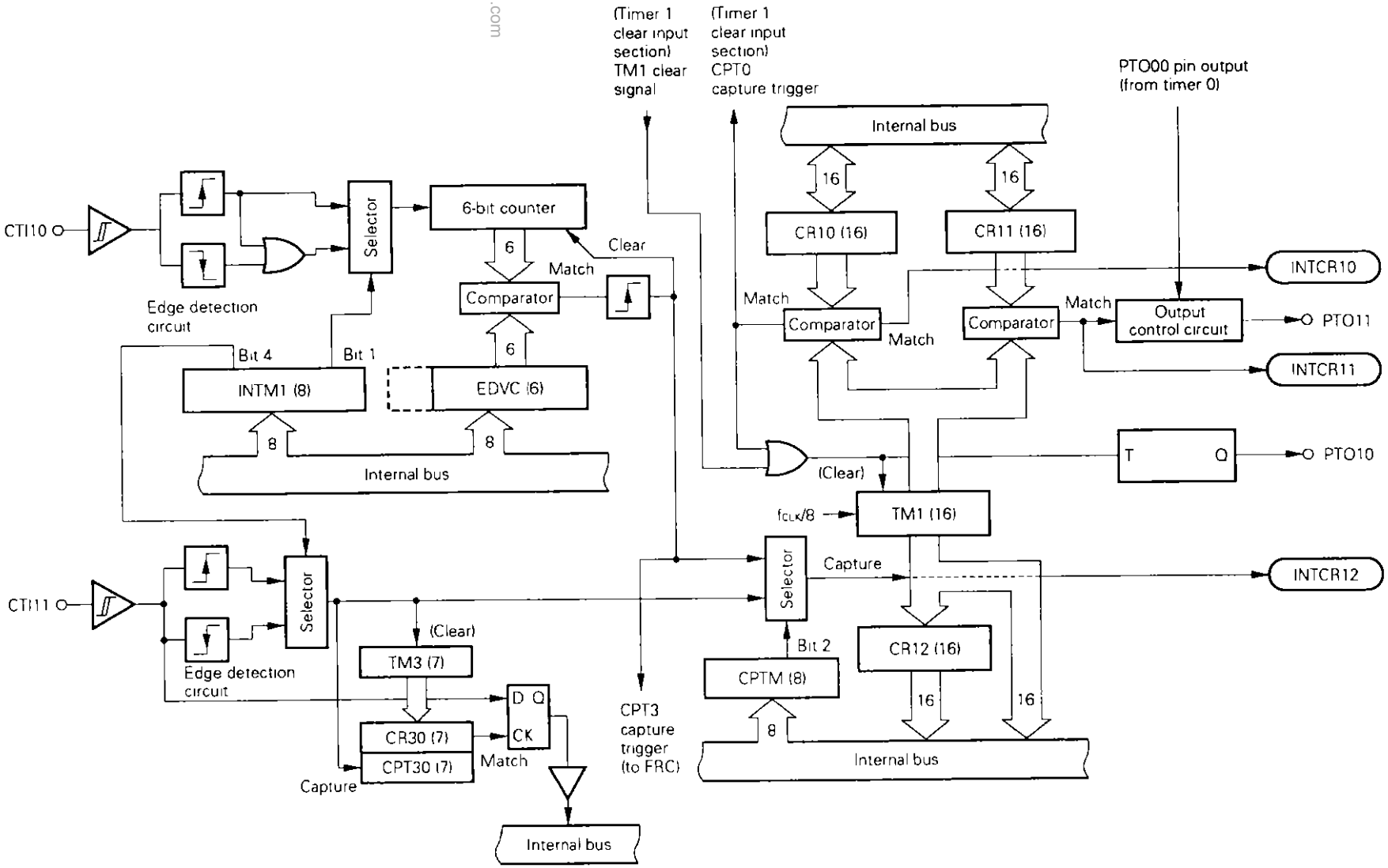
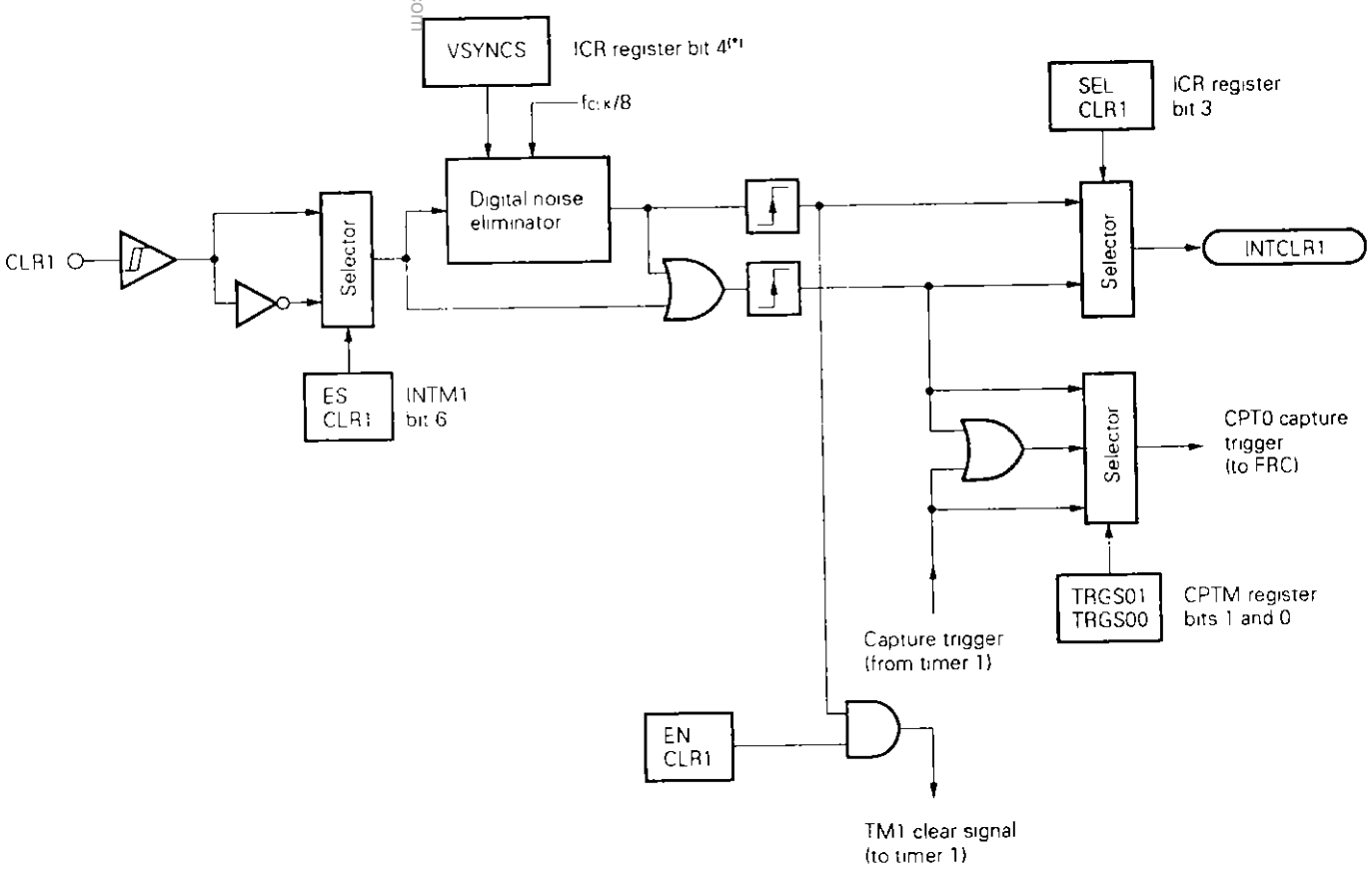


Fig. 3-8 Configuration of Timer 1 Clear Signal Input Section

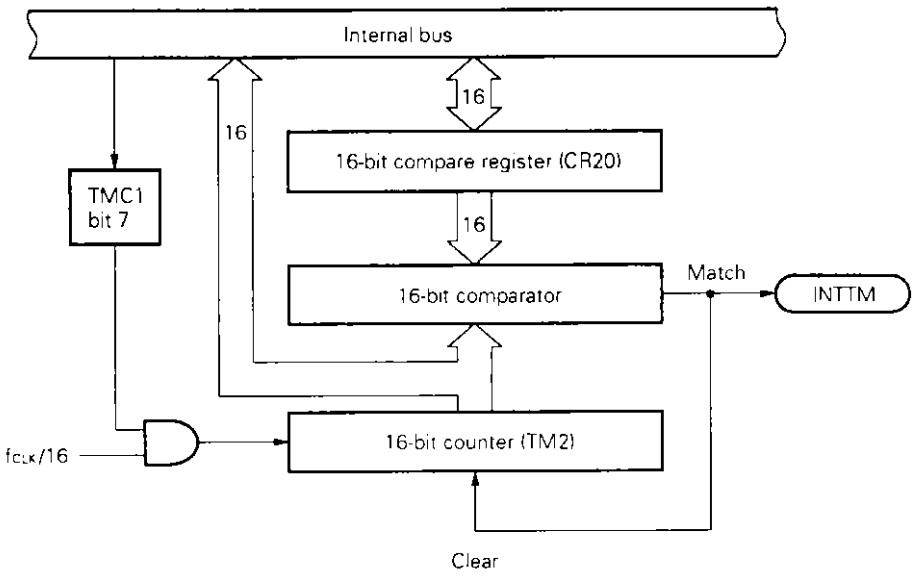


- Pulses with a width in either of the following ranges are removed as noise. Either range is selected according to the setting of bit 4 of the ICR register.
  - Less than ($f_{CLK}/8$) × 5
  - Less than ($f_{CLK}/8$) × 10

(4) Timer 2 (TM2): 16-bit timer

Timer 2 is a general-purpose 16-bit timer unit. When the contents of the compare register match the contents of timer 2, timer 2 is automatically cleared, and functions as an interval timer to initiate an interrupt at the same time.

Fig. 3-9 Block Diagram of Timer 2

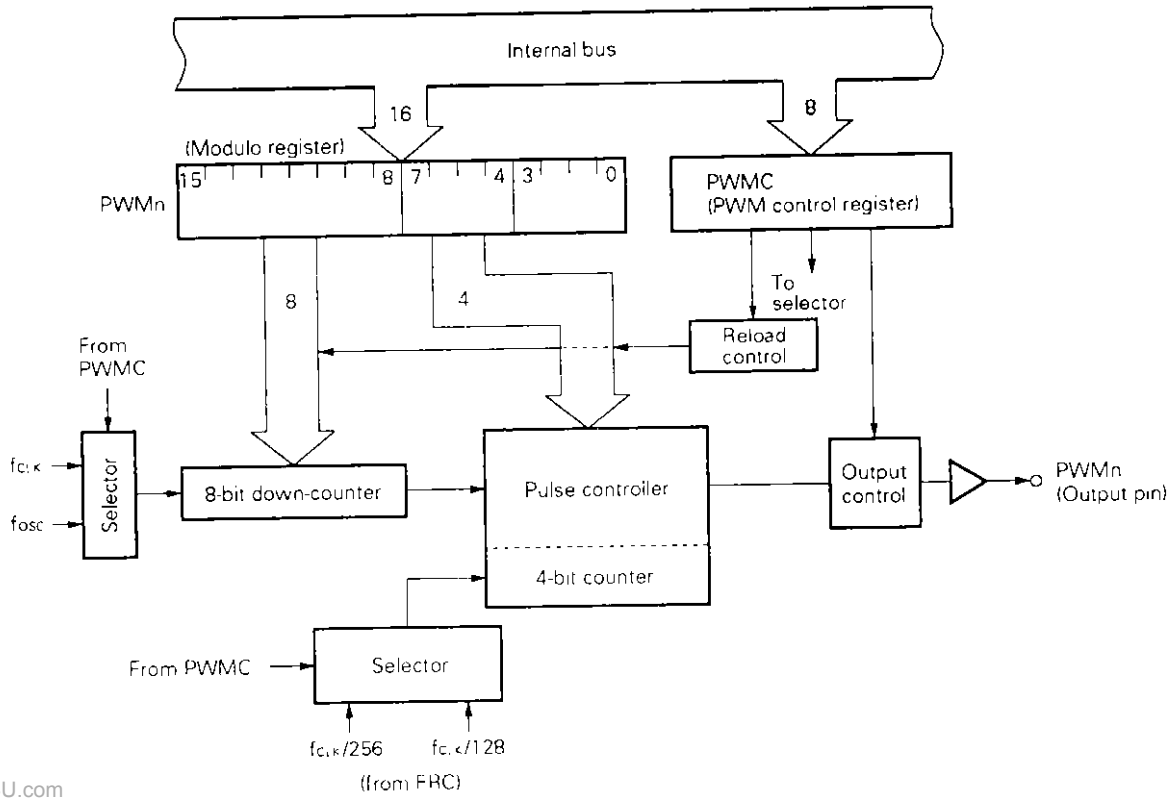


(5) PWM output (PWM0, PWM1): 12-bit PWM

This unit contains 2 channels of a pulse width modulation (PWM) output circuit with a 12-bit resolution. An active level, high or low, can be selected for each channel independently. This unit is most suitable for DC motor speed control.

Either 23.4 kHz or 46.9 kHz (at 12 MHz) can be selected as the carrier frequency of the PWM output of the μPD78138. This 46.9-kHz PWM output especially enables the servo control with better responsibility.

Fig. 3-10 Block Diagram of the PWM Output Unit (n = 0, 1)



Remark $f_{CLK} = f_{osc}/2$

3.4 A/D CONVERTER

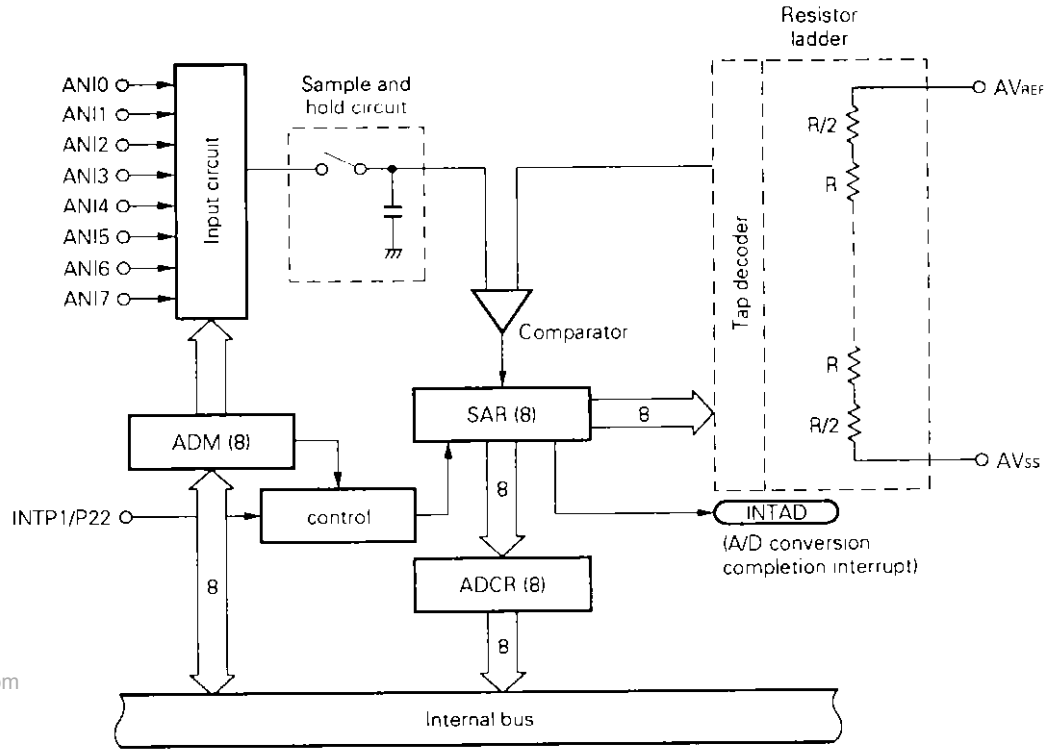
The μPD78138 contains a fast, high-accuracy 8-bit analog-to-digital (A/D) converter with eight multiplexed analog inputs (ANI0 to ANI7). Fig. 3-11 shows the block diagram of the A/D converter.

The A/D converter uses successive approximation system, and provides the A/D conversion result register (ADCR) to store the converted result.

There are two operation modes: Select mode converts analog input on a particular pin continuously. Scan mode converts analog inputs sequentially. A/D conversion can also be started with the synchronization with the external interrupt request signal.

Conversion requires 30 μs when the system operates at 12 MHz.

Fig. 3-11 Block Diagram of A/D Converter



Caution Be careful not to apply voltage exceeding the reference voltage for A/D converter (AV_{REF}) to the ANI0 to ANI7 pins in any circumstance.

3.5 SERIAL INTERFACE

The μPD78138 is provided with one channel for synchronous serial interface. The serial interface operates in one of the following two modes:

- Three-wire serial interface
- Serial bus interface (SBI)

Remark Serial bus interface (SBI)
The serial bus interface mode enables communication with more than one device by using the two lines including serial clock ($\overline{SCK}$) and serial data bus (SB0). Following the NEC original format, the transmitted data can be identified as "address," "command," or "data" by serial data manipulation.

Fig. 3-12 Example of SBI Connection

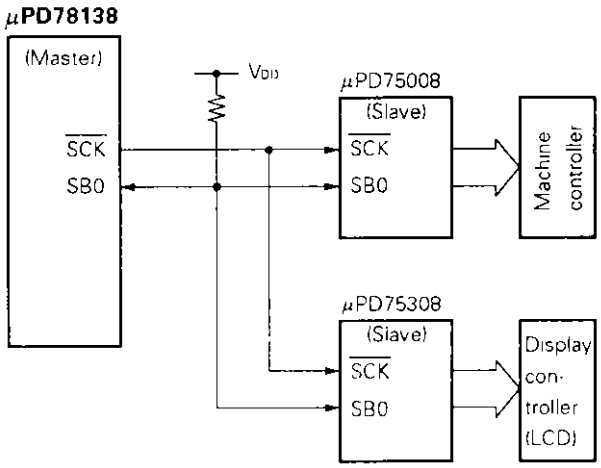
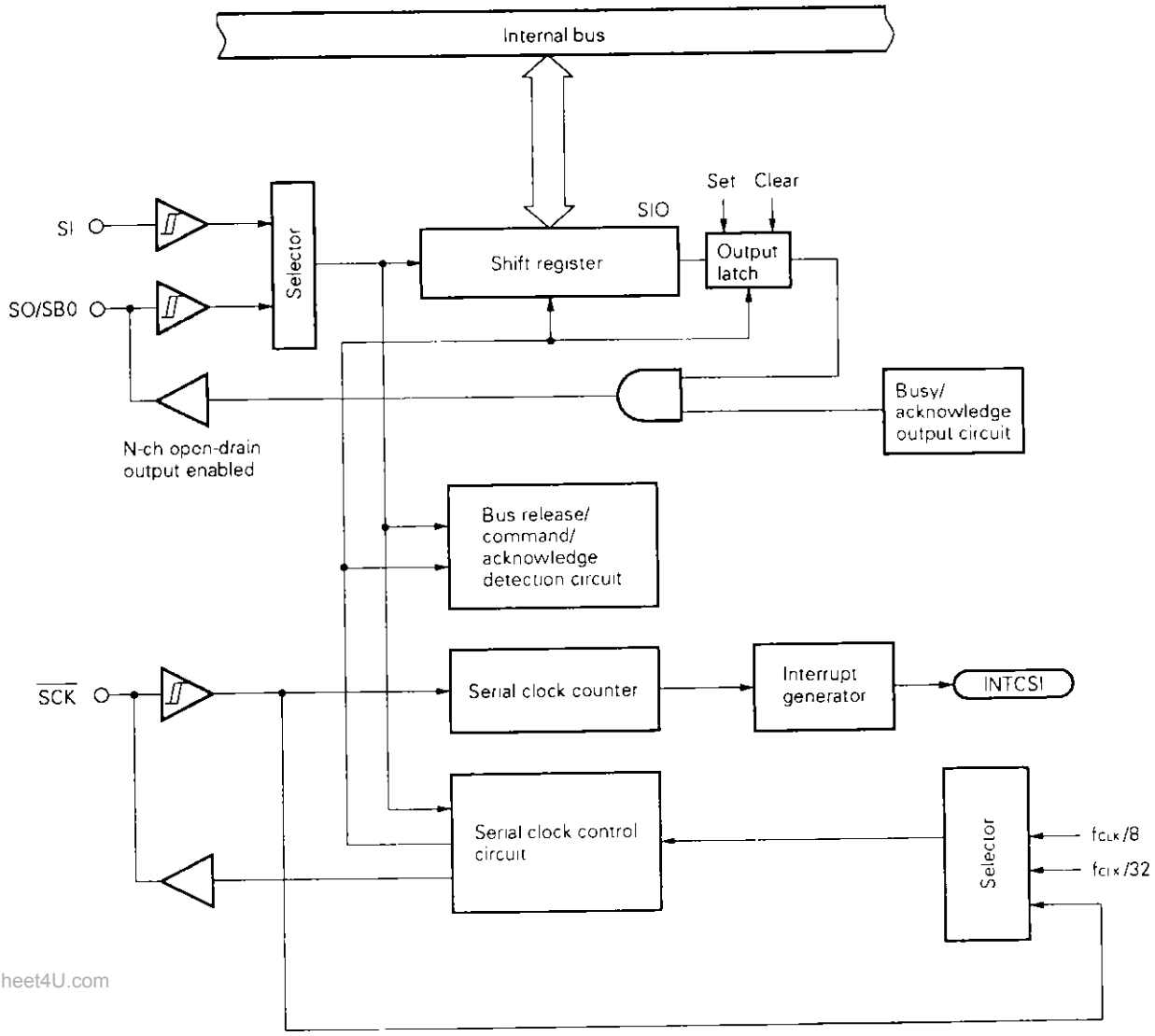


Fig. 3-13 Block Diagram of Synchronous Serial Interface



4. INTERRUPT FUNCTIONS

The μPD78138 has a powerful interrupt function that can handle up to 17 interrupt requests (internal and external). Processing for each interrupt request is divided into two processing modes, vectored interrupt handling and macro service, one of which can be selected by software.

4.1 VECTORED INTERRUPT AND MACRO SERVICE

(1) Vectored interrupt

When an interrupt is accepted, the interrupt service routine is executed according to the data contained in the vector table area. The data indicates the start address of the interrupt service routine created by the user. In the vectored interrupt of the μPD78138, two priority levels can be specified by software. This facilitates the control of multiple interrupts.

(2) Macro service

When an interrupt is accepted, a service set beforehand by firmware is executed.

The following macro services are provided.

- Data transfer mode in which data is transferred between the memory and the special function register (SFR)
- Real-time output port control mode in which the real-time output port can be controlled easily
- Counter mode in which the number of interrupt occurrence is counted
- Data pattern identification mode in which data string from external source is identified

Since these services are not performed via the CPU, the CPU statuses (such as SP and PSW) need not be saved and returned. Thus, CPU service time is greatly improved.

4.2 INTERRUPT SOURCES

Table 4-1 lists the interrupt request sources of the μPD78138. An interrupt vector table is assigned to each source. All the maskable interrupts are provided with macro service routines.

Table 4-1 Interrupt Request Sources

Type	Default priority	Interrupt source		Macro service	Vectors table address
		Name	Trigger		
Nonmaskable	–	NMI	Pin input edge detection	Not provided	0002H
Maskable	0 (highest)	INTP0	Pin input edge detection	Provided	0004H
	1	INTCPT3	EDVC output (CPT3 capture)		0006H
	2	INTCPT2	Pin input edge detection (CPT2 capture)		0008H
	3	INTCR12	Pin input/EDVC (CR12 capture)		000AH
	4	INTCR00	TM0-CR00 match		000CH
	5	INTCLR1	Pin input edge detection		000EH
	6	INTCR10	TM1-CR10 match signal		0010H
	7	INTCR01	TM0-CR01 match signal		0012H
	8	INTCR02	TM0-CR02 match signal		0014H
	9	INTCR11	TM1-CR11 match signal		0016H
	10	INTCPT1	Pin input/ECC0, ECC1		0018H
	11	INTTM	TM2-CR20 match signal		001AH
	12	INTCSI	Serial transfer end		001CH
	13	INTTB	FRC overflow		001EH
	14	INTP1/INTAD	Pin input/A/D conversion end[*]		0020H
	15 (lowest)	INTP2	Pin input edge detection		0022H

EDVC : Event divider compare register
ECC0, ECC1: Event counter compare registers 0 and 1
TM0, TM1 : 16-bit timers 0 and 1
CRxx : Compare register (xx=00, 01, 02, 11, 20)
CR12: Capture register
FRC : 18-bit free running counter

* An external interrupt (INTP1) is also used as an INTAD interrupt (A/D conversion end interrupt).

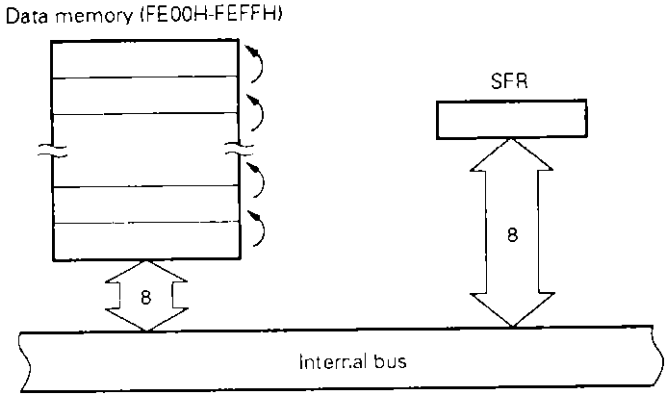
Remark The default priority indicates the priority used when two or more interrupts occur simultaneously.

4.3 MACRO SERVICE FUNCTIONS

One of the interrupt processing functions of the μPD78138 is a built-in macro service function set by the firmware. There are four kinds of macro service function, each of which can be selected by software.

(1) Data transfer mode

When an interrupt occurs, 8-bit data is transmitted between the SFR corresponding to the interrupt and data memory. The address stored in data memory changes every time an interrupt occurs, so that consecutive data can be transmitted. The address in data memory (one from FE00H to FEFFH) and the number of transfer are controlled by software. After the specified number of transfer is completed, a vectored interrupt occurs.

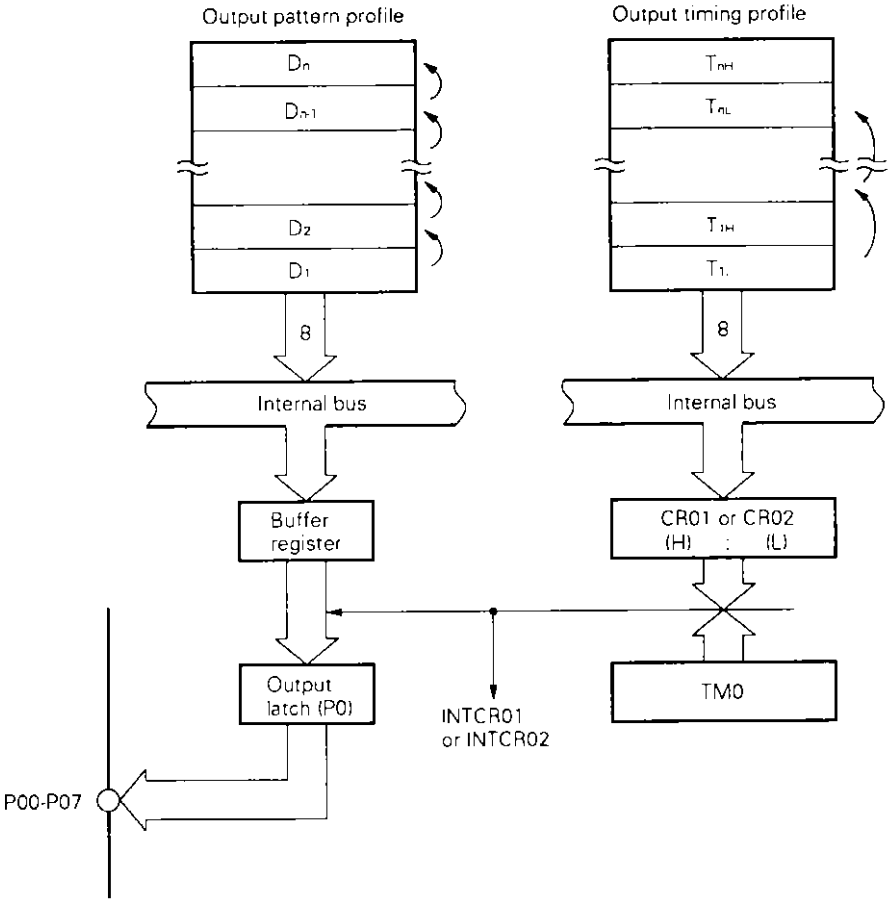


(2) Real-time output port control mode

This macro service is effective the real-time output port. Whenever INTCR01 or INTCR02 occurs, the output pattern profile is transmitted to the buffer registers (P0L, P0H), and the output timing profile is transmitted to the compare register (CR01 or CR02).

After the data transfer of each profile is completed, a vectored interrupt occurs.

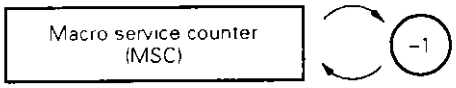
This macro service enables various application such as the control of two stepping motors independently, or control of the PWM output or the output of the Head switching signal of the VCR.



(3) Counter mode

This mode decrements the macro service counter (MSC) according to an interrupt occurrence. This can be used to count the number of interrupt occurrence or to divide an interrupt.

When the counter reaches 0, a vectored interrupt occurs.

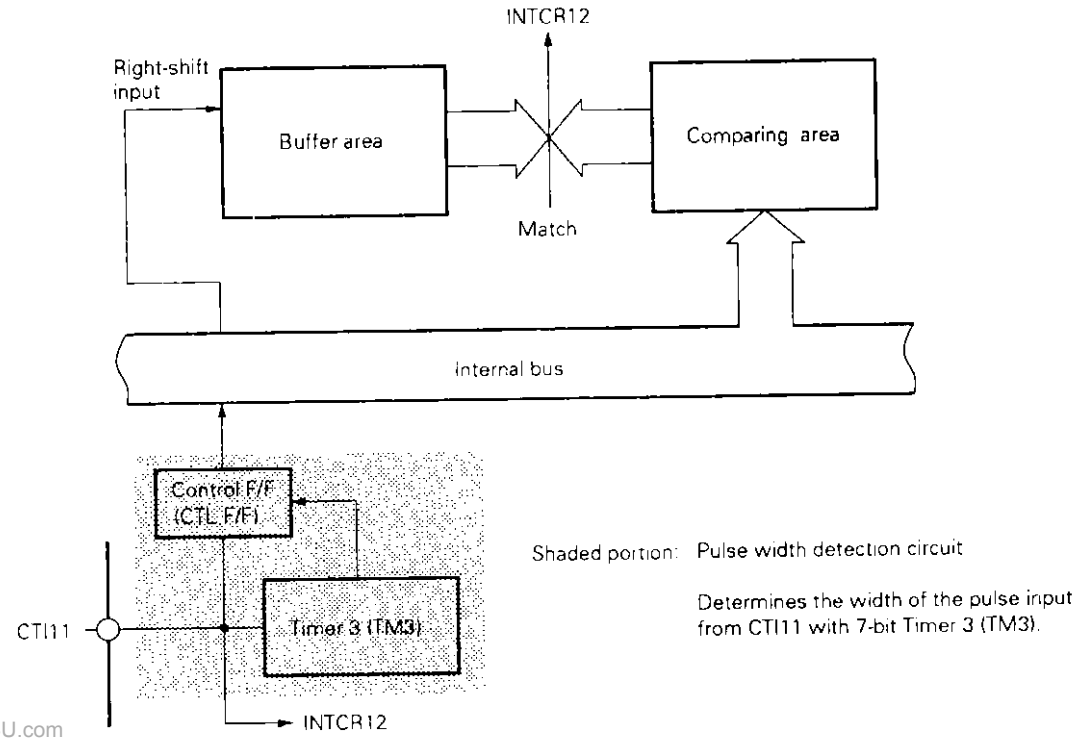


(4) Data pattern identification mode (VCR index search control etc.)

This macro service transfers the data from the control F/F (CTL F/F) to the buffer area when INTCR12 occurs. Then, the buffer area stores the transferred data by shifting it to the right. And if stored data in the buffer area matches the data in the comparing area which is set beforehand, a vectored interrupt occurs (data pattern identification function).

This macro service can be applicable to easy control of the index search of a VCR. It can also be used to issue a vectored interrupt whenever a number of data storing specified by software is reached. In this case, data of the buffer area and data of the comparing area need not match.

Caution Storing of data to the buffer area is fixed to right shift.



4.3.1 Macro Service Modes and Interrupt Requests

The macro service mode depends upon the type of interrupt request source. Table 4-2 lists the correspondence between the macro service modes and interrupt request sources.

Table 4-2 Macro Service Modes and Interrupt Request Sources

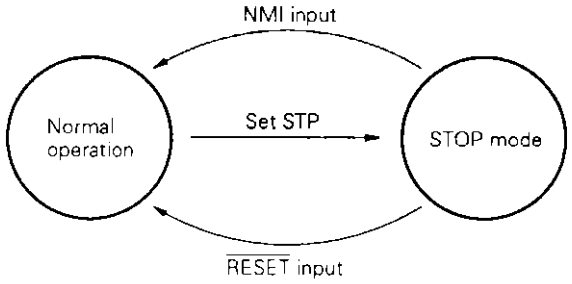
Macro service mode	Interrupt request source which can process the corresponding macro service
Data transfer mode	INTCSI, INTAD
Real-time output port control mode	INTCR01, INTCR02
Counter mode	All maskable interrupt requests
Data pattern identification mode	INTCR12

5. STANDBY FUNCTION

The μPD78138 has a STOP mode as a standby function to reduce power consumption of the system. In this mode, the oscillator is stopped to stop the entire system. Data can be held at very low power consumption in which only a leakage current flows.

The STOP mode is set when the STOP flag (STP) is set to 1 by software. The STOP mode is released by a nonmaskable interrupt (NMI) or reset ($\overline{\text{RESET}}$) input. Fig. 5-1 shows standby status transition.

Fig. 5-1 Standby Status Transition



6. RESET FUNCTION

When input to pin $\overline{\text{RESET}}$ becomes low, the system is reset and each hardware component is put in the status shown in Table 6-1.

When input to pin $\overline{\text{RESET}}$ becomes high, the reset status is released. The contents at address 0000H in a reset vector table are then set in bits 7 to 0 in the program counter (PC) and the contents of bits 4 to 0 at address 0001H are set in bits 12 to 8 in the PC. The branch is taken in this way and program execution starts at the branch destination address. Reset start is possible at any address.

Initialize the contents of registers in the program as required.

A noise eliminator using analog delay is provided for the $\overline{\text{RESET}}$ input pin to prevent miss-operation due to noise (see Fig. 6-1).

For the reset operation at power-on, reserve time for the oscillation to stabilize from power-on to releasing the reset signal as shown in Fig. 6-2.

Fig. 6-1 Accepting a Reset Signal

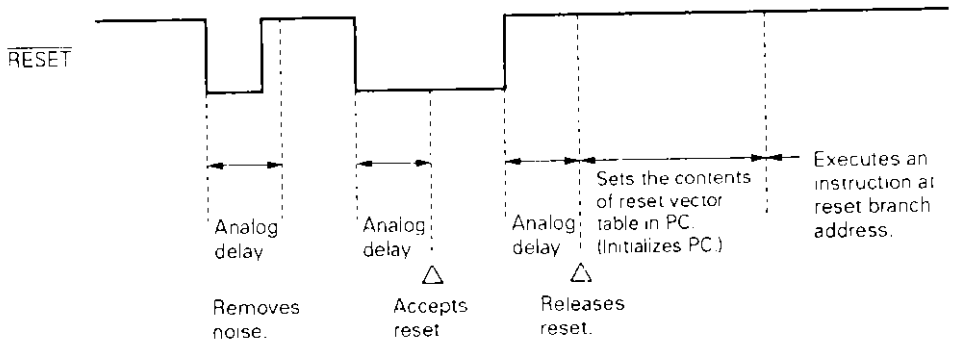


Fig. 6-2 Reset at Power-On

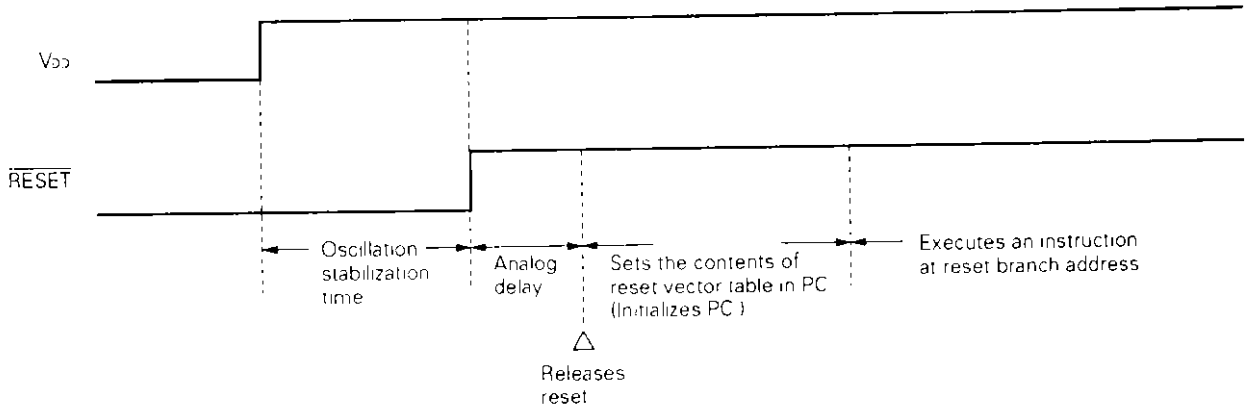


Table 6-1 Hardware Statuses after Reset (1/2)

Hardware			Status after reset
Program counter (PC)			The contents of a reset vector table (0000H, 0001H) are set.
Stack pointer (SP)			Undefined
Program status word (PSW)			02H
Built-in RAM	Data memory		Undefined *
	General registers (X, A, C, B, E, D, L, and H)		
I/O line	P00-P07		High impedance (output buffer off)
	P20-P27, and P34-P37		Input
	P10-P17, P30-P33, P40-P47, P50-P57, P64-67, P70, and P71		Input (output buffer off)
	P60-P63		Low-level output
Output latch	Ports 0, 1, 3, 4, 5, and 7		Undefined
	Port 6		××××0000
Port mode register	PM0, PM1, PM3, PM5, and PM7		FFH
	PM6		F0H
Port 3 mode control register (PMC3)			30H
Memory mapping register (MM)			20H
Register for optional pull-up resistor (PUO)			00H
Super timer unit	Counters (TM0, TM1, FRC, and TM2)		Undefined for up to 16 clock pulses after releasing the reset signal. Zero clear after 17th clock pulse after releasing the reset signal.
	Compare registers (CR00, CR01, CR02, CR10, CR11, and CR20)		Undefined
	Capture registers (CR12, CPT0, CPT1, CPT2H, CPT2L, and CPT3)		
	Timer control register 0 (TMC0)		0××00000
	Timer control register 1 (TMC1)		00H
	Capture mode register (CPTM)		×××××0×0
	Input control register (ICR)		0×0×0×××
	External capture input mode register (INTM1)		0000××01
	Event counters (CTI00 input section)	EC	××000000
		ECC0	××111111
		ECC1	××111111
	Event divider control register (CTI10 input section)	EDVC	Undefined
	Pulse width detection circuit (CTI11 input section)	TM3	00H
		PRM3	0××××000
		CR30	×1111111
		CPT30	Undefined
	Timer output mode register	TOM0	××000000
		TOM1	××××0000
	Timer output control register	TOC0	××000000
		TOC1	××××0000

* When the STOP mode is released by $\overline{\text{RESET}}$ input, the values before the STOP mode was set are restored.

Table 6-1 Hardware Statuses after Reset (2/2)

Hardware			Status after reset
Super timer unit	PTO10 output (PTO10)		High-level output
	PWM outputs (PWM0 and PWM1)		Low-level output
	PWM control register (PWMC)		05H
	PWM modulo registers (PWM0 and PWM1)		Undefined
Real-time output port	Port 0 buffer registers (P0L and P0H)		Undefined
	Real-time output port control register (RTPC)		00H
AD converter	Mode register (ADM)		00H
	AD conversion result register (ADCR)		Undefined
Serail interface	Mode register (CSIM)		00H
	Shift register (SIO)		Undefined
	Serial bus control register (SBIC)		00H
Interrupt	Interrupt request flag registers	(IF0H)	00H
		(IF0L)	00H
	Interrupt mask registers	(MK0H)	FFH
		(MK0L)	FFH
	Interrupt priority specification flag registers	(PR0H)	FFH
		(PR0L)	FFH
	Interrupt service mode registers	(ISM0H)	00H
		(ISM0L)	00H
	External interrupt mode registers	(INTM0)	50H
Standby control register (STBC)			00H
Clock output mode register (CLOM)			00H
ASTB/CLO output			Low-level output

7. INSTRUCTION SET

7.1 INSTRUCTION SET AND OPERATIONS OF INDIVIDUAL INSTRUCTIONS

(1) Operand notation and coding format

Operands are coded in the operand field of each instruction as listed in the coding column of the table below. 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. These symbols have the following meanings.

- \+ : Auto increment
- # : Immediate data
- ! : Address by immediate addressing
- $: Address by relative addressing
- / : Bit inversion
- [ ] : Indirect addressing

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

Notation	Coding
r, r'	X(R0), A(R1), C(R2), B(R3), E(R4), D(R5), L(R6), H(R7)
r1	A, B
r2	B, C
r3	D, E, E+
r4	D, E
rp, rp'	AX(RP0), BC(RP1), DE(RP2), HL(RP3)
sfr	Special function register abbreviation
sfrp	Special function register abbreviation (16-bit manipulation register)
saddr	FE20H-FF1FH Immediate data or label
saddrp	FE20H-FF1EH Immediate data (bit 0=0, however) or label (for 16-bit manipulation)
!addr16	0000H-FFFFH Immediate data or label: Immediate addressing
$addr16	0000H-FFFFH Immediate data or label: Relative addressing
addr11	800H-FFFH Immediate data or label
addr5	40H-7EH Immediate data (bit 0=0, however) 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)
RBn	RB0-RB3

Remarks 1. Absolute names (R0 to R7 and RP0 to RP3) can be specified in r, r', rp, and rp', as well as functional names (X, A, C, B, E, D, L, H, AX, BC, DE, and HL).

2. 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	:	Register A; 8-bit accumulator
X	:	Register X
B	:	Register B
C	:	Register C
D	:	Register D
E	:	Register E
H	:	Register H
L	:	Register L
R0-R7	:	Register 0 to register 7 (absolute name)
AX	:	Register pair (AX); 16-bit accumulator
BC	:	Register pair (BC)
DE	:	Register pair (DE)
HL	:	Register pair (HL)
RP0-RP3	:	Register pair 0 to register pair 3 (absolute name)
PC	:	Program counter
SP	:	Stack pointer
PSW	:	Program status word
CY	:	Carry flag
AC	:	Auxiliary carry flag
Z	:	Zero flag
RBS0-RBS1	:	Register bank select flag
IE	:	Interrupt request enable flag
STBC	:	Standby control register
jdisp8	:	Signed 8-bit data (displacement: −128 to +127)
( )	:	Contents at an address enclosed in parentheses or at an address indicated in a register enclosed in parentheses
××H	:	Hexadecimal number
×H, ×L	:	Eight high-order bits and eight low-order bits of 16-bit register pair

(3) Descriptions in clock field

(i) The digit in the clock filed is the number of clocks when the program is fetched from the internal ROM.

(ii) The number of clocks varies according to the memory area to be accessed.
Refer to the User's Manual for details.

(4) 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.
R	Saved values are restored.

Instruction set	Mnemonic	Operand	Byte	Clock	Operation	Flag Z	Flag AC	Flag CY
8-bit data transfer	MOV	r, #byte	2	2	r←byte			
		saddr, #byte	3	3/5	(saddr)←byte			
		sfr, #byte (*1)	3	5	sfr←byte			
		r, r'	2	2	r←r'			
		A, r	1	2	A←r			
		A, saddr	2	2/4	A←(saddr)			
		saddr, A	2	3/5	(saddr)←A			
		A, sfr	2	4	A←sfr			
		sfr, A	2	5	sfr←A			
		A, [r3] (*2)	1	5	A←(FE00H+r3) r3=00H-FFH			
		[r3], A (*2)	1	5	(FF00H+r3)←A r3=00H-FFH			
		A, [HL]	1	5-7	A←(HL)			
		[HL], A	1	5/7	(HL)←A			
		A, [HL+]	1	8-10	A←(HL), HL←HL+1			
		[HL+], A	1	8/10	(HL)←A, HL←HL+1			
		A, [DE]	1	5-7	A←(DE)			
		[DE], A	1	5/7	(DE)←A			
		A, [DE+]	1	8-10	A←(DE), DE←DE+1			
		[DE+], A	1	8/10	(DE)←A, DE←DE+1			
		A, !addr16	4	6-8	A←!addr16			
		!addr16, A	4	6-8	addr16←A			
		A, word[r1]	4	7-9	A←(word+r1)			
		word[r1], A	4	7/9	(word+r1)←A			
		PSW, #byte	3	5	PSW←byte	×	×	×
		PSW, A	2	5	PSW←A	×	×	×
		A, PSW	2	4	A←PSW			
	XCH	A, r	1	4	A↔r			
		A, saddr	2	4/8	A↔(saddr)			
		A, sfr	3	10	A↔sfr			
		A, [r4]	1	9	A↔(FE00H+r4) r4=00H-FFH			
		A, [HL]	2	9/13	A↔(HL)			
		A, [DE]	2	9/13	A↔(DE)			
		A, word[r1]	4	9/13	A↔(word+r1)			

*1 If STBC is written in sfr, a different dedicated instruction having the different byte and clock counts is generated. (See CPU control instructions.)

*2 If E+ is written in r3, the contents of register E are incremented by 1 after instruction execution, and the number of clocks is 6.

Instruction set	Mnemonic	Operand	Byte	Clock	Operation	Flag Z	Flag AC	Flag CY
16-bit data transfer	MOVW	rp, #word	3	3	rp←word			
		saddrp, #word	4	4/8	(saddrp+1)(saddrp)←word			
		sfrp, #word	4	8	sfrp←word			
		rp, rp'	2	4	rp←rp'			
		AX, saddrp	2	6/10	AX←(saddrp+1)(saddrp)			
		saddrp, AX	2	5/9	(saddrp+1)(saddrp)←AX			
		AX, sfrp	2	10	AX←sfrp			
		sfrp, AX	2	9	sfrp←AX			

Instruction set	Mnemonic	Operand	Byte	Clock	Operation	Flag Z	Flag AC	Flag CY
8-bit arithmetic/logical	ADD	A, #byte	2	2	A, CY←A+byte	×	×	×
		saddr, #byte	3	4/8	(saddr), CY←(saddr)+byte	×	×	×
		sfr, #byte	4	10	sfr, CY←sfr+byte	×	×	×
		r, r'	2	3	r, CY←r+r'	×	×	×
		A, saddr	2	3/5	A, CY←A+(saddr)	×	×	×
		A, sfr	3	7	A, CY←A+sfr	×	×	×
		A, [r4]	2	7	A, CY←A+(FE00H+r4) r4=00H-FFH	×	×	×
		A, [HL]	2	8-10	A, CY←A+(HL)	×	×	×
		A, [DE]	2	8-10	A, CY←A+(DE)	×	×	×
		A, word[r1]	4	8-10	A, CY←A+(word+r1)	×	×	×
	ADDC	A, #byte	2	2	A, CY←A+byte+CY	×	×	×
		saddr, #byte	3	4/8	(saddr), CY←(saddr)+byte+CY	×	×	×
		sfr, #byte	4	10	sfr, CY←sfr+byte+CY	×	×	×
		r, r'	2	3	r, CY←r+r'+CY	×	×	×
		A, saddr	2	3/5	A, CY←A+(saddr)+CY	×	×	×
		A, sfr	3	7	A, CY←A+sfr+CY	×	×	×
		A, [r4]	2	7	A, CY←A+(FE00H+r4)+CY r4=00H-FFH	×	×	×
		A, [HL]	2	8-10	A, CY←A+(HL)+CY	×	×	×
		A, [DE]	2	8-10	A, CY←A+(DE)+CY	×	×	×
		A, word[r1]	4	8-10	A, CY←A+(word+r1)+CY	×	×	×
	SUB	A, #byte	2	2	A, CY←A-byte	×	×	×
		saddr, #byte	3	4/8	(saddr), CY←(saddr)-byte	×	×	×
		sfr, #byte	4	10	sfr, CY←sfr-byte	×	×	×
		r, r'	2	3	r, CY←r-r'	×	×	×
		A, saddr	2	3/5	A, CY←A-(saddr)	×	×	×
		A, sfr	3	7	A, CY←A-sfr	×	×	×
		A, [r4]	2	7	A, CY←A-(FE00H+r4) r4=00H-FFH	×	×	×
		A, [HL]	2	8-10	A, CY←A-(HL)	×	×	×
		A, [DE]	2	8-10	A, CY←A-(DE)	×	×	×
		A, word[r1]	4	8-10	A, CY←A-(word+r1)	×	×	×
	SUBC	A, #byte	2	2	A, CY←A-byte-CY	×	×	×
		saddr, #byte	3	4/8	(saddr), CY←(saddr)-byte-CY	×	×	×
		sfr, #byte	4	10	sfr, CY←sfr-byte-CY	×	×	×
		r, r'	2	3	r, CY←r-r'-CY	×	×	×
		A, saddr	2	3/5	A, CY←A-(saddr)-CY	×	×	×
		A, sfr	3	7	A, CY←A-sfr-CY	×	×	×
		A, [r4]	2	7	A, CY←A-(FE00H+r4)-CY r4=00H-FFH	×	×	×
		A, [HL]	2	8-10	A, CY←A-(HL)-CY	×	×	×
		A, [DE]	2	8-10	A, CY←A-(DE)-CY	×	×	×
		A, word[r1]	4	8-10	A, CY←A-(word+r1)-CY	×	×	×

Instruction set	Mnemonic	Operand	Byte	Clock	Operation	Flag Z	Flag AC	Flag CY
8-bit arithmetic/logical	AND	A, #byte	2	2	A←A∧byte	×		
		saddr, #byte	3	4/8	(saddr)←(saddr)∧byte	×		
		sfr, #byte	4	10	sfr←sfr∧byte	×		
		r, r'	2	3	r←r∧r'	×		
		A, saddr	2	3/5	A←A∧(saddr)	×		
		A, sfr	3	7	A←A∧sfr	×		
		A, [r4]	2	7	A←A∧(FE00H+r4) r4=00H-FFH	×		
		A, [HL]	2	8-10	A←A∧(HL)	×		
		A, [DE]	2	8-10	A←A∧(DE)	×		
		A, word[r1]	4	8-10	A←A∧(word+r1)	×		
	OR	A, #byte	2	2	A←A∨byte	×		
		saddr, #byte	3	4/8	(saddr)←(saddr)∨byte	×		
		sfr, #byte	4	10	sfr←sfr∨byte	×		
		r, r'	2	3	r←r∨r'	×		
		A, saddr	2	3/5	A←A∨(saddr)	×		
		A, sfr	3	7	A←A∨sfr	×		
		A, [r4]	2	7	A←A∨(FE00H+r4) r4=00H-FFH	×		
		A, [HL]	2	8-10	A←A∨(HL)	×		
		A, [DE]	2	8-10	A←A∨(DE)	×		
		A, word[r1]	4	8-10	A←A∨(word+r1)	×		
	XOR	A, #byte	2	2	A←A▽byte	×		
		saddr, #byte	3	4/8	(saddr)←(saddr)▽byte	×		
		sfr, #byte	4	10	sfr←sfr▽byte	×		
		r, r'	2	3	r←r▽r'	×		
		A, saddr	2	3/5	A←A▽(saddr)	×		
		A, sfr	3	7	A←A▽sfr	×		
		A, [r4]	2	7	A←A▽(FE00H+r4) r4=00H-FFH	×		
		A, [HL]	2	8-10	A←A▽(HL)	×		
		A, [DE]	2	8-10	A←A▽(DE)	×		
		A, word[r1]	4	8-10	A←A▽(word+r1)	×		
	CMP	A, #byte	2	2	A-byte	×	×	×
		saddr, #byte	3	3/5	(saddr)-byte	×	×	×
		sfr, #byte	4	7	sfr-byte	×	×	×
		r, r'	2	3	r-r'	×	×	×
		A, saddr	2	3/5	A-(saddr)	×	×	×
		A, sfr	3	7	A-sfr	×	×	×
		A, [r4]	2	7	A-(FE00H+r4) r4=00H-FFH	×	×	×
		A, [HL]	2	8-10	A-(HL)	×	×	×
		A, [DE]	2	8-10	A-(DE)	×	×	×
		A, word[r1]	4	8-10	A-(word+r1)	×	×	×

Instruction set	Mnemonic	Operand	Byte	Clock	Operation	Flag Z	Flag AC	Flag CY
16-bit arithmetic/logical	ADDW	AX, #word	3	4	AX, CY←AX+word	×	×	×
		AX, rp	2	6	AX, CY←AX+rp	×	×	×
		AX, saddrp	2	7/11	AX, CY←AX+(saddrp+1) (saddrp)	×	×	×
		AX, sfrp	3	13	AX, CY←AX+sfrp	×	×	×
	SUBW	AX, #word	3	4	AX, CY←AX−word	×	×	×
		AX, rp	2	6	AX, CY←AX−rp	×	×	×
		AX, saddrp	2	7/11	AX, CY←AX−(saddrp+1) (saddrp)	×	×	×
		AX, sfrp	3	13	AX, CY←AX−sfrp	×	×	×
	CMPW	AX, #word	3	3	AX−word	×	×	×
		AX, rp	2	5	AX−rp	×	×	×
		AX, saddrp	2	6/10	AX−(saddrp+1) (saddrp)	×	×	×
		AX, sfrp	3	12	AX−sfrp	×	×	×
Multiply/divide	MULSW	r(*)	2	47	AX(16 high-order bits), r1(8 low-order bits)← AX (signed 16 bits) × r1(absolute 8 bits)			
	MULUW	r(*)	2	47	AX(16 high-order bits), r(8 low-order bits)←AX × r			
	DIVUW	r(*)	2	74	AX (quotient), r (remainder)←AX ÷ r			
Increment/decrement	INC	r	1	2	r←r+1	×	×	
		saddr	2	3/7	(saddr)←(saddr)+1	×	×	
	DEC	r	1	2	r←r−1	×	×	
		saddr	2	3/7	(saddr)←(saddr)−1	×	×	
	INCW	rp	1	3	rp←rp+1			
	DECW	rp	1	3	rp←rp−1			
Shift/rotate	ROR	r, n	2	3+2n	(CY, r_7←r_0, r_{m-1}←r_m) × n n=0–7			×
	ROL	r, n	2	3+2n	(CY, r_0←r_7, r_{m+1}←r_m) × n n=0–7			×
	RORC	r, n	2	3+2n	(CY←r_0, r_7←CY, r_{m-1}←r_m) × n n=0–7			×
	ROLC	r, n	2	3+2n	(CY←r_7, r_0←CY, r_{m+1}←r_m) × n n=0–7			×
	SHR	r, n	2	3+2n	(CY←r_0, r_7←0, r_{m-1}←r_m) × n n=0–7	×	0	×
	SHL	r, n	2	3+2n	(CY←r_7, r_0←0, r_{m+1}←r_m) × n n=0–7	×	0	×
	SHRW	rp, n	2	3+3n	(CY←rp_0, rp_{15}←0, rp_{m+1}←rp_m) × n n=0–7	×	0	×
	SHLW	rp, n	2	3+3n	(CY←rp_{15}, rp_0←0, rp_{m+1}←rp_m) × n n=0–7	×	0	×
	ROR4	[r4]	2	22	A_{3-0}←$(FE00+r4)_{3-0}$, $(FE00+r4)_{7-4}$←A_{3-0}, $(FE00+r4)_{3-0}$←$(FE00+r4)_{7-4}$			
	ROL4	[r4]	2	23	A_{3-0}←$(FE00+r4)_{7-4}$, $(FE00+r4)_{3-0}$←A_{3-0}, $(FE00+r4)_{7-4}$←$(FE00+r4)_{3-0}$			
BCD correction	ADJBA		1	3	Decimal Adjust Accumulator after Addition	×	×	×
	ADJBS		1	3	Decimal Adjust Accumulator after Subtract	×	×	×

* Excluding registers A and X.

Remark n in the shift/rotate instructions indicates the number of bits to be shifted.

Instruction set	Mnemonic	Operand	Byte	Clock	Operation	Flag Z	Flag AC	Flag CY
Bit manipulation	MOV1	CY, saddr.bit	3	5/7	CY←(saddr.bit)			×
		CY, sfr.bit	3	7	CY←sfr.bit			×
		CY, A.bit	2	5	CY←A.bit			×
		CY, X.bit	2	5	CY←X.bit			×
		CY, PSW.bit	2	5	CY←PSW.bit			×
		saddr.bit, CY	3	8/12	(saddr.bit)←CY			
		sfr.bit, CY	3	12	sfr.bit←CY			
		A.bit, CY	2	8	A.bit←CY			
		X.bit, CY	2	8	X.bit←CY			
		PSW.bit, CY	2	7	PSW.bit←CY	×	×	
	AND1	CY, saddr.bit	3	5/7	CY←CY∧(saddr.bit)			×
		CY, /saddr.bit	3	5/7	CY←CY∧$\overline{(saddr.bit)}$			×
		CY, sfr.bit	3	7	CY←CY∧sfr.bit			×
		CY, /sfr.bit	3	7	CY←CY∧$\overline{sfr.bit}$			×
		CY, A.bit	2	5	CY←CY∧A.bit			×
		CY, /A.bit	2	5	CY←CY∧$\overline{A.bit}$			×
		CY, X.bit	2	5	CY←CY∧X.bit			×
		CY, /X.bit	2	5	CY←CY∧$\overline{X.bit}$			×
		CY, PSW.bit	2	5	CY←CY∧PSW.bit			×
		CY, /PSW.bit	2	5	CY←CY∧$\overline{PSW.bit}$			×
	OR1	CY, saddr.bit	3	5/7	CY←CY∨(saddr.bit)			×
		CY, /saddr.bit	3	5/7	CY←CY∨$\overline{(saddr.bit)}$			×
		CY, sfr.bit	3	7	CY←CY∨sfr.bit			×
		CY, /sfr.bit	3	7	CY←CY∨$\overline{sfr.bit}$			×
		CY, A.bit	2	5	CY←CY∨A.bit			×
		CY, /A.bit	2	5	CY←CY∨$\overline{A.bit}$			×
		CY, X.bit	2	5	CY←CY∨X.bit			×
		CY, /X.bit	2	5	CY←CY∨$\overline{X.bit}$			×
		CY, PSW.bit	2	5	CY←CY∨PSW.bit			×
		CY, /PSW.bit	2	5	CY←CY∨$\overline{PSW.bit}$			×
	XOR1	CY, saddr.bit	3	5/7	CY←CY⊻(saddr.bit)			×
		CY, sfr.bit	3	7	CY←CY⊻sfr.bit			×
		CY, A.bit	2	5	CY←CY⊻A.bit			×
		CY, X.bit	2	5	CY←CY⊻X.bit			×
		CY, PSW.bit	2	5	CY←CY⊻PSW.bit			×
	SET1	saddr.bit	2	3/7	(saddr.bit)←1			
		sfr.bit	3	10	sfr.bit←1			
		A.bit	2	6	A.bit←1			
		X.bit	2	6	X.bit←1			
		PSW.bit	2	5	PSW.bit←1	×	×	×

Instruction set	Mnemonic	Operand	Byte	Clock	Operation	Flag Z	Flag AC	Flag CY
Bit manipulation	CLR1	saddr.bit	2	3/7	(saddr.bit)←0			
		sfr.bit	3	10	sfr.bit←0			
		A.bit	2	6	A.bit←0			
		X.bit	2	6	X.bit←0			
		PSW.bit	2	5	PSW.bit←0	×	×	×
	NOT1	saddr.bit	3	6/10	(saddr.bit)←$\overline{\text{(saddr.bit)}}$			
		sfr.bit	3	10	sfr.bit←$\overline{\text{sfr.bit}}$			
		A.bit	2	6	A.bit←$\overline{\text{A.bit}}$			
		X.bit	2	6	X.bit←$\overline{\text{X.bit}}$			
		PSW.bit	2	5	PSW.bit←$\overline{\text{PSW.bit}}$	×	×	×
	SET1	CY	1	2	CY←1			1
	CLR1	CY	1	2	CY←0			0
	NOT1	CY	1	2	CY←$\overline{\text{CY}}$			×
Call/return	CALL	!addr16	3	11/15	(SP−1) (SP−2)←PC+3, PC←addr16 SP←SP−2			
		rp	2	12/16	(SP−1) (SP−2)←PC+2, PC←rp SP←SP−2			
	CALLF	!addr11	2	11/15	(SP−1) (SP−2)←PC+2, PC_{12-11}←01, PC_{10-0}←addr11, SP←SP−2			
	CALLT	[addr5]	1	14/18	(SP−1) (SP−2)←PC+1, PC_H←(addr5+1), PC_L←(addr5), SP←SP−2			
	RET		1	10/14	PC_L←(SP), PC_H←(SP+1), SP←SP+2			
	RETI		1	15/21	PC_L←(SP), PC_H←(SP+1), PSW←(SP+2), SP←SP+3	R	R	R
Stack manipulation	PUSH	rp	1	8/12	(SP−1)←rp_H, (SP−2)←rp_L, SP←SP−2			
		PSW	1	5/7	(SP−1)←PSW, SP←SP−1			
	POP	rp	1	11/15	rp_L←(SP), rp_H←(SP+1), SP←SP+2			
		PSW	1	6/8	PSW←(SP), SP←SP+1	R	R	R
	MOVW	SP, #word	4	8	SP←word			
		SP, AX	2	9	SP←AX			
		AX, SP	2	10	AX←SP			
Unconditional branch	BR	!addr16	3	5	PC←addr16			
		rp	2	6	PC_H←rp_H, PC_L←rp_L			
		$addr16	2	4	PC←PC+2+jdisp8			

Remark When high-order 8 bits (SP8-SP15) are changed in the call return or the stuck manipulation instructions, the number of clocks is incremented by one or two.

Instruction set	Mnemonic	Operand	Byte	Clock	Operation	Flag Z	Flag AC	Flag CY
Conditional branch	BC BL	$addr16	2	4(2)	PC←PC+2+jdisp8 if CY=1			
	BNC BNL	$addr16	2	4(2)	PC←PC+2+jdisp8 if CY=0			
	BZ BE	$addr16	2	4(2)	PC←PC+2+jdisp8 if Z=1			
	BNZ BNE	$addr16	2	4(2)	PC←PC+2+jdisp8 if Z=0			
	BT	saddr.bit, $addr16	3	7(5)	PC←PC+3+jdisp8 if (saddr.bit)=1			
		sfr.bit, $addr16	4	9(7)	PC←PC+4+jdisp8 if sfr.bit=1			
		A.bit, $addr16	3	7(5)	PC←PC+3+jdisp8 if A.bit=1			
		X.bit, $addr16	3	7(5)	PC←PC+3+jdisp8 if X.bit=1			
		PSW.bit, $addr16	3	7(5)	PC←PC+3+jdisp8 if PSW.bit=1			
	BF	saddr.bit, $addr16	4	7(5)	PC←PC+4+jdisp8 if (saddr.bit)=0			
		sfr.bit, $addr16	4	9(7)	PC←PC+4+jdisp8 if sfr.bit=0			
		A.bit, $addr16	3	7(5)	PC←PC+3+jdisp8 if A.bit=0			
		X.bit, $addr16	3	7(5)	PC←PC+3+jdisp8 if X.bit=0			
		PSW.bit, $addr16	3	7(5)	PC←PC+3+jdisp8 if PSW.bit=0			
	BTCLR	saddr.bit, $addr16	4	9(5)	PC←PC+4+jdisp8 if (saddr.bit)=1 then reset (saddr.bit)			
		sfr.bit, $addr16	4	13(7)	PC←PC+4+jdisp8 if sfr.bit=1 then reset sfr.bit			
		A.bit, $addr16	3	9(5)	PC←PC+3+jdisp8 if A.bit=1 then reset A.bit			
		X.bit, $addr16	3	9(5)	PC←PC+3+jdisp8 if X.bit=1 then reset X.bit			
		PSW.bit, $addr16	3	8(5)	PC←PC+3+jdisp8 if PSW.bit=1 then reset PSW.bit	×	×	×
	DBNZ	r2, $addr16	2	5(3)	r2←r2−1, then PC←PC+2+jdisp8 if r2≠0			
		saddr, $addr16	3	6(4)	saddr←saddr−1, then PC←PC+3+jdisp8 if saddr≠0			
CPU control	MOV	STBC, #byte	4	9	STBC←byte			
	SEL	RBn	2	2	RBS1-0←n n=0–3			
	NOP		1	2	No Operation			
	EI		1	2	IE←1 (Enable Interrupt)			
	DI		1	2	IE←0 (Disable Interrupt)			

Remark Values in parentheses in the clock field indicate the number of clocks when there is no branch.

7.2 INSTRUCTION CODES OF EACH INSTRUCTION

(1) Symbols of instruction codes

r, r'

R2	R1	R0	reg	
R6	R5	R4		
0	0	0	R0	X
0	0	1	R1	A
0	1	0	R2	C
0	1	1	R3	B
1	0	0	R4	E
1	0	1	R5	D
1	1	0	R6	L
1	1	1	R7	H

r1

R5	reg
0	A
1	B

r2

R0	reg
0	C
1	B

r3

R1	R0	reg
0	0	E
0	1	E+
1	0	D

r4

R1	reg
R2	
R4	
0	E
1	D

rp, rp'

P1	P0	reg-pair	
P2	P1		
P6	P5		
0	0	RP0	AX
0	1	RP1	BC
1	0	RP2	DE
1	1	RP3	HL

Bn	:	Immediate data for the bit operand
Nn	:	Immediate data for the n operand
Data	:	8-bit immediate data for the byte operand
Low/High Byte	:	16-bit immediate data for the word operand
Saddr-offset	:	Offset data for eight low-order bits of 16-bit address for the saddr operand
Sfr-offset	:	Offset data for eight low-order bits of 16-bit address of special function register (sfr)
Low/High Offset	:	16-bit offset data for the word operand in indirect addressing
Low/High Addr.	:	16-bit immediate data for the addr16 operand
jdisp	:	Signed 2's complement data (8 bits) indicating the relative address displacement from the starting address of the instruction next to the branch address
fa	:	11 low-order bits of immediate data for the addr11 operand
ta	:	Five low-order bits of immediate data for (add5 × 1/2)

Caution **If registers or register pairs are specified as both the first and second operands in the operand field, the instruction code is as follows.**
In a register specification byte, four high-order bits are used for the first operand specification code, and our low-order bits are used for the second operand specification code.

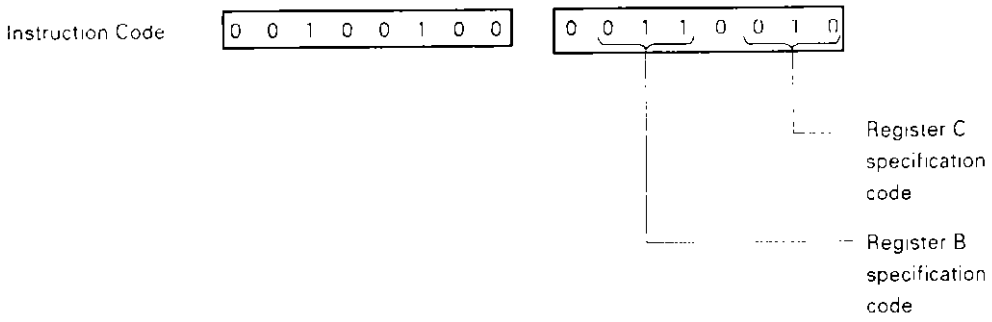
Example MOV r, r'

Instruction Code | 0 0 1 0 0 1 0 0 | 0 R6 R5 R4 0 R2 R1 R0

If register B is specified as the first operand and register C as the second operand, the following instruction must be written:

MOV B, C

Then, the instruction code is as follows:



Instruction set	Mnemonic	Operand	Instruction code B1	B2	B3	B4
8-bit data transfer	MOV	r, #byte	1 0 1 1 1 R2 R1 R0	← Data →		
		saddr, #byte	0 0 1 1 1 0 1 0	← Saddr-offset →	Data	
		sfr, #byte	0 0 1 0 1 0 1 1	← Sfr-offset →	Data	
		r, r'	0 0 1 0 0 1 0 0	0 R6 R5 R4 0 R2 R1 R0		
		A, r	1 1 0 1 0 R2 R1 R0			
		A, saddr	0 0 1 0 0 0 0 0	← Saddr-offset →		
		saddr, A	0 0 1 0 0 0 1 0	← Saddr-offset →		
		A, sfr	0 0 0 1 0 0 0 0	← Sfr-offset →		
		sfr, A	0 0 0 1 0 0 1 0	← Sfr-offset →		
		A, [r3]	0 1 1 1 1 1 R1 R0			
		[r3], A	0 1 1 1 1 0 R1 R0			
		A, [HL]	0 1 0 1 1 1 0 1			
		[HL], A	0 1 0 1 0 1 0 1			
		A, [HL+]	0 1 0 1 1 0 0 1			
		[HL+], A	0 1 0 1 0 0 0 1			
		A, [DE]	0 1 0 1 1 1 0 0			
		[DE], A	0 1 0 1 0 1 0 0			
		A, [DE+]	0 1 0 1 1 0 0 0			
		[DE+], A	0 1 0 1 0 0 0 0			
		A, !addr16	0 0 0 0 1 0 0 1	1 1 1 1 0 0 0 0	Low Addr.	High Addr.
		!addr16, A	0 0 0 0 1 0 0 1	1 1 1 1 0 0 0 1	Low Addr.	High Addr.
		A, word [r1]	0 0 0 0 1 0 1 0	0 0 R5 1 0 0 0 0	Low Offset	High Offset
		word [r1], A	0 0 0 0 1 0 1 0	1 0 R5 1 0 0 0 0	Low Offset	High Offset
		PSW, #byte	0 0 1 0 1 0 1 1	1 1 1 1 1 1 1 0	Data	
		PSW, A	0 0 0 1 0 0 1 0	1 1 1 1 1 1 1 0		
		A, PSW	0 0 0 1 0 0 0 0	1 1 1 1 1 1 1 0		
	XCH	A, r	1 1 0 1 1 R2 R1 R0			
		A, saddr	0 0 1 0 0 0 0 1	← Saddr-offset →		
		A, sfr	0 0 0 0 0 0 0 1	0 0 1 0 0 0 0 1	Sfr-Offset	
		A, [r4]	0 1 1 1 1 R2 1 1			
		A, [HL]	0 0 0 1 0 1 1 0	0 1 0 1 0 1 0 0		
		A, [DE]	0 0 0 1 0 1 1 0	0 1 0 0 0 1 0 0		
		A, word [r1]	0 0 0 0 1 0 1 0	0 0 R5 1 0 1 0 0	Low Offset	High Offset
16-bit data transfer	MOVW	rp, #word	0 1 1 0 0 P2 P1 0	← Low Byte →	High Byte	
		saddrp, #word	0 0 0 0 1 1 0 0	← Saddr-offset →	Low Byte	High Byte
		sfrp, #word	0 0 0 0 1 0 1 1	← Sfr-offset →	Low Byte	High Byte
		rp, rp'	0 0 1 0 0 1 0 0	0 P6 P5 0 1 P2 P1 0		
		AX, saddrp	0 0 0 1 1 1 0 0	← Saddr-offset →		
		saddrp, AX	0 0 0 1 1 0 1 0	← Saddr-offset →		
		AX, sfrp	0 0 0 1 0 0 0 1	← Sfr-Offset →		
		sfrp, AX	0 0 0 1 0 0 1 1	← Sfr-offset →		

Instruction set	Mnemonic	Operand	Instruction code B1	B2	B3	B4
8-bit arithmetic/logical	ADD	A, #byte	1 0 1 0 1 0 0 0	← Data →		
		saddr, #byte	0 1 1 0 1 0 0 0	← Saddr-offset →	Data	
		sfr, #byte	0 0 0 0 0 0 0 1	0 1 1 0 1 0 0 0 ← Sfr-Offset →	Data	
		r, r'	1 0 0 0 1 0 0 0	0 R6 R5 R4 0 R2 R1 R0		
		A, saddr	1 0 0 1 1 0 0 0	← Saddr-offset →		
		A, sfr	0 0 0 0 0 0 0 1	1 0 0 1 1 0 0 0	Sfr-Offset	
		A, [r4]	0 0 0 1 0 1 1 0	0 1 1 R4 1 0 0 0		
		A, [HL]	0 0 0 1 0 1 1 0	0 1 0 1 1 0 0 0		
		A, [DE]	0 0 0 1 0 1 1 0	0 1 0 0 1 0 0 0		
		A, word [r1]	0 0 0 0 1 0 1 0	0 0 R5 1 1 0 0 0	Low Offset	High Offset
	ADDC	A, #byte	1 0 1 0 1 0 0 1	← Data →		
		saddr, #byte	0 1 1 0 1 0 0 1	← Saddr-offset →	Data	
		sfr, #byte	0 0 0 0 0 0 0 1	0 1 1 0 1 0 0 1	Sfr-offset	Data
		r, r'	1 0 0 0 1 0 0 1	0 R6 R5 R4 0 R2 R1 R0		
		A, saddr	1 0 0 1 1 0 0 1	← Saddr-offset →		
		A, sfr	0 0 0 0 0 0 0 1	1 0 0 1 1 0 0 1	Sfr-offset	
		A, [r4]	0 0 0 1 0 1 1 0	0 1 1 R4 1 0 0 1		
		A, [HL]	0 0 0 1 0 1 1 0	0 1 0 1 1 0 0 1		
		A, [DE]	0 0 0 1 0 1 1 0	0 1 0 0 1 0 0 1		
		A, word [r1]	0 0 0 0 1 0 1 0	0 0 R5 1 1 0 0 1	Low Offset	High Offset
	SUB	A, #byte	1 0 1 0 1 0 1 0	← Data →		
		saddr, #byte	0 1 1 0 1 0 1 0	← Saddr-offset →	Data	
		sfr, #byte	0 0 0 0 0 0 0 1	0 1 1 0 1 0 1 0	Sfr-offset	Data
		r, r'	1 0 0 0 1 0 1 0	0 R6 R5 R4 0 R2 R1 R0		
		A, saddr	1 0 0 1 1 0 1 0	← Saddr-offset →		
		A, sfr	0 0 0 0 0 0 0 1	1 0 0 1 1 0 1 0	Sfr-offset	
		A, [r4]	0 0 0 1 0 1 1 0	0 1 1 R4 1 0 1 0		
		A, [HL]	0 0 0 1 0 1 1 0	0 1 0 1 1 0 1 0		
		A, [DE]	0 0 0 1 0 1 1 0	0 1 0 0 1 0 1 0		
		A, word [r1]	0 0 0 0 1 0 1 0	0 0 R5 1 1 0 1 0	Low Offset	High Offset
	SUBC	A, #byte	1 0 1 0 1 0 1 1	← Data →		
		saddr, #byte	0 1 1 0 1 0 1 1	← Saddr-offset →	Data	
		sfr, #byte	0 0 0 0 0 0 0 1	0 1 1 0 1 0 1 1	Sfr-offset	Data
		r, r'	1 0 0 0 1 0 1 1	0 R6 R5 R4 0 R2 R1 R0		
		A, saddr	1 0 0 1 1 0 1 1	← Saddr-offset →		
		A, sfr	0 0 0 0 0 0 0 1	1 0 0 1 1 0 1 1	Sfr-offset	
		A, [r4]	0 0 0 1 0 1 1 0	0 1 1 R4 1 0 1 1		
		A, [HL]	0 0 0 1 0 1 1 0	0 1 0 1 1 0 1 1		
		A, [DE]	0 0 0 1 0 1 1 0	0 1 0 0 1 0 1 1		
		A, word [r1]	0 0 0 0 1 0 1 0	0 0 R5 1 1 0 1 1	Low Offset	High Offset

Instruction set	Mnemonic	Operand	Instruction code B1	B2	B3	B4
8-bit arithmetic/logical	AND	A, #byte	1 0 1 0 1 1 0 0	← Data →		
		saddr, #byte	0 1 1 0 1 1 0 0	← Saddr-offset →	Data	
		sfr, #byte	0 0 0 0 0 0 0 1	0 1 1 0 1 1 0 0	Sfr-offset	Data
		r, r'	1 0 0 0 1 1 0 0	0 R6 R5 R4 0 R2 R1 R0		
		A, saddr	1 0 0 1 1 1 0 0	← Saddr-offset →		
		A, sfr	0 0 0 0 0 0 0 1	1 0 0 1 1 1 0 0	Sfr-offset	
		A, [r4]	0 0 0 1 0 1 1 0	0 1 1 R4 1 1 0 0		
		A, [HL]	0 0 0 1 0 1 1 0	0 1 0 1 1 1 0 0		
		A, [DE]	0 0 0 1 0 1 1 0	0 1 0 0 1 1 0 0		
		A, word [r1]	0 0 0 0 1 0 1 0	0 0 R5 1 1 1 0 0	Low Offset	High Offset
	OR	A, #byte	1 0 1 0 1 1 1 0	← Data →		
		saddr, #byte	0 1 1 0 1 1 1 0	← Saddr-offset →	Data	
		sfr, #byte	0 0 0 0 0 0 0 1	0 1 1 0 1 1 1 0	Sfr-offset	Data
		r, r'	1 0 0 0 1 1 1 0	0 R6 R5 R4 0 R2 R1 R0		
		A, saddr	1 0 0 1 1 1 1 0	← Saddr-offset →		
		A, sfr	0 0 0 0 0 0 0 1	1 0 0 1 1 1 1 0	Sfr-offset	
		A, [r4]	0 0 0 1 0 1 1 0	0 1 1 R4 1 1 1 0		
		A, [HL]	0 0 0 1 0 1 1 0	0 1 0 1 1 1 1 0		
		A, [DE]	0 0 0 1 0 1 1 0	0 1 0 0 1 1 1 0		
		A, word [r1]	0 0 0 0 1 0 1 0	0 0 R5 1 1 1 1 0	Low Offset	High Offset
	XOR	A, #byte	1 0 1 0 1 1 0 1	← Data →		
		saddr, #byte	0 1 1 0 1 1 0 1	← Saddr-offset →	Data	
		sfr, #byte	0 0 0 0 0 0 0 1	0 1 1 0 1 1 0 1	Sfr-offset	Data
		r, r'	1 0 0 0 1 1 0 1	0 R6 R5 R4 0 R2 R1 R0		
		A, saddr	1 0 0 1 1 1 0 1	← Saddr-offset →		
		A, sfr	0 0 0 0 0 0 0 1	1 0 0 1 1 1 0 1	Sfr-offset	
		A, [r4]	0 0 0 1 0 1 1 0	0 1 1 R4 1 1 0 1		
		A, [HL]	0 0 0 1 0 1 1 0	0 1 0 1 1 1 0 1		
		A, [DE]	0 0 0 1 0 1 1 0	0 1 0 0 1 1 0 1		
		A, word [r1]	0 0 0 0 1 0 1 0	0 0 R5 1 1 1 0 1	Low Offset	High Offset
	CMP	A, #byte	1 0 1 0 1 1 1 1	← Data →		
		saddr, #byte	0 1 1 0 1 1 1 1	← Saddr-offset →	Data	
		sfr, #byte	0 0 0 0 0 0 0 1	0 1 1 0 1 1 1 1	Sfr-offset	Data
		r, r'	1 0 0 0 1 1 1 1	0 R6 R5 R4 0 R2 R1 R0		
		A, saddr	1 0 0 1 1 1 1 1	← Saddr-offset →		
		A, sfr	0 0 0 0 0 0 0 1	1 0 0 1 1 1 1 1	Sfr-offset	
		A, [r4]	0 0 0 1 0 1 1 0	0 1 1 R4 1 1 1 1		
		A, [HL]	0 0 0 1 0 1 1 0	0 1 0 1 1 1 1 1		
		A, [DE]	0 0 0 1 0 1 1 0	0 1 0 0 1 1 1 1		
		A, word [r1]	0 0 0 0 1 0 1 0	0 0 R5 1 1 1 1 1	Low Offset	High Offset

Instruction set	Mnemonic	Operand	Instruction code B1	B2	B3	B4
16-bit arithmetic/logical	ADDW	AX, #word	0 0 1 0 1 1 0 1	← Low Byte →	High Byte	
		AX, rp	1 0 0 0 1 0 0 0	0 0 0 0 1 P_2 P_1 0		
		AX, saddrp	0 0 0 1 1 1 0 1	← Saddr-offset →		
		AX, sfrp	0 0 0 0 0 0 0 1	0 0 0 1 1 1 0 1	Sfr-offset	
	SUBW	AX, #word	0 0 1 0 1 1 1 0	← Low Byte →	High Byte	
		AX, rp	1 0 0 0 1 0 1 0	0 0 0 0 1 P_2 P_1 0		
		AX, saddrp	0 0 0 1 1 1 1 0	← Saddr-offset →		
		AX, sfrp	0 0 0 0 0 0 0 1	0 0 0 1 1 1 1 0	Sfr-offset	
	CMPW	AX, #word	0 0 1 0 1 1 1 1	← Low Byte →	High Byte	
		AX, rp	1 0 0 0 1 1 1 1	0 0 0 0 1 P_2 P_1 0		
		AX, saddrp	0 0 0 1 1 1 1 1	← Saddr-offset →		
		AX, sfrp	0 0 0 0 0 0 0 1	0 0 0 1 1 1 1 1	Sfr-offset	
Multiply/divide	MULSW	r	0 0 0 0 0 1 0 1	0 0 1 1 0 R_2 R_1 R_0		
	MULUW	r	0 0 0 0 0 1 0 1	0 0 0 0 1 R_2 R_1 R_0		
	DIVUW	r	0 0 0 0 0 1 0 1	0 0 0 1 1 R_2 R_1 R_0		
Increment/decrement	INC	r	1 1 0 0 0 R_2 R_1 R_0			
		saddr	0 0 1 0 0 1 1 0	← Saddr-offset →		
	DEC	r	1 1 0 0 1 R_2 R_1 R_0			
		saddr	0 0 1 0 0 1 1 1	← Saddr-offset →		
	INCW	rp	0 1 0 0 0 1 P_1 P_0			
	DECW	rp	0 1 0 0 1 1 P_1 P_0			
Shift/rotate	ROR	r, n	0 0 1 1 0 0 0 0	0 1 N_2 N_1 N_0 R_2 R_1 R_0		
	ROL	r, n	0 0 0 1	0 1 N_2 N_1 N_0 R_2 R_1 R_0		
	RORC	r, n	0 0 0 0	0 0 N_2 N_1 N_0 R_2 R_1 R_0		
	ROLC	r, n	0 0 0 1	0 0 N_2 N_1 N_0 R_2 R_1 R_0		
	SHR	r, n	0 0 0 0	1 0 N_2 N_1 N_0 R_2 R_1 R_0		
	SHL	r, n	0 0 0 1	1 0 N_2 N_1 N_0 R_2 R_1 R_0		
	SHRW	rp, n	0 0 0 0	1 1 N_2 N_1 N_0 P_2 P_1 0		
	SHLW	rp, n	0 0 0 1	1 1 N_2 N_1 N_0 P_2 P_1 0		
	ROR4	[r4]	0 0 0 0 0 1 0 1	1 0 0 0 1 0 R_1 1		
	ROL4	[r4]	0 0 0 0 0 1 0 1	1 0 0 1 1 0 R_1 1		
BCD correction	ADJBA		0 0 0 0 1 1 1 0			
	ADJBS		0 0 0 0 1 1 1 1			

Instruction set	Mnemonic	Operand	Instruction code B1	B2	B3	B4
Bit manipulation	MOV1	CY, saddr.bit	0 0 0 0 1 0 0 0	0 0 0 0 0 B2 B1 B0	Saddr-offset	
		CY, sfr.bit	1 0 0 0	1 B2 B1 B0	Sfr-offset	
		CY, A.bit	0 0 1 1	1 B2 B1 B0		
		CY, X.bit	0 0 1 1	0 B2 B1 B0		
		CY, PSW.bit	0 0 1 0	0 B2 B1 B0		
		saddr.bit, CY	1 0 0 0	0 0 0 1 0 B2 B1 B0	Saddr-offset	
		sfr.bit, CY	1 0 0 0	1 B2 B1 B0	Sfr-offset	
		A.bit, CY	0 0 1 1	1 B2 B1 B0		
		X.bit, CY	0 0 1 1	0 B2 B1 B0		
		PSW.bit, CY	0 0 1 0	0 B2 B1 B0		
	AND1	CY, saddr.bit	0 0 0 0 1 0 0 0	0 0 1 0 0 B2 B1 B0	Saddr-offset	
		CY, /saddr.bit		0 0 1 1 0 B2 B1 B0	Saddr-offset	
		CY, sfr.bit		0 0 1 0 1 B2 B1 B0	Sfr-offset	
		CY, /sfr.bit		0 0 1 1 1 B2 B1 B0	Sfr-offset	
		CY, A.bit	0 0 1 1	0 0 1 0 1 B2 B1 B0		
		CY, /A.bit		0 0 1 1 1 B2 B1 B0		
		CY, X.bit		0 0 1 0 0 B2 B1 B0		
		CY, /X.bit		0 0 1 1 0 B2 B1 B0		
		CY, PSW.bit	0 0 1 0	0 0 1 0 0 B2 B1 B0		
		CY, /PSW.bit	0 0 1 0	0 0 1 1 0 B2 B1 B0		
	OR1	CY, saddr.bit	0 0 0 0 1 0 0 0	0 1 0 0 0 B2 B1 B0	Saddr-offset	
		CY, /saddr.bit		0 1 0 1 0 B2 B1 B0	Saddr-offset	
		CY, sfr.bit		0 1 0 0 1 B2 B1 B0	Sfr-offset	
		CY, /sfr.bit		0 1 0 1 1 B2 B1 B0	Sfr-offset	
		CY, A.bit	0 0 1 1	0 1 0 0 1 B2 B1 B0		
		CY, /A.bit		0 1 0 1 1 B2 B1 B0		
		CY, X.bit		0 1 0 0 0 B2 B1 B0		
		CY, /X.bit		0 1 0 1 0 B2 B1 B0		
		CY, PSW.bit	0 0 1 0	0 1 0 0 0 B2 B1 B0		
		CY, /PSW.bit	0 0 1 0	0 1 0 1 0 B2 B1 B0		
	XOR1	CY, saddr.bit	0 0 0 0 1 0 0 0	0 1 1 0 0 B2 B1 B0	Saddr-offset	
		CY, sfr.bit	1 0 0 0	1 B2 B1 B0	Sfr-offset	
		CY, A.bit	0 0 1 1	1 B2 B1 B0		
		CY, X.bit	0 0 1 1	0 B2 B1 B0		
		CY, PSW.bit	0 0 1 0	0 B2 B1 B0		

Instruction set	Mnemonic	Operand	Instruction code B1	B2	B3	B4
Bit manipulation	SET1	saddr.bit	1 0 1 1 0 B_2 B_1 B_0	Saddr-offset		
		sfr.bit	0 0 0 0 1 0 0 0	0 0 0 0 1 B_2 B_1 B_0	Sfr-offset	
		A.bit	0 0 1 1	1 B_2 B_1 B_0		
		X.bit	0 0 1 1	0 B_2 B_1 B_0		
		PSW.bit	0 0 1 0	0 B_2 B_1 B_0		
	CLR1	saddr.bit	1 0 1 0 0 B_2 B_1 B_0	← Saddr-offset →		
		sfr.bit	0 0 0 0 1 0 0 0	1 0 0 1 1 B_2 B_1 B_0	Sfr-offset	
		A.bit	0 0 1 1	1 B_2 B_1 B_0		
		X.bit	0 0 1 1	0 B_2 B_1 B_0		
		PSW.bit	0 0 1 0	0 B_2 B_1 B_0		
	NOT1	saddr.bit	0 0 0 0 1 0 0 0	1 1 1 1 0 B_2 B_1 B_0	Saddr-offset	
		sfr.bit	1 0 0 0	1 B_2 B_1 B_0	Sfr-offset	
		A.bit	0 0 1 1	1 B_2 B_1 B_0		
		X.bit	0 0 1 1	0 B_2 B_1 B_0		
		PSW.bit	0 0 1 0	0 B_2 B_1 B_0		
	SET1	CY	0 1 0 0 0 0 0 1			
	CLR1	CY	0 1 0 0 0 0 0 0			
	NOT1	CY	0 1 0 0 0 0 1 0			
Call/return	CALL	!addr16	0 0 1 0 1 0 0 0	← Low Addr. →	High Addr.	
		rp	0 0 0 0 0 1 0 1	0 1 0 1 1 P_2 P_1 0		
	CALLF	!addr11	1 0 0 1 0 ←	fa →		
	CALLT	[addr5]	1 1 1 ← ta →			
	RET		0 1 0 1 0 1 1 0			
	RETI		0 1 0 1 0 1 1 1			
Stack manipulation	PUSH	rp	0 0 1 1 1 1 P_1 P_0			
		PSW	0 1 0 0 1 0 0 1			
	POP	rp	0 0 1 1 0 1 P_1 P_0			
		PSW	0 1 0 0 1 0 0 0			
		SP, #word	0 0 0 0 1 0 1 1	1 1 1 1 1 1 0 0	Low Byte	High Byte
	MOVW	SP, AX	0 0 0 1 0 0 1 1	1 1 1 1 1 1 0 0		
		AX, SP	0 0 0 1 0 0 0 1	1 1 1 1 1 1 0 0		

Instruction set	Mnemonic	Operand	Instruction code B1	B2	B3	B4
Unconditional branch	BR	!addr16	0 0 1 0 1 1 0 0	← Low Addr. →	High Addr.	
		rp	0 0 0 0 0 1 0 1	0 1 0 0 1 P_2 P_1 0		
		$addr16	0 0 0 1 0 1 0 0	← jdisp →		
Conditional branch	BC BL	$addr16	1 0 0 0 0 0 1 1	← jdisp →		
	BNC BNL	$addr16	0 0 1 0	← jdisp →		
	BZ BE	$addr16	0 0 0 1	← jdisp →		
	BNZ BNE	$addr16	0 0 0 0	← jdisp →		
	BT	saddr.bit, $addr16	0 1 1 1 0 B_2 B_1 B_0	← Saddr-offset →	jdisp	
		sfr.bit, $addr16	0 0 0 0 1 0 0 0	1 0 1 1 1 B_2 B_1 B_0	Sfr-offset	jdisp
		A.bit, $addr16	0 0 1 1	1 B_2 B_1 B_0	jdisp	
		X.bit, $addr16	0 0 1 1	0 B_2 B_1 B_0	jdisp	
		PSW.bit, $addr16	0 0 1 0	0 B_2 B_1 B_0	jdisp	
	BF	saddr.bit, $addr16	0 0 0 0 1 0 0 0	1 0 1 0 0 B_2 B_1 B_0	Saddr-offset	jdisp
		sfr.bit, $addr16	1 0 0 0	1 B_2 B_1 B_0	Sfr-offset	jdisp
		A.bit, $addr16	0 0 1 1	1 B_2 B_1 B_0	jdisp	
		X.bit, $addr16	0 0 1 1	0 B_2 B_1 B_0	jdisp	
		PSW.bit, $addr16	0 0 1 0	0 B_2 B_1 B_0	jdisp	
	BTCLR	saddr.bit, $addr16	0 0 0 0 1 0 0 0	1 1 0 1 0 B_2 B_1 B_0	Saddr-offset	jdisp
		sfr.bit, $addr16	1 0 0 0	1 B_2 B_1 B_0	Sfr-offset	jdisp
		A.bit, $addr16	0 0 1 1	1 B_2 B_1 B_0	jdisp	
		X.bit, $addr16	0 0 1 1	0 B_2 B_1 B_0	jdisp	
		PSW.bit, $addr16	0 0 1 0	0 B_2 B_1 B_0	jdisp	
	DBNZ	r2, $addr16	0 0 1 1 0 0 1 R_0	← jdisp →		
		saddr, $addr16	0 0 1 1 1 0 1 1	← Saddr-offset →	jdisp	
CPU control	MOV	STBC, #byte	0 0 0 0 1 0 0 1	1 1 0 0 1 0 0 0	Data	Data
	SEL	RBn	0 0 0 0 0 1 0 1	1 0 1 0 1 0 N_1 N_0		
	NOP		0 0 0 0 0 0 0 0			
	EI		0 1 0 0 1 0 1 1			
	DI		0 1 0 0 1 0 1 0			

8. ELECTRICAL SPECIFICATIONS

ABSOLUTE MAXIMUM RATINGS (T_a = 25 °C)

Parameter	Symbol	Conditions	Rating	Unit
Supply voltage	V_{DD}		−0.5 to +7.0	V
	AV_{REF}		−0.5 to V_{DD}	V
	AV_{SS}		−0.5 to +0.5	V
Input voltage	V_I		−0.5 to V_{DD} + 0.5	V
Output voltage	V_O		−0.5 to V_{DD} + 0.5	V
Low output current	I_{OL}	1 pin	15	mA
		Total of all output pins	100	mA
High output current	I_{OH}	1 pin	−10	mA
		Total of all output pins	−50	mA
Operating temperature	T_{OPT}		−10 to +70	°C
Storage temperature	T_{STG}		−65 to +150	°C

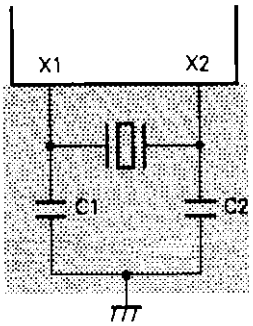
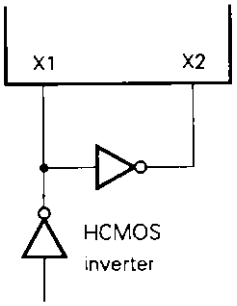
OPERATING CONDITIONS

Clock frequency	Operating temperature (T_a)	Supply voltage (V_{DD})
4 MHz ≤ fxx ≤ 12 MHz	−10 °C to +70 °C	+5.0 V ±10 %

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

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

OSCILLATOR CHARACTERISTICS (T_a = -10 °C to +70 °C, V_{DD} = +5.0 V ±10 %, V_{SS} = 0 V)

Resonator	Recommended circuit	Parameter	Min.	Max.	Unit
Ceramic or crystal resonator (*1)		Oscillation frequency (fxx)	4 or 6 (*2)	12	MHz
External clock		X1 input frequency (fx)	4 or 6 (*2)	12	MHz
		X1 input rising (falling) time (t_{XR}, t_{XF})	0	30	ns
		X1 input high-level (low-level) width (t_{WXH}, t_{WXL})	30	130	ns ★

Cautions 1. The oscillation circuit must be as near pins X1 and X2 as possible.

2. No signals are allowed in the shaded portion.

*1 When a crystal resonator is used, the external circuit with a capacity of C1 = C2 = 15 pF is recommended.

*2 When an A/D converter is used : 6 MHz
When an A/D converter is not used : 4 MHz

RECOMMENDED CONSTANTS FOR THE OSCILLATION CIRCUIT ★

Resonator	Manufacturer	Part number	Frequency [MHz]	Recommended constants	
				C1 [pF]	C2 [pF]
Ceramic resonator	Murata Mfg. Co., Ltd.	CAT12.0MTZ	12	30	30
		CST12.0MTW		Built-in capacitor type	

DC CHARACTERISTICS (T_a = -10 °C to +70 °C, V_{DD} = +5.0 V ±10 %, Vss = 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}	Pins other than those shown in (*1)	2.2		V_{DD}	V
	V_{IH2}	Pins shown in (*1)	0.8V_{DD}		V_{DD}	V
Low-level output voltage	V_{OL1}	I_{OL} = 2.0 mA, except for (*2)			0.45	V
	V_{OL2}	I_{OL} = 8.0 mA (*2)			1.0	V
High-level output voltage	V_{OH1}	I_{OH} = -1.0 mA, except for (*3)	V_{DD} - 1.0			V
	V_{OH2}	I_{OH} = -100 μA, except for (*3)	V_{DD} - 0.5			V
	V_{OH3}	I_{OH} = -5.0 mA (*3)	2.0			V
Input leakage current	I_{LI}	0 V ≤ V_I ≤ V_{DD}			±10	μA
Output leakage current	I_{LO}	0 V ≤ V_O ≤ V_{DD}			±10	μA
AV_{REF} current	AI_{REF}	Operation mode, fxx = 12 MHz		1.5	5.0	mA
V_{DD} supply current	I_{DD1}	Operation mode, fxx = 12 MHz		20	40	mA
Data retention voltage	V_{DDDR}	STOP mode	2.0		5.5	V
Data retention current	I_{DDDR}	STOP mode: V_{DDDR} = 2.0 V		1	10	μA
		STOP mode: V_{DDDR} = 5 V ±10 %		2	20	μA
Pull-up resistor	R_L	V_I = 0 V	15	40	80	kΩ

*1 Pins X1, X2, $\overline{\text{RESET}}$, P20/NMI, P21/INTP0, P22/INTP1, P23/INTP2, P24/CTI10, P25/CTI00, P26/CTI11, P27/CLR1, P34/CLR0, P35/SI, P36/SO/SB0, P37/$\overline{\text{SCK}}$, and $\overline{\text{EA}}$

*2 Pins P10-P17

*3 Pins P00-P07

AC CHARACTERISTICS (T_a = -10 °C to +70 °C, V_{DD} = +5 V ±10 %, V_{SS} = 0 V)

Read/Write Operation

Parameter	Symbol	Conditions	Min.	Max.	Unit
X1 input clock cycle time	t_{CYX}		82	250	ns
Address setup time (to ASTB↓)	t_{SAST}		52		ns
Address hold time (to ASTB↓)	t_{HSTA}	R_L = 5 kΩ, C_L = 50 pF	25		ns
Address →$\overline{RD}$↓ delay time	t_{DAR}		129		ns
Address float time (to $\overline{RD}$↓)	t_{FAR}		11		ns
Address → data input time	t_{DAID}			228	ns
ASTB↓ → data input time	t_{DSTID}			181	ns
$\overline{RD}$↓ → data input time	t_{DRID}			99	ns
ASTB↓ → $\overline{RD}$↓ delay time	t_{DSTR}		52		ns
Data hole time (to $\overline{RD}$↑)	t_{HRID}		0		ns
$\overline{RD}$↑ → address active time	t_{DRA}		124		ns
$\overline{RD}$↑→ ASTB↑ delay time	t_{DRST}		124		ns
$\overline{RD}$ low-level width	t_{WRL}		124		ns
ASTB high-level width	t_{WSTH}		52		ns
Address → $\overline{WR}$↓ delay time	t_{DAW}		129		ns
ASTB↓ → data output time	t_{DSTOD}			142	ns
$\overline{WR}$↓ → data output time	t_{DWOD}			60	ns
ASTB↓ → $\overline{WR}$↓ delay time	t_{DSTW}		52		ns
Data setup time (to $\overline{WR}$↑)	t_{SODWR}		146		ns
Data hold time (to $\overline{WR}$↑)[*]	t_{HWOD}		22		ns
$\overline{WR}$↑ → ASTB↑ delay time	t_{DWST}		42		ns
$\overline{WR}$ low-level width	t_{WWL}		196		ns

Remark The values listed in the above table are obtained when f_{XX} = 12 MHz and C_L = 100 pF.

* The hold time includes the time for holding V_{OH} and V_{OL} on the load conditions of C_L = 100 pF and R_L = 2 kΩ.

t_{CYX}-dependent Bus Timing Definition

Parameter	Symbol	Calculation formula	Max./Min.	12 MHz	Unit
X1 input clock cycle time	t_{CYX}		Min.	82	ns
Address setup time (to ASTB↓)	t_{SAST}	t_{CYX} – 30	Min.	52	ns
Address →$\overline{RD}$ ↓ delay time	t_{DAR}	2t_{CYX} – 35	Min.	129	ns
Address float time (to $\overline{RD}$ ↓)	t_{FAR}	t_{CYX}/2 – 30	Min.	11	ns
Address → data input time	t_{DAID}	(4 + 2n)t_{CYX} – 100	Max.	228	ns
ASTB↓ → data input time	t_{DSTID}	(3 + 2n)t_{CYX} – 65	Max.	181	ns
$\overline{RD}$ ↓ → data input time	t_{DRID}	(2 + 2n)t_{CYX} – 65	Max.	99	ns
ASTB↓ → $\overline{RD}$ ↓ delay time	t_{DSTR}	t_{CYX} – 30	Min.	52	ns
$\overline{RD}$ ↑ → address active time	t_{DRA}	2t_{CYX} – 40	Min.	124	ns
$\overline{RD}$ ↑→ ASTB↑ delay time	t_{DRST}	2t_{CYX} – 40	Min.	124	ns
$\overline{RD}$ low-level width	t_{WRL}	(2 + 2n)t_{CYX} – 40	Min.	124	ns
ASTB high-level width	t_{WSTH}	t_{CYX} – 30	Min.	52	ns
Address → $\overline{WR}$ ↓ delay time	t_{DAW}	2t_{CYX} – 35	Min.	129	ns
ASTB↓ → data output time	t_{DSTOD}	t_{CYX} + 60	Max.	142	ns
ASTB↓ → $\overline{WR}$ ↓ delay time	t_{DSTW}	t_{CYX} – 30	Min.	52	ns
Data setup time (to $\overline{WR}$ ↑)	t_{SODWR}	(3 + 2n)t_{CYX} – 100	Min.	146	ns
Data setup time (to $\overline{WR}$ ↓)	t_{SODWF}	t_{CYX} – 60	Min.	22	ns
$\overline{WR}$ ↑ → ASTB↑ delay time	t_{DWST}	t_{CYX} – 40	Min.	42	ns
$\overline{WR}$ low-level width	t_{WWL}	(3 + 2n)t_{CYX} – 50	Min.	196	ns

Remarks 1. n represents the number of wait cycles inserted according to the specification of the memory mapping register (MM).

2. The standard values for 12 MHz are obtained when n is 0.

3. The items that are not listed in the above table do not depend on the clock frequency (fxx).

Serial Operation

Parameter	Symbol	Conditions		Min.	Max.	Unit
Serial clock cycle time	tCYSK	Input	External clock	1.0		μs
		Output	fCLK divided by 8	1.3		μs
			fCLK divided by 32	5.3		μs
Serial clock low-level width	tWSKL	Input	External clock	420		ns
		Output	fCLK divided by 8	556		ns
			fCLK divided by 32	2.5		μs
Serial clock high-level width	tWSKH	Input	External clock	420		ns
		Output	fCLK divided by 8	556		ns
			fCLK divided by 32	2.5		μs
SI, SB0 setup time (to $\overline{SCK}$ ↑)	tSSSK			150		ns
SI, SB0 hold time (to $\overline{SCK}$ ↑)	tHSSK			400		ns
SO/SB0 output delay time (to $\overline{SCK}$ ↓)	tDSBSK1	CMOS push-pull output (three-wire serial I/O mode)		0	300	ns
	tDSBSK2	Open-drain output (SBI mode), RL = 1kΩ		0	800	ns
SB0 high hold time (to $\overline{SCK}$ ↑)	tHSBSK	SBI mode		4		tCYX
SB0 low setup time (to $\overline{SCK}$ ↓)	tSSBSK			4		tCYX
SB0 low-level width	tWSBL			4		tCYX
SB0 high-level width	tWSBH			4		tCYX

Remarks 1. The values listed in the above table are obtained when fxx = 12 MHz and CL = 100 pF.

2. fCLK indicates the internal system clock (fx or fxx divided by 2).

Other Operations

Parameter		Symbol	Conditions		Min.	Max.	Unit
CTI00, CTI10, CTI11 low-level width		tWCTL			4		tCYX
CTI00, CTI10, CTI11 high-level width		tWCTH			4		tCYX
★ CLR1 low-level width		tWCR1L	Digital noise eliminator not used		4		tCYX
			Digital noise eliminator used	ICR.4[*] = 0	40		tCYX
★				ICR.4[*] = 1	80		tCYX
★ CLR1 high-level width		tWCR1H	Digital noise eliminator not used		4		tCYX
			Digital noise eliminator used	ICR.4[*] = 0	40		tCYX
★				ICR.4[*] = 1	80		tCYX
Digital noise eliminator	Removed pulse width	tWSEP	ICR.4[*] = 0			32	tCYX
	Passed pulse width				40		tCYX
	Removed pulse width		ICR.4[*] = 1			72	tCYX
	Passed pulse width				80		tCYX
NMI low-level width		tWNIL			10		μs
NMI high-level width		tWNIH			10		μs
INTP0-INTP2 low-level width		tWIPL			4		tCYX
INTP0-INTP2 high-level width		tWIPH			4		tCYX
$\overline{\text{RESET}}$ low-level width		tWRSL			10		μs
$\overline{\text{RESET}}$ high-level width		tWRSH			10		μs

* Bit 4 of the input control register (ICR)

Clock Output Operation

Parameter	Symbol	Calculation formula	Min.	Max.	Unit
CLO cycle time	tCYCL		333	2667	ns
CLO low-level width	tCLL	tCYCL/2 ±50	116	1384	ns
CLO high-level width	tCLH	tCYCL/2 ±50	116	1384	ns
CLO rising time	tCLR			50	ns
CLO falling time	tCLF			50	ns

Remark The values in the table are obtained when fxx = 12 MHz and C_L = 100 pF.

External Clock Timing

Parameter	Symbol	Conditions	Min.	Max.	Unit
X1 input low-level width	tWXL		30	130	ns
X1 input high-level width	tWXH		30	130	ns
X1 input rising time	tXR		0	30	ns
X1 input falling time	tXF		0	30	ns
X1 input clock cycle time	tCYX		82	250	ns

A/D CONVERTER CHARACTERISTICS

(T_a = −10 °C to +70 °C, V_{DD} = +5.0 V ±10 %, 3.8 V ≤ AV_{REF} ≤ V_{DD}, AVss = Vss = 0 V)

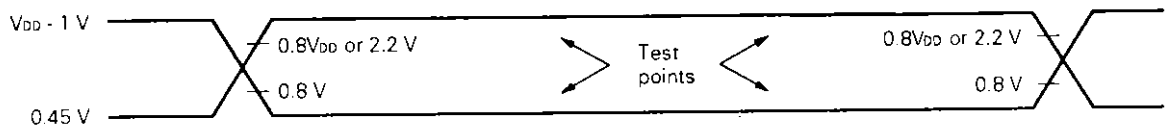
Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Resolution			8			bit
Total error		4.0 V ≤ AV_{REF} ≤ V_{DD}			0.4	%
		3.8 V ≤ AV_{REF} ≤ V_{DD}			0.8	%
Quantization error					±1/2	LSB
Conversion time	t_{CONV}	83 ns ≤ t_{CYX} ≤ 125 ns	360			t_{CYX}
		125 ns ≤ t_{CYX} ≤ 250 ns	240			t_{CYX}
Sampling time	t_{SAMP}	83 ns ≤ t_{CYX} ≤ 125 ns	72			t_{CYX}
		125 ns ≤ t_{CYX} ≤ 250 ns	48			t_{CYX}
Analog input voltage	V_{IAN}		-0.3		AV_{REF} + 0.3	V
Analog input impedance	R_{AN}			1000		MΩ
Reference voltage	AV_{REF}		3.8		V_{DD}	V
AV_{REF} current	AI_{REF}	Normal operation mode, fxx = 12 MHz		1.5	5.0	mA
		STOP mode		0.7	1.5	mA

DATA MEMORY LOW-VOLTAGE DATA RETENTION CHARACTERISTICS (T_a = -10 °C to +70 °C)

Parameter	Symbol	Conditions	Min.	Typ.	Max.	Unit
Data retention voltage	V_{DDDR}	STOP mode	2.0		5.5	V
Data retention current	I_{DDDR}	V_{DDDR} = 2.0 V		1	10	μA
		V_{DDDR} = 5 V ± 10 %		2	20	μA
V_{DD} rising time	t_{RVD}		200			μs
V_{DD} falling time	t_{FVD}		200			μs
STOP release signal input time	t_{DREL}		0			ms
Low-level input voltage	V_{IL}	Specified pins[*1]	0		0.1V_{DD}	V
High-level input voltage	V_{IH}		0.9V_{DD}		V_{DD}	V

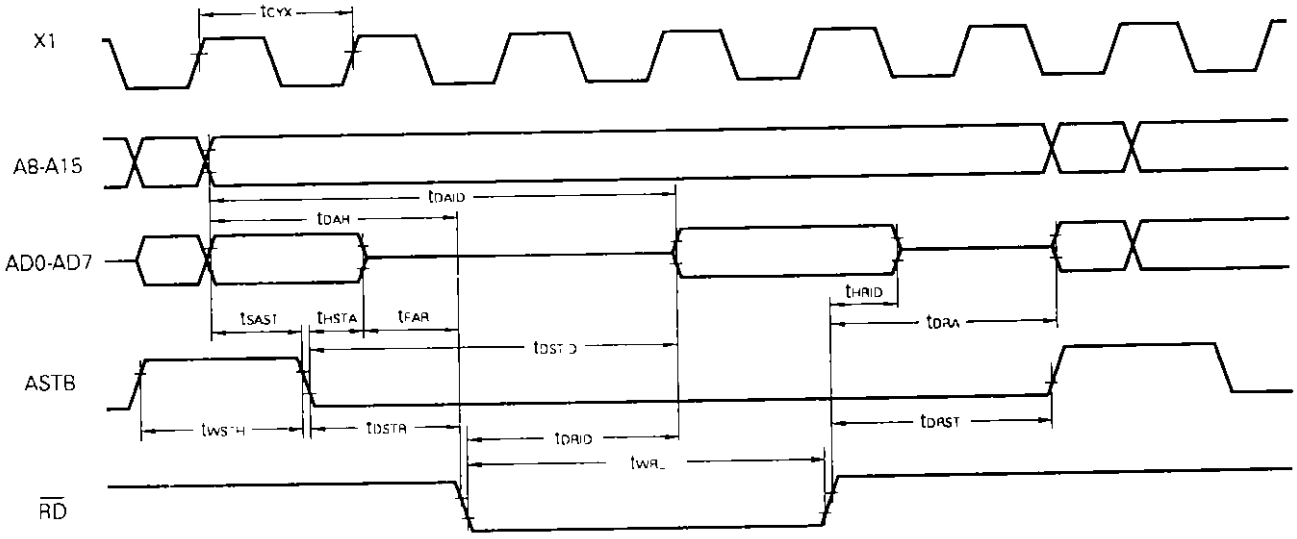
* Pins $\overline{\text{RESET}}$, P20/NMI, P21/INTP0, P22/INTP1, P23/INTP2, P24/CTI10, P25/CTI00, P26/CTI11, P27/CLR1, P34/CLR0, P35/SI, P36/SO/SB0, P37/$\overline{\text{SCK}}$, and $\overline{\text{EA}}$

AC Timing Test Points

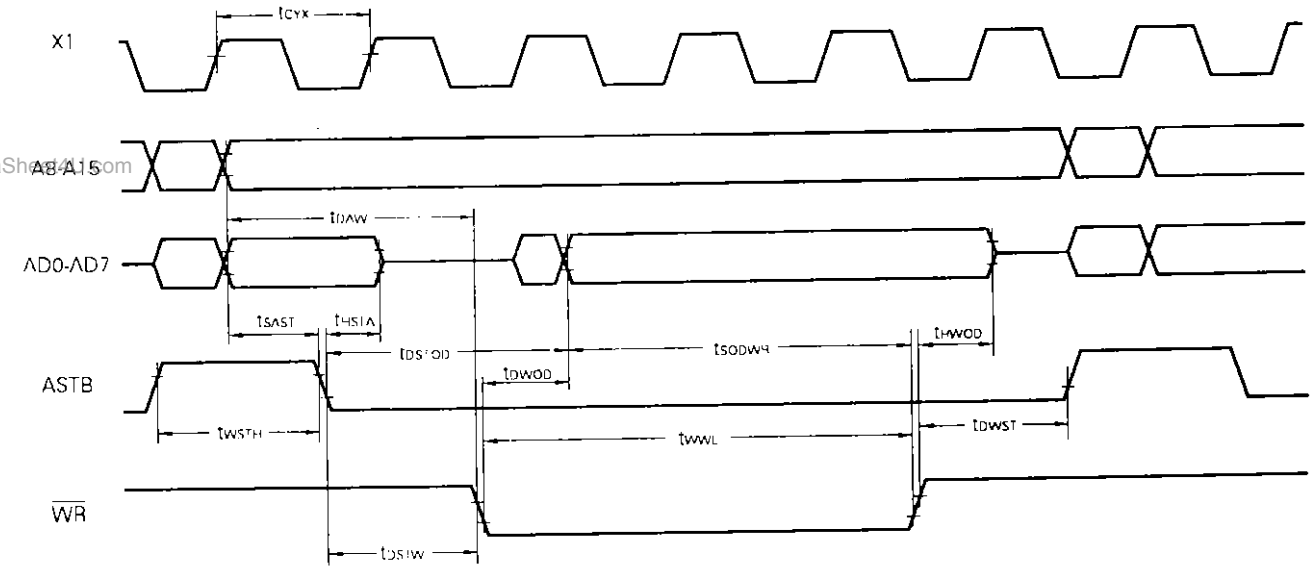


Timing Waveform

Read operation:

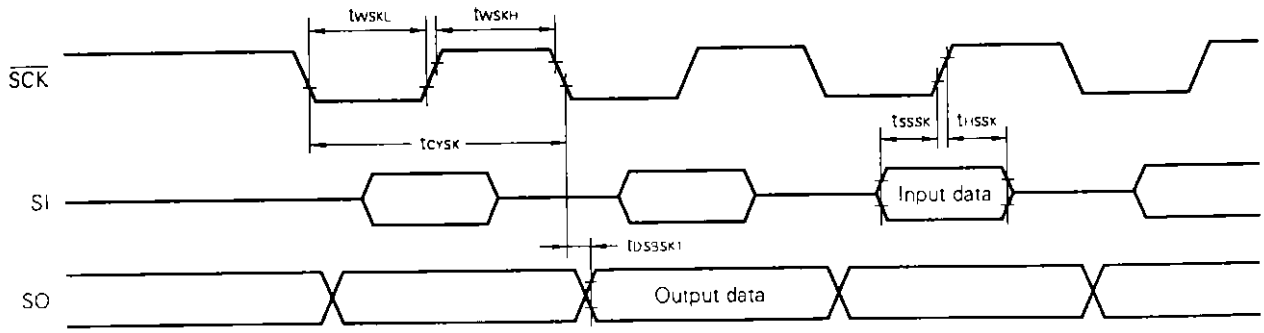


Write operation:



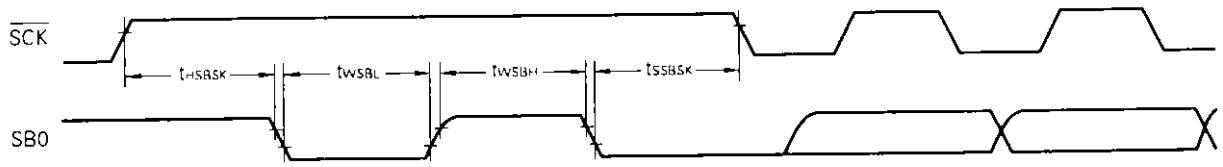
Serial Operation

Three-wire serial I/O mode:

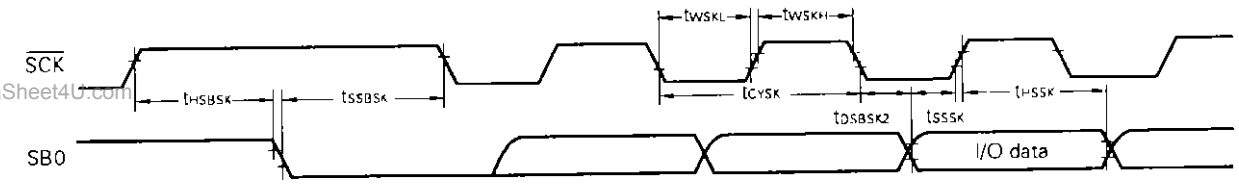


SBI Mode

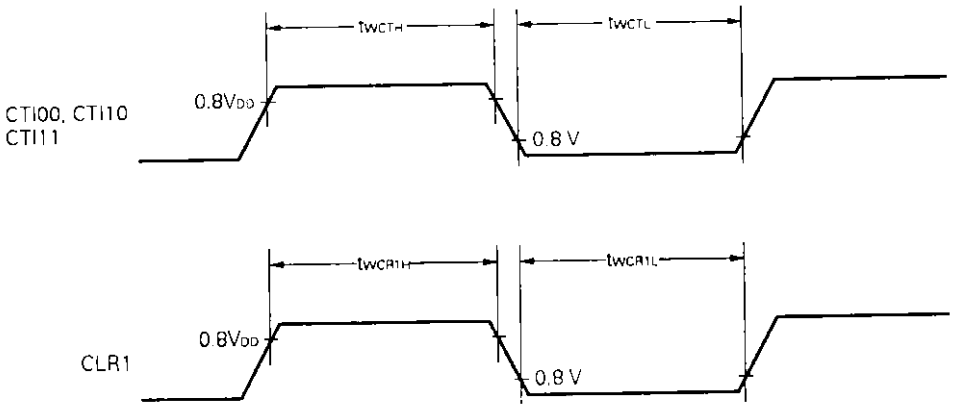
Bus release signal transfer:



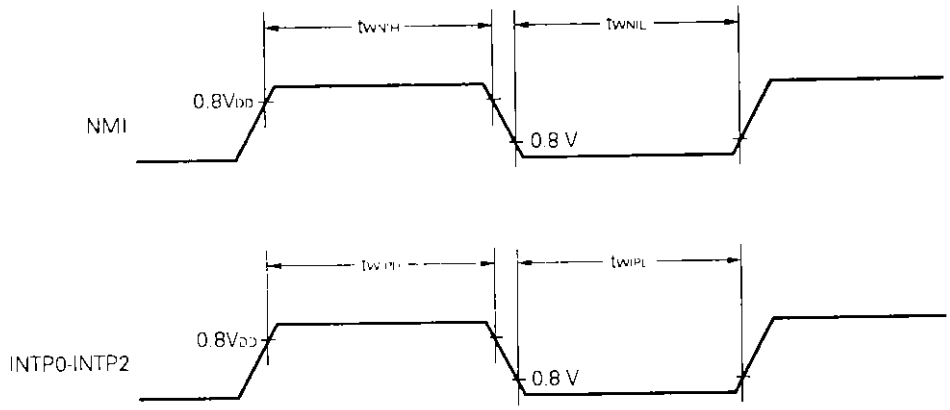
Command signal transfer:



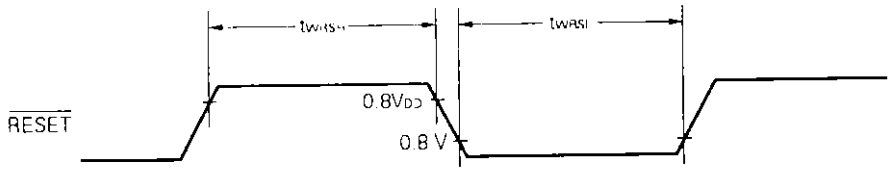
Super Timer Unit Input Timing



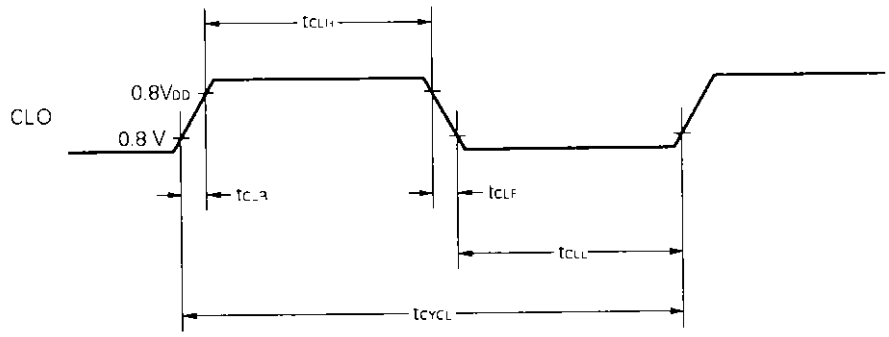
Interrupt Input Timing



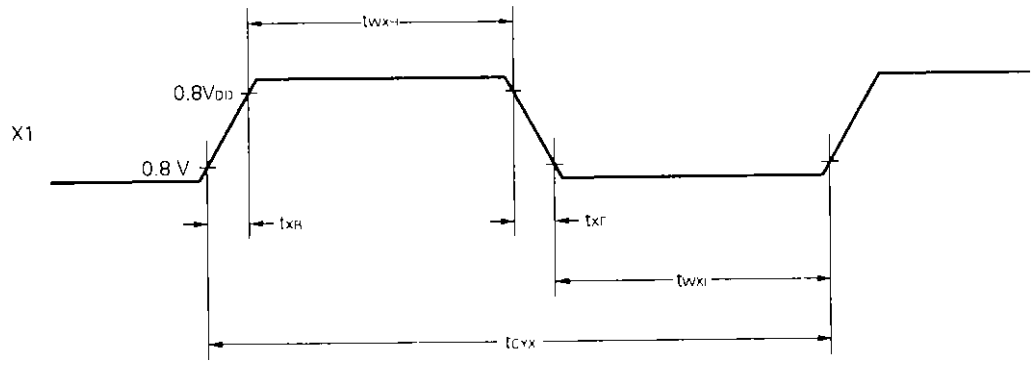
Reset Input Timing



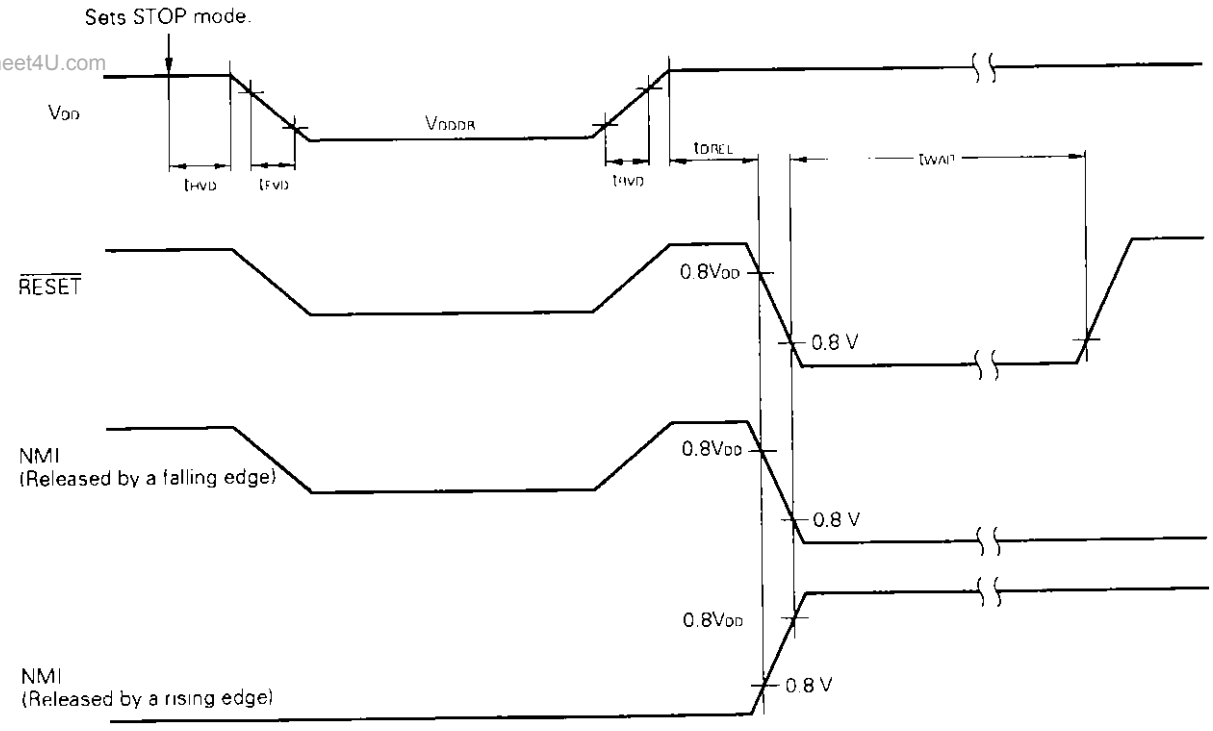
Clock Output Timing



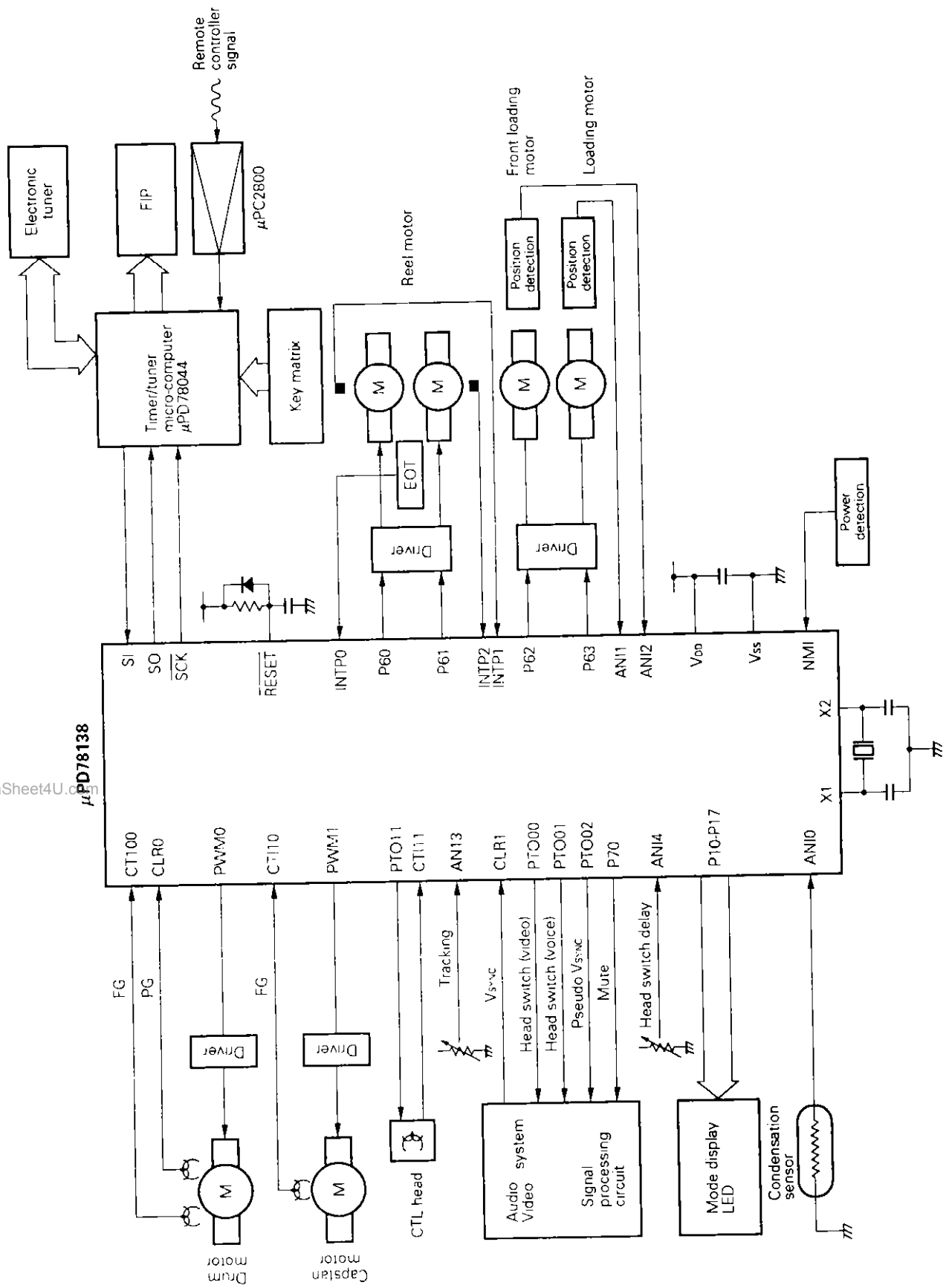
External Clock Timing



Data Retention Timing

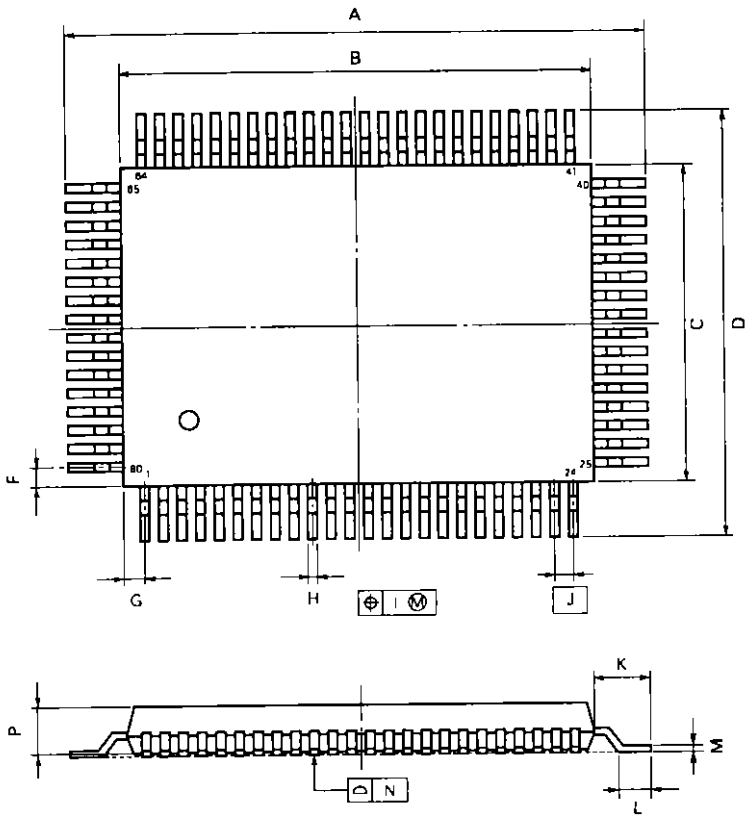


9. APPLICATION EXAMPLE (NORMAL-TYPE VIDEO CASSETTE RECORDER)

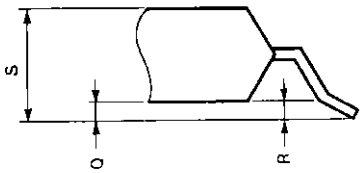


10. PACKAGE DIMENSIONS

80-PIN PLASTIC QFP (14 × 20) (UNIT: mm)



Detail of lead end



P80GF-80-3B9-2E

ITEM	MILLIMETERS	INCHES
A	$23.6^{\pm 0.4}$	$0.929^{\pm 0.016}$
B	$20.0^{\pm 0.2}$	$0.795^{+0.009}_{-0.008}$
C	$14.00^{\pm 0.2}$	$0.551^{+0.009}_{-0.008}$
D	$17.6^{\pm 0.4}$	$0.693^{\pm 0.016}$
F	1.0	0.039
G	0.8	0.031
H	$0.35^{\pm 0.10}$	$0.014^{+0.004}_{-0.005}$
I	0.15	0.006
J	0.8 (T.P.)	0.031 (T.P.)
K	$1.8^{\pm 0.2}$	$0.071^{+0.008}_{-0.009}$
L	$0.8^{\pm 0.2}$	$0.031^{+0.009}_{-0.008}$
M	$0.15^{+0.10}_{-0.05}$	$0.006^{+0.004}_{-0.003}$
N	0.15	0.006
P	2.7	0.106
Q	$0.1^{\pm 0.1}$	$0.004^{\pm 0.004}$
R	$0.1^{\pm 0.1}$	$0.004^{\pm 0.004}$
S	3.0 MAX.	0.119 MAX.

NOTE

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

★ 11. RECOMMENDED SOLDERING CONDITIONS

The following conditions (see table below) must be met when soldering μPD78138.

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

Table 11-1 Recommended Soldering Conditions

Part number	Package	Symbol
μPD78134AGF-xxx-3B9	80-pin plastic QFP	IR30-107-1
μPD78136GF-xxx-3B9		VP15-107-1
μPD78138GF-xxx-3B9		WS60-107-1
		Partial heating method

Table 11-2 Soldering Conditions

Symbol	Soldering process	Soldering conditions
IR30-107-1	Infrared ray reflow	Peak package's surface temperature: 230 °C Reflow time: 30 seconds or less (210 °C or higher) Number of reflow processes: 1 Exposure limit: 7 days (*) (10 hours of pre-baking is required at 125 °C afterward.)
VP15-107-1	VPS	Peak package's surface temperature: 215 °C Reflow time: 40 seconds or less (200 °C or higher) Number of reflow processes: 1 Exposure limit: 7 days (*) (10 hours of pre-baking is required at 125 °C afterward.)
WS60-107-1	Wave soldering	Solder temperature: 260 °C or less Flow time: 10 seconds or less Number of flow processes: 1 Exposure limit: 7 days (*) (10 hours of pre-baking is required at 125 °C afterward.) Pre-baking temperature: 120 °C max. (package surface temperature)
Partial heating method	Partial heating method	Terminal temperature: 300 °C or less Flow time: 3 seconds or less (one side per device)

* 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."

Remark For the details of the recommended soldering conditions refer to our document "SMT MANUAL" (IEI-1207).

APPENDIX A DIFFERENCES BETWEEN THE μPD78138 SERIES AND μPD78134 ★

The differences between the μPD78138 series (μPD78134A, μPD78136, μPD78138, and μPD78P138) and μPD78134 are as follows:

① ROM/RAM size

Item	μPD78134	μPD78134A	μPD78136	μPD78138	μPD78P138
ROM	16K bytes (Mask ROM)		24K bytes (Mask ROM)	32K bytes (Mask ROM)	32K bytes (PROM)
RAM	384 bytes		640 bytes		

② Added instructions

The following 15 instructions are added in the μPD78138 series.

- Signed multiply instruction : MULSW r
- 8-bit data transfer instruction : MOV A, [HL+]
  MOV [HL+], A
  MOV A, [DE]
  MOV [DE], A
  MOV A, [DE+]
  MOV [DE+], A
  MOV A, !addr16
  MOV !addr16, A
  XCH A, [HL]
  XCH A, [DE]
  XCH A, word [r1]
- 8-bit arithmetic/logical instruction : ALU A, [DE]
  ALU A, word [r1]
- Call instruction : CALL rp

Remark Legend

A	:	Register A
rp	:	AX, BC, DE, or HL
r	:	A, X, B, C, D, E, H, or L
r1	:	A or B
word	:	16-bit immediate data or label
!addr16	:	0000H-FFFFH immediate data or label
HL	:	Register pair
DE	:	Register pair
ALU	:	Generic for all the mnemonics of the 8-bit arithmetic/logical instructions (ADD, ADDC, SUB, SUBC, AND, OR, XOR, CMP)

③ **Operation when a value is written in an event counter compare register (ECCO or ECC1)**

- μPD78134 : Clears the event counter (EC).
- μPD78138 series: Clears the event counter (EC) when 0xxxxxxxB is written. Does not clear event counter (EC) when 1xxxxxxxB is written.

④ **Threshold width for pulse elimination for the digital noise eliminator**

- μPD78134 : 40/f_{CLK}
- μPD78138 series: 32/f_{CLK} or 72/f_{CLK} selectable

⑤ **PWM carrier frequency**

- μPD78134 : 23.4 kHz
- μPD78138 series: 23.4 kHz or 46.9 kHz selectable

⑥ **Restrictions are lifted for the real-time output port control mode in macro service**

In the μPD78138 series, the restriction that the output timing data must be stored in ROM in the real-time output control mode in macro service in the μPD78134 is lifted.

Remark Refer to the **μPD78134 DATA SHEET (IC-7839)** for the details of the μPD78134.

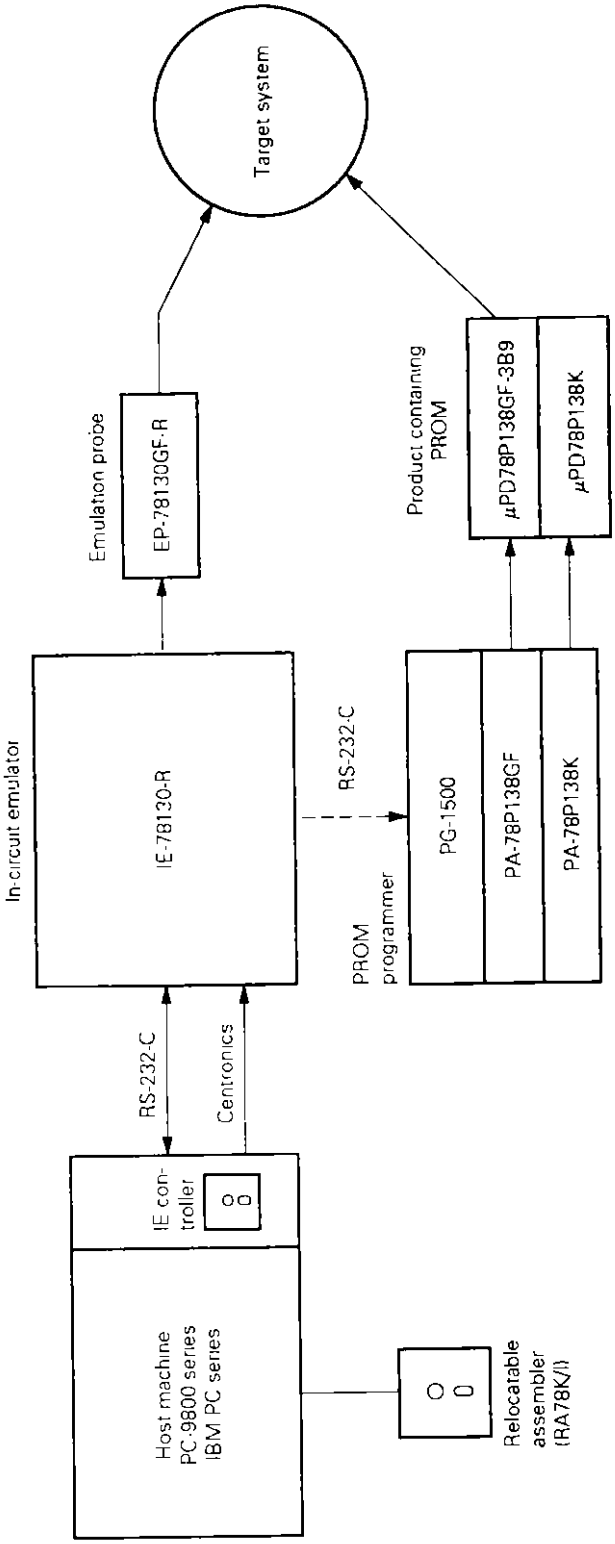
APPENDIX B DEVELOPMENT TOOLS ★

The following development tools are readily available for development of systems using the μPD78138.

Hardware	IE-78130-R	IE-78130-R is an in-circuit emulator which can be used for developing and debugging application systems for μPD78138. This emulator is connected to the host machine or console when debugging is performed. The connection of the emulator to the host machine enables symbolic debugging and file transfer between the emulator and the host machine for enhancement of debugging efficiency. IE-78130-R has the Centronics interface in addition to two channels of the RS-232-C serial interfaces, so that it can be connected to the PROM programmer such as PG-1500.			
	EP-78130GF-R	EP-78130GF-R is an emulation probe for connecting the IE-78130-R to a target system.			
	PA-78P138GF PA-78P138K	PA-78P138GF and PA-78P138K are socket adapters for writing a program into μPD78P138GF/K using PG-1500.			
Software	IE-78130-R control program (IE-controller)	Host machine	OS	Distribution media	Part number
		PC-9800 series	MS-DOS™ (Ver. 3.10 to Ver 3.30C)	5-inch 2HD	μS5A10IE78130
				3.5-inch 2HD	μS5A13IE78130
		IBM PC series	PC DOS™ (Ver. 3.1 to Ver 3.3)	5-inch 2HC	μS7B10IE78130
	RA78K/I relocatable assembler	Host machine	OS	Distribution media	Part number
		PC-9800 series	MS-DOS (Ver. 3.10 to Ver 3.30C)	5-inch 2HD	μS5A10RA78K1
				3.5-inch 2HD	μS5A13RA78K1
		IBM PC series	PC DOS (Ver. 3.1 to Ver 3.3)	5-inch 2HC	μS7B10RA78K1

Remark IE-controller and assembler operations are guaranteed only on the host machine and by the OS mentioned above.

CONFIGURATION OF DEVELOPMENT TOOLS



Remark PA-78P138GF/K can be directly connected to the PG-2000 when the PG-2000 is used.
(The PG-2000 is no longer manufactured.)

APPENDIX C SERIES PRODUCT FUNCTIONS

Product	μPD78134A	μPD78136	μPD78138	μPD78P138
Number of basic instructions	64			
Minimum instruction execution time	0.33 μs (at 12 MHz)			
Internal memory ROM	16K bytes (Mask ROM)	24K bytes (Mask ROM)	32K bytes (Mask ROM)	32K bytes (PROM)
Internal memory RAM	384 bytes		640 bytes	
Memory expansion	Externally expandable up to 64K bytes			
General register	8 bits × 8 × 4 banks (memory mapping)			
Instruction set	• Signed multiplication (signed 16 bits × absolute 8 bits) • Unsigned multiplication/division (16 bits × 8 bits, 16 bits ÷ 8 bits) • 16-bit addition, subtraction, comparison • Bit manipulation (transfer, Boolean operation, set, reset, test) • BCD correction • Addressing enabling data access to 64K-byte space			
I/O line	66 Total: Input port : 10; Output port : 12; I/O port : 36; Analog input : 8			
Super timer unit	• Timer, counter: Timer : 16 bits × 3, 7 bits × 1; Counter : 18 bits × 1 • Register: Capture register : 18 bits × 1, 16 bits × 4, 7 bits × 1; Compare register : 16 bits × 6, 7 bits × 1 • PWM: 12 bits × 2 (variable active level, two selectable carrier frequencies (23.4/46.9 kHz)) (at 12 MHz) • Clear operation of event counter (EC) added			
Real-time output port	• Timer-connected port output function • 4 bits × 2 or 8 bits × 1			
Serial interface	• Either NEC format serial bus interface (SBI) or 3-wire serial interface can be selected.			
A/D converter	• 8-bit precision × 8 inputs • Conversion time: 30 μs/1 analog input (at 12 MHz)			
Interrupt	• Interrupt source: 17 (5 external and 12 internal) • One of the two service modes can be selected (macro service/vector interrupt). • Variable 2-level interrupt priority			
Standby	• STOP mode			
Pull-up resistor	44, built-in (enable/disable built-in can be specified via software)			
Package	• 80-pin plastic QFP (14 × 20 m excluding the dimensions of the pins) • 80-pin LCC with window (μPD78P138 only)			

[MEMO]