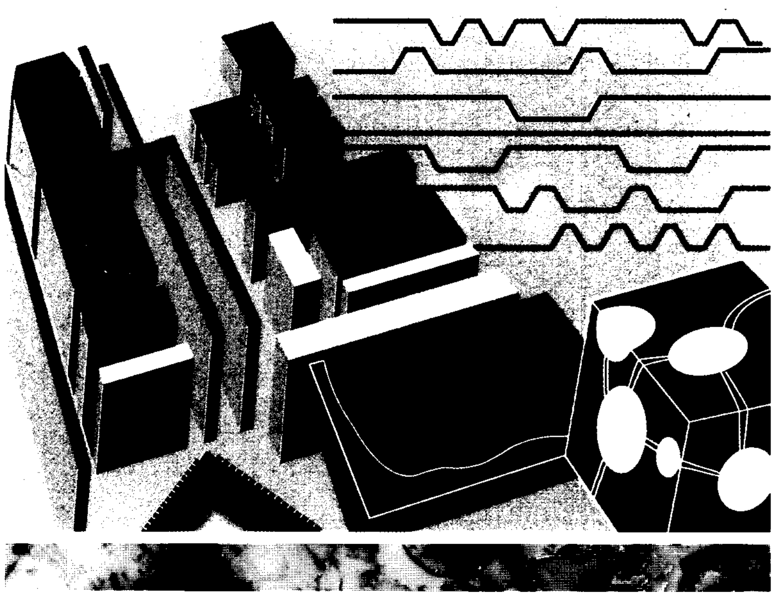
March 1992

K-Series



μPD78350

8-/16-Bit, Single-Chip Microcomputers

Data Sheet

NEC

The uPD78350 is a product of the 16/8-bit single-chip microcomputer 78K/III series. It contains a 16-bit high-performance CPU.

The uPD78350 contains only hardware necessary for operating as an ASIC controller so that a unique application system with the ASIC connected can be developed. And, since the sum-of-products instruction is added to enhance operation functions, the uPD78350 can be used in many fields as high-speed, simple CPU.

Features

- o 16-bit internal architecture, 8-bit external data bus

- o High-speed data processing using the pipeline control system and high-speed operation clock
  - . Instruction cycle: 160 ns (internal clock frequency: 12.5 MHz)

- o Internal memory: ROM: Not provided
  RAM: 640 bytes

- o An instruction set suited for control applications (uPD78322 upward compatible)
  - . Multiply/divide instruction (16 bits x 16 bits, 32 bits ÷ 16 bits)
  - . Sum-of-products operation instruction
  - . Bit manipulation instruction and so on

- o Built-in high-speed interrupt controller
  - . A 4-level priority can be specified.
  - . One interrupt processing mode can be selected out of three types: vector interrupt function, macro service function, and context switching function.

- Wait control for a bus cycle is possible from the external device.
  - External wait pin

- 8-bit PWM signal output function: 2 channels

Application

- Office automation (OA) field such as for hard disk drive or floppy disk drive control

- Factory automation (FA) field

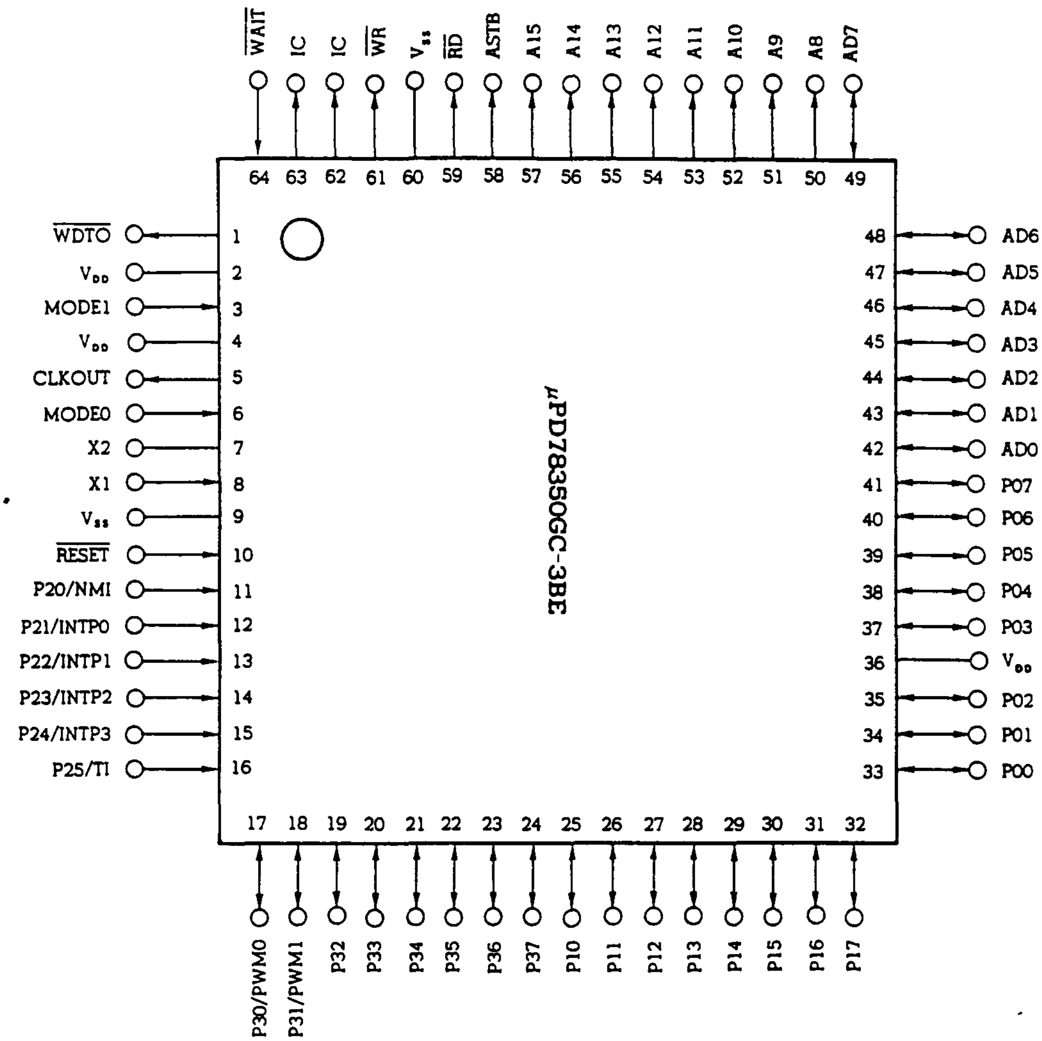
Ordering information

Order code	Package	Quality grade
uPD78350GC-3BE	64-pin plastic QFP (14 x 14 mm)	Standard

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)

64-pin plastic QFP (14 x 14 mm)



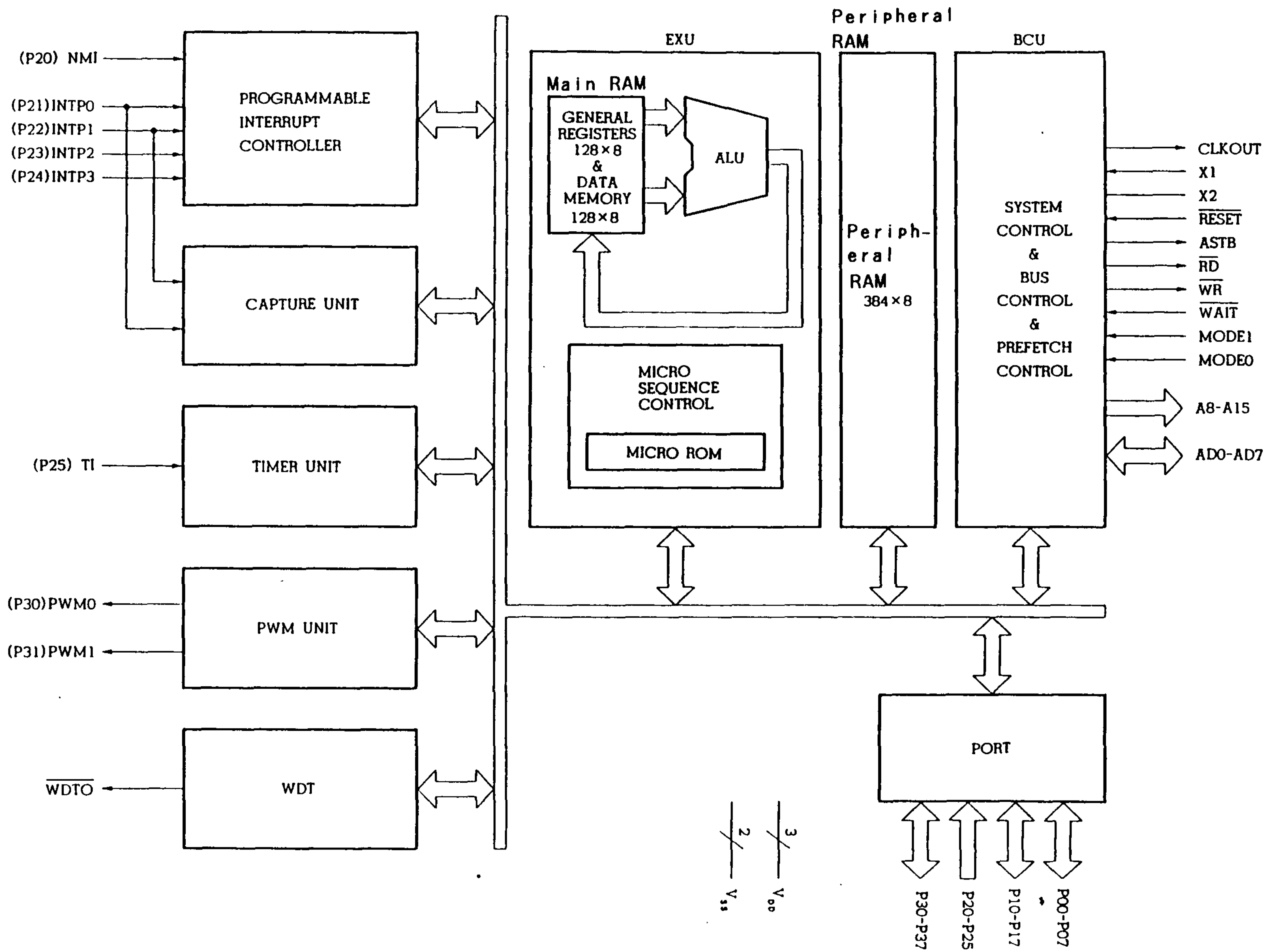
Caution: Leave the IC pins open.

P00-P07:	Port 0
P10-P17:	Port 1
P20-P25:	Port 2
P30-P37:	Port 3
NMI:	Nonmaskable interrupt
INTP0-INTP3:	Interrupt from peripherals
TI:	Timer input
PWM0, PWM1:	Pulse width modulation output
$\overline{WDTO}$:	Watchdog timer output
MODE0, MODE1:	Mode
AD0-AD7:	Address/data bus
A8-A15:	Address bus
ASTB:	Address strobe
$\overline{RD}$:	Read strobe
$\overline{WR}$:	Write strobe
CLKOUT:	Clock output
$\overline{WAIT}$:	Wait
$\overline{RESET}$:	Reset
X1, X2:	Crystal
V_{DD}:	Power supply
V_{SS}:	Ground
IC:	Internaliy connected

Function overview

Item	Description
Number of basic instructions	113
Minimum instruction execution time	160 ns (internal clock frequency: 12.5 MHz, external clock frequency: 25.0 MHz)
Internal memory	. ROM: Not provided . RAM: 640 bytes
Memory space	64K bytes (can externally be extended)
General register	8 bits x 16 x 8 banks
Instruction set	. 16-bit transfer or arithmetic/logical instruction . Unsigned multiply/divide instruction (16 bits x 16 bits, 32 bits ÷ 16 bits) . Bit manipulation instruction . String instruction . Sum-of-products instruction
Capture/timer unit	. 16-bit timers: 3 channels 16-bit capture registers: 2 16-bit compare registers: 2
Interrupt function	. A 4-level priority can be specified by software. . One interrupt processing mode can be selected out of three types: vector interrupt function, macro service function, and context switching function.
I/O line	. Input ports: 6 . I/O ports: 24
PWM unit	8-bit PWM outputs: 2 channels
Package	. 64-pin plastic QFP (14 x 14 mm)
Others	. Watchdog timer function . Standby function (HALT, STOP) . External wait pin

Block diagram



1. PIN FUNCTIONS

1.1 Port Pins

Pin name	I/O	Function	Dual-function pin
P00-P07	I/O	Port 0 8-bit I/O port Can be specified as input or output bit by bit.	–
P10-P17	I/O	Port 1 8-bit I/O port Can be specified as input or output bit by bit.	–
P20	I	Port 2 6-bit input dedicated port	NMI
P21			INTP0
P22			INTP1
P23			INTP2
P24			INTP3
P25			TI
P30	I/O	Port 3 8-bit I/O port Can be specified as input or output bit by bit.	PWM0
P31			PWM1
P32-P37			–

1.2 Non-port Pins

Pin name	I/O	Function	Dual-function pin
NMI	I	Nonmaskable interrupt request input	P20
INTP0		External interrupt request input	P21
INTP1			P22
INTP2			P23
INTP3			P24
TI		External count input to timer 1 (TM1)	P25
PWM0	O	PWM signal output	P30
PWM1			P31
$\overline{WDTO}$		Signal output which indicates the occurrence of a watchdog timer interrupt	–
MODE0	I	Control signal input to set an operation mode. Normally, connect the MODE0 to V_{DD} and the MODE1 to V_{SS}.	–
MODE1			
AD0-AD7	I/O	Multiplexed address/data bus when an external memory is expanded	–
A8-A15	O	Address bus when an external memory is expanded	–
ASTB	O	Address strobe signal output	–
$\overline{RD}$		Read strobe signal output to the external device	–
$\overline{WR}$		Write strobe signal output to the external memory	–
CLKOUT		System clock output	–
$\overline{WAIT}$	I	Control signal input to set a bus cycle to the wait state	–
$\overline{RESET}$	I	System reset input	–

(to be continued)

(Cont'd)

Pin name	I/O	Function	Dual-function pin
X1	I	Crystal input pin for system clock oscillation: A clock signal provided externally is input to the X1 pin.	−
X2	−		−
V_{DD}	−	Positive power supply	−
V_{SS}	−	Ground	−
IC	−	Internally connected pin. Leave open.	−

1.3 Input/Output Circuits of Each Pin

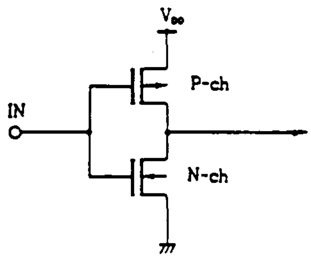
Table 1-1 and Figure 1-1 show the input and output circuits of each pin in a simplified format.

Table 1-1 Input/Output Circuits of Each Pin

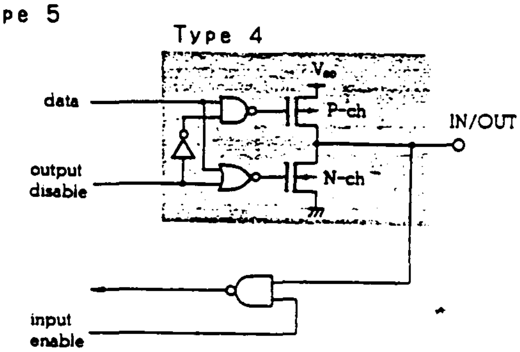
Pin	I/O circuit type
P00-P07	5
P10-P17	
P30-P37	
AD0-AD7	
A8-A15	
P20-P25	2
ASTB	4
$\overline{RD}$	
$\overline{WR}$	
$\overline{WDTO}$	14
CLKOUT	3
MODE0, MODE1	1
$\overline{WAIT}$	
$\overline{RESET}$	2

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

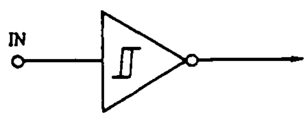
Type 1



Type 5

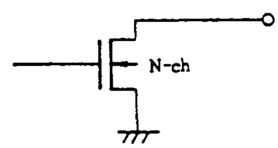


Type 2

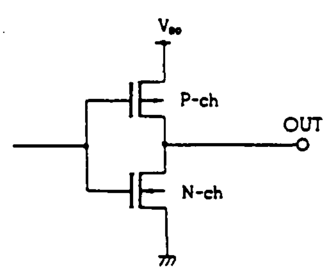


Schmitt-triggered input with hysteresis characteristics

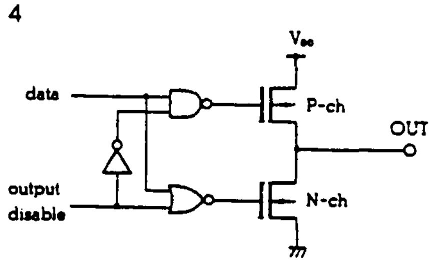
Type 14



Type 3



Type 4



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

1.4 Handling Unused Pins

Table 1-2 Handling Unused Pins

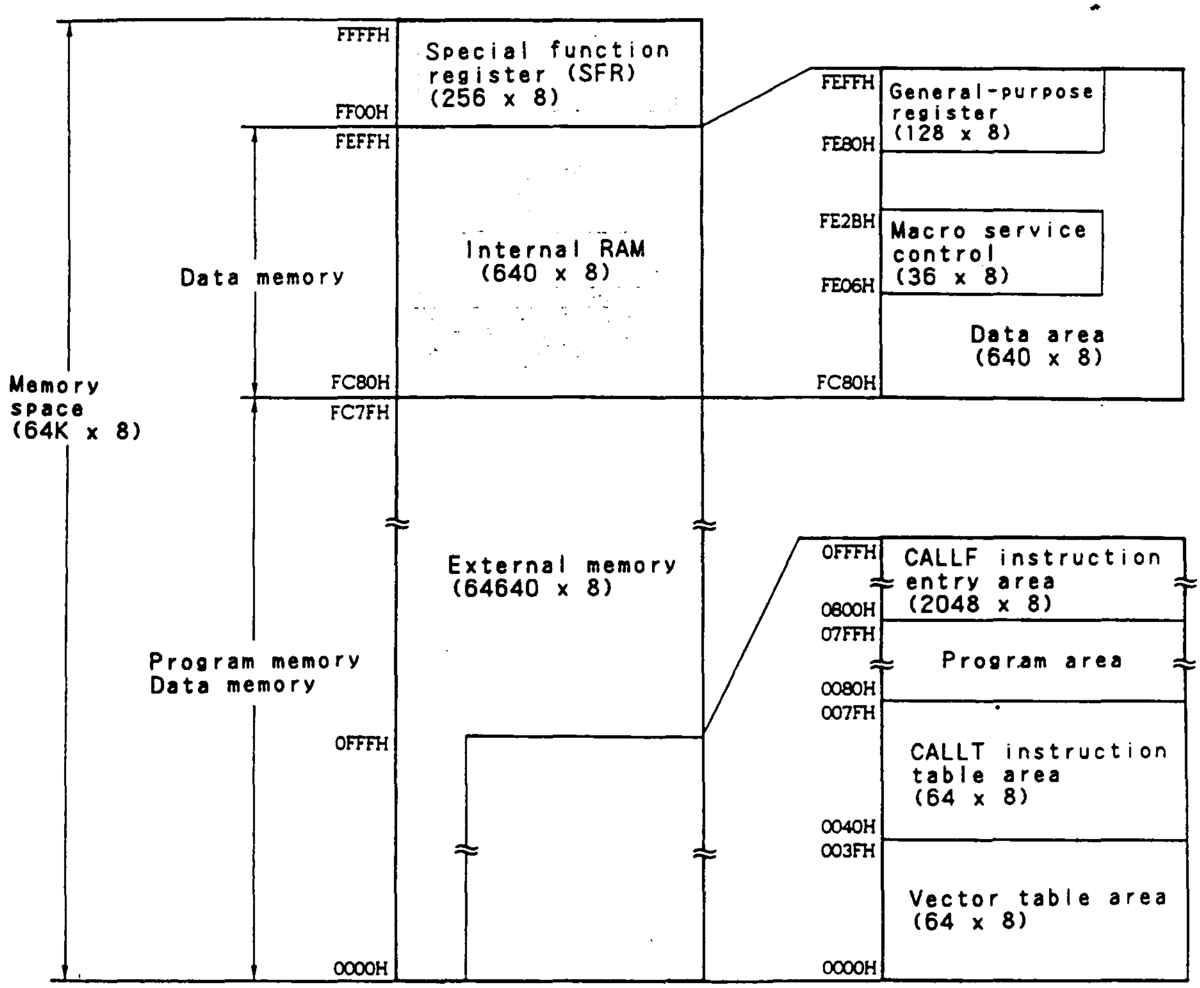
<table>
<tr><th>Pin</th><th>Recommended connections</th></tr>
<tr><td>P00-P07</td><td rowspan="2">Input status: To be connected to the V_{DD} or V_{SS} pin via a resistor.
Output status: Open</td></tr>
<tr><td>P10-P17</td></tr>
<tr><td>P20-P25</td><td>To be connected to the V_{SS} pin.</td></tr>
<tr><td>P30-P37</td><td rowspan="3">Input status: To be connected to the V_{DD} or V_{SS} pin via a resistor.
Output status: Open</td></tr>
<tr><td>AD0-AD7</td></tr>
<tr><td>A8-A15</td></tr>
<tr><td>ASTB</td><td rowspan="5">Open</td></tr>
<tr><td>$\overline{RD}$</td></tr>
<tr><td>$\overline{WR}$</td></tr>
<tr><td>$\overline{WDTO}$</td></tr>
<tr><td>CLKOUT</td></tr>
<tr><td>$\overline{WAIT}$</td><td>To be connected to the V_{DD} pin.</td></tr>
<tr><td>IC</td><td>Open</td></tr>
</table>

2. CPU ARCHITECTURE

2.1 Memory Space

The uPD78350 can access memory of up to 64K bytes. Figure 2-1 shows the memory map.

Fig. 2-1 Memory Map



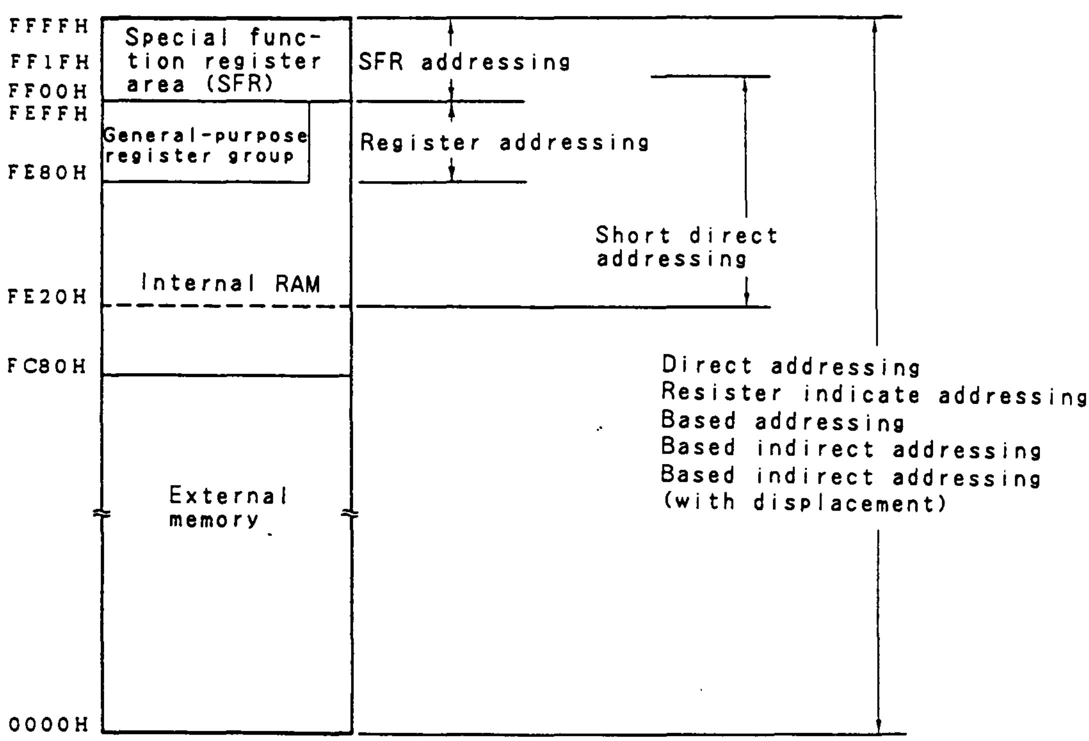
Remark: Shaded portions indicate internal memory.

2.2 Data Memory Addressing

Various addressing modes are provided for the uPD78350 to improve memory operability or to enable the use of a high-level language. Special addressing is applicable, in particular, to the space of data memory from FC80H to FFFFH according to each function of the special function register (SFR) group and general register group.

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

Fig. 2-2 Addressing Space of Data Memory



2.3 Processor Registers

The uPD78350 contains three processor register groups.

2.3.1 Control registers

(1) Program counter (PC)

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

(2) Program status word (PSW)

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

(3) Stack pointer (SP)

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

(4) CPU control word (CCW)

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

Fig. 2-3 Control Register Configuration

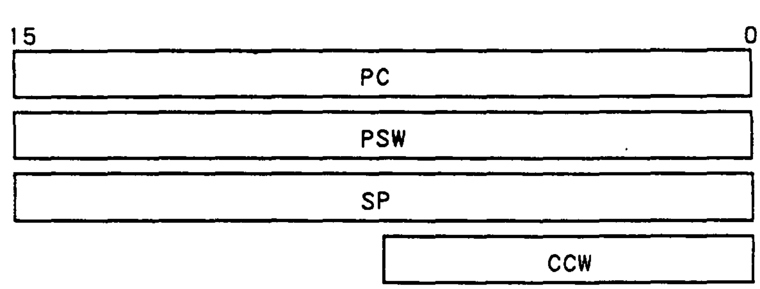


Fig. 2-4 PSW Configuration

PSW

15							8
UF	RBS2	RBS1	RBS0	0	0	0	0
7							**0**
S	Z	RSS	AC	IE	P/V	0	CY

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

Fig. 2-5 CCW Configuration

CCW

7							0
0	0	0	0	0	0	TPF	0

TPF: Table position flag

2.3.2 General register

The general register group consists of eight banks (one bank: 8 words x 16 bits). Figure 2-6 shows general register configuration. The general register group is mapped into addresses from FE80H to FFEFH, and functions as a 16-bit register as well as an 8-bit register (see Figure 2-7). The use of this register enables easy control of complicated multitask processing.

Fig. 2-6 General Register Configuration

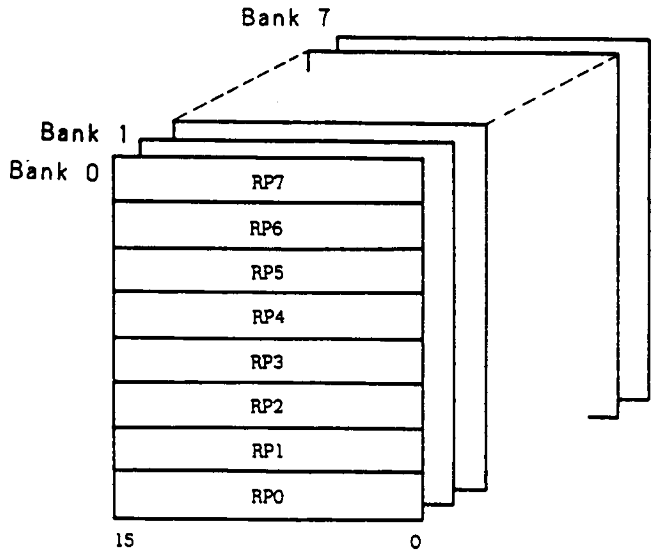
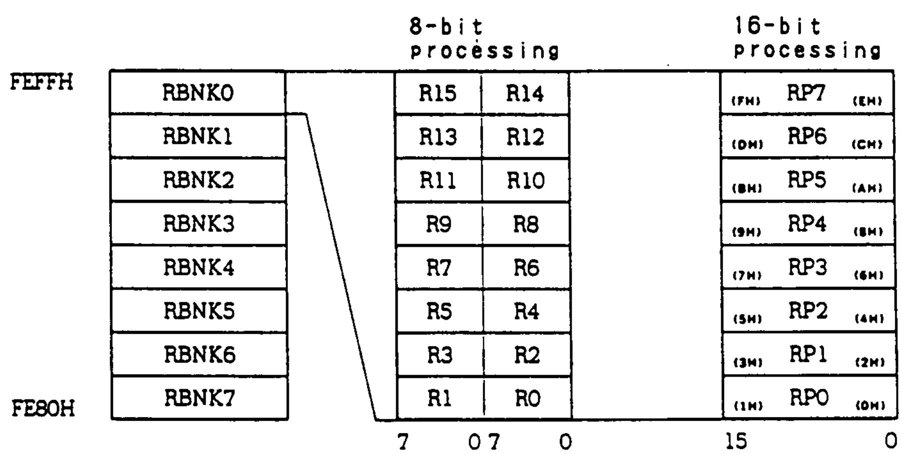


Fig. 2-7 Bit Processing for General Register



2.3.3 Special function registers (SFR)

The special function register group consists of the registers for control of the peripheral hardware the uPD78350 contains. This register group is mapped into addresses from FF00H to FFFFH. The operation of these registers enables control of ports, a timer, and PWM unit.

Table 2-1 Special Function Registers

Address	Special function register (SFR) name	Abbre-viation	R/W	Manipulation bit unit			At resetting
				1	8	16	
FF00H	Port 0	P0	R/W	o	o	–	*
FF01H	Port 1	P1		o	o	–	
FF02H	Port 2	P2	R	o	o	–	
FF03H	Port 3	P3	R/W	o	o	–	
FF10H FF11H	Capture register 00	CT00	R/W	–	–	o	Undefined
FF12H FF13H	Capture register 01	CT01		–	–	o	
FF14H FF15H	Compare register 10	CM10		–	–	o	
FF1EH FF1FH	Compare register 20	CM20		–	–	o	
FF20H	Port 0 mode register	PM0	R/W	o	o	–	FFH
FF21H	Port 1 mode register	PM1		o	o	–	
FF23H	Port 3 mode register	PM3		o	o	–	
FF30H FF31H	Timer register 0	TM0	R	–	–	o	00H
FF32H FF33H	Timer register 1	TM1		–	–	o	
FF34H FF35H	Timer register 2	TM2		–	–	o	
FF38H	Timer control register 0	TMC0	R/W	o	o	–	
FF39H	Timer control register 1	TMC1		o	o	–	
FF3CH	External interrupt mode register 0	INTM0		o	o	–	
FF3DH	External interrupt mode register 1	INTM1		o	o	–	

(to be continued)

Table 2-1 Special Function Registers (Cont'd)

Address	Special function register (SFR) name	Abbreviation	R/W	Manipulation bit unit			At resetting
				1	8	16	
FF43H	Port 3 mode control register	PMC3	R/W	o	o	−	00H
FF62H	Port read control register	PRDC		o	o	−	
FF64H	PWM control register	PWMC		o	o	−	
FF66H	PWM buffer register 0	PWM0		o	o	−	Undefined
FF6EH	PWM buffer register 1	PWM1		o	o	−	
FFA8H	In-service priority register	ISPR	R	o	o	−	00H
FFAAH	Interrupt mode control register	IMC	R/W	o	o	−	80H
FFACH	Interrupt mask flag register	MKL		o	o	−	7FH
FFACH FFADH	Interrupt mask flag register	MK(*)		−	−	o	xx7FH
FFC0H	Standby control register	STBC		−	o	−	0000 x000B
FFC1H	CPU control word	CCW		o	o	−	00H
FFC2H	Watchdog timer mode register	WDM		−	o	−	
FFC4H	Memory expansion mode register	MM		o	o	−	0xxx xxxxB
FFC6H FFC7H	Programmable wait control register	PWC		−	−	o	C0AAH
FFD0H | FFDFH	External SFR area	−		o	o	−	Undefined
FFE0H	Interrupt control register (INTOV)	OVIC		o	o	−	43H
FFE1H	Interrupt control register (INTP0)	PIC0		o	o	−	
FFE2H	Interrupt control register (INTP1)	PIC1		o	o	−	
FFE3H	Interrupt control register (INTCM10)	CMIC10		o	o	−	
FFE4H	Interrupt control register (INTCM20)	CMIC20		o	o	−	
FFE5H	Interrupt control register (INTP2)	PIC2		o	o	−	
FFE6H	Interrupt control register (INTP3)	PIC3		o	o	−	

* Used only when a word is accessed by an instruction with the sfrp operand.

3. BLOCK FUNCTION

3.1 Bus Control Unit (BCU)

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

3.2 Execution Unit (EXU)

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

The 256-byte main RAM in the EXU can be accessed at higher speed with an instruction than 384-byte peripheral RAM.

3.3 RAM

The built-in peripheral RAM consists of 384 bytes.

3.4 Interrupt Controller

The interrupt controller processes various interrupt requests (NMI and INTP0 to INTP3) issued from peripheral hardware and external device with the vector interrupt, macro service, or context switching.

The interrupt controller also specifies the 4-level interrupt priority.

3.5 Capture/Timer Unit

The capture/timer unit consists of the following hardware.

. 16-bit timers/counters: 3 channels
  16-bit capture registers: 2
  16-bit compare registers: 2

The capture/timer unit can output a programmable pulse and measure a pulse width and frequency.

3.6 PWM Unit

The uPD78350 has two channels of 8-bit PWM signal outputs. By connecting an external low-pass filter, a PWM output can be used as an analog voltage output.

3.7 Watchdog Timer (WDT)

The 8-bit watchdog timer is built into the CPU to detect a program crash and system error. This microcomputer has the $\overline{\text{WDTO}}$ pin to notify the external device that a watchdog timer interrupt occurs.

3.8 Port

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

Table 3-1 Pin Function

Port	I/O	Function	
P0	8-bit I/O	General port	−
P1	8-bit I/O		−
P2	6-bit input		External interrupt and capture trigger
P3	8-bit I/O		PWM signal output

4. PERIPHERAL HARDWARE FUNCTIONS

4.1 Port Functions

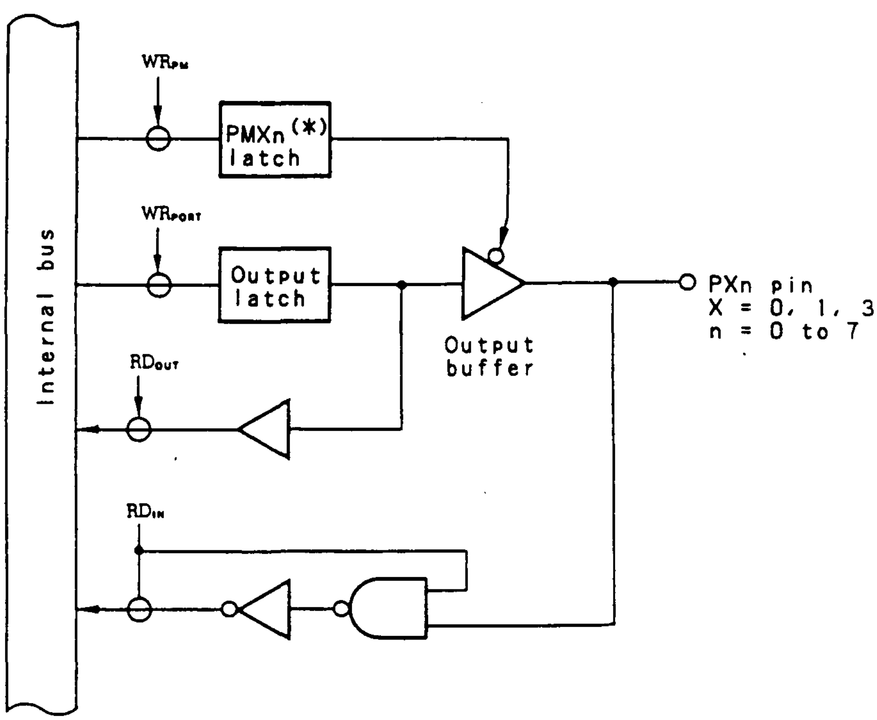
4.1.1 Hardware configuration

As shown in Figure 4-1, three-state bidirectional ports are basically used for the ports of the uPD78350.

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

Figure 4-2 shows the port configuration.

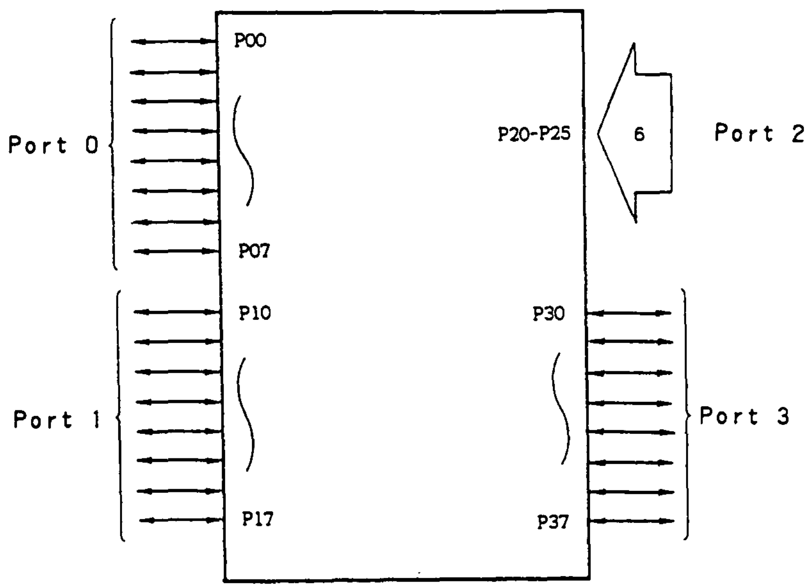
Fig. 4-1 Basic I/O Port Configuration



* PMXn latch: Bit n (n = 0 to 7) of port mode register PMX (X = 0, 1, 3)

Remark: Port 2 is used only for 6-bit input.

Fig. 4-2 Port Configuration



4.1.2 Functions of the digital I/O ports

Table 4-1 lists the ports of the uPD78350.

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

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

Port name	Port function	Additional function
Port 0	8-bit I/O port. Specifiable as input or output bit by bit.	—
Port 1	8-bit I/O port. Specifiable as input or output bit by bit.	—
Port 2	Port used only for 6-bit input	External interrupt input, capture trigger input, and count pulse input in control mode
Port 3	8-bit I/O port. Specifiable as input or output bit by bit.	PWM signal output in control mode

4.1.3 Port output check function

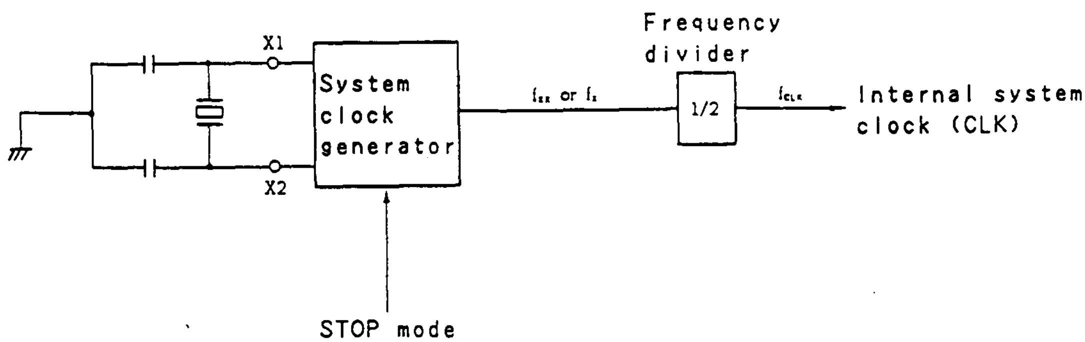
The uPD78350 has a function of reading pin state (pin access mode) to improve system application reliability in port output mode. With this function, output data (output latch data) and actual pin state can be checked as required. For frequent port state checking, special instructions (CHKL and CHKLA) are available.

4.2 Clock Generator

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

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

Fig. 4-3 Block Diagram of the Clock Generator



Remarks 1. f_{XX}: Crystal oscillator frequency

2. f_X: External clock frequency

3. f_{CLK}: Internal system clock frequency

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

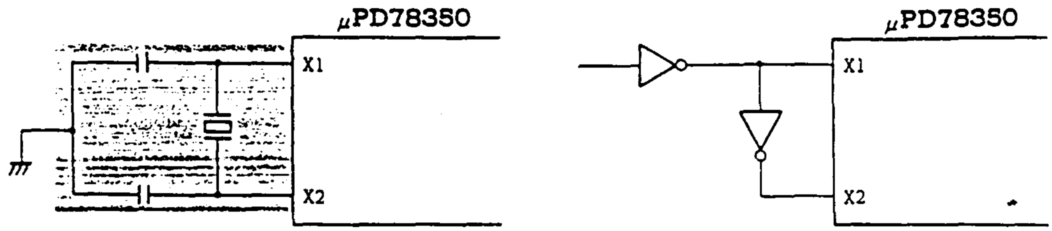
An external clock can be applied. In this case, a clock signal is to be applied to the X1 pin, with the inverted signal to be applied to the X2 pin.

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

The frequency divider divides system clock generator output (f_{XX} for the crystal oscillator or f_X for an external clock) by two to produce an internal system clock (f_{CLK}).

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

(a) Crystal oscillator (b) External clock



Caution: When using the system clock generator, run wires in the shaded area (▭) in Figure 4-4 according to the following rules to avoid effects such as stray capacitance:

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

4.3 Capture/Timer Unit

The capture/timer unit can output programmable pulses and can also measure pulse intervals and frequencies.

The capture/timer unit mainly consists of three timers and four registers.

4.3.1 Configuration of the capture/timer unit

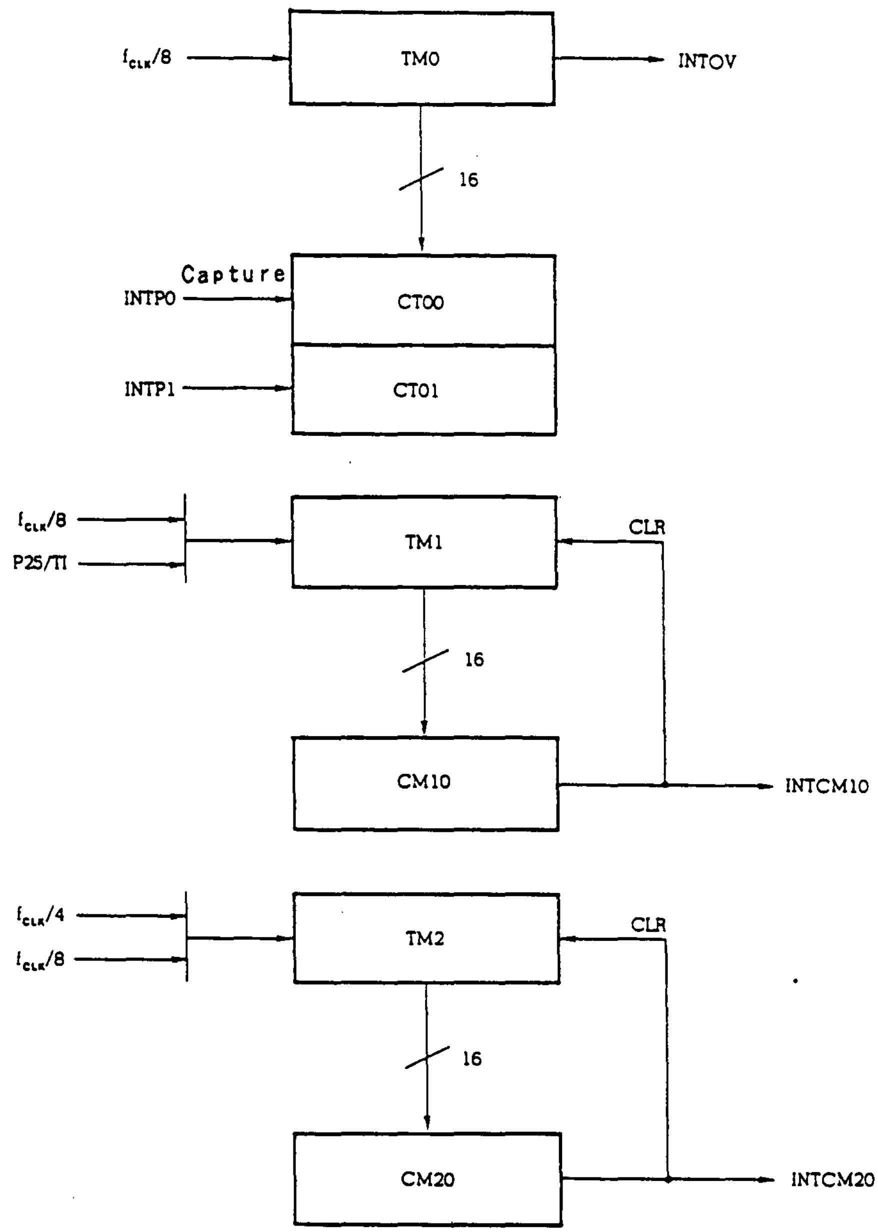
The capture/timer unit consists of the hardware components listed in Table 4-2. Figure 4-5 shows the configuration of the capture/timer unit.

Table 4-2 Components of the Capture/Timer Unit

Timer	Count clock	Register	Compare register match interrupt	Capture trigger
16-bit timer (TM0)	$f_{CLK}/8$	16-bit capture register (CT00) 16-bit capture register (CT01)	– –	INTP0 INTP1
16-bit timer (TM1)	$f_{CLK}/8$ TI pin input	16-bit compare register (CM10)	INTCM10	–
16-bit timer (TM2)	$f_{CLK}/4$ $f_{CLK}/8$	16-bit compare register (CM20)	INTCM20	–

Remarks 1. f_{CLK}: Internal system clock
2. INTP0, INTP1: External interrupt
3. Timer 0 has an overflow interrupt function.
4. Timer 1 is cleared by INTCM10.
5. Timer 2 is cleared by INTCM20.

Fig. 4-5 Configuration of the Capture/Timer Unit



4.3.2 Function

(1) Timer 0 (TM0)

Timer 0 is a 16-bit free-running timer.

Timer 0 counts an internal clock, and generates an overflow interrupt (INTOV) when a timer overflow occurs.

(2) Timer 1 (TM1)

Timer 1 is a 16-bit timer/event counter. Timer 1 can count an internal clock or external event applied to the TI pin.

Timer 1 can be by a match interrupt (INTCM10) from the compare register CM10.

(3) Timer 2 (TM2)

Timer 2 is a 16-bit interval timer. Timer 2 counts an internal clock. Timer 2 is cleared by a match interrupt (INTCM20) from the compare register CM20.

(4) 16-bit compare registers (CM10 and CM20)

A 16-bit compare register compares the contents of each timer with the data held in the compare register at all times, and generates a match signal when a match is found.

See Table 4-2 for detailed information about the configuration of the timers and compare registers, and the correspondence between the compare registers and interrupt sources.

(5) 16-bit capture registers (CT00 and CT01)

A 16-bit capture register takes in (captures) the contents of timer 0 when a capture trigger signal occurs. As a capture trigger, an external interrupt (INTP0 or INTP1) can be used.

See Table 4-2 for the correspondence between the registers and capture triggers.

The occurrence of a capture trigger also means the occurrence of an interrupt. By using a capture register, the pulse width and period of an externally applied pulses can be easily measured.

4.4 PWM Unit

The uPD78350 has two PWM signal outputs of 8-bit resolution. By externally connecting a low-pass filter, a PWM output can be used as a digital-to-analog conversion output. The PWM outputs are most suitable, for example, for a control signal for the actuator of a motor.

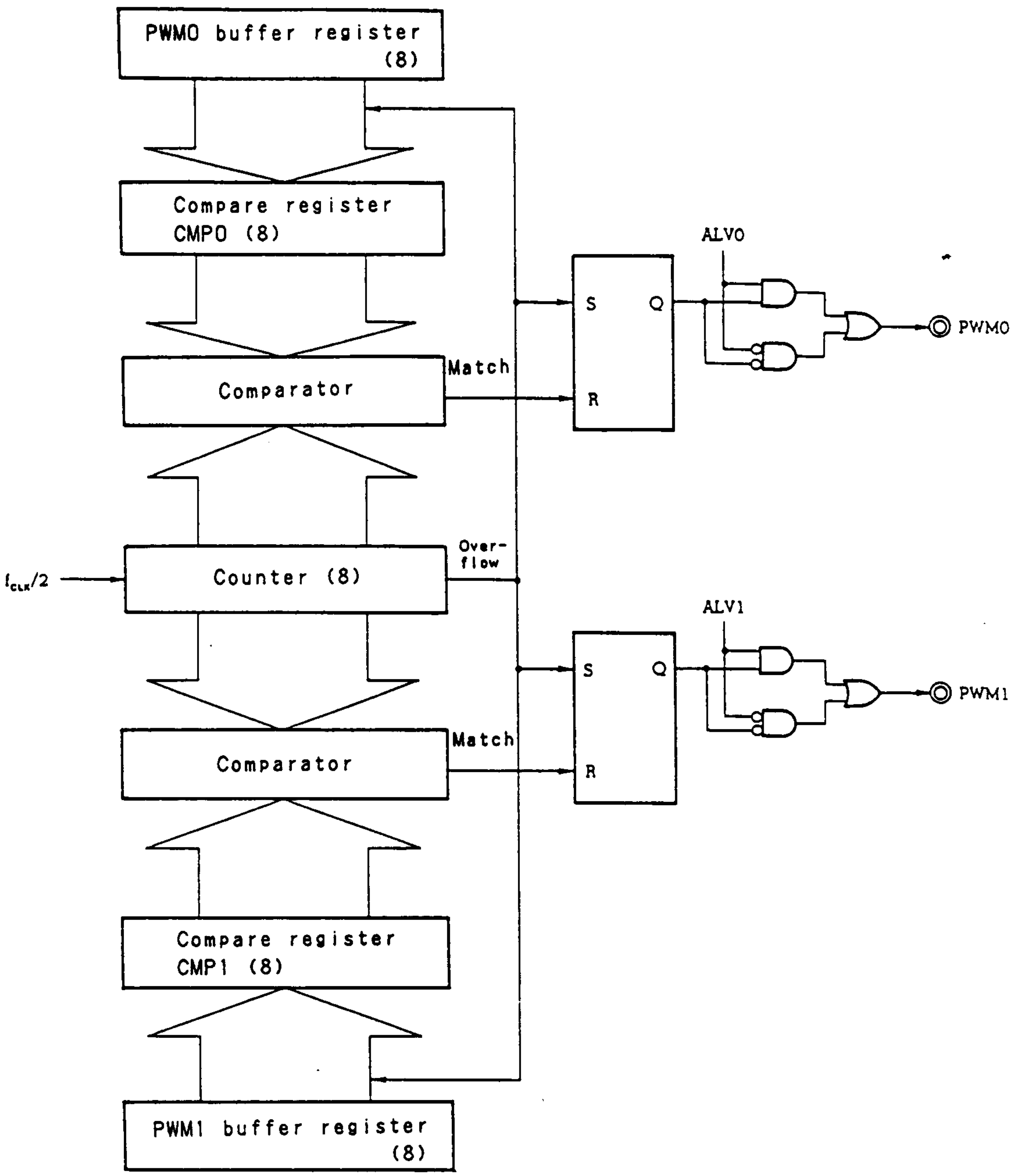
Table 4-3 lists PWM signal output repetition frequencies. Figure 4-6 shows the configuration of the PWM output function.

Table 4-3 PWM Signal Repetition Frequencies

Resolution per bit	Repetition frequency
$2/f_{CLK}$ (0.16 us)	$f_{CLK}/2^9$ (24.4 kHz)

Remark: The values in parentheses are for f_{CLK} = 12.5 MHz.

Fig. 4-6 Configuration of the PWM Output Function



4.5 Watchdog Timer (WDT)

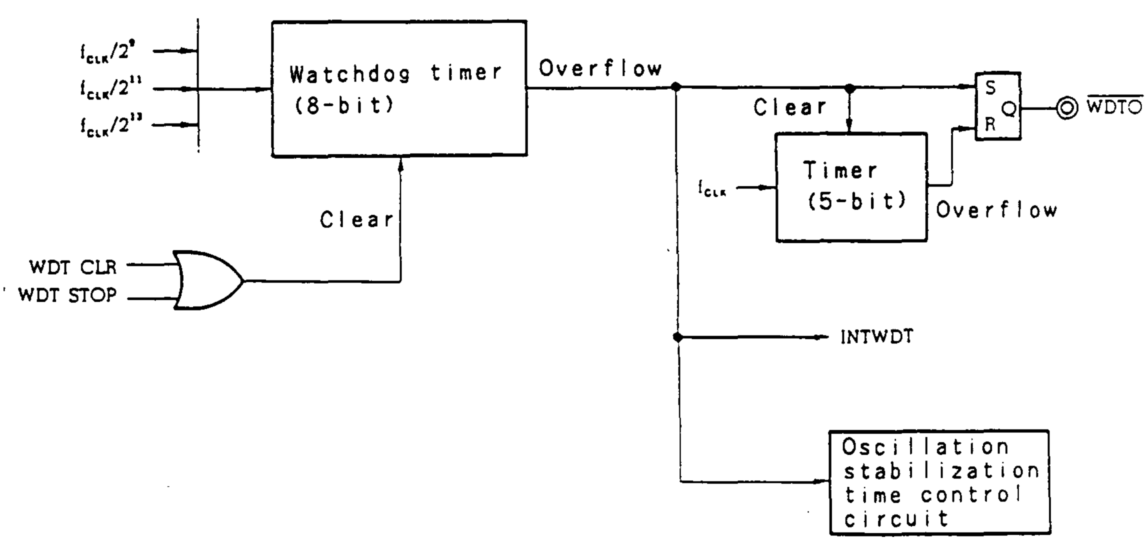
The watchdog timer is a free-running counter with a nonmaskable interrupt function designed to prevent crashes or deadlocks. A program error can be detected when a watchdog timer overflow interrupt (INTWDT) is generated or when the watchdog timer output pin ($\overline{\text{WDTO}}$) goes low. By connecting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

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

4.5.1 WDT configuration

Figure 4-7 shows the configuration of the watchdog timer.

Fig. 4-7 Configuration of the Watchdog Timer



4.5.2 WDT operation

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

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

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

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

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

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

5. INTERRUPT FUNCTION

The uPD7835 has a powerful interrupt function that can handle interrupt requests from the peripheral hardware or other external devices. Three interrupt handling modes are available:

. Vectored interrupt handling
. Macro service
. Context switching

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

Table 5-1 Types of Interrupt Requests and Handling Modes

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

5.1 Types of Interrupt Requests

With the uPD78350, four types of interrupt requests are used:

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

Each type of interrupt request is explained below.

(1) Nonmaskable interrupt

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

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

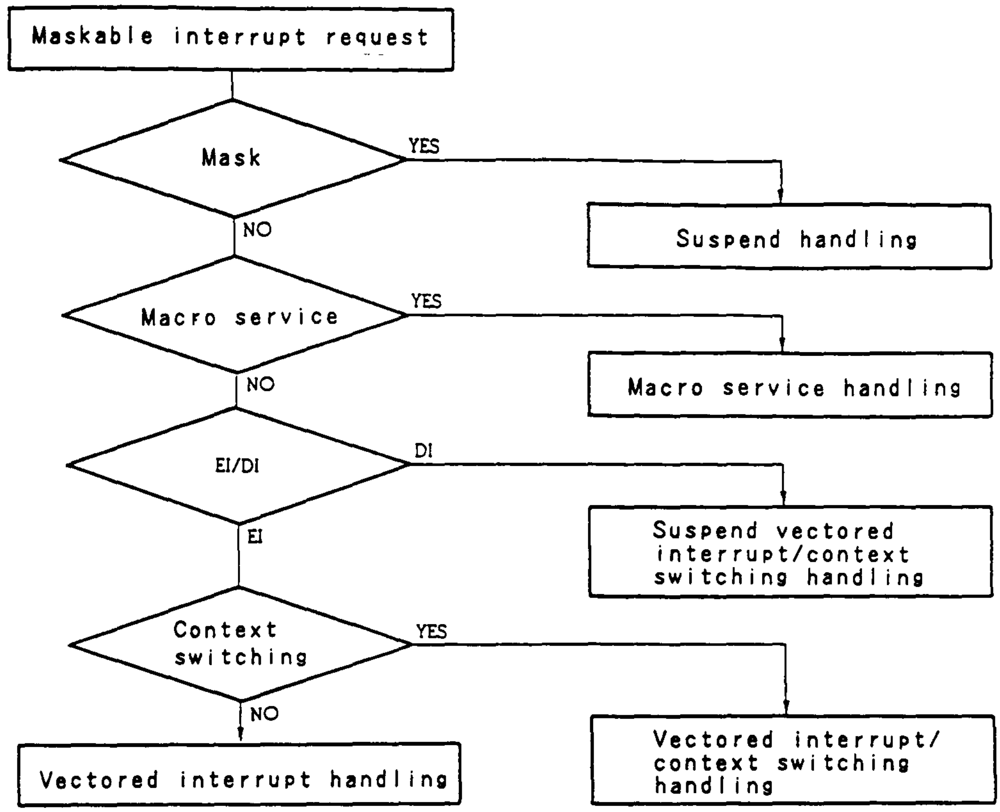
For a nonmaskable interrupt, vectored interrupt handling can be performed.

(2) Maskable interrupt

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

- Vectored interrupt handling
- Macro service
- Context switching

Fig. 5-1 Maskable Interrupt Handling



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

(3) Software interrupt

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

- BRK: Causes a branch to the address indicated by the contents of memory addresses 003EH and 003FH.

- BRKCS: Causes a branch by context switching processing for switching to the register bank specified in the instruction.

(4) Exception trap

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

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

5.2 Interrupt Handling Modes

With the uPD78350, three interrupt handling modes are available:

. Vectored interrupt handling
. Macro service
. Context switching

(1) Vectored interrupt handling

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

(2) Macro service

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

(3) Context switching

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

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

Table 5-2 lists the interrupt sources.

Table 5-2 Interrupt Source List

Type	(*)	Interrupt source		Unit requesting interrupt	Vector table address	Macro service	Context switch
		Name	Trigger				
Non-maskable	–	NMI	NMI pin input	External	0002H	No	No
	–	WDT	Watchdog timer	WDT	0004H		
Maskable	0	INTOV	Timer 0 overflow	Capture/ timer unit	0006H	Yes	Yes
	1	INTP0	INTP0 pin input	External	0008H		
	2	INTP1	INTP1 pin input	External	000AH		
	3	INTCM10	CM10 match signal	Capture/ timer unit	000CH		
	4	INTCM20	CM20 match signal	Capture/ timer unit	000EH		
	5	INTP2	INTP2 pin input	External	0010H		
	6	INTP3	INTP3 pin input	External	0012H		
Software	–	BRK	BRK instruction	–	003EH	No	No
	–	BRKCS	BRKCS instruction	–	–		Yes
Exception	–	TRAP	Invalid op code trap	–	003CH		No
Reset	–	RESET	RESET input	–	0000H		

* Default priority: Priority used when multiple maskable interrupts occur at the same time, with 0 for the highest priority and 6 for the lowest priority

5.3 Macro Service

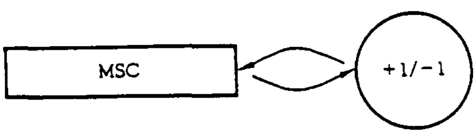
The uPD78350 has five types of macro services. Each macro service is explained below.

(1) Counter mode:EVTCNT

. Operation

(a) This mode increments or decrements the 8-bit macro service counter (MSC).

(b) When the MSC reaches 0, a vectored interrupt request occurs.



. Sample application

This mode can be used as the event counter or capture counter.

(2) Block transfer mode:BLKTRS

. Operation

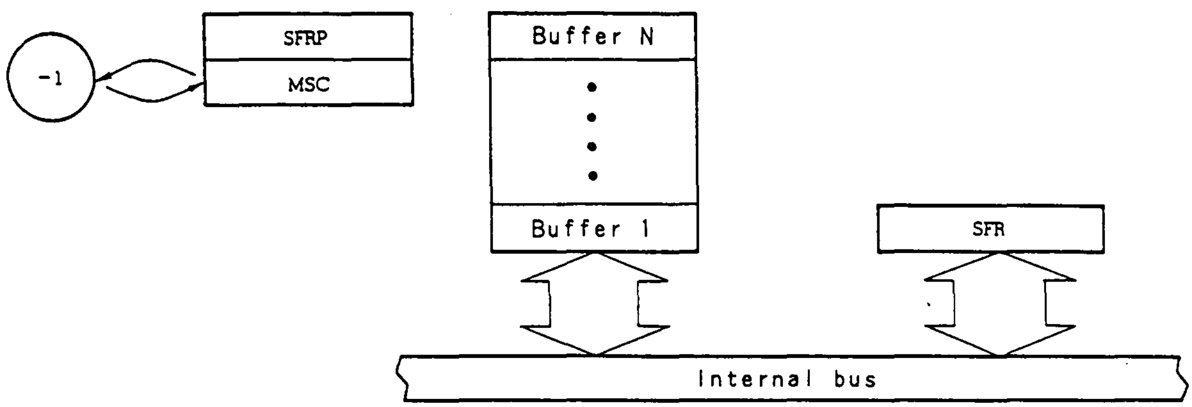
(a) This mode transfers a data block between the buffer and the SFR pointed to by the SFR pointer (SFRP).

(b) Either an SFR or buffer area can be specified as a transfer source or transfer destination. In addition, either the byte or word can be selected as the length of transfer data.

(c) The MSC is used to specify the number of data transfers (block size).

(d) Each time the macro service is executed, the MSC is automatically decremented by one.

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



. Sample application

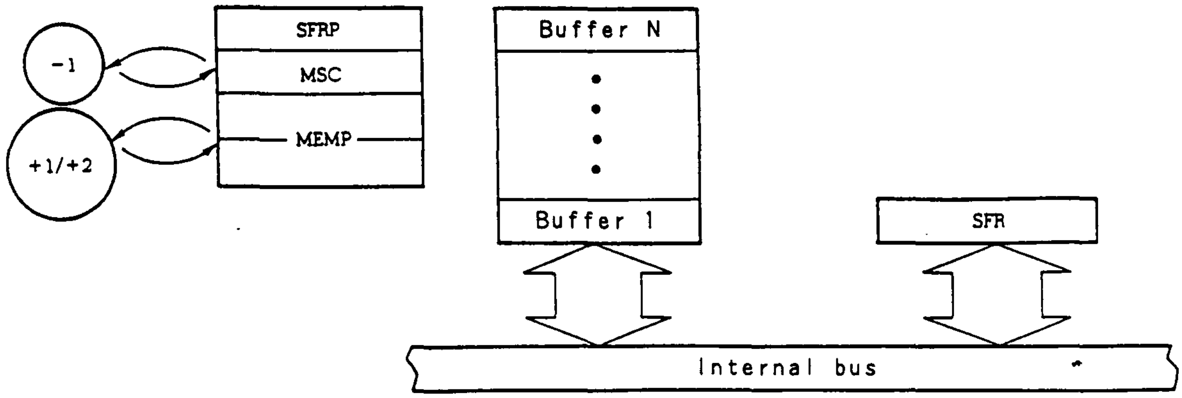
This mode can be used to read port data in response to an external interrupt request.

(3) Block transfer mode (with memory pointer):BLKTRS-P

. Operation

This mode is the block transfer mode with a memory pointer (MEMP) added. The additional buffer area for MEMP can be freely set in memory space.

Remark: Each time the macro service is executed, the MEMP is automatically incremented (by one for a byte-data transfer or by two for a word-data transfer).



. Sample application

Same as (2) above

(4) Data difference mode:DTADIF

. Operation

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

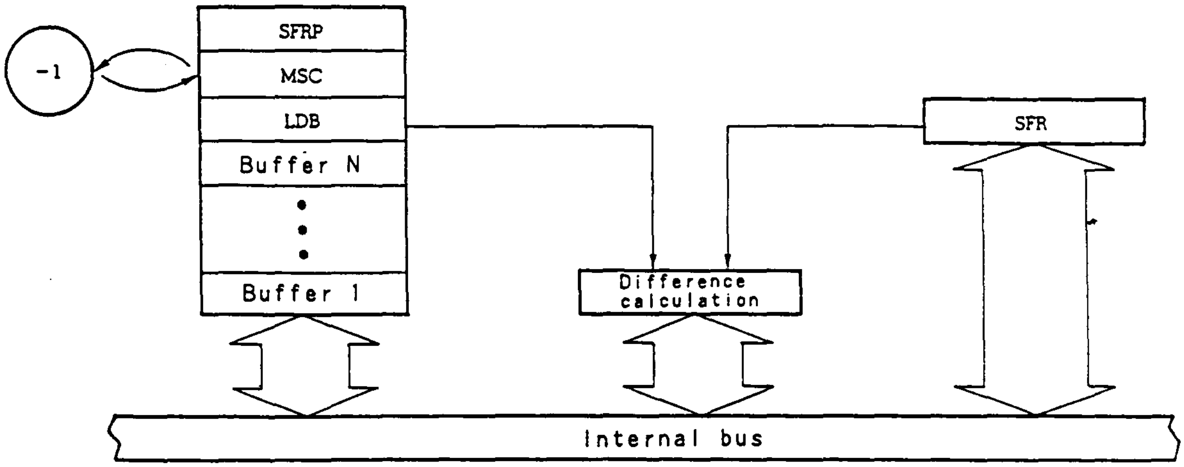
(b) The result of calculation is stored in a buffer area specified beforehand.

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

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

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

Remark: The difference can be calculated only for a 16-bit SFR.



. Sample application

This mode can be used to measure periods or pulse widths of the capture/timer unit.

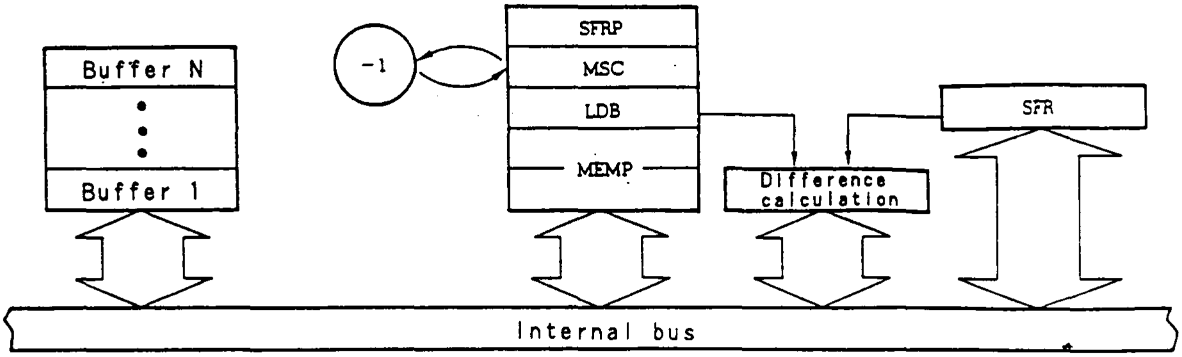
(5) Data difference mode (with memory pointer):DTADIF-P

. Operation

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

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

Note: MEMP - (MSC x 2) + 2



. Sample application

Same as (4) above

5.4 Context Switching

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

5.4.1 Context switching function based on an interrupt request

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

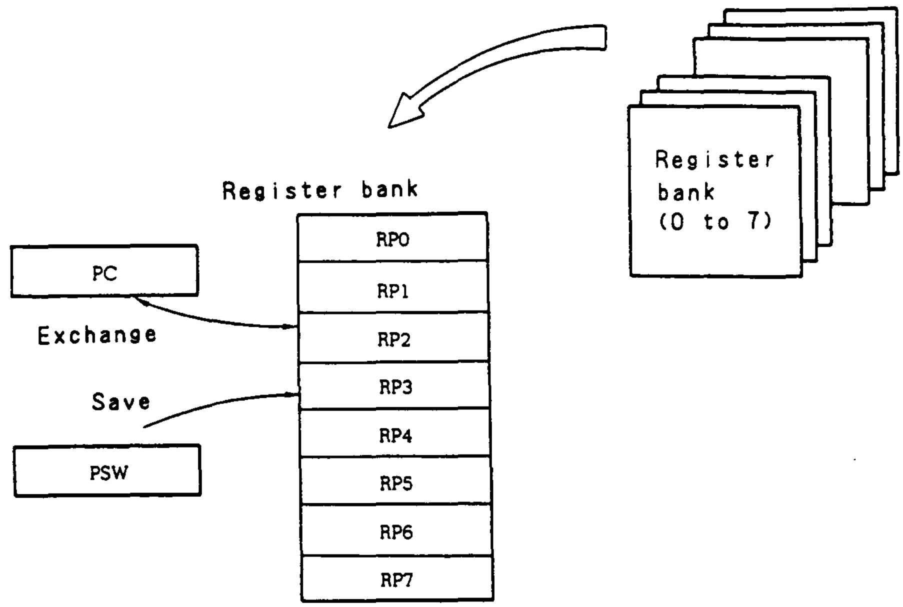
Context switching operation based on an interrupt request is performed as described below.

(1) When an interrupt request occurs, a register bank subject to context switching is specified from the contents of the lower three bits of the row address (even address) of the corresponding vector table.

(2) The vector address set beforehand in the register bank subject to context switching is transferred to PC, and the contents of PC and PSW present immediately before switching operation are saved in the register bank.

(3) A branch is made to the address pointed to by the newly set contents of PC.

Fig. 5-2 Context Switching Operation



5.4.2 Context switching function based on the BRKCS instruction

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

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

(1) An 8-bit register is specified in an operand of the BRKCS instruction. The contents of the register determine a register bank subject to context switching. (Only the low-order three bits of the eight bits are used.)

(2) The vector address set beforehand in the register bank subject to context switching is transferred to PC, and the contents of PC and PSW present immediately before switching operation are saved in the register bank at the same time.

(3) A branch is made to the address pointed to by the newly set contents of PC.

5.4.3 Return from context switching

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

Table 5-3 Return from Context Switching

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

6. EXTERNAL DEVICE EXPANSION FUNCTION

The uPD78350 does not contain ROM, but has extended built-in functions to connect external devices.

Connectable external devices are a general-purpose memory and I/O device.

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

Pin	Function
AD0-AD7	Multiplexed address/data bus
A8-A15	Data bus
$\overline{RD}$	Read strobe
$\overline{WR}$	Write strobe
ASTB	Address strobe
CLKOUT	System clock output

7. STANDBY FUNCTION

The uPD78350 has a standby function to reduce power consumption of the system. With the standby function, two modes are available:

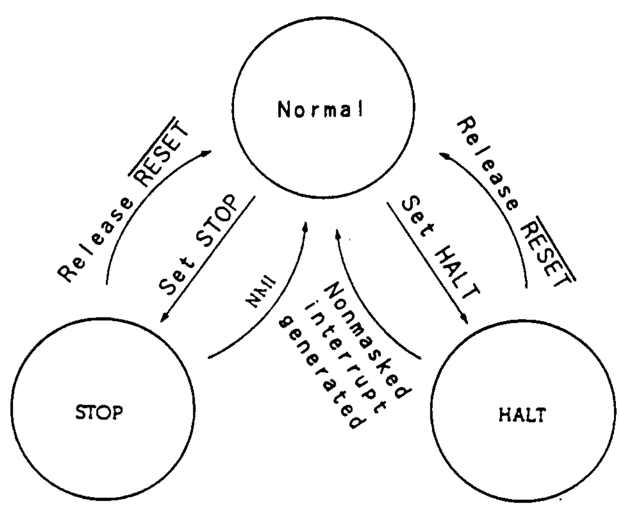
. HALT mode: In this mode, the CPU operation clock is stopped. Intermittent operation, when combined with the normal operation mode, can reduce overall system power consumption.

. STOP mode: In this mode, the oscillator is stopped to stop the entire system.

Since only leakage currents may flow in this mode, system power consumption can be minimized.

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

Fig. 7-1 Transition Diagram of the Standby Modes



8. RESET FUNCTION

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

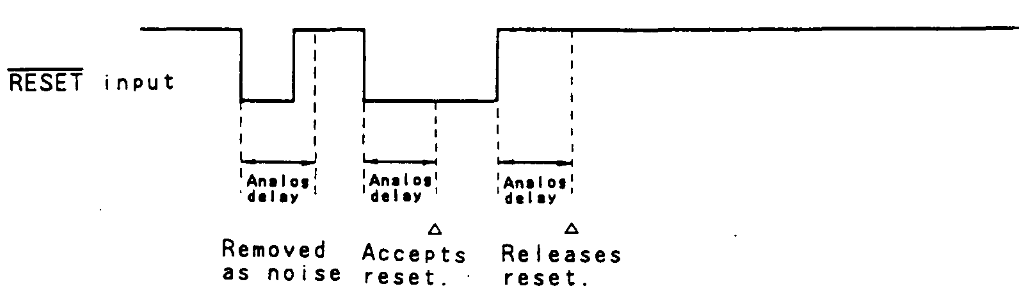
In particular, the number of cycles specified in the programmable wait control register must be changed as required.

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

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

2. When RAM is expanded externally, attach a pull-up resistor to the $\overline{\text{RD}}$ pin and $\overline{\text{WR}}$ pin. Otherwise, these pins may go into the high-impedance state, and the contents of the external RAM may be lost or the pins may be damaged.

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



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

Fig. 8-2 Reset Operation at Power-on

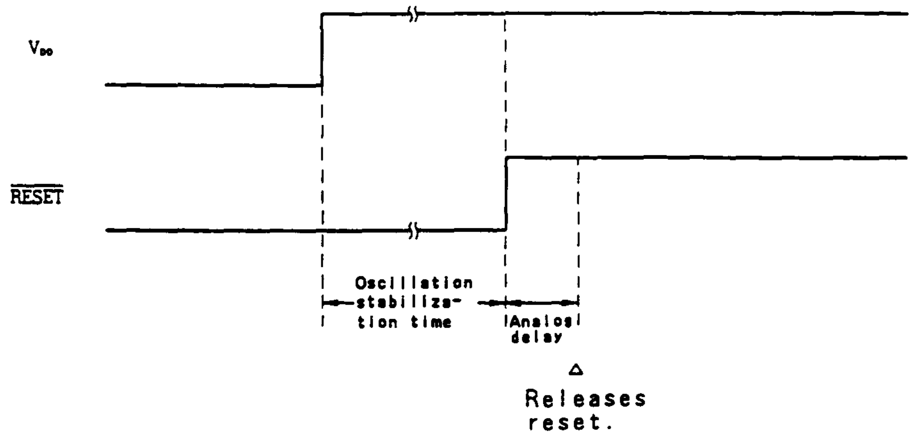


Table 8-1 Hardware Statuses after Reset

Hardware		Status after reset
Control registers	Program counter (PC)	The contents of a reset vector table (0000H, 0001H) are set.
	Stack pointer (SP)	Undefined
	Program status word (PSW)	0000H
	CPU control word (CCW)	00H
Internal RAM	Data memory	Undefined
	General registers (R0-R15)	
Ports	Output latches (P0, P1, P3)	Undefined
	Mode registers (PM0, PM1, PM3)	FFH
	Mode control register (PMC3)	00H
	Port read control register (PRDC)	00H
Capture/timer unit	Timers (TM0, TM1, TM2)	00H
	Timer control registers (TMC0, TMC1)	00H
	Capture registers (CT00, CT01)	Undefined
	Compare registers (CM10, CM20)	Undefined
PWM output function	PWM control register (PWMC)	00H
	PWM buffer registers (PWM0, PWM1)	Undefined
External expansion function	Memory expansion mode register (MM)	0xxx xxxxB
	Programmable wait control register (PWC)	C0AAH
Watchdog timer	Watchdog timer mode register (WDM)	00H

(to be continued)

Table 8-1 Hardware Statuses after Reset (Cont'd)

Hardware			Status after reset
Interrupt function	External interrupt mode registers (INTM0, INTM1)		00H
	Interrupt mode control register (IMC)		80H
	Interrupt mask flag registers	(MKL)	7FH
		(MK)	xx7FH
	Interrupt control registers (OVIC, PIC0, PIC1, CMIC10, CMIC20, PIC2, PIC3)		43H
	In-service priority register (ISPR)		00H
CPU control	Standby control register (STBC)		0000 x000H

9. INSTRUCTION SET

(1) Operand identifier and description

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

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

Table 9-1 Operand Identifier and Description

Identi-fier	Description
r r1 r2	R0, R1, R2, R3, R4, R5, R6, R7, R8, R9, R10, R11, R12, R13, R14, R15 R0, R1, R2, R3, R4, R5, R6, R7 C, B
rp rp1 rp2	RP0, RP1, RP2, RP3, RP4, RP5, RP6, RP7 RP0, RP1, RP2, RP3, RP4, RP5, RP6, RP7 DE, HL, VP, UP
sfr sfrp	Special function register abbreviation (See Table 2-1.) Special function register abbreviation (16-bit manipulation register. See T¬ble 2-1.)
post	RP0, RP1, RP2, RP3, RP4, RP5/PSW, RP6, RP7 (Can be coded more than once. However, RP5 can only be used in a PUSH or POP instruction and PSW can only be used in a PUSHU or POPU instruction.)

(to be continued)

Table 9-1 Operand Identifier and Description (Cont'd)

Identifier	Description
mem	[DE], [HL], [DE+], [HL+], [DE-], [HL-], [VP], [UP]: Register indirect mode [DE+A], [HL+A], [DE+B], [HL+B], [VP+DE], [VP+HL]: Based indexed mode [DE+byte], [HL+byte], [VP+byte], [UP+byte], [SP+byte]: Based mode word[A], word[B], word[DE], word[HL]: Indexed mode
saddr saddrp	FE20H-FF1FH Immediate data or label FE20H-FF1EH Immediate data (bit 0 = 0, however) or label (for 16-bit manipulation)
$addr16 !addr16 addr11 addr5	0000H-FDFFH Immediate data or label: Relative addressing 0000H-FDFFH Immediate data or label: Immediate addressing (Data at an address up to FFFFH can be coded in an MOV instruction. Data at an address from FE00H to FEFFH can be coded in an MOVTBLW instruction.) 800H-FFFH Immediate data or label 40H-7EH Immediate data (bit 0 = 0, however)(*) or label
word byte bit n	16-bit immediate data or label 8-bit immediate data or label 3-bit immediate data or label 3-bit immediate data (0 to 7)

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

Remarks 1. The same register name can be specified in rp and rp1, but different codes are generated.

2. Functional names (X, A, C, B, E, D, L, H, AX, BC, DE, HL, VP, and UP) can be specified in r, r1, rp, rp1, and post, as well as absolute names (R0 to R15 and RP0 to RP7).

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

(2) Legend

A:	A register; 8-bit accumulator
X:	X register
B:	B register
C:	C register
D:	D register
E:	E register
H:	H register
L:	L register
R0-R15	Register 0 to register 15 (absolute name)
AX:	Register pair (AX); 16 bit accumulator
BC:	Register pair (BC)
DE:	Register pair (DE)
HL:	Register pair (HL)
RP0-RP7:	Register pair 0 to register pair 7 (absolute name)
PC:	Program counter
SP:	Stack pointer
UP:	User stack pointer
PSW:	Program status word
CY:	Carry flag
AC:	Auxiliary carry flag
Z:	Zero flag
P/V:	Parity/overflow flag
S:	Sign flag
TPF:	Table position flag
RBS:	Register bank selecting flag
RSS:	Register set selecting flag
IE:	Interrupt enable flag
STBC:	Standby control register
WDM:	Watchdog timer mode register
jdisp8:	Signed 8-bit data (displacement value: -128 to +127)

( ):	Contents at an address enclosed in parentheses or at an address indicated in a register indicated in parentheses. (+) and (-) indicate that an address or the contents of a register indicated in parentheses are incremented and decremented by one after execution of the instruction, respectively.
(( )):	Contents at an address indicated by the contents at an address indicated in parentheses (( )).
xxH:	Hexadecimal number
x_H, x_L:	High-order 8 bits and low-order 8 bits of 16-bit register

(3) Notational symbols in flag operation field

Table 9-2 Notational Symbols in Flag Operation Field

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

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

(to be continued)

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

(Cont'd)

Instruction set	Mnemonic	Operand	Byte	Operation	Flag S Z AC P/V CY
8-bit data transfer	XCH	A,r1	1	A←→r1	
		r,r1	2	r←→r1	
		A,mem	2-4	A←→(mem)	
		A,saddr	2	A←→(saddr)	
		A,sfr	3	A←→sfr	
		A,[saddrp]	2	A←→((saddrp))	
		saddr,saddr	3	(saddr)←→(saddr)	
16-bit data transfer	MOVW	rp1,#word	3	rp1←word	
		saddrp,#word	4	(saddrp)←word	
		sfrp,#word	4	sfrp←word	
		rp,rp1	2	rp←rp1	
		AX,saddrp	2	AX←(saddrp)	
		saddrp,AX	2	(saddrp)←AX	
		saddrp,saddrp	3	(saddrp)←(saddrp)	
		AX,sfrp	2	AX←sfrp	
		sfrp,AX	2	sfrp←AX	
		rp1,!addr16	4	rp1←(addr16)	
		!addr16,rp1	4	(addr16)←rp1	
		AX,mem	2-4	AX←(mem)	
		mem,AX	2-4	(mem)←AX	
	XCHW	AX,saddrp	2	AX←→(saddrp)	
		AX,sfrp	3	AX←→sfrp	
		saddrp,saddrp	3	(saddrp)←→(saddrp)	
		rp,rp1	2	rp←→rp1	
		AX,mem	2-4	AX←→(mem)	

(to be continued)

(Cont'd)

Instruction set	Mnemonic	Operand	Byte	Operation	Flag				
					S	Z	AC	P/V	CY
8-bit arithmetic/logical operation	ADD	A,#byte	2	A,CY←A+byte	x	x	x	V	x
		saddr,#byte	3	(saddr),CY←(saddr)+byte	x	x	x	V	x
		sfr,#byte	4	sfr,CY←sfr+byte	x	x	x	V	x
		r,r1	2	r,CY←r+r1	x	x	x	V	x
		A,saddr	2	A,CY←A+(saddr)	x	x	x	V	x
		A,sfr	3	A,CY←A+sfr	x	x	x	V	x
		saddr,saddr	3	(saddr),CY←(saddr)+(saddr)	x	x	x	V	x
		A,mem	2-4	A,CY←A+(mem)	x	x	x	V	x
		mem,A	2-4	(mem),CY←(mem)+A	x	x	x	V	x
	ADDC	A,#byte	2	A,CY←A+byte+CY	x	x	x	V	x
		saddr,#byte	3	(saddr),CY←(saddr)+byte+CY	x	x	x	V	x
		sfr,#byte	4	sfr,CY←sfr+byte+CY	x	x	x	V	x
		r,r1	2	r,CY←r+r1+CY	x	x	x	V	x
		A,saddr	2	A,CY←A+(saddr)+CY	x	x	x	V	x
		A,sfr	3	A,CY←A+sfr+CY	x	x	x	V	x
		saddr,saddr	3	(saddr), CY←(saddr)+(saddr)+CY	x	x	x	V	x
		A,mem	2-4	A,CY←A+(mem)+CY	x	x	x	V	x
		mem,A	2-4	(mem),CY←(mem)+A+CY	x	x	x	V	x

(to be continued)

(Cont'd)

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

(to be continued)

(Cont'd)

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

(to be cont

(Cont'd)

Instruction set	Mnemonic	Operand	Byte	Operation	Flag				
					S	Z	AC	P/V	CY
16-bit arithmetic/logical operation	ADDW	AX,#word	3	AX,CY←AX+word	x	x	x	V	x
		saddrp,#word	4	(saddrp),CY←(saddrp)+word	x	x	x	V	x
		sfrp,#word	5	sfrp,CY←sfrp+word	x	x	x	V	x
		rp,rp1	2	rp,CY←rp+rp1	x	x	x	V	x
		AX,saddrp	2	AX,CY←AX+(saddrp)	x	x	x	V	x
		AX,sfrp	3	AX,CY←AX+sfrp	x	x	x	V	x
		saddrp,saddrp	3	(saddrp), CY←(saddrp)+(saddrp)	x	x	x	V	x
	SUBW	AX,#word	3	AX,CY←AX-word	x	x	x	V	x
		saddrp,#word	4	(saddrp),CY←(saddrp)-word	x	x	x	V	x
		sfrp,#word	5	sfrp,CY←sfrp-word	x	x	x	V	x
		rp,rp1	2	rp,CY←rp-rp1	x	x	x	V	x
		AX,saddrp	2	AX,CY←AX-(saddrp)	x	x	x	V	x
		AX,sfrp	3	AX,CY←AX-sfrp	x	x	x	V	x
		saddrp,saddrp	3	(saddrp), CY←(saddrp)-(saddrp)	x	x	x	V	x
	CMPW	AX,#word	3	AX-word	x	x	x	V	x
		saddrp,#word	4	(saddrp)-word	x	x	x	V	x
		sfrp,#word	5	sfrp-word	x	x	x	V	x
		rp,rp1	2	rp-rp1	x	x	x	V	x
		AX,saddrp	2	AX-(saddrp)	x	x	x	V	x
		AX,sfrp	3	AX-sfrp	x	x	x	V	x
		saddrp,saddrp	3	(saddrp)-(saddrp)	x	x	x	V	x

(to be continued)

(Cont'd)

Instruction set	Mnemonic	Operand	Byte	Operation	Flag S	Z	AC	P/V	CY
Multiply/divide operation	MULU	r1	2	AX←AXxr1					
	DIVUW	r1	2	AX (quotient), r1 (remainder)←AX÷r1					
	MULUW	rp1	2	AX (high-order 16 bits), rp1 (low-order 16 bits) ←AXxrp1					
	DIVUX	rp1	2	AXDE (quotient), rp1 (remainder)←AXDE÷rp1					
Signed multiply operation	MULW	rp1	2	AX (high-order 16 bits), rp1 (low-order 16 bits) ←AXxrp1					
Sum-of-products operation	MACW	n	3	AXDE←(B)x(C)+AXDE B←B+2, C←C+2, n←n-1 End if n=0 or P/V=1	x	x	x	V	x
Table shift	MOVTBLW	!addr16, n	4	(addr16+2)←(addr16), n←n-1 addr16←addr16-2, End if n=0					
Increment/decrement	INC	r1	1	r1←r1+1	x	x	x	V	
		saddr	2	(saddr)←(saddr)+1	x	x	x	V	
	DEC	r1	1	r1←r1-1	x	x	x	V	
		saddr	2	(saddr)←(saddr)-1	x	x	x	V	
	INCW	rp2	1	rp2←rp2+1					
		saddrp	3	(saddrp)←(saddrp)+1					
	DECW	rp2	1	rp2←rp2-1					
		saddrp	3	(saddrp)←(saddrp)-1					

(to be continued)

Remark: The addressing range of the table shift instruction is FE00H to FEFFH.

(Cont'd)

Instruction set	Mnemonic	Operand	Byte	Operation	Flag S	Z	AC	P/V	CY
Shift/rotate	ROR	r1,n	2	$(CY, r1_7 \leftarrow r1_0, r1_{m-1} \leftarrow r1_m)$ x n times				P	x
	ROL	r1,n	2	$(CY, r1_0 \leftarrow r1_7, r1_{m+1} \leftarrow r1_m)$ x n times				P	x
	RORC	r1,n	2	$(CY \leftarrow r1_0, r1_7 \leftarrow CY, r1_{m-1} \leftarrow r1_m)$ x n times				P	x
	ROLC	r1,n	2	$(CY \leftarrow r1_7, r1_0 \leftarrow CY, r1_{m+1} \leftarrow r1_m)$ x n times				P	x
	SHR	r1,n	2	$(CY \leftarrow r1_0, r1_7 \leftarrow 0, r1_{m-1} \leftarrow r1_m)$ x n times	x	x	0	P	x
	SHL	r1,n	2	$(CY \leftarrow r1_7, r1_0 \leftarrow 0, r1_{m+1} \leftarrow r1_m)$ x n times	x	x	0	P	x
	SHRW	rp1,n	2	$(CY \leftarrow rp1_0, rp1_{15} \leftarrow 0, rp1_{m-1} \leftarrow rp1_m)$ x n times	x	x	0	P	x
	SHLW	rp1,n	2	$(CY \leftarrow rp1_{15}, rp1_0 \leftarrow 0, rp1_{m+1} \leftarrow rp1_m)$ x n times	x	x	0	P	x
	ROR4	[rp1]	2	$A_{3-0} \leftarrow (rp1)_{3-0}$, $(rp1)_{7-4} \leftarrow A_{3-0}$, $(rp1)_{3-0} \leftarrow (rp1)_{7-4}$					
	ROL4	[rp1]	2	$A_{3-0} \leftarrow (rp1)_{7-4}$, $(rp1)_{3-0} \leftarrow A_{3-0}$, $(rp1)_{7-4} \leftarrow (rp1)_{3-0}$					
BCD correction	ADJBA		2	Decimal Adjust Accumulator	x	x	x	P	x
	ADJBS								
Data conversion	CVTBW		1	When $A_7=0$, $X \leftarrow A, A \leftarrow 00H$ When $A_7=1$, $X \leftarrow A, A \leftarrow FFH$					

(to be continued)

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

(Cont'd)

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

(to be continued)

(Cont'd)

Instruction set	Mnemonic	Operand	Byte	Operation	S	Z	AC	P/V	CY
Bit manipulation	OR1	CY,saddr.bit	3	$CY \leftarrow CY \vee (saddr.bit)$					x
		CY,/saddr.bit	3	$CY \leftarrow CY \vee (\overline{saddr.bit})$					x
		CY,sfr.bit	3	$CY \leftarrow CY \vee sfr.bit$					x
		CY,/sfr.bit	3	$CY \leftarrow CY \vee \overline{sfr.bit}$					x
		CY,A.bit	2	$CY \leftarrow CY \vee A.bit$					x
		CY,/A.bit	2	$CY \leftarrow CY \vee \overline{A.bit}$					x
		CY,X.bit	2	$CY \leftarrow CY \vee X.bit$					x
		CY,/X.bit	2	$CY \leftarrow CY \vee \overline{X.bit}$					x
		CY,PSWH.bit	2	$CY \leftarrow CY \vee PSW_H.bit$					x
		CY,/PSWH.bit	2	$CY \leftarrow CY \vee \overline{PSW_H.bit}$					x
		CY,PSWL.bit	2	$CY \leftarrow CY \vee PSW_L.bit$					x
		CY,/PSWL.bit	2	$CY \leftarrow CY \vee \overline{PSW_L.bit}$					x
	XOR1	CY,saddr.bit	3	$CY \leftarrow CY \veebar (saddr.bit)$					x
		CY,sfr.bit	3	$CY \leftarrow CY \veebar sfr.bit$					x
		CY,A.bit	2	$CY \leftarrow CY \veebar A.bit$					x
		CY,X.bit	2	$CY \leftarrow CY \veebar X.bit$					x
		CY,PSWH.bit	2	$CY \leftarrow CY \veebar PSW_H.bit$					x
		CY,PSWL.bit	2	$CY \leftarrow CY \veebar PSW_L.bit$					x
	SET1	saddr.bit	2	$(saddr.bit) \leftarrow 1$					
		sfr.bit	3	$sfr.bit \leftarrow 1$					
		A.bit	2	$A.bit \leftarrow 1$					
		X.bit	2	$X.bit \leftarrow 1$					
		PSWH.bit	2	$PSW_H.bit \leftarrow 1$					
		PSWL.bit	2	$PSW_L.bit \leftarrow 1$	x	x	x	x	x

(to be continued)

(Cont'd)

Instruction set	Mnemonic	Operand	Byte	Operation	Flag S Z AC P/V CY
Bit manipulation	CLR1	saddr.bit	2	(saddr.bit)←0	
		sfr.bit	3	sfr.bit←0	
		A.bit	2	A.bit←0	
		X.bit	2	X.bit←0	
		PSWH.bit	2	PSW_H.bit←0	
		PSWL.bit	2	PSW_L.bit←0	x x x x x
	NOT1	saddr.bit	3	(saddr.bit)←$\overline{\text{(saddr.bit)}}$	
		sfr.bit	3	sfr.bit←$\overline{\text{sfr.bit}}$	
		A.bit	2	A.bit←$\overline{\text{A.bit}}$	
		X.bit	2	X.bit←$\overline{\text{X.bit}}$	
		PSWH.bit	2	PSW_H.bit←$\overline{PSW_H\text{.bit}}$	
		PSWL.bit	2	PSW_L.bit←$\overline{PSW_L\text{.bit}}$	x x x x x
	SET1	CY	1	CY←1	1
	CLR1	CY	1	CY←0	0
	NOT1	CY	1	CY←$\overline{CY}$	x
Call/return	CALL	!addr16	3	(SP-1)←$(PC+3)_H$, (SP-2)←$(PC+3)_L$, PC←addr16,SP←SP-2	
	CALLF	!addr11	2	(SP-1)←$(PC+2)_H$, (SP-2)←$(PC+2)_L$, PC_{15-11}←00001, PC_{10-0}←addr11,SP←SP-2	
	CALLT	[addr5]	1	(SP-1)←$(PC+1)_H$, (SP-2)←$(PC+1)_L$, PC_H←(TPF,00000000,addr5+1), PC_L←(TPF,00000000,addr5), SP←SP-2	
	CALL	rp1	2	(SP-1)←$(PC+2)_H$, (SP-2)←$(PC+2)_L$, PC_H←$rp1_H$,PC_L←$rp1_L$, SP←SP-2	
		[rp1]	2	(SP-1)←$(PC+2)_H$, (SP-2)←$(PC+2)_L$, PC_H←(rp1+1),PC_L←(rp1), SP←SP-2	

(to be continued)

(Cont'd)

Instruction set	Mnemonic	Operand	Byte	Operation	Flag S	Z	AC	P/V	CY
Call/return	BRK		1	(SP-1)←PSW_H, (SP-2)←PSW_L (SP-3)←$(PC+1)_H$, (SP-4)←$(PC+1)_L$, PC_L←(003EH), PC_H←(003FH), SP←SP-4, IE←0					
	RET		1	PC_L←(SP), PC_H←(SP+1), SP←SP+2					
	RETB		1	PC_L←(SP), PC_H←(SP+1) PSW_L←(SP+2), PSW_H←(SP+3) SP←SP+4	R	R	R	R	R
	RETI		1	PC_L←(SP), PC_H←(SP+1) PSW_L←(SP+2), PSW_H←(SP+3) SP←SP+4	R	R	R	R	R
Stack manipulation	PUSH	sfrp	3	(SP-1)←sfr_H (SP-2)←sfr_L SP←SP-2					
		post	2	{(SP-1)←$post_H$, (SP-2)← $post_L$, SP←SP-2} x n times					
		PSW	1	(SP-1)←PSW_H, (SP-2)←PSW_L, SP←SP-2					
	PUSHU	post	2	{(UP-1)←$post_H$, (UP-2)← $post_L$, UP←UP-2} x n times					
	POP	sfrp	3	sfr_L←(SP) sfr_H←(SP+1) SP←SP+2					
		post	2	{$post_L$←(SP), $post_H$←(SP+1), SP←SP+2} x n times					
		PSW	1	PSW_L←(SP), PSW_H←(SP+1), SP←SP+2	R	R	R	R	R
	POPU	post	2	{$post_L$←(UP), $post_H$←(UP+1), UP←UP+2} x n times					
	MOVW	SP, #word	4	SP←word					
		SP, AX	2	SP←AX					
		AX, SP	2	AX←SP					
	INCW	SP	2	SP←SP+1					
	DECW	SP	2	SP←SP-1					

(to be continued)

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

(Cont'd)

Instruction set	Mnemonic	Operand	Byte	Operation	Flag S	Flag Z	Flag AC	Flag P/V	Flag CY
Special	CHKL	sfr	3	(Pin level) ⊻ (Signal level before output buffer)	x	x		P	
	CHKLA	sfr	3	A←{(Pin level) ⊻ (Signal level before output buffer)}	x	x		P	
Unconditional branch	BR	!addr16	3	PC←addr16					
		rp1	2	PC_H←$rp1_H$, PC_L←$rp1_L$					
		[rp1]	2	PC_H←(rp1+1), PC_L←(rp1)					
		$addr16	2	PC←PC+2+jdisp8					
Conditional branch	BC BL	$addr16	2	PC←PC+2+jdisp8 if CY=1					
	BNC BNL	$addr16	2	PC←PC+2+jdisp8 if CY=0					
	BZ BE	$addr16	2	PC←PC+2+jdisp8 if Z=1					
	BNZ BNE	$addr16	2	PC←PC+2+jdisp8 if Z=0					
	BV BPE	$addr16	2	PC←PC+2+jdisp8 if P/V=1					
	BNV BPO	$addr16	2	PC←PC+2+jdisp8 if P/V=0					
	BN	$addr16	2	PC←PC+2+jdisp8 if S=1					
	BP	$addr16	2	PC←PC+2+jdisp8 if S=0					
	BGT	$addr16	3	PC←PC+3+jdisp8 if (P/V ⊻ S) ∨ Z=0					
	BGE	$addr16	3	PC←PC+3+jdisp8 if P/V ⊻ S=0					
	BLT	$addr16	3	PC←PC+3+jdisp8 if P/V ⊻ S=1					
	BLE	$addr16	3	PC←PC+3+jdisp8 if (P/V ⊻ S) ∨ Z=1					
	BH	$addr16	3	PC←PC+3+jdisp8 if Z ∨ CY=0					
	BNH	$addr16	3	PC←PC+3+jdisp8 if Z ∨ CY=1					

(to be continued)

(Cont'd)

Instruction set	Mnemonic	Operand	Byte	Operation	Flag S	Z	AC	P/V	CY
Conditional branch	BT	saddr.bit, $addr16	3	PC←PC+3+jdisp8 if (saddr.bit)=1					
		sfr.bit, $addr16	4	PC←PC+4+jdisp8 if sfr.bit=1					
		A.bit,$addr16	3	PC←PC+3+jdisp8 if A.bit=1					
		X.bit,$addr16	3	PC←PC+3+jdisp8 if X.bit=1					
		PSWH.bit, $addr16	3	PC←PC+3+jdisp8 if PSW_H.bit=1					
		PSWL.bit, $addr16	3	PC←PC+3+jdisp8 if PSW_L.bit=1					
	BF	saddr.bit, $addr16	4	PC←PC+4+jdisp8 if (saddr.bit)=0					
		sfr.bit, $addr16	4	PC←PC+4+jdisp8 if sfr.bit=0					
		A.bit,$addr16	3	PC←PC+3+jdisp8 if A.bit=0					
		X.bit,$addr16	3	PC←PC+3+jdisp8 if X.bit=0					
		PSWH.bit, $addr16	3	PC←PC+3+jdisp8 if PSW_H.bit=0					
		PSWL.bit, $addr16	3	PC←PC+3+jdisp8 if PSW_L.bit=0					
	BTCLR	saddr.bit, $addr16	4	PC←PC+4+jdisp8 if (saddr.bit)=1 then reset (saddr.bit)					
		sfr.bit, $addr16	4	PC←PC+4+jdisp8 if sfr.bit=1 then reset sfr.bit					
		A.bit,$addr16	3	PC←PC+3+jdisp8 if A.bit=1 then reset A.bit					
		X.bit,$addr16	3	PC←PC+3+jdisp8 if X.bit=1 then reset X.bit					
		PSWH.bit, $addr16	3	PC←PC+3+jdisp8 if PSW_H.bit=1 then reset PSW_H.bit					
		PSWL.bit, $addr16	3	PC←PC+3+jdisp8 if PSW_L.bit=1 then reset PSW_L.bit	x	x	x	x	x

(to be continued)

(Cont'd)

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

(to be continued)

(Cont'd)

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

(to be continued)

(Cont'd)

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

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

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

10. ELECTRICAL CHARACTERISTICS

Absolute maximum ratings (T_a = 25°C)

Item	Symbol	Condition	Rating	Unit
Supply voltage	V_{DD}		-0.5 to +7.0	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-level output current	I_{OL}	Each pin	4.0	mA
		Total of all output pins	100	mA
High-level output current	I_{OH}	Each pin	-1.0	mA
		Total of all output pins	-20	mA
Operating temperature	T_{opt}		-10 to +70	°C
Storage temperature	T_{stg}		-65 to +150	°C

Recommended operating conditions

Oscillator frequency	T_a	V_{DD}
8 MHz < f_{XX} ≦ 25 MHz	-10 to +70°C	+5.0 V ±10%

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

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

DC characteristics

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

Item	Symbol	Condition		Min.	Typ.	Max.	Unit
Low-level input voltage	V_{IL}			0		0.8	V
High-level input voltage	V_{IH1}	(*1)		2.2			V
	V_{IH2}	(*2)		0.8 V_{DD}			
Low-level output voltage	V_{OL}	I_{OL} = 2.0 mA				0.45	V
High-level output voltage	V_{OH}	I_{OH} = -400 uA		V_{DD} - 1.0			V
Input leakage current	I_{LI}	0 V ≦ V_I ≦ V_{DD}				±10	uA
Output leakage current	I_{LO}	0 V ≦ V_O ≦ V_{DD}				±10	uA
V_{DD} supply current	I_{DD1}	Operation mode			50	90	mA
	I_{DD2}	HALT mode			25	40	mA
Data retention voltage	V_{DDDR}	STOP mode		2.5			V
Data retention current	I_{DDDR}	STOP mode	V_{DDDR} = 2.5 V		2	10	uA
			V_{DDDR} = 5.0 V ±10%		10	50	uA

*1 Other than pins in *2.

*2 $\overline{\text{RESET}}$, X1, X2, P20/NMI, P21/INTP0, P22/INTP1, P23/INTP2, P24/INTP3, P25/TI.

AC characteristics (T_a = -10 to +70°C, V_{DD} = +5.0 V ±10%,
V_{SS} = 0 V, C_L = 100 pF, f_X = 25 MHz)

Read/write operation (when the general memory is connected)

Item	Symbol	Condition	Min.	Max.	Unit
System clock cycle time	t_{CYK}	C_L = 50 pF	80	250	ns
Address setup time (to ASTB↓)	t_{SAST}		16		ns
Address hold time (to ASTB↓)	t_{HSTA}		26		ns
$\overline{RD}$↓ → address float time	t_{FRA}			24	ns
Address → data input time	t_{DAID}			144	ns
$\overline{RD}$↓ → data input time	t_{DRID}			76	ns
ASTB↓ → $\overline{RD}$↓ delay time	t_{DSTR}		24		ns
Data hold time (to $\overline{RD}$↑)	t_{HRID}		0		ns
$\overline{RD}$↑ → address active time	t_{DRA}		26		ns
$\overline{RD}$ low-level width	t_{WRL}		90		ns
ASTB high-level width	t_{WSTH}		23		ns
$\overline{WR}$↓ → data output time	t_{DWOD}			29	ns
ASTB↓ → $\overline{WR}$↓ delay time	t_{DSTW}		24		ns
Data setup time (to $\overline{WR}$↑)	t_{SODW}		75		ns
Data hold time (to $\overline{WR}$↑)	t_{HWOD}		8		ns
$\overline{WR}$ low-level width	t_{WWL}		90		ns
$\overline{WAIT}$ setup time (to an address)	t_{SAWT}			65	ns
$\overline{WAIT}$ hold time (to an address)	t_{HAWT}		110		ns

t_{CYK}-dependent bus timing definition

Item	Expression	Min./Max.	Unit
t_{SAST}	(0.5 + a) T - 24	Min.	ns
t_{HSTA}	0.5T - 14	Min.	ns
t_{WSTH}	(0.5 + a) T-17	Min.	ns
t_{DSTR}	0.5T - 16	Min.	ns
t_{WRL}	(1.5 + n) T - 30	Min.	ns
t_{DAID}	(2.5 + a + n) T - 56	Max.	ns
t_{DRID}	(1.5 + n) T - 44	Max.	ns
t_{DRA}	0.5T - 14	Min.	ns
t_{DSTW}	0.5T - 16	Min.	ns
t_{WWL}	(1.5 + n) T - 30	Min.	ns
t_{DWOD}	0.5T - 11	Max.	ns
t_{SODW}	(1 + n) T - 5	Min.	ns
t_{SAWT}	(a + n) T - 15	Max.	ns
t_{HAWT}	(0.5 + a + n) T - 10	Min.	ns

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

2. When an address wait is inserted, the value of a is 1. Otherwise, it is 0.

3. The number n represents the number of wait cycles specified by the external wait pin ($\overline{WAIT}$) or PWC register.

4. Only the bus timing items listed above are dependent on t_{CYK}.

Other operations

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

Item	Symbol	Condition	Min.	Max.	Unit
NMI high/low level width	t_{WNIH} t_{WNIL}		2.5		us
INTP0 high/low level width	t_{WI0H} t_{WI0L}		640		ns
INTP1 high/low level width	t_{WI1H} t_{WI1L}		640		ns
INTP2 high/low level width	t_{WI2H} t_{WI2L}		640		ns
INTP3 high/low level width	t_{WI3H} t_{WI3L}		640		ns
$\overline{\text{RESET}}$ high/low level width	t_{WRSH} t_{WRSL}		2.5		us
TI high/low level width	t_{WTIH} t_{WTIL}		640		ns

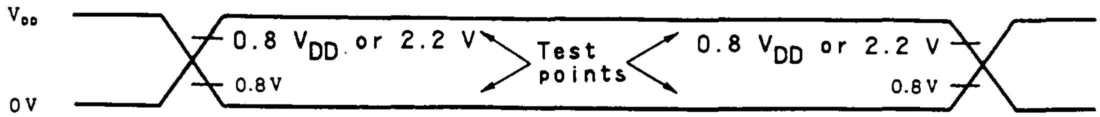
Other t_{CYK}-dependent operations

Item	Expression	Min./Max.	Unit
t_{WI0H}	8T	Min.	ns
t_{WI0L}	8T	Min.	ns
t_{WI1H}	8T	Min.	ns
t_{WI1L}	8T	Min.	ns
t_{WI2H}	8T	Min.	ns
t_{WI2L}	8T	Min.	ns
t_{WI3H}	8T	Min.	ns
t_{WI3L}	8T	Min.	ns
t_{WTIH}	8T	Min.	ns
t_{WTIL}	8T	Min.	ns

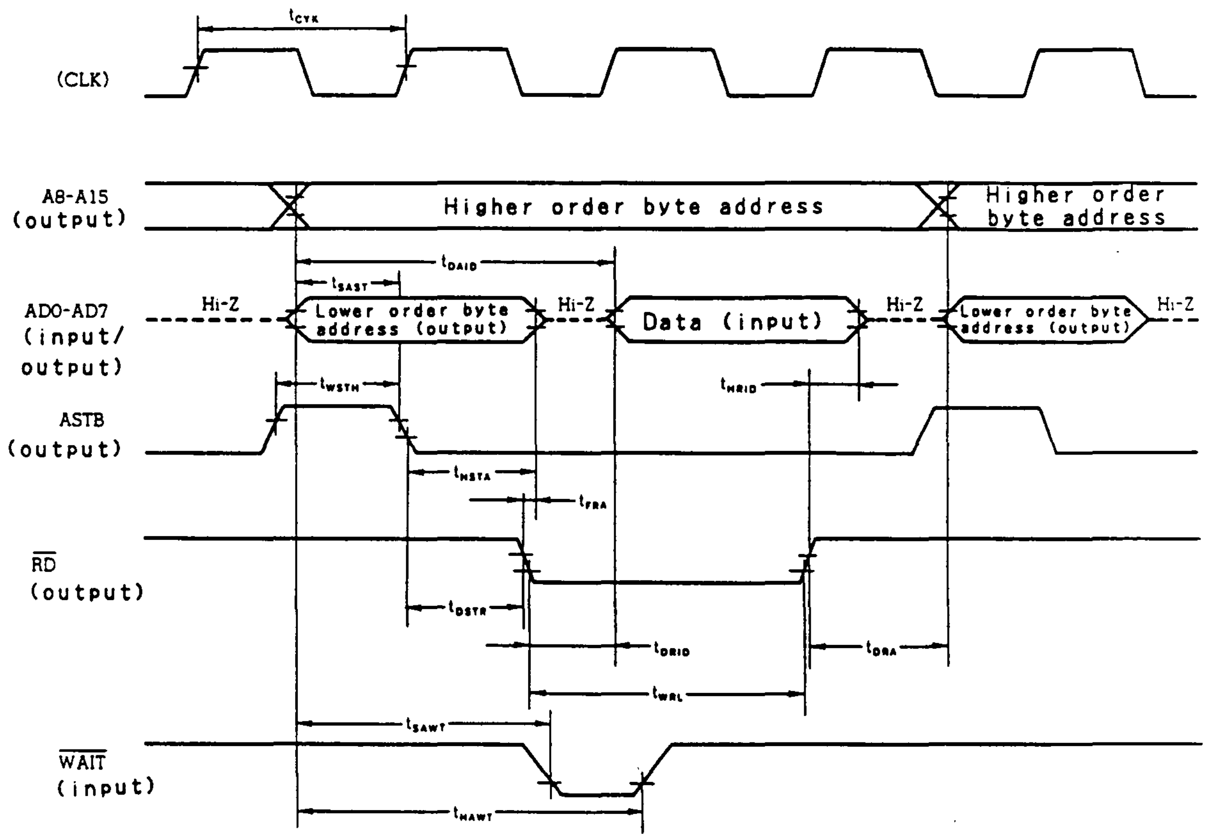
Remarks 1. $T = t_{CYK} = 1/f_{CLK}$ (f_{CLK} is the internal system clock frequency.)

2. Besides the bus timing items, only the items listed above are dependent on t_{CYK}.

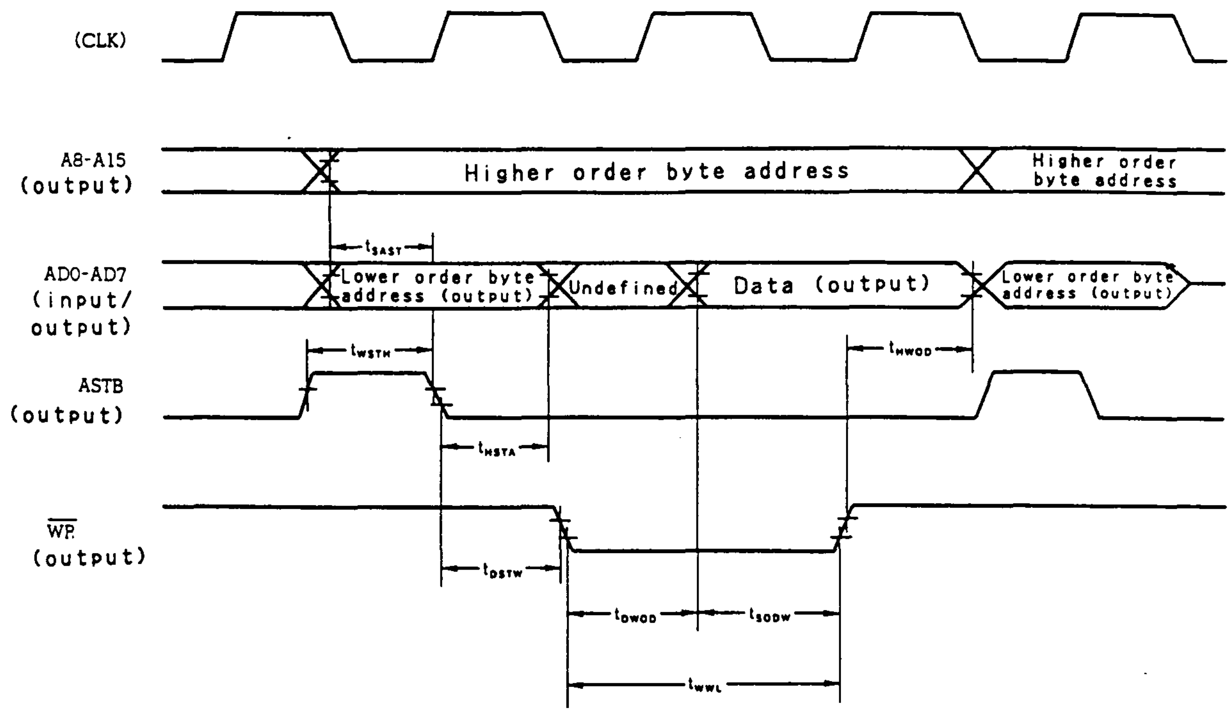
AC timing test points:



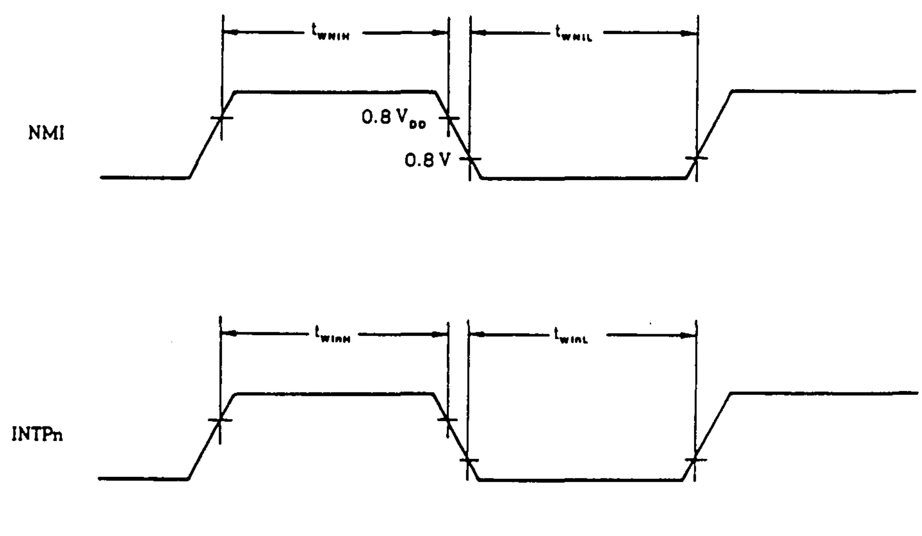
Read operation:



Write operation:

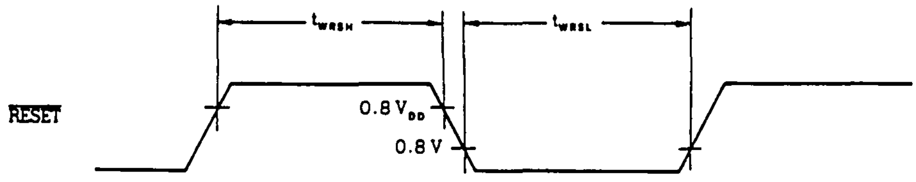


Interrupt input timing:

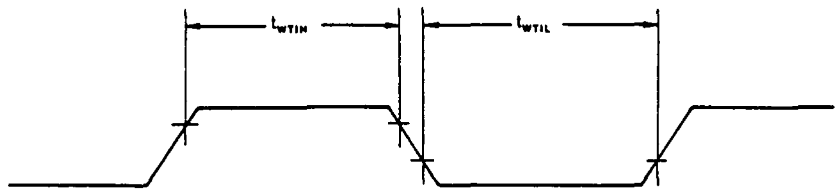


Remark: n = 0 to 3

Reset input timing:

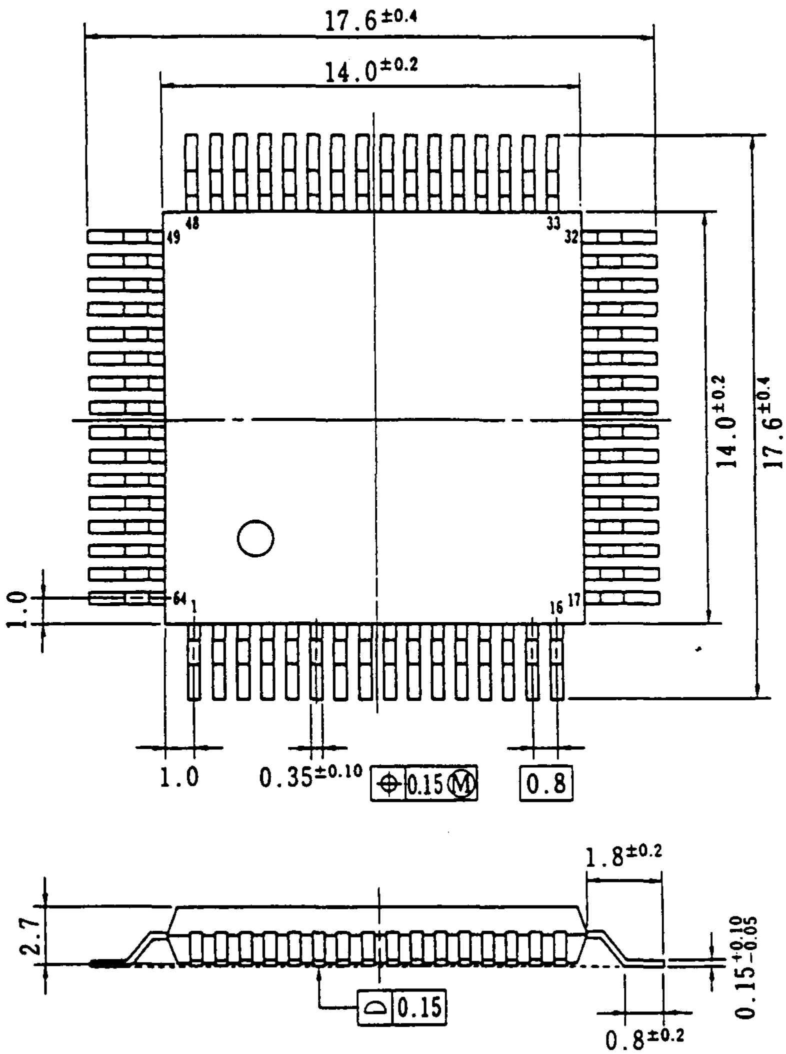


TI pin input timing:

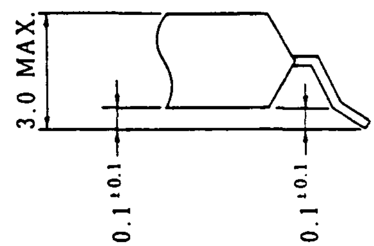


11. PACKAGE DIMENSION

64-pin plastic QFP (14 x 14) (Units: mm)



Detail of pin lead



P64GC-80-3BE

12. RECOMMENDED SOLDERING CONDITIONS

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

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

Table 12-1 Recommended Soldering Conditions

Product	Package	Symbol
uPD78350GC-3BE	64-pin plastic QFP	IR30-107
		VP15-107
		Partial heating method

Table 12-2 Soldering Conditions

Symbol	Soldering process	Soldering conditions
IR30-107	Infrared ray reflow	Peak package's surface temperature: 230°C or below Reflow time: 30 seconds or below (210°C or higher) Number of reflow processes: 1 Exposure limit(*): 7 days (10-hour pre-baking is required at 125°C afterwards.)
VP15-107	VPS	Peak package's surface temperature: 215°C or below Reflow time: 40 seconds or below (200°C or higher) Number of reflow processes: 1 Exposure limit(*): 7 days (10-hour pre-baking is required at 125°C afterwards.)
Partial heating method	Terminal to be heated	Terminal temperature: 300°C or below Flow time: 10 seconds or below

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

Caution: Do not apply two or more different soldering methods to one chip (except for partial heating method for terminal sections).

Remark: For details of the recommended soldering conditions for surface mount type products, refer to our document "SMT MANUAL" (IEI-1207).

APPENDIX A DIFFERENCE BETWEEN uPD78350 AND uPD78322

Item		uPD78350	uPD78322	uPD78320
Number of basic instructions		113	111	
Minimum instruction execution time		160 ns (when internal clock operates at 12.5 MHz or when external clock operates at 25.0 MHz)	250 ns (when internal clock operates at 8 MHz or when external clock operates at 16 MHz)	
Internal memory	ROM	–	16384 x 8 bits	–
	RAM	640 x 8 bits		
Memory space		64K bytes		
I/O line	Input	6	24 (analog input: 8)	
	Output	–		
	I/O	24	39	21
Real-time pulse unit (Capture/timer unit)		. 16-bit timers/counters: 3 . 16-bit capture registers: 2 . 16-bit compare registers: 2	. 18/16-bit free running timer: 1 . 16-bit timer/event counter: 1 . 16-bit compare registers: 6 . 18-bit capture registers: 4 . 18-bit capture/compare registers: 2 . Real-time output ports: 8	
Serial interface		–	. Serial interface with dedicated baud rate generator: 2 channels UART: 1 channel Clock synchronous serial interface/SBI: 1 channel	
A/D converter		–	10-bit resolution, 8 inputs	

(to be continued)

(Cont'd)

Item \ Product	uPD78350	uPD78322	uPD78320
Interrupt	. External: 5, internal: 4 . 4-level programmable priority	. External: 8, internal: 13 . 3-level programmable priority	
	. Three processing modes: Vectored interrupt function, macro service function, and context switching function		
Instruction set	Sum-of-products instruction is added as compared with the uPD78320 and uPD78322.	Much more instructions are added as compared with the uPD78310 and uPD78312.	
Other specifications	. Watchdog timer: Provided . Standby function (STOP/HALT)		
	. External wait pin	–	
	-	. Can directly be connected to the access manager (TAM II: uPD71P301).	
Package	. 64-pin plastic QFP (14 x 14 mm)	. 68-pin PLCC . 74-pin plastic QFP (20 x 20 mm)	

APPENDIX B TOOLS

The following tools are provided for developing a system that uses the uPD78350:

Hardware	IE-78350-R(*)	In-circuit emulator for developing and debugging the application system. For debugging, connect the emulator to the host machine. Since object files can be transferred between them and symbolic debugging can be performed, it enables effective debugging results. IE-78350-R has two channels of RS-232-C serial interfaces so that it can be connected to the PROM programmer such as PG-1500. The IE-78350-R also has the Centronics interface so that files in the object/symbol/debugging environment can be downloaded at high speed.			
	(*) IE-78350-R-EM1	Emulation board for emulating the peripheral hardware such as the I/O port of the uPD78350.			
	EP-78240GC-R	Emulation probe for connecting the IE-78350-R to the user system. Use it with the IE-78350-R-EM1.			
Software	IE-78350-R control program (IE-controller)(*)	Host machine	OS	Media	Order code (part no.)
		PC-9800 series	MS-DOS™	3.5-inch 2HD	uS5A13IE78350
				5-inch 2HD	uS5A10IE78350
		IBM PC series	PC DOS™	5-inch 2HC	uS7B10IE78350
	78K/III series relocatable assembler	Host machine	OS	Media	Order code (part no.)
		PC-9800 series	MS-DOS	3.5-inch 2HD	uS5A13RA78K3
				5-inch 2HD	uS5A10RA78K3
		IBM PC series	PC DOS	5-inch 2D	uS7B11RA78K3

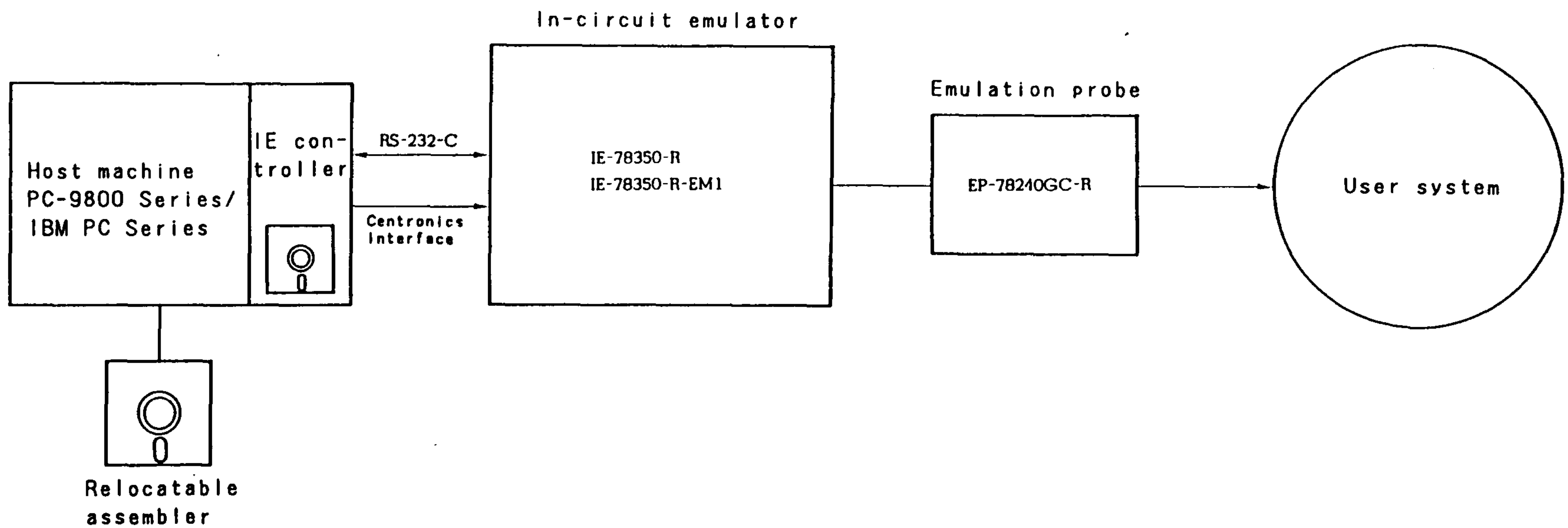
* Under development

Remark: Operations of each software are guaranteed only on the host machine and by the OS mentioned above.

The following evaluation tools are provided for evaluating the function of the uPD78350:

EB-78350-98 (applicable for PC-9800 series) or EB-78350-PC (applicable for IBM PC series)	When the evaluation tool is connected to the host machine (PC-9800 series or IBM PC series), the function of the uPD78350 can easily be evaluated. As the command system of these products conforms to that of the IE-78350-R, the migration can easily be made to the development of the application system with the IE-78350-R.

Caution: These products are not tools for the application system that uses the uPD78350.



Configuration of development tools

NEC

NEC Electronics Inc.

CORPORATE HEADQUARTERS
401 Ellis Street
P.O. Box 7241
Mountain View, CA 94039
TEL 415-960-6000
TLX 3715792

For literature, call toll-free 8 a.m. to 4 p.m. Pacific time:
1-800-632-3531

IC-2845

96

029692 V _ R