NEC

μPD78C10CW/G
μPD78C11CW/G

PRODUCT DESCRIPTION

NEC ELECTRONICS (EUROPE) GMBH

μPD78C10CW/G μPD78C11CW/G

PRODUCT DESCRIPTION

DESCRIPTION

The μPD78C11 is a single-chip, CMOS 8-bit microcomputer in which a 16-bit ALU, a ROM, a RAM, an A/D converter, a multi-function timer/event counter, and a serial interface are all integrated. Moreover, a 60k-byte external expansion memory (ROM/RAM) can be connected.

The μPD78C10 is identical to the μPD78C11 minus the ROM and direct addressing to an external memory up to 64k bytes is possible.

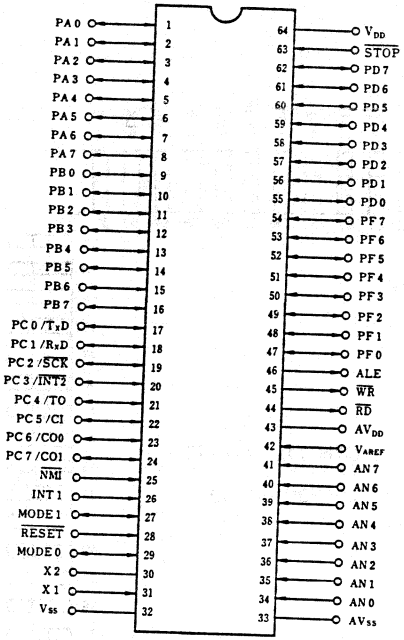
Since the μPD78C11 and μPD78C10 have CMOS construction, their operations are performed with low power consumption. By using the standby function, they perform functions such as data retention with lower power consumption.

FEATURES

° One-chip microcomputer (μCOM-87AD)
° 159 instructions
  - uCOM-87 upward compatible
  - Multiplication and division instructions
  - 16-bit arithmetic operation instructions
° Instruction cycle: 1μs/12MHz
° Internal ROM : 4096W x 8 (μPD78C11)
° Internal RAM : 256W x 8
° Direct addressing to an external memory (ROM/RAM) up to 64K bytes
° Highly accurate 8-bit A/D converter
  - Eight analog inputs
° General-purpose serial interface
  - Asynchronous, synchronous, and I/O interface modes
° Multifunction 16-bit timer/event counter
° Two 8-bit timers
° I/O lines : 44 (μPD78C11)
  : 32 (μPD78C10)
° Interrupt functions : Three external, eight internal
  - Nonmaskable interrupt: 1
  - Maskable interrupts : 10
° Zero-cross detection function (two inputs)
° Standby functions
  - HALT mode
  - Hardware/software STOP mode
° CMOS
° Single power supply
° 64-pin plastic shrink DIP (750 mil)
  (μPD78C11CW/μPD78C10CW)
  64-pin plastic flat package (μPD78C11G)
  64-pin plastic QUIP (μPD78C11G/μPD78C10G)

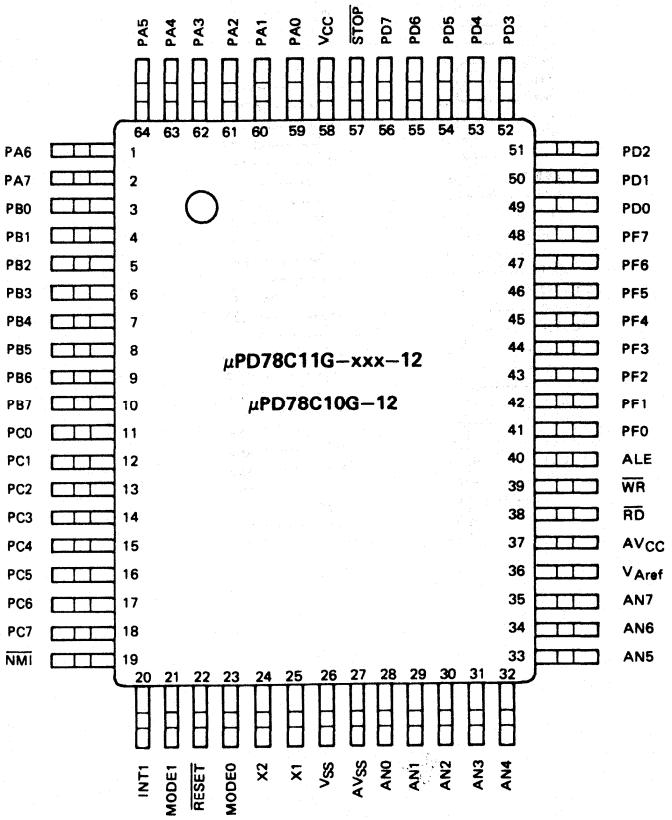
Pin Configuration (Top View)

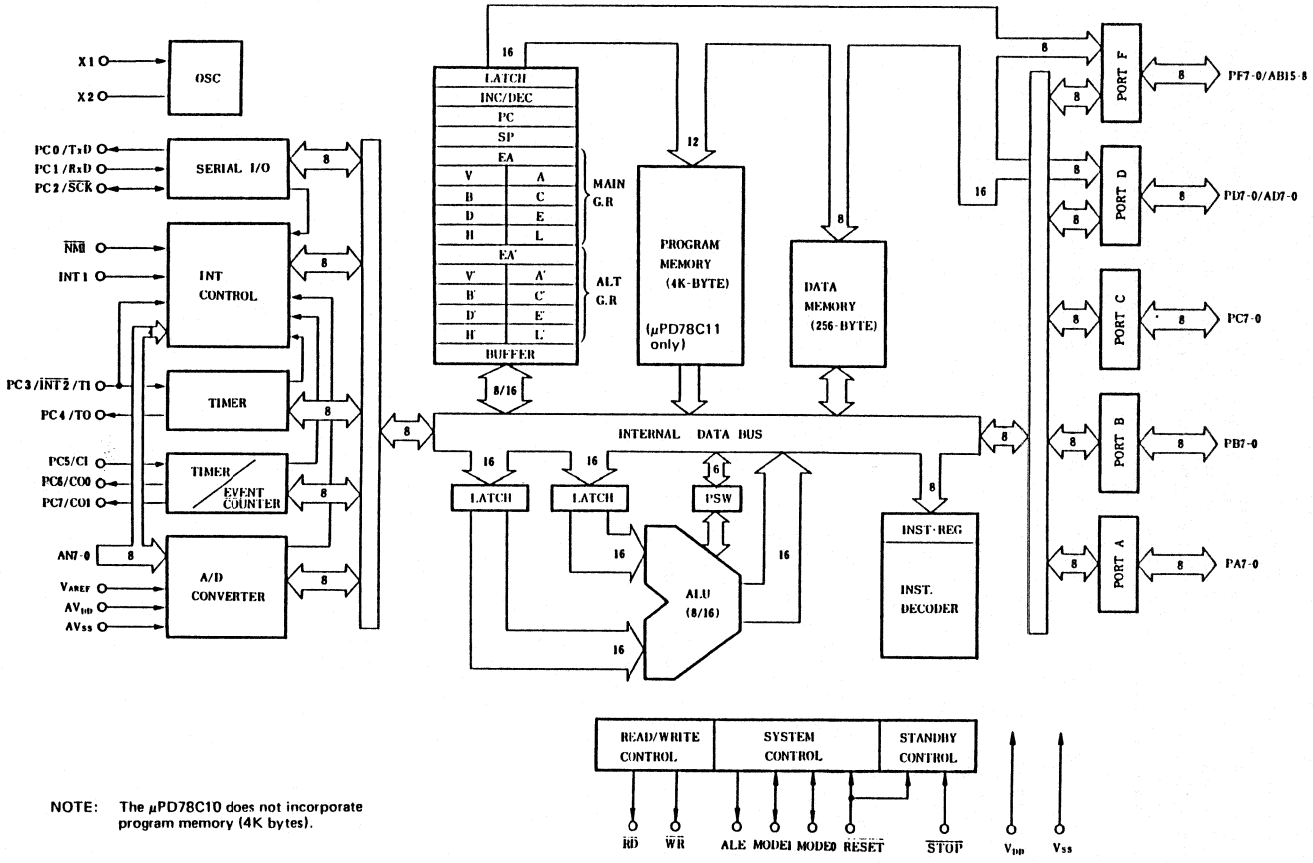
Quil/Shrink Dip Package



Pin Configuration(Top View)

Flat Package





NOTE: The μPD78C10 does not incorporate program memory (4K bytes).

Block Diagram of μPD78C11/μPD78C10

Table of Contents

1. PIN FUNCTIONS

Name	Input/Output	Function	
PA7-PA0 (Port A)	Input/Output	These 8 pins constitute an 8-bit I/O port and input/output can be specified in bit units.	
PB7-PB0 (Port B)	Input/Output	These 8 pins constitute an 8-bit I/O port and input/output can be specified in bit units.	
PC0/TxD	Input/Output/ Output	Port C These 8 pins constitute an 8-bit I/O port and input/output can be specified in bit units.	Transmit Data This pin outputs serial data.
PC1/RxD	Input/Output/ Inpout		Receive Data This pin inputs serial data.
PC2/$\overline{SCK}$	Input/Output/ Input/Output		Serial Clock This pin inputs/outputs serial clock. It becomes an output pin when an internal clock is used or an input pin when an external clock is used.
PC3/$\overline{INT2}$/ TI	Input/Output/ Input/Input		Interrupt Request/Timer Input This pin inputs edge triggering (falling edge) maskable interrupt or external clock for timer. This pin is also shared with zero-cross detection pin for AC input.

Name	Input/Output	Function	
PC4/TO	Input/Output/ Output		Timer Output This pin outputs square waves in which one cycle of the internal clock forms a half cycle, indicating the timer's counting time.
PC5/CI	Input/Output/ Input		Counter Input This pin inputs external pulse for timer/ event counter.
PC6/CO0 PC7/CO1	Input/Output/ Output		Counter Output 0,1 This pin outputs programmable square wave by timer/event counter.
PD7-PD0/ AD7-AD0	Input/Output/ Input/Output	Port D These 8 pins constitute an 8-bit I/O port and input/output can be specified in bit units (μPD78C11).	Address/Data Bus These pins function as multiplexed address/ data bus when using an external memory.
PF7-PF0/ AB15-AB8	Input/Output/ Output	Port F These 8 pins constitute an 8-bit I/O port and input/output can be specified in bit units.	Address Bus These pins function as address bus when using an external memory.

<table>
<tr><th>Name</th><th>Input/Output</th><th>Function</th></tr>
<tr><td>$\overline{WR}$ (Write Strobe)</td><td>Output</td><td>This is a strobe signal output to write data in external memory. This signal becomes high level except the data write machine cycle for external memory. This signal becomes output high impedance when the $\overline{RESET}$ signal low or in the hardware STOP mode.</td></tr>
<tr><td>$\overline{RD}$ (Read Strobe)</td><td>Output</td><td>This is a strobe signal output to read data from external memory. This signal becomes high level except the data read machine cycle from external memory. This signal becomes output high impedance when the $\overline{RESET}$ signal is low or in the hardware STOP mode.</td></tr>
<tr><td>ALE (Address Latch Enable)</td><td>Output</td><td>This is a strobe signal to externally latch the low-order address information output to pins PD7-PD0 to access the external memory. This signal becomes output high impedance when the $\overline{RESET}$ signal is low or in the hardware STOP mode.</td></tr>
<tr><td>MODE0
MODE1
(Mode)</td><td>Input/Output</td><td>In the μPD78C11, pin MODE 0 is set to "0" (low level) and pin MODE1 to "1" (high level).
In the μPD78C10, the external memory capacity can be selected by setting pins MODE0 and MODE1 as follows.

<table><tr><th>MODE0</th><th>MODE1</th><th>External memory</th></tr><tr><td>0</td><td>0</td><td>4K bytes</td></tr><tr><td>1</td><td>0</td><td>16K bytes</td></tr><tr><td>1</td><td>1</td><td>- 64K bytes</td></tr></table>
When both pins MODE0 and MODE1 are set to "1", these pins synchronize to the ALE and a control signal is output.</td></tr>
</table>

Name	Input/Output	Function
$\overline{\text{NMI}}$ (Non-maskable Interrupt)	Input	This pin inputs the edge triggering (falling edge) nonmaskable interrupt.
INT1 (Interrupt Request)	Input	This pin inputs edge triggering (rising edge) maskable interrupt. This pin is also shared with zero-cross detection pin for AC input.
AN7-AN0 (Analog Input)	Input	These eight pins input analog signals for the A/D converter. Pins AN7-AN4 can be used as edge detection (falling edge) inputs.
V_{AREF} (Reference Voltage)	Input	This pin inputs the reference voltage for the A/D converter.
AV_{DD} (Analog V_{DD})		Power supply pin for the A/D converter
AV_{SS} (Analog V_{SS})		Ground pin for the A/D converter
X1, X2 (Crystal)		These are crystal connecting pins for the system clock oscillation. When clock is externally supplied, input it through pin X1.
$\overline{\text{RESET}}$ (Reset)	Input	This pin inputs the active-low reset input signal.
$\overline{\text{STOP}}$ (Stop)	Input	This pin inputs control signal of the hardware STOP mode. When low level of this signal is input, the oscillator stops to operate.
V_{DD}		Positive power supply pin
V_{SS}		Ground pin

2. DIFFERENCE BETWEEN μPD78C11 AND μPD78C10

A difference between μPD78C11 and μPD78C10 is the provision or otherwise of the internal mask programmable ROM. Depending on this, the memory mapping will differ as follows.

(1) μPD78C11

The μPD78C11 incorporates a mask programmable ROM in addresses 0000H to 0FFFH and a RAM in addresses FF00H to FFFFH. Also, external memory can be gradually expanded up to 60K bytes (addresses 1000H to FEFFH). By setting the MEMORY MAPPING register (refer to Fig. 3-11), the external memory size can be selected from among no external memory, 256 bytes, 4K bytes, 16K bytes, and 60K bytes. The external memory can be accessed by using PD7 to PD0 (multiplexed address/data bus), PF7 to PF0 (address bus), $\overline{RD}$, $\overline{WR}$, and ALE signals. Both programs and data can be stored in the external memory. The PF7 to PF0 become address lines depending on the external memory size and the remaining pins can be used as general-purpose I/O ports as shown below.

PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0	External memory
Port	Port	Port	Port	Port	Port	Port	Port	256 bytes max.
Port	Port	Port	Port	AB11	AB10	AB9	AB8	4K bytes max.
Port	Port	AB13	AB12	AB11	AB10	AB9	AB8	16K bytes max.
AB15	AB14	AB13	AB12	AB11	AB10	AB9	AB8	60K bytes max.

(2) μPD78C10

Since the μPD78C10 does not incorporate a ROM, all memory areas excluding the internal RAM area (FF00H to FFFFH) can be set as external memory. By setting pins MODE0 and MODE1, external memory sizes can be selected from among 4K bytes (0000H to 0FFFH), 16K bytes (0000H to 3FFFH), and 64K bytes (0000H to FEFFH) as shown below and in Fig. 2-1.

Operation mode	Control pin		External memory	Internal RAM
	MODE1	MODE0		
4K-byte access	0	0	4K bytes (0000H to 0FFFH)	FF00H to FFFFH
16K-byte access	0	1	16K bytes (0000H to 3FFFFH)	FF00H to FFFFH
64K-byte access	1	1	64K bytes (0000H to FEFFH)	FF00H to FFFFH

The external memory can be accessed by using PD7 to PD0 (multiplexed address/data bus), PF7 to PF0 (address bus), $\overline{RD}$, $\overline{WR}$, and ALE signals.
When accessing 4K- or 16K-byte external memory, among pins PF7 to PF0, those not used as address lines can be used as general-purpose I/O ports. External memory size can be selected by setting pins MODE0 and MODE1. Be sure to set "0" to bits MM2, MM1, and MM0 of the MEMORY MAPPING register.

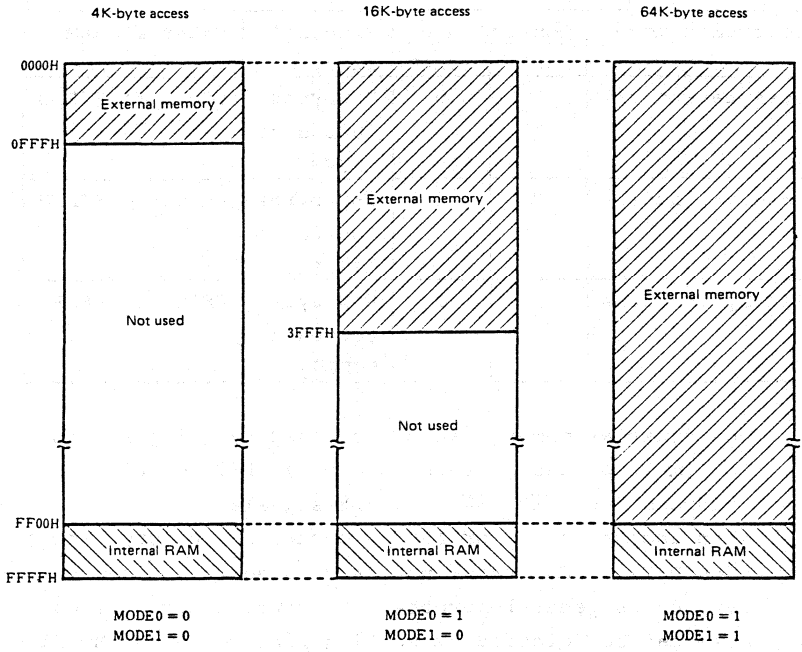


Fig. 2-1 μPD78C10 Memory Map

3. INTERNAL BLOCK FUNCTIONS

3.1 Registers

In total 32 registers are provided: 16 8-bit registers and 4 16-bit registers, as shown in Fig. 3-1 below.

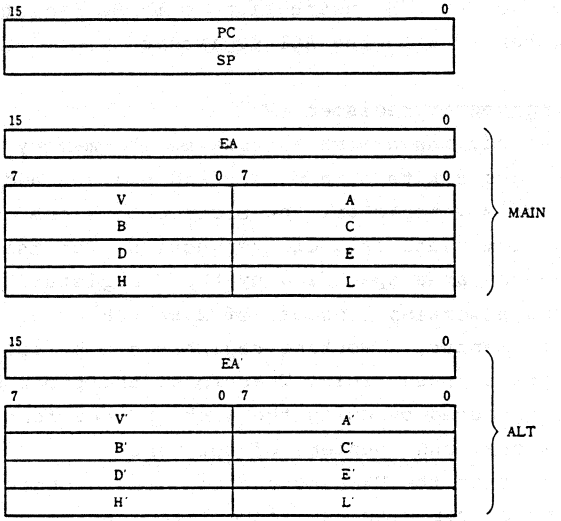


Fig. 3-1 Register Configuration

(1) General-purpose registers (B, C, D, E, H, L)

Two sets of general-purpose registers (MAIN: B, C, D, E, H, L and ALT: B', C', D', E', H', L') are provided. Besides having functions such as auxiliary registers of the accumulator, these registers function as data pointers to address the data memory when used in pairs (BC, DE, HL; B'C', D'E', H'L'). Especially, four register pairs (DE, D'E', HL, H'L') function as base registers. By using the two register sets, when an interrupt occurs, the interrupt is serviced by saving the contents of register to another register set without saving them in memory. This other register set can be used as expansion registers of data pointer. For the register pairs DE, HL, D'E',

and H'L', 1-step autoincrement/decrement and 2-step autoincrement addressing modes are available, thereby reducing the data processing time. By executing the EXX instruction, registers BC, DE, and HL can be exchanged with the ALT registers at one time. Also by executing the EXH instruction, only HL register can be exchanged with the ALT registers.

(2) Working/vector register (V)

When specifying a working area on the memory area, high-order 8 bits of a memory address are addressed by using this V register. Low-order 8 bits are addressed by the immediate data of instruction. For this reason, the memory area specified by the V register can be used as a working register of 256W x 8.

In this manner, a working register can be specified in a 1-byte address field. Therefore, the program size can be reduced by using the working area for flags, parameters, and counters of software.

By executing the EXA instruction, the V register forming a pair with the accumulator can be exchanged with the ALT registers.

(3) Accumulator (A)

Since the μPD78C11 and μPD78C10 are of accumulator type architecture, the accumulator plays an important role in 8-bit data processings such as 8-bit arithmetic operations, logical operations, etc. By executing the EXA instruction, the accumulator forming a pair with the vector register (V) can be exchanged with the ALT registers.

(4) Expansion Accumulator (EA)

This accumulator plays an important role in 16-bit data processing such as 16-bit arithmetic operations, logical operations, etc. By executing the EXA instruction, this register can be exchanged with the EA' register in the ALT registers.

(5) Program Counter (PC)

This is a 16-bit register that retains address information of the program to be executed next. Normally, the program counter is automatically incremented by the number of bytes of the fetched instruction. When an instruction with branch is executed, the immediate data or the contents of register are loaded to the program counter. The PC is cleared to 0000H when the $\overline{\text{RESET}}$ signal is input.

(6) Stack Pointer (SP)

This is a 16-bit register that retains the starting address information in the stack area (LIFO format) in memory.

The contents of the stack pointer is decremented when a call or PUSH instruction is executed or when an interrupt occurs. They are incremented when a return or POP instruction is executed.

3.2 Arithmetic Logic Unit (ALU) ... 16 bits

The arithmetic logic unit performs the following operations.

° 8-bit data processing such as
  - Arithmetic operation
  - Logical operation
  - Shift
  - Rotation

° 16-bit data processing such as
  - Arithmetic operation
  - Logical operation
  - Shift

° 8-bit multiplication

° Division
  16-bit divide by 8-bit

3.3 Program Status Word (PSW)

The program status word consists of 6 kinds of flags, of which 3 kinds of flags (Z, HC, and CY) can be tested by executing an instruction. The contents of PSW are automatically saved to the stack when an interrupt (external, internal, or SOFTI instruction) is generated and they are restored by the RETI instruction. All the bits are reset to 0 by the $\overline{\text{RESET}}$ signal input.

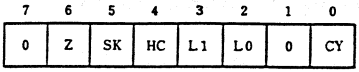


Fig. 3-2 Configuration of Program Status Word

(1) Z (Zero)
This flag is set to 1 if an operation result is zero. Otherwise, this flag is set to 0.

(2) SK (Skip)
This flag is set to 1 when the skip conditions have been satisfied. Otherwise, this flag is set to 0.

(3) HC (Half Carry)
The half carry flag is set to 1 if a carry/borrow is generated from/to bit 3 of the ALU as a result of an 8-bit arithmetic operation. Otherwise, this flag is set to 0.

(4) L1
The L1 flag is set to 1 when the MVI A, byte instruction is executed continuously (string effect). Otherwise, this flag is set to 0.

(5) L0
The L0 flag is set to 1 when MVI L, byte or LXI H, word instruction is executed continuously (string effect). Otherwise this flag is set to 0.

(6) CY (Carry)

The CY flag is set to 1 if a carry/borrow is generated from/to bit 7 or 15 of the ALU as a result of an arithmetic operation. Otherwise, this flag is set to 0. When any of 35 ALU instructions, or rotation or carry operation instructions is executed, each of the flags is affected as listed in Table 3-1.

Table 3-1 Flag Operation

Operation						D6	D5	D4	D3	D2	D0
reg, memory			immediate		skip	Z	SK	HC	L1	L0	CY
ADD ADC SUB SBB DADD DADC DSUB DSBB EADD ESUB	ADDW ADCW SUBW SBBW	ADDX ADCX SUBX SBBX	ADI ACI SUI SBI			‡	0	‡	0	0	‡
ANA ORA XRA DAN DOR DXR	ANAW ORAW XRAW	ANAX ORAX XRAX	ANI ORI XRI	ANIW ORIW		‡	0	●	0	0	●
ADDNC SUBNB GTA LTA DADDNC DSUBNB DGT DLT	ADDNCW SUBNBW GTAW LTAW	ADDNCX SUBNBX GTAX LTAX	ADINC SUINB GTI LTI	GTIW LTIW		‡	‡	‡	0	0	‡
ONA OFFA DON DOFF	ONAW OFFAW	ONAX OFFAX	ONI OFFI	ONIW OFFIW		‡	‡	●	0	0	●
NEA EQA DNE DEQ	NEAW EQAW	NEAX EQAX	NEI EQI	NEIW EQIW		‡	‡	‡	0	0	‡
INR DCR	INRW DCRW					‡	‡	‡	0	0	●
DAA						‡	0	‡	0	0	‡
RLR RLL SLP SLL DRLR DRLL DSLR DSLL						●	0	●	0	0	‡
SLRC SLLC						●	‡	●	0	0	‡
STC						●	0	●	0	0	1
CLC						●	0	●	0	0	0
			MVI A, byte			●	0	●	1	0	●
			MVI L, byte LXI H, word			●	0	●	0	1	●
					BIT SK SKN SKIT SKNIT	●	‡	●	0	0	●
					RETS	●	1	●	0	0	●
All other instructions						●	0	●	0	0	●

‡ ····· Affected (set or reset)

1 ····· Set

0 ····· Reset

● ····· Not affected

3.4 Memory

A maximum of 64K-byte memory can be addressed to the μPD78C11 and μPD78C10. Memory mapping is shown in Fig. 3-3. The external memory and internal RAM areas can be freely used as program memory and data memory. High-speed data processing is possible because access timings of internal memory and external memory are identical.

(1) Interrupt starting addresses

The interrupt starting addresses are fixed as follows.

$\overline{\text{NMI}}$	0004H
INTT0/INTT1	0008H
INT1/$\overline{\text{INT2}}$	0010H
INTE0/INTE1	0018H
INTEIN/INTAD	0020H
INTSR/INTST	0028H
SOFTI	0060H

(2) Call address table

The call addresses of 1-byte instruction (CALT) can be stored in a 64-byte area of addresses 0080H to 00BFH.

(3) Special area in memory

Since the reset starting addresses, interrupt starting addresses, and call tables are assigned to addresses 0000H to 00BFH, use this area bearing in mind this assignment. Addresses 0800H to 0FFFH can be directly addressed by executing a 2-byte call instruction (CALF).

The μPD78C11 incorporates a mask programmable ROM in addresses from 0000H to 0FFFFH.

Since the μPD78C10 does not incorporate a ROM in addresses 0000H to 0FFFH, this special area can be externally set.

(4) Internal data memory area

The 256-byte RAM is incorporated in locations FF00H to FFFFH. The all 256-byte RAM contents are retained in standby mode.

(5) External memory area

In the μPD78C11, the external memory can be gradually expanded up to 60K bytes in addresses 1000H to FEFFH by setting the MEMORY MAPPING register. In the μPD78C10, a 64K-byte external memory can be set in addresses 0000H to FEFFH by setting the MODE0 and MODE1 pins as shown in Fig. 2-1.

The external memory can be accessed by PD7 to PD0 (multiplexed address/data bus), PF7 to PF0 (address bus, $\overline{RD}$, $\overline{WR}$, and ALE signals. Both programs and data can be stored in the external memory.

(6) Working register area

A 256-byte working register can be assigned in any part of memory by the V register. Thus, the working register addressing is possible.

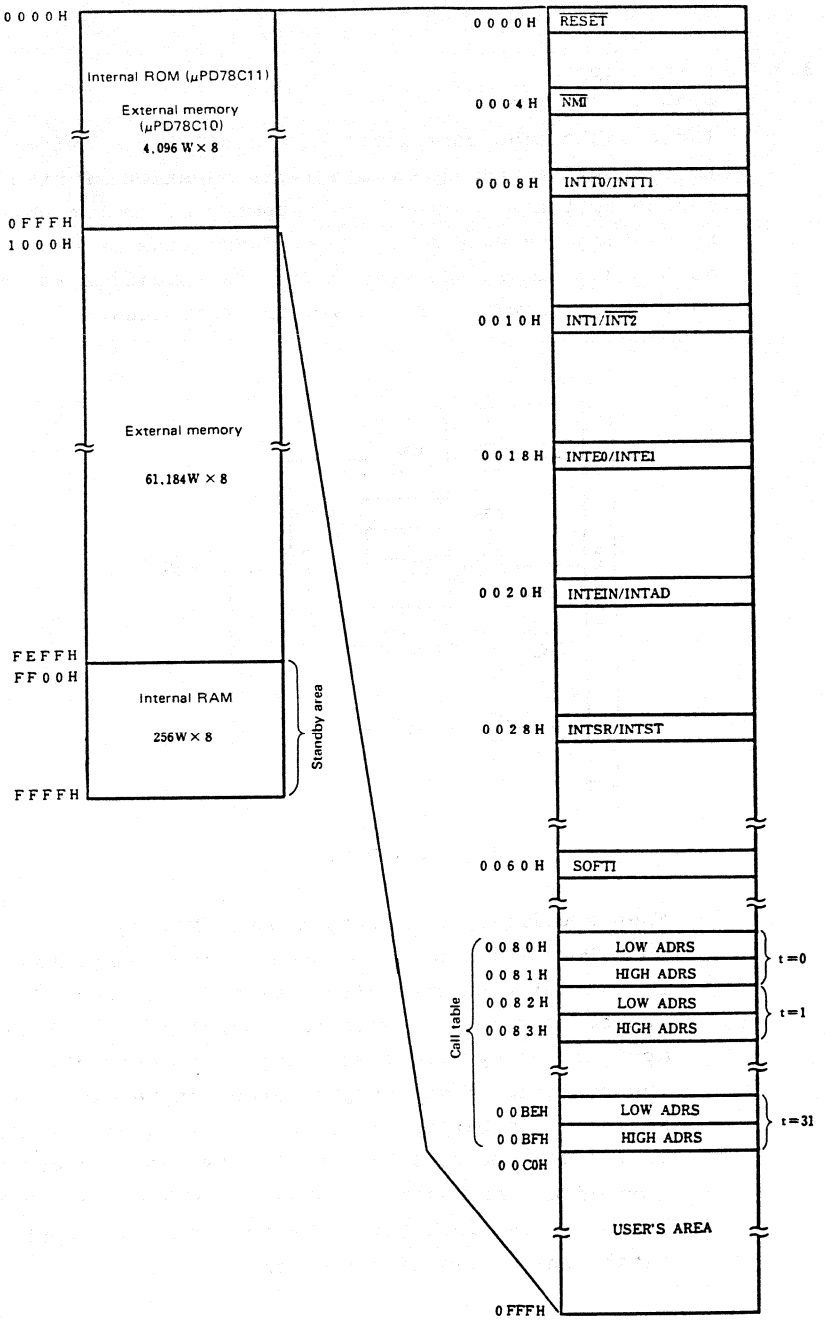


Fig. 3-3 Memory Mapping

3.5 Port Functions

(1) PA7-PA0 (Port A)

These eight pins constitute an 8-bit I/O port with an I/O buffer function and a latch function. Port A can be specified as input or output port in bit units by setting the MODE A register. These pins become output high impedance when Port A is specified as an input port or when a reset signal is applied.

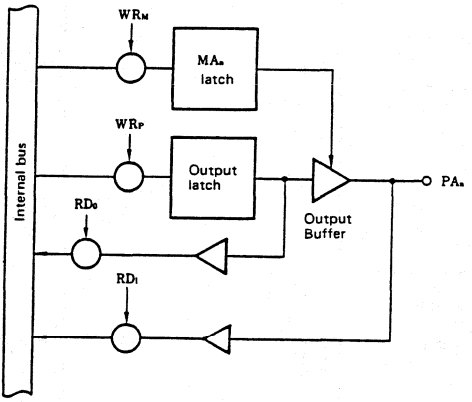


Fig. 3-4 Port A

(a) When specified as an output port (MA_n=0)

When Port A is specified as an output port, the output latch becomes effective and data transfer can be performed between the output latch and the accumulator by executing a transfer instruction. The contents of the output latch can be directly set/reset in bit units by executing an arithmetic or logical operation instruction without intervention of the accumulator. Data written to the output latch are retained until the next Port A output latch manipulating instruction is executed.

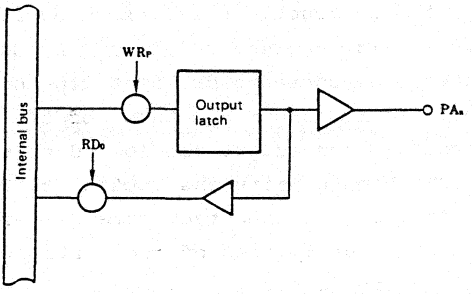


Fig. 3-5 Port A when Specified as an Output Port

(b) When specified as an input port (MA_n=1)
When Port A is specified as an input port, the contents of the PA line can be loaded to the accumulator by executing a transfer instruction. The contents of the PA line can be directly tested in bit units by executing an arithmetic or logical operation without intervention of the accumulator.

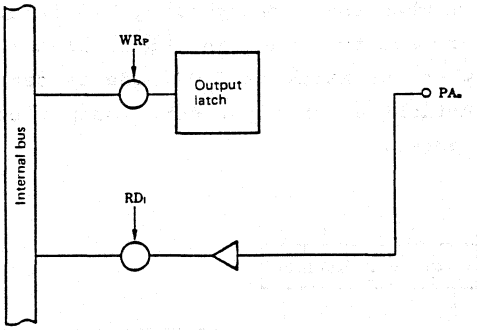


Fig. 3-6 Port A when Specified as an Input Port

Actually, the instructions are executed in 8-bit units. If a read instruction of Port A (MOV A, PA) is executed, the contents of input line of the port specified for input and those of the output latch specified for output are loaded to the accumulator. When a write instruction of Port A (MOV PA, A, etc.) is executed, data are written to both the output latches of the ports specified for input and for output.
However, the contents of the output latch of the bits specified as an input port cannot be loaded to the accumulator neither are they output to the external pins (which are functioning as input pins) because the output buffer is in high impedance state.

° MODE A register (MA)
The MA register is an 8-bit register used to specify input/output of Port A. The input/output of Port A can be specified in bit units and Port A functions as an input port when the corresponding bit of the MA register is 1, and functions as an output port when the corresponding bit is 0. All the bits are set to 1 when the $\overline{\text{RESET}}$ signal is input or when the hardware STOP mode is set and Port A functions as an input port (output becomes high impedance).

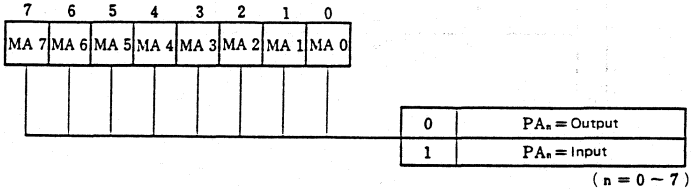


Fig. 3-7 MODE A Register Format

(2) PB7-PB0 (Port B)

Same as Port A, these eight pins constitute an 8-bit I/O port with an I/O buffer and output latch functions. Port B can be specified as an input or output port in bit units by setting the MODE B register. These pins become output high impedance when Port B is specified as an input port and when a reset signal is applied. The operations of Port B are identical to those of Port A and the contents of Port B can be directly set/reset and tested in bit units by executing an arithmetic or logical operation instruction without the intervention of the accumulator. Data transfer between the accumulator and Port B can be also performed.

° MODE B register (MB)

Same as the MODE A register of Port A, the MODE B register is an 8-bit register used to specify input/output to/from Port B in bit units.

All the bits are set to 1 when the $\overline{\text{RESET}}$ signal is input or in the hardware STOP mode and Port B functions as an input port (output becomes high impecance).

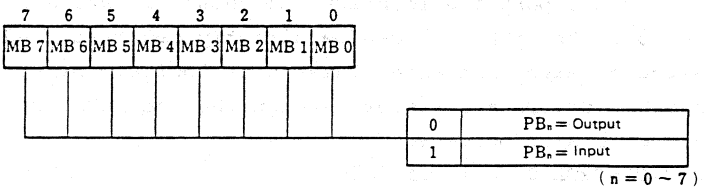


Fig. 3-8 MODE B Register Format

(3) PC7-PC0 (Port C)

The Port C is an 8-bit special I/O port and input/output can be specified in bit units same as Port A. Besides this function as a general-purpose I/O port, these pins function as control signals. These functions are selected by setting the MODE C register and the MODE CONTROL C register in bit units as listed below.

Table 3-2 Functions of PC7-PC0

	$MCC_n = 1$	$MCC_n = 0$	
	$MC_n = \times$	$MC_n = 0$	$MC_n = 1$
PC0	T_XD Output	Output	Input
PC1	R_XD Input	Output	Input
PC2	$\overline{SCK}$ I/O	Output	Input
PC3	$\overline{INT2}$/TI Input	Output	Input
PC4	TO Output	Output	Input
PC5	CI Input	Output	Input
PC6	CO 0 Output	Output	Input
PC7	CO 1 Output	Output	Input

($n = 0 \sim 7$)

The functions of Port C when specified as a general-purpose I/O port are identical to those of Port A and the contents of Port C can be directly set/reset and tested by executing an arithmetic or logical operation instruction. Data transfer between the accumulator and Port C is also possible.

° MODE CONTROL C register (MCC)

The MCC register is an 8-bit register used to specify the port/control signal I/O modes of Port C in bit units.

Pins PC7-PC0 are set to the control signal I/O mode when the corresponding bit of the MODE CONTROL C register is 1, and are set to the port mode when the corresponding bit is 0.

All the bits of the MODE CONTROL C register are set to 0 when the $\overline{RESET}$ signal is input or in the hardware STOP mode, and the register is set to port mode.

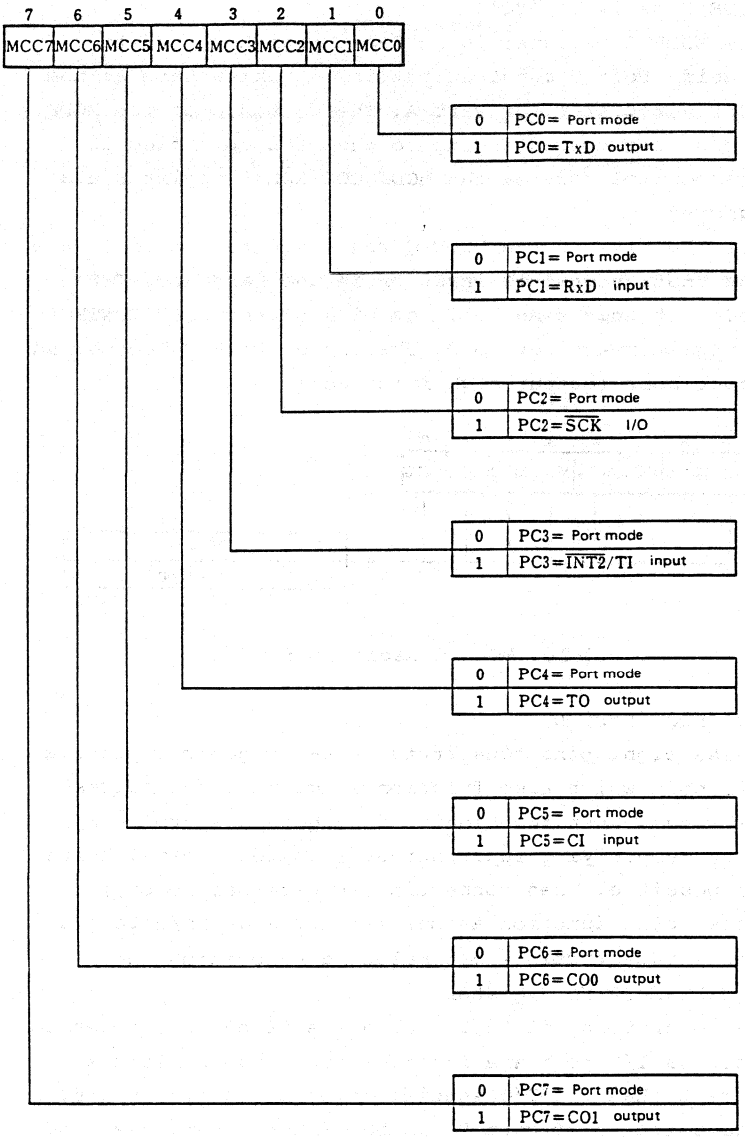


Fig. 3-9 MODE CONTROL C Register Format

° MODE C register (MC)

The MODE C register is an 8-bit register used to specify Port C input/output in bit units same as the MODE A register for Port A. The contents of the MODE C register corresponding to the bits specified to the control mode by the MODE CONTROL C register are ignored.

All the bits of the MODE C register are set to 1 when the $\overline{\text{RESET}}$ signal is input or in the hardware STOP mode. At this time, all the bits of the MODE CONTROL C register are set to 0. Therefore, Port C becomes an input port (output high impedance).

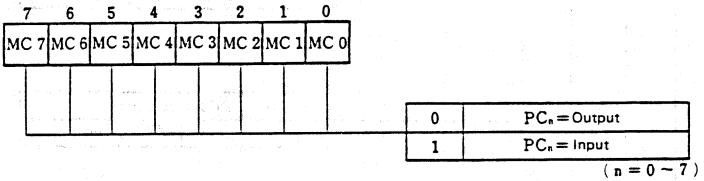


Fig. 3-10 MODE C Register Format

(4) PD7-PD0 (Port D)

These eight pins constitute an 8-bit general purpose I/O port which also function as multiplexed address/data bus. When functioning as a general-purpose I/O port, specifying input/output in byte (8 bits) units is possible. When connecting an external memory, these pins function as multiplexed address/data bus. These two functions are selected by setting the MEMORY MAPPING register.

The operations of Port D when specified as a general-purpose I/O port are identical to those of Port A except that the I/O specification is in byte units. The contents of Port D can be directly set/reset and tested in bit units by executing an arithmetic operation or logical arithmetic operation instruction without intervention of the accumulator. Data transfer between Port D and the accumulator is also possible.

(5) PF7-PF0 (Port F)

These eight pins constitute an 8-bit general-purpose I/O port and they function also as an address bus. When functioning as a general-purpose I/O port, input/output can be specified in bit units. When accessing an externally expansion memory larger than 256 bytes, address signals are output through these pins depending on the external expansion memory size. These two functions are selected by setting the MEMORY MAPPING register and the MODE F register.

PF7	PF6	PF5	PF4	PF3	PF2	PF1	PF0	External memory
Port	Port	Port	Port	Port	Port	Port	Port	256 bytes max.
Port	Port	Port	Port	AB11	AB10	AB9	AB8	4K bytes max.
Port	Port	AB13	AB12	AB11	AB10	AB9	AB8	16K bytes max.
AB15	AB14	AB13	AB12	AB11	AB10	AB9	AB8	60K bytes max.

The operations of Port F when specified as a general-purpose I/O port are identical to those of Port A and the contents of Port F can be directly set/reset and tested in bit units by executing arithmetic operation or logical arithmetic operation instruction without intervention of the accumulator. Data transfer between Port F and the accumulator is also possible.

° MEMORY MAPPING REGISTER (MM)

The MM register is a 4-bit register used to specify port/expansion mode of pins PD7-PD0 and PF7-PF0 and to control enable/disable of the internal RAM access. Bits 0, 1, and 2 (MM0, MM1, and MM2) of the MEMORY MAPPING register are used to specify port/expansion mode and input/output of pins PD7-PD0 and specify address line of pins PF7-PF0. When bits MM1 and MM2 of the MEMORY MAPPING register are set to 0, both pins PD7-PD0 and pins PF7-PF0 function as general-purpose I/O ports. Bit MM0 is used to specify input/output of PD7-PD0 while the MODE F register is used to specify input/output of pins PF7-PF0.

Four external expansion memory sizes are selectable: 256, 4K, 16K, and 60K bytes, as shown in Fig. 3-11. Among pins PF7-PF0, the ports which are not used as address lines can be used as general-purpose I/O ports.

Bit 3 (RAE) of the MEMORY MAPPING register is used to control enable/disable of the internal RAM access. When expanding an external memory, if the external memory uses the expanded area without using the internal RAM, set the RAE bit to 0 to inhibit the internal RAM access.

Each of bits MM0, MM1, and MM2 of the MEMORY MAPPING register is set to 0 when the $\overline{\text{RESET}}$ signal is input or in the hardware STOP mode and pins PD7-PD0 become input port (output high impedance). Even if the $\overline{\text{RESET}}$ signal is input during the normal operation, the contents of RAE bit at that moment are retained. However, with power-on reset, the RAE bit is undefined. Therefore, initialize the RAE bit by executing an instruction.

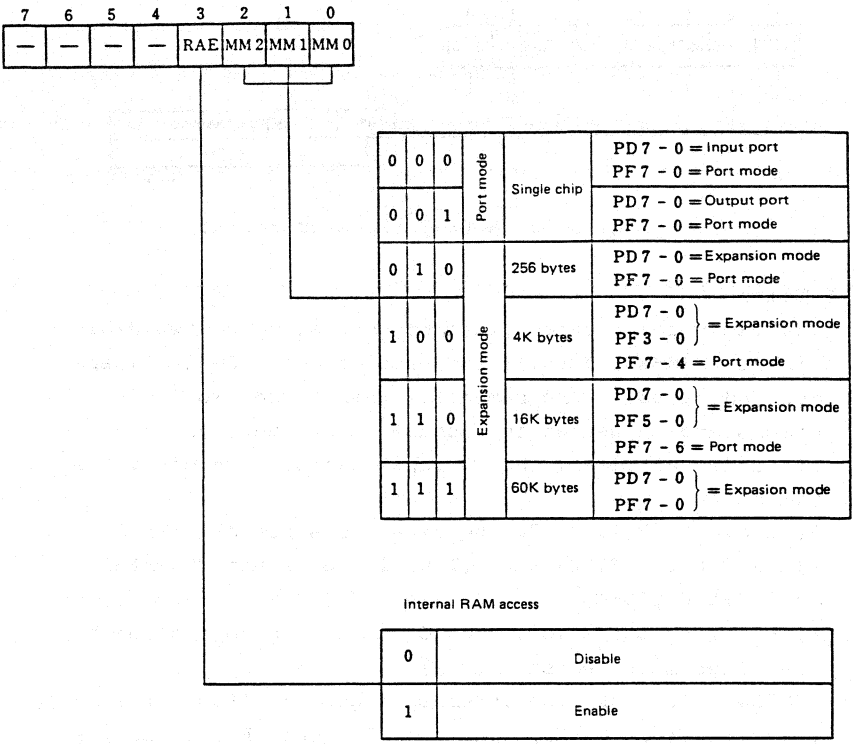


Fig. 3-11 MEMORY MAPPING Register Format

° MODE F register (MF)

The MF register is used to specify input/output of Port F same as the MODE A register of Port A. However, the contents of the MODE F register corresponding to the bits of Port F specified as address lines by the MEMORY MAPPING register become output mode. All the bits of the MODE F register are set to 1 when the $\overline{\text{RESET}}$ signal is input or in the hardware STOP mode and Port F functions as an input port (output high impedance).

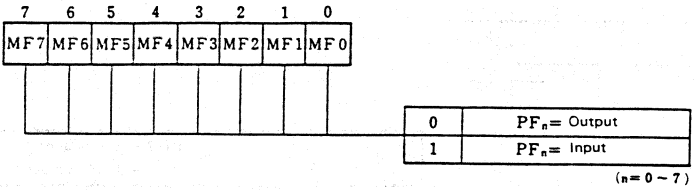


Fig. 3-12 MODE F Register Format

3.6 Timer

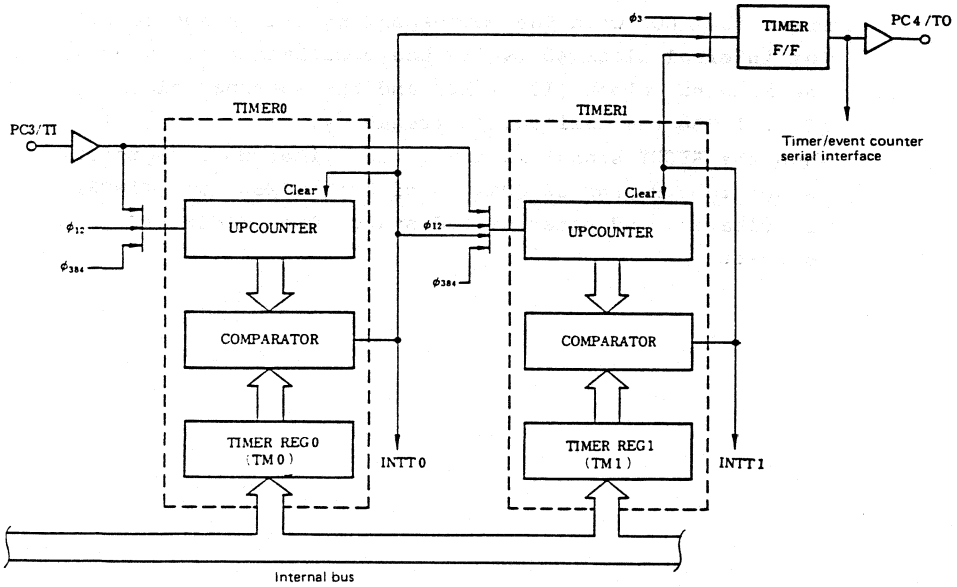
The timer is an interval timer consisting of two 8-bit timers (TIMER0,TIMER1), and each of these 8-bit timers are individually programmable. When these two 8-bit timers are connected in cascade, they function as a 16-bit interval timer. The TI input can be counted by the timer.

As shown in Fig. 3-13, the timer consists of TIMER0, TIMER1, 8-bit TIMER REGs (TM0, 1), an 8-bit COMPARATOR, an 8-bit UPCOUNTER, and an 8-bit TIMER F/F. Input selection, timer operation, and TO output are controlled by the timer mode register (TMM).

TIMER0 inputs are the internal clock ϕ12 (1μs: at 12MHz), ϕ384 (32μs: at 12MHz), and the TI input. For TIMER1, ϕ12 and ϕ384 are the same as for TIMER0. Furthermore, the TIMER0 coincidence signal is also used as input to TIMER1 in addition to the TI input.

Because TIMER0 and TIMER1 operate in the same way, only the TIMER0 operation is described here. TIMER0 starts operating when the number of counts is set to TIMER REG0 and necessary data (bit 4 of Timer Mode register=0) are written into the Timer Mode register. The upcounter is incremented by each input clock while the comparator constantly compares the contents of both the upcounter and TIMER REG0. If both coincide, a coincidence signal (internal interrupt request: INTT0) is generated. When this happens, the upcounter is cleared and starts counting again from 00H. In this way, TIMER0 functions as an interval timer to repeatedly generate interrupt requests at count intervals set to the TIMER REG0.

The internal interrupt request (INTT0) can be disabled by setting bit 1 (MKT0) of Interrupt Mask register (MKL) to 1. In the output of TO, the TIMER F/F is provided and it is inverted each time the coincidence signal of the COMPARATOR of each timer is generated or the internal clock $\phi 3$ (250μs: at 12MHz) is input. Therefore, square waves whose cycles are half the count time or $\phi 3$ are obtained. This output can be used as the reference time for the timer/event counter when setting the timer/event count mode register (ETMM). This output can be also used as the serial clock ($\overline{SCK}$) for the serial interface by setting the serial mode register (SMH).



Remarks: $\phi_3 = f_{XTAL} \times \frac{1}{3}$

$\phi_{12} = f_{XTAL} \times \frac{1}{12}$ f_{XTAL} : Oscillation frequency (MHz)

$\phi_{384} = f_{XTAL} \times \frac{1}{384}$

Fig. 3-13 Timer Block Diagram

(1) Timer mode register (TMM)

This is an 8-bit register which specifies the operation modes of TIMER0, TIMER1, and TIMER F/F (refer to Fig. 3-15).

Bits 0 and 1 (TF0, 1) of the Timer Mode register specify the operation mode of TIMER F/F, and bits 2 and 3 (CK00, CK01) specify the input clock for TIMER0, and bit 4 (TS0) specifies the operation mode of TIMER0. Bits 5 and 6 (CK10,CK11) specify the input clock for TIMER1, and bit 7 (TS1) specifies the operation mode of TIMER1.

When both TS0 and TS1 are 1, the corresponding UP-COUNTER is cleared to 00H, and upcounting will then start from 00H when the corresponding bit is set to 0. The internal clock $\phi 3$ is 1/3 the oscillation frequency, the internal clock $\phi 12$, 1/12, and the internal clock $\phi 384$, 1/384 the oscillation frequency.

When the $\overline{\text{RESET}}$ signal is input, the Timer Mode register is set to FFH, and UPCOUNTERS of both TIMER0 and TIMER1 are cleared, and enter halted state. Then TIMER F/F is reset.

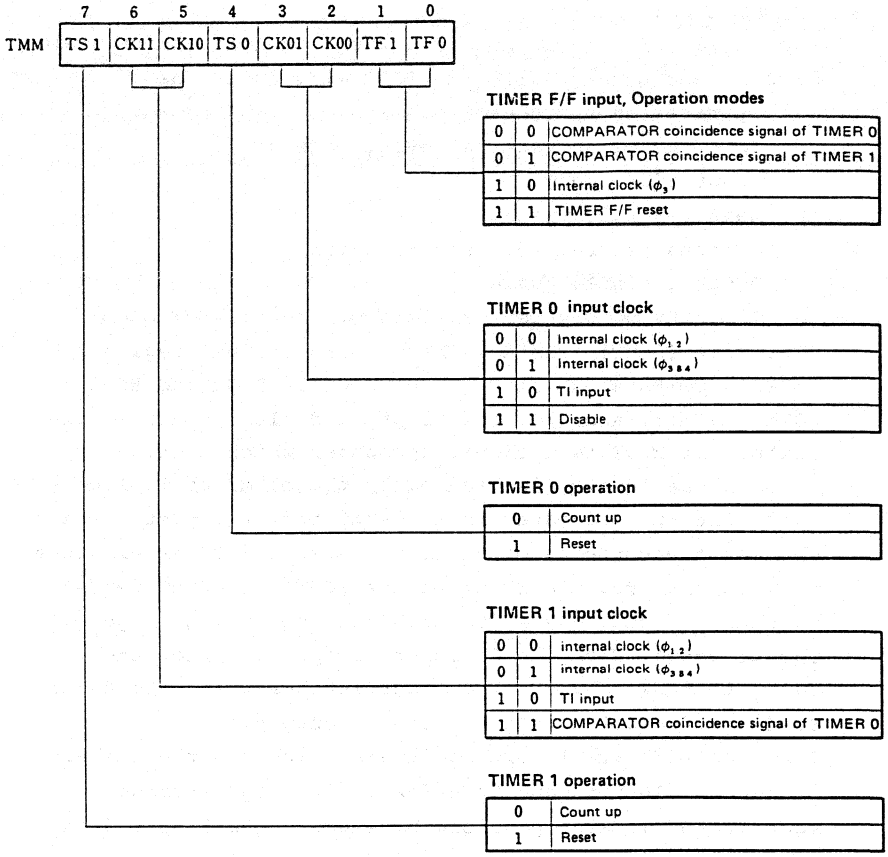


Fig. 3-14 Timer Mode Register (TMM) Format

3.7 Timer/Event Counter

The μPD78C11/78C10 is provided with a multifunction 16-bit timer/event counter with the following functions:

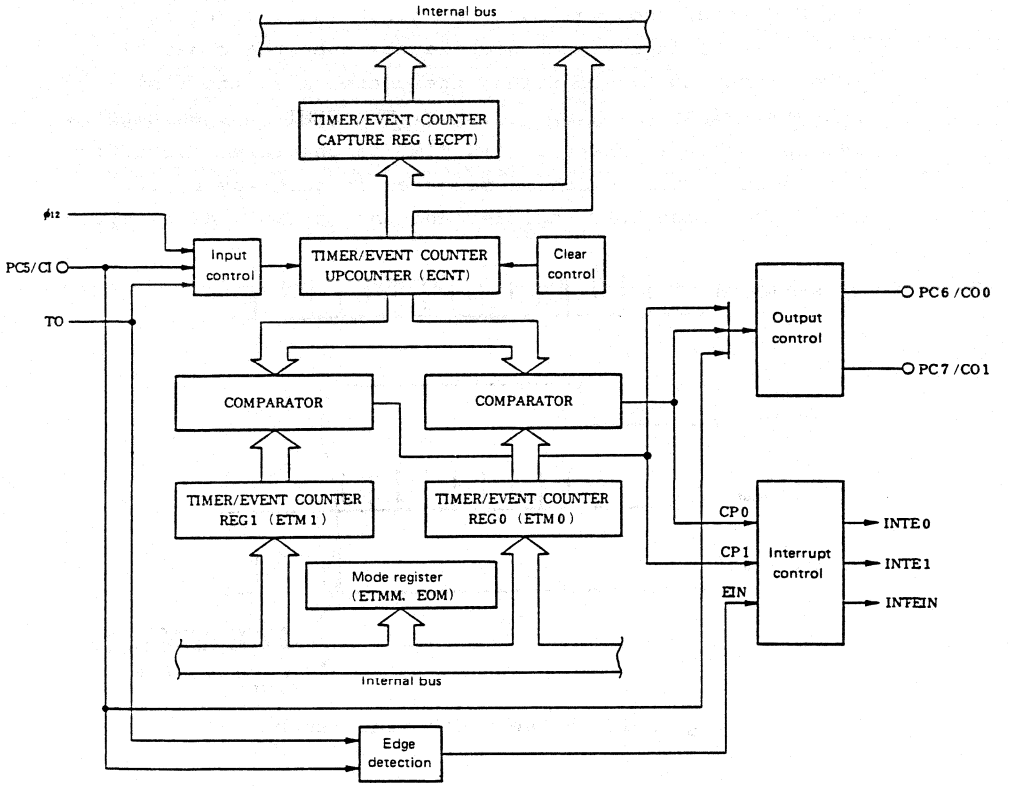
- Operates as an interval timer
- Operates as an external event counter
- Counts frequency
- Measures pulse width
- Outputs programmable square waves
- Outputs single pulse

The Timer/Event Counter consists of TIMER/EVENT COUNTER UPCOUNTER (ECNT), TIMER EVENT COUNTER CAPTURE register (ECPT), COMPARATOR, TIMER/EVENT COUNTER REG0 and REG1 (ETM0, ETM1), and I/O, interrupt, and clear control circuits. The ECNT is a 16-bit upcounter which counts input pulses. The counter is cleared by the clear control circuit. The ECPT register is a 16-bit buffer register which retains the contents of the ECNT. When the internal clock is specified for the input to the ECNT, the ECPT register latches the contents of the ECNT at the falling edge of the CI input. When the CI input is specified for the input to ECNT, the ECPT register latches the contents of ECNT at the falling edge of the TO output.

ETM0 and ETM1 are 16-bit registers which set the numbers of counts. 16-bit data transfer to/from the expanded accumulator is also possible by executing a 16-bit data transfer instruction.

The COMPARATOR compares the contents of the ECNT to the contents of ETM0 and ETM1 registers. When the data in the registers coincide, the comparator generates a coincidence signal.

The interrupt control circuit controls the timer/event counter interrupt. The following three interrupt sources are generated by the interrupt control circuit; coincidence signal (INTE0) of ECNT and ETM0 register, coincidence signal (INTE1) of ECNT and ETM1 register, and falling edge (INTEIN) of the CI input or timer output (TO).



Remarks: $\phi_{12}=f_{XTAL}\times\frac{1}{12}$, f_{XTAL} : Oscillation frequency

Fig. 3-15 Timer/Event Counter Block Diagram

The following describes the operation of the timer/event counter taking pulse width measurement as an example. In pulse width measurement mode, the high level width of the external pulse input to the CI pin is measured. This operation can be started by setting 09H to the Timer/ Event Counter Mode register (ETMM).

The ECNT continues the internal clock (ϕ12) counting while the CI input is at high level. When the external pulse input to the CI falls, the contents of the ECNT are transferred to the ECPT register. The ECNT is then cleared and an internal interrupt (INTEIN) is generated (refer to Fig. 3-16). Pulse width is measured in this way by the contents of the ECPT register and the internal clock cycle.

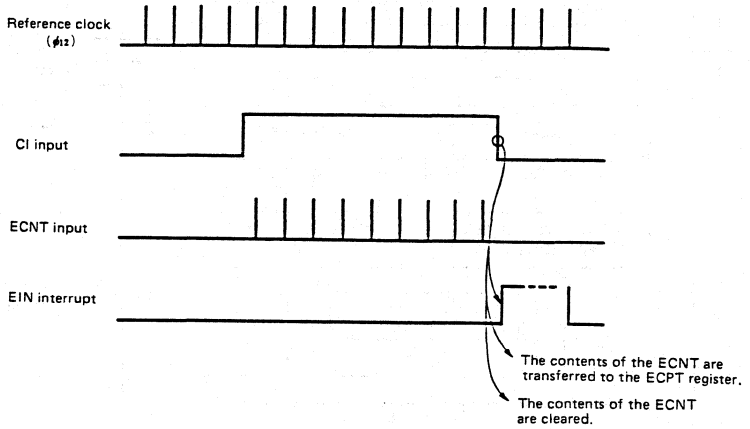


Fig. 3-16 Pulse Width Measurement

The μPD78C11/78C10 has an output control circuit that outputs pulses whose width and frequency can be changed, interlocking with the timer/event counter.
Two outputs are available from the output control circuit, CO0 and CO1, whose configurations are identical. Therefore, only CO0 output is described here. Fig. 3-17 shows the configuration of the CO0 output. The CO0 output is a master-slave configured output. The level flip-flop (LV0) in the first stage retains the level to be output next and the output latch in the second stage outputs the level of LV0 to an external device.
LV0 can be set/reset by setting the Timer/Event Counter Output Mode register (EOM). LV0 is provided with a level invert pin (INV) so that the level of LV0 can be inverted at the output timing determined by the setting of the

Timer/Event Counter Mode register. The level of LV0 is output from the output latch at the timing set by the Timer/Event Counter Mode register.

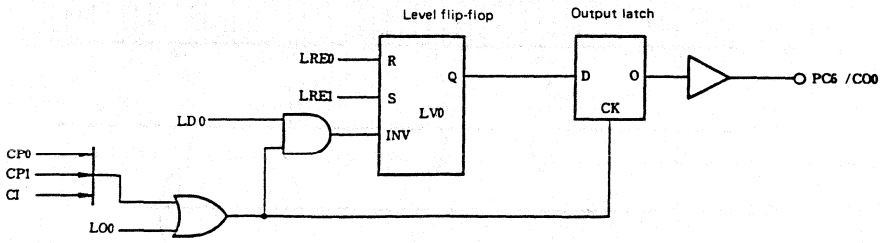


Fig. 3-17 Output Control Circuit

The following describes the operation to output square waves to CO0 pin.

To output square waves to CO0 pin, first, the ECNT is cleared, and numbers of counts (ETM0<ETM1) are set to ETM0 and ETM1 registers. Then, data to enable LV0 initialization and inversion of the level of LV0 are set to the Timer/Event Counter Output Mode register.

The timer/event counter starts to operate by setting the ϕ12 (1μs: at 12MHz) internal clock to ECNT, the clear mode of ECNT to the coincidence signal of ECNT and ETM1 register, and output timing to CO0 pin at the issuance of the coincidence signal of ECNT and ETM0 register or of ECNT and EMT1 register.

The ECNT counts up by each ϕ12 internal clock while the comparators compare the contents of both ECNT and ETM0 register as well as ECNT and EMT1 register and each generates a coincidence signal (CP0, CP1) if both coincide.

Each coincidence signal permits outputting the level of LV0 to CO0 pin and inverting its level.

The ECNT is cleared by the coincidence signal of ECNT and ETM1 register (CP1) and starts counting again from 0000H.

These operating procedures are repeated continuously (refer to Fig. 3-18).

In this way, programmable square waves with pulse width set by the numbers of counts of ETM0 and ETM1 registers can be output.

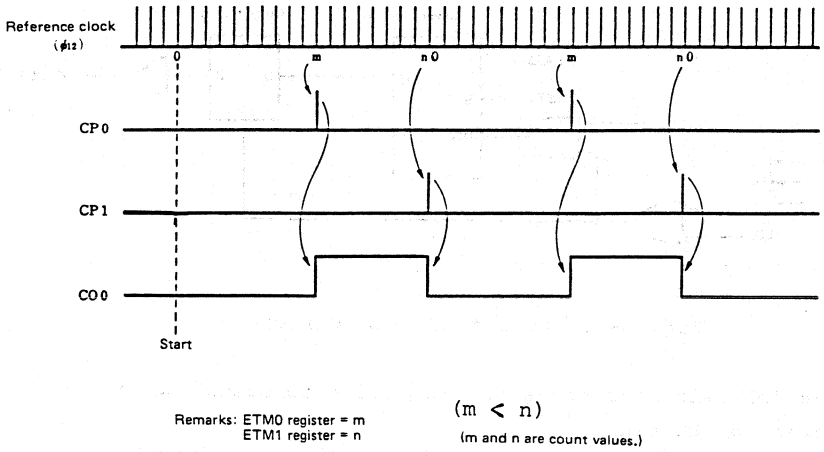


Fig. 3-18 Square Wave Output

(1) Timer/Event Counter Mode register (ETMM)

This is an 8-bit register which specifies operation modes of the timer/event counter (refer to Fig. 3-19). Bits 0 and 1 (ET0, ET1) of the Timer/Event Counter Mode register specify the input clock of the Timer/Event Counter Upcounter (ECNT). Bits 2 and 3 (EM0, EM1) specify the ECNT clearing mode. Bits 4 and 5 (CO00, CO01) specify the timing for outputting the contents of the output latch to Counter Output 0 (CO0). Bits 6 and 7 specify the CO1 output timing. The internal clock φ12 is 1/12 the oscillation frequency. When the $\overline{\text{RESET}}$ signal is input or in the hardware STOP mode, the Timer/Event Counter Mode register is reset to 00H.

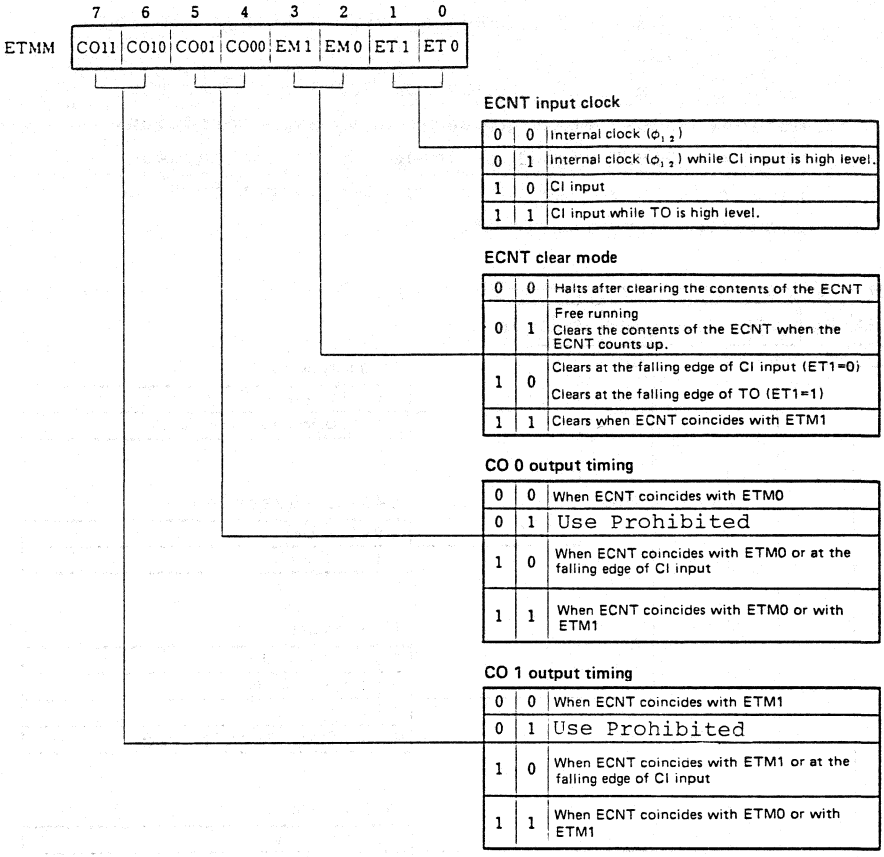


Fig. 3-19 Timer/Event Counter Mode Register Format

(2) Timer/Event Counter Output Mode register (EOM)

This is an 8-bit register which controls the operation modes: CO0 and CO1 (Counter Outputs 0, 1) of the Timer/Event Counter.

Bits 0 and 4 (LO0, LO1) of the Timer/Event Counter Output Mode register specify whether or not the levels of LV0 and LV1 are output to CO0 and CO1. Bits 1 and 5 (LD0, LD1) specify whether or not the levels of LV0 and LV1 are inverted at the timing specified by the Timer/Event Counter Mode register. Bits 2, 3, 6, and

7 (LRE0, LRE1, LRE2, LRE3) specify setting/reseting LV0 and LV1.

LO0, LO1, LRE0, LRE1, LRE2, and LRE3 bits are automatically reset to 0 at the end of each operation. When the $\overline{\text{RESET}}$ signal is input, or in the hardware STOP mode, the Timer/Event Counter Output Mode register is reset to 00H.

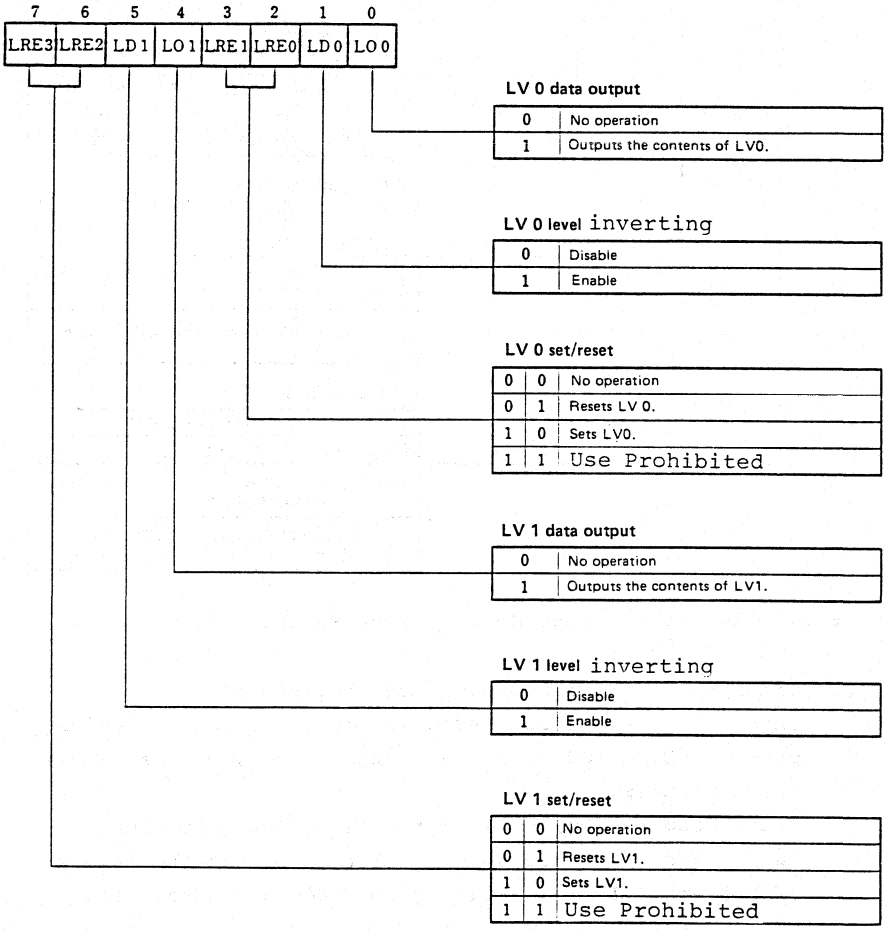


Fig. 3-20 Timer/Event Counter Output Mode Register (EOM) Format

3.8 Serial Interface

The μPD78C11/78C10 is provided with a serial interface with the following three operation modes.

- Asynchronous mode (start-stop synchronization)
  Data transmit/receive by using the start and stop bits. Bit and character of data are synchronized by the start bit.
- Synchronous mode
  Data transfer is performed in synchronization with the serial clock.
- I/O interface mode
  In the same way as the serial data transfer in the μPD7801, μPD78C06, etc., data transfer is performed in synchronization with a controlled serial clock.

As shown in Fig.3-21, the serial interface consists of: serial data input pin (RxD), serial data output pin (TxD), serial clock I/O pin ($\overline{SCK}$), transfer controller, two 8-bit serial registers (one each for transmit and receive), 8-bit transmit buffer, and 8-bit receive buffer. Serial registers and buffers are discretely provided for transmit and receive operations. Therefore, data transmit and receive can be performed independently (full-duplexed, double buffer transmitter/receiver).
However, since the serial clock ($\overline{SCK}$) is used commonly in data transmit and receive, communication system becomes half duplex mode in synchronous or I/O interface mode.

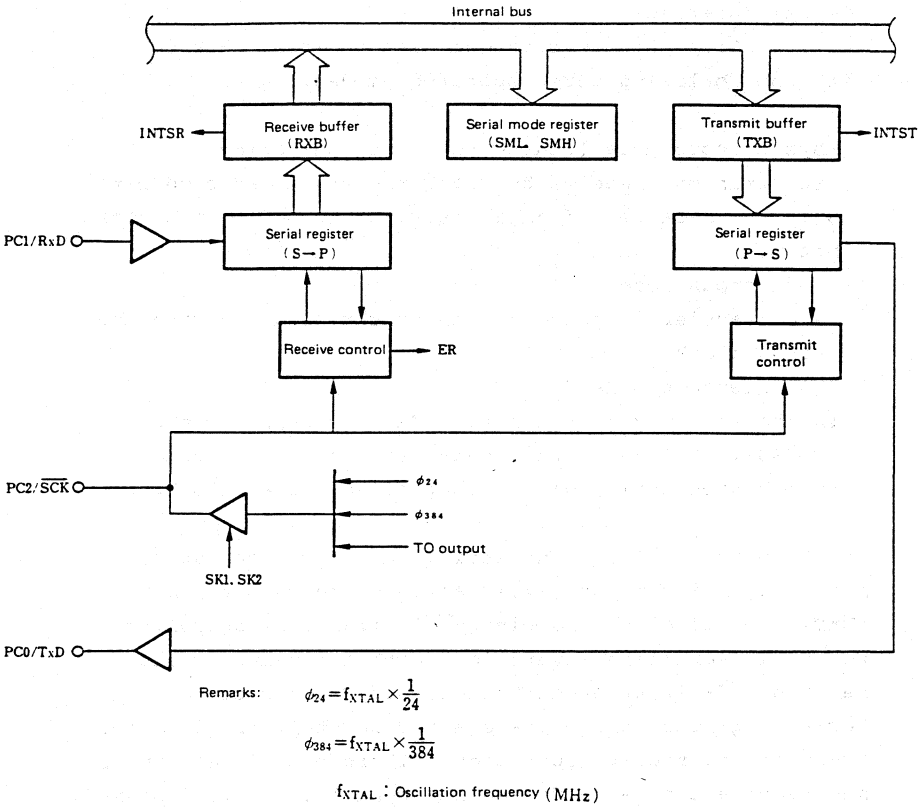


Fig. 3-21 Serial Interface Block Diagram

(1) Asynchronous mode

In the asynchronous mode, clock rate, character length, number of stop bits, parity enable, odd/even parity are specified by the Serial Mode register (SML). Data transmit is enabled by setting bit 2 (TxE) of the Serial Mode register (SMH) to 1.

When data is written to the transmit buffer by the MOV TXB, A instruction and the previous data transfer is complete, the contents of the transmit buffer is automatically transferred to the serial

register. The start bit (1 bit), parity bit (odd/even, no parity), and stop bits (1 or 2 bits) are automatically added to the data transferred to the serial register. The transferred data is then output from the TxD pin from the least significant bit (LSB).
When the transmit buffer becomes empty, an internal interrupt (INTST) is generated.
The transmit data is output from the TxD pin at the falling edge of $\overline{SCK}$ at the clock rate of serial clock ($\overline{SCK}$) X 1, X 1/16, or X 1/64.
The maximum transfer rate at 12MHz is determined by $\overline{SCK}$ and clock rate as shown in the following table.

$\overline{SCK}$ / Clock rate	Internal clock		External clock	
	$\overline{SCK}$	Transfer rate	$\overline{SCK}$	Transfer rate
× 1	500kHz	500kbps	660kHz	660kbps
×16	2 MHz	125kbps	2 MHz	125kbps
×64		31.25kbps		31.25kbps

The TxD pin becomes mark state (1) when TxE is "0" or no data to be transmitted exists in the serial register.
The internal interrupt (INTST) is disabled by setting bit 2 (MKST) of the Interrupt Mask register (MKH) to 1.

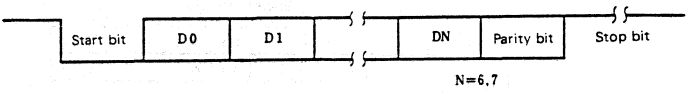


Fig. 3-22 Asynchronous Data Format

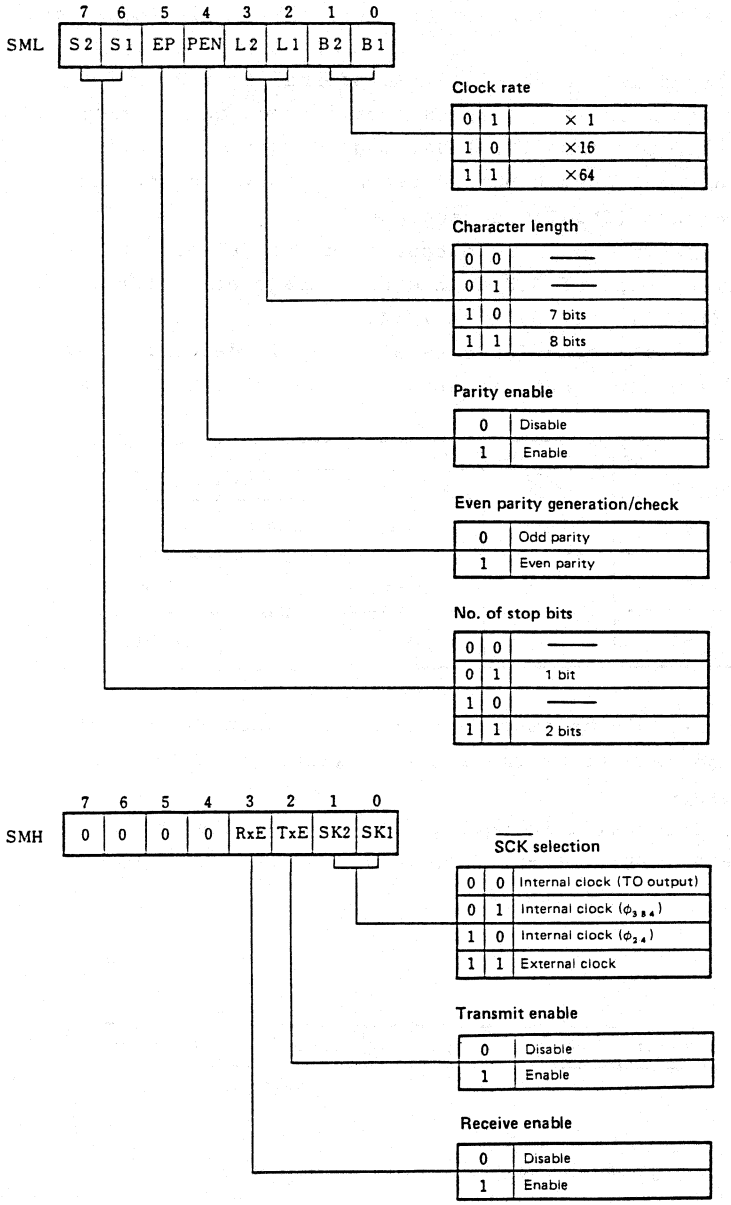


Fig. 3-23 Serial Mode Register Format in Asynchronous Mode

Receive operation is enabled by setting bit 3 (RxE) of the serial mode register (SMH) to 1. The start bit is confirmed by detecting low-level of RxD input and detecting its low level again 1/2 bit after sensing it for the first time. Sampling is done at the middle of the following character bits, parity bits, and stop bits to receive data. When the specified data is input to the serial register from RxD pin, the data are transferred to the receive buffer. An internal interrupt (INTSR) is generated when the receive buffer becomes full. The internal interrupt (INTSR) is disabled by setting bit 1 (MKSR) of the interrupt mask register (MKH) to 1.

The error flag is set to 1 during receive operation when

- odd/even parity check is done (when PEN bit = 1) and no coincidence is obtained (parity error),
- stop bit is low (framing error)
- next data is transferred to the receive buffer when the receive buffer is full (overrun error)

However, error interrupt function is not provided; therefore, this is tested on the program by executing a skip instruction (SKIT, SKNIT).

Either the external or internal clock can be selected as serial clock ($\overline{SCK}$) by setting the Serial Mode register (SMH).

ϕ24, ϕ384, or TO output can be selected as an internal clock and can be output externally. An external serial clock can also be input.

When the internal clock (TO output) is used as $\overline{SCK}$, the data transfer rate can be freely changed by programming.

The maximum data transfer rate during receive operation at 12 MHz is determined by $\overline{SCK}$ and clock rate as follows.

SCK / Clock rate	Internal clock		External clock	
	SCK	Transfer rate	SCK	Transfer rate
×1 *2	500kHz	500kbps	660kHz 1 MHz	660kbps 1 Mbps *1
×16	2 MHz	125kbps	2 MHz	125kbps
×64		31.25kbps		31.25kbps

*1 To receive data transferred at a rate of 660K to 1M bps, 2 stop bits become necessary.

*2 When the clock rate x1 is used, RxD and SCK must be synchronized externally.

As an example, when data transfer is performed at a transfer rate of 110 to 9,600bps and the internal clock (φ12) is specified for timer input clock, the timer count values (C) are as shown in the table below.

Oscillation frequency (MHz) / Transfer rate (bps) \ N	7.3728		11.0592	
	16	64	16	64
9600	C = 2	–	C = 3	–
4800	4	C = 1	6	–
2400	8	2	12	C = 3
1200	16	4	24	6
600	32	8	48	12
300	64	16	96	24
150	128	32	192	48
110	175	44	262	65

(2) Synchronous mode

In synchronous mode, data can be transferred with character length fixed to 8 bits and without parity bit. Therefore, the Serial Mode register (SML) should be set to 0CH (refer to Fig. 3-24).

Transmit operation is enabled by setting bit 3 (TxE) of the Serial Mode register (SMH) to 1.

When data is written to the transmit buffer by executing the MOV TXB, A instruction and the previous data transfer is complete, the contents of the transmit buffer is automatically transferred to the serial register and converted to serial data. The serial data are output from TxD pin from the least significant bit in synchronization with the falling edge of $\overline{SCK}$. The serial data are output at the same rate as $\overline{SCK}$. In transmission, the data transfer rate is 500K bps maximum when the internal clock is used as $\overline{SCK}$ and 1M bps maximum when the external clock is used as $\overline{SCK}$ (at 12MHz).

An internal interrupt (INTST) is generated when data is transferred from the transmit buffer to the serial register and transmit buffer becomes empty.

TxD pin becomes mark state (1) when TxE is 0 or no data to be transmitted exists in the serial register.

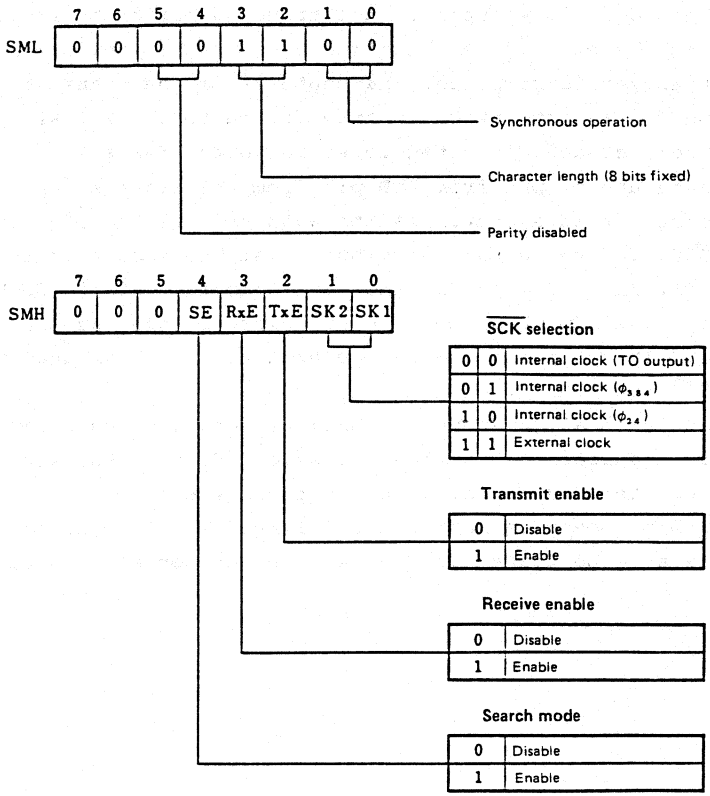


Fig. 3-24 Serial Mode Register Format in Synchronous Mode

In the synchronous mode, two operation modes for data reception are available and these modes are specified by setting the SE bit of the serial mode register (SMH). The search mode is specified when the SE bit is set to 1. In this mode, each time 1 bit is received from the RxD pin, the contents of the serial register are transferred to the receive buffer and an internal interrupt (INTSR) is generated. Since the μPD78C11/78C10 is not provided with synch. character detection circuit by means of hardware, synch. character should be detected by means of software.

When the sync. character is detected and the receive operation is synchronized, the SE bit is reset to 0. When the SE bit is reset, the operation mode changes to character receive mode, the contents of the serial register are transferred to the receive buffer, and the internal interrupt (INTSR) is generated each time an 8-bit data is received.

The internal interrupt (INTSR) is disabled by setting the MKSR bit of the interrupt mask register (MKH) to 1.

In the synchronous mode, data are output from TxD pin at the falling edge of $\overline{SCK}$, and data are input RxD pin at the rising edge of $\overline{SCK}$.

Either the internal clock or external clock can be selected for $\overline{SCK}$ by setting the serial mode register (SMH).

The data transfer rate in the receive operation is 500K bps maximum when the internal clock is used as $\overline{SCK}$, and 660K bps maximum when the external clock is used as $\overline{SCK}$ (at 12MHz).

(3) I/O interface mode

This mode is identical to the serial interface mode of the μCOM-87, and is very effective when expanding I/O devices externally or connecting I/O controllers such as A/D converter and LCD controller. In this mode, data transfer is performed with no parity bit and with character length fixed to 8. The most significant bit (MSB) is transferred first in this mode. Therefore, the serial mode register (SML) should be set to 0CH and bit 5 (IOE) of the serial mode register (SMH) should be set to 1. In this mode, characters are synchronized by the controlled $\overline{SCK}$ (eight serial clock pulses). Therefore, $\overline{SCK}$ should be set to high when data transfer is not performed.

Data transmit operation in the I/O interface mode is enabled by setting bit 2 (TxE) of the serial mode register (SMH) to 1.

When data is written to the transmit buffer register using the MOV TXB, A instruction, after the previous data is transmitted, the contents of the transmit buffer register are automatically transferred to the serial register and output from TxD pin at the falling edge of $\overline{SCK}$. When the transmit buffer register becomes empty, the internal interrupt (INTST) is generated.

The data transfer rate in the transmit operation is 500K bps maximum when the internal clock is used as the $\overline{SCK}$, or 1M bps maximum when the external clock is used as the $\overline{SCK}$ (12MHz).

Receive operation is enabled by setting bit 3 (RxE) of the serial mode register (SMH) to 1. Receive data is input to the serial register at the rising edge of $\overline{SCK}$. When the serial register receives an 8-bit data, the data is transferred from the serial register to the receive buffer register and the internal interrupt (INTSR) is generated.

Either the external clock or internal clock can be selected as the $\overline{SCK}$ by the serial mode register (SMH).

The data transfer rate in the receive operation is 500K bps maximum when the internal clock is used as the $\overline{SCK}$, and 660K bps when the external clock is used as the $\overline{SCK}$ (at 12MHz). However, the high-level width of the eighth $\overline{SCK}$ must be at least 6 states. *1

*1 1 state = 250 nsec at fosc = 12 MHz

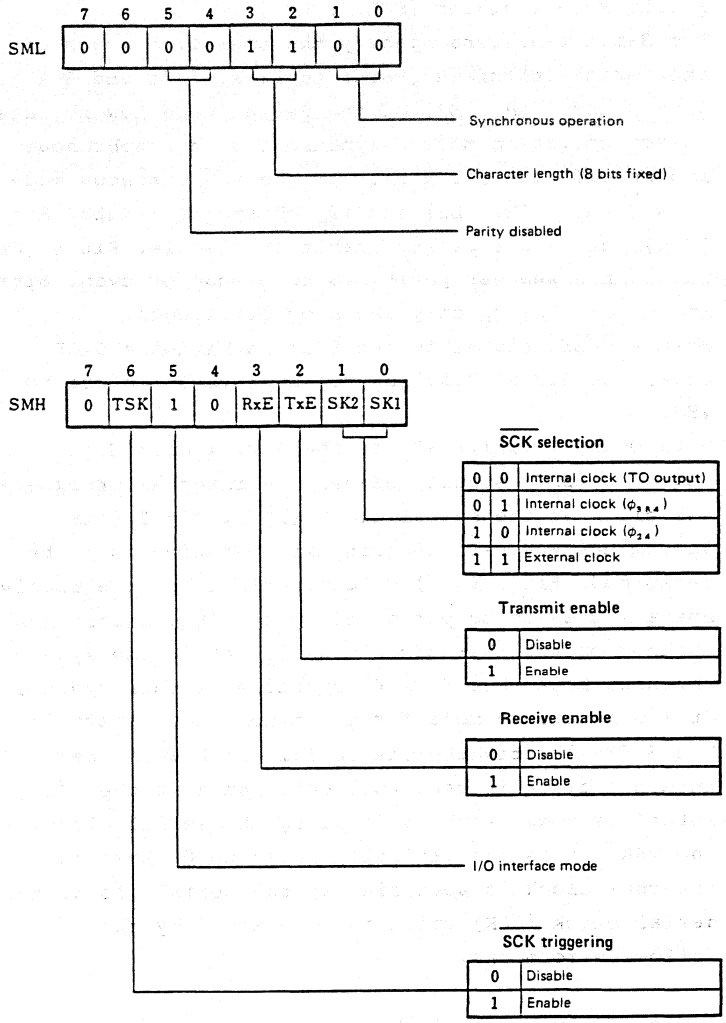


Fig. 3-25 Serial Mode Register Format in I/O Interface Mode

(4) Serial Mode register (SML, SMH)

Two 8-bit registers specify the operation modes of the serial interface (refer to Figs. 3-26 and 3-27). Bits 0 and 1 (B1, B2) of the Serial Mode Low register select operation mode: asynchronous or synchronous and specify the clock rate in the asynchronous mode. Bits 2 and 3 (L1, L2) specify character length. Bit 4 (PEN) determines parity enable or disable. Bit 5 (EP) determines whether parity is to be odd or even. Bits 6 and 7 (S1, S2) specify the stop bit length.

When a $\overline{\text{RESET}}$ signal is input or in hardware STOP mode, the Serial Mode Low register (SML) is set to 48H.

Bits 0 and 1 (SK1, SK2) of the Serial Mode High register (SMH) specify either the internal or external clock as the serial clock ($\overline{\text{SCK}}$). Bit 2 (TxE) determines whether the transmit operation is to be performed. Bit 3 (RxE) determines whether the receive operation is to be performed. Bit 4 (SE) determines whether or not the search mode is set in the synchronous mode. Bit 5 (IOE) specifies either synchronous or I/O interface mode for the synchronous operation. Bit 6 (TSK) activates the serial clock when the internal clock is used to receive data in the I/O interface mode. After activating the serial clock, the TSK bit is automatically reset to 0. When the internal clock is specified as the serial clock, the serial clock ($\overline{\text{SCK}}$) value is determined by the following formulas.

- For ϕ24 internal clock
  $\overline{\text{SCK}}$ = fXTAL/24
- For ϕ384 internal clock
  $\overline{\text{SCK}}$ = fXTAL/384

° For TO output internal clock

. When the timer input clock is ϕ12:

$\overline{SCK}$ = fXTAL/24xC

. When the timer internal clock is ϕ384:

$\overline{SCK}$ = fXTAL/768xC

. When timer F/F input is ϕ3:

$\overline{SCK}$ = fXTAL/6

Where fXTAL is the oscillation frequency, $\overline{SCK}$ is the serial clock frequency, and C is the number of counts of the timer. If the Timer F/F input is ϕ3, the internal clock (TO output) can be used only in the asynchronous mode and the clock rate is either 16 or 64.

When a $\overline{RESET}$ signal is input or when in the hardware STOP mode, the Serial Mode High register (SMH) is reset to 00H.

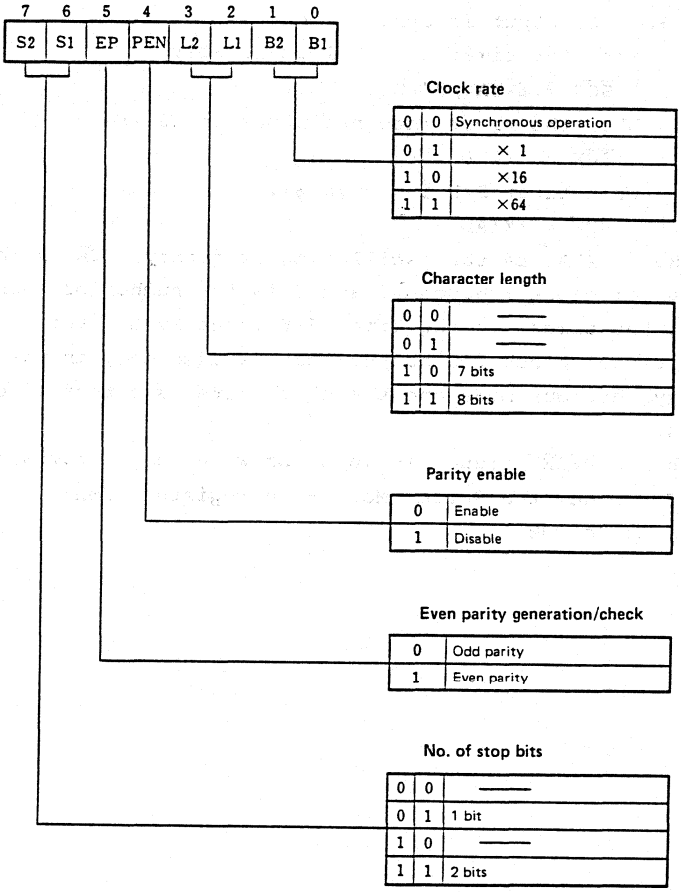


Fig. 3-26 Serial Mode Low Register (SML) Format

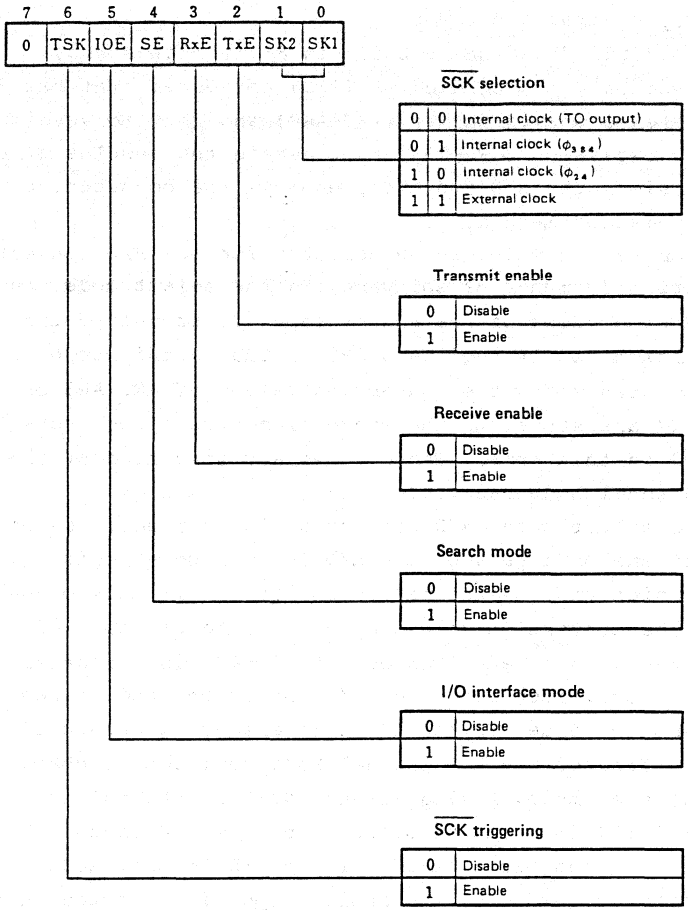


Fig. 3-27 Serial Mode High Register (SMH) Format

3.9 Analog/Digital Converter

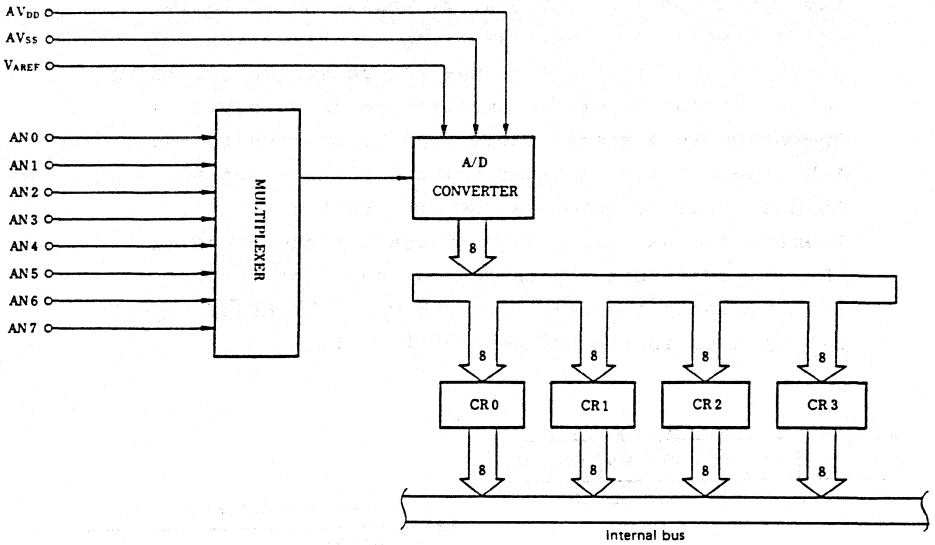
The μPD78C11/78C10 has a built-in 8-bit high-speed, high-accuracy analog/digital (A/D) converter that has 8 multiplexed analog inputs (AN7-AN0) and four Conversion Result registers (CR0-CR3) that retain the results of the conversion. The A/D converter employs the consecutive approximation method.

Scan or select mode can be selected for the A/D converter operation by means of software. In the select mode, the conversion value of one analog input is stored to the conversion result registers CR0 to CR3 in this order. In the scan mode, the conversion values of AN0-AN3 or AN4-AN7 are stored to the conversion result registers CR0 to CR3 in this order. These modes are specified by the A/D Channel Mode register.

In the select mode, A/D conversion is started by selecting one of analog inputs by the A/D Channel Mode register. The conversion value is stored to CR0 to CR3 in this order. When the conversion values are stored to all four CR registers, an internal interrupt (INTAD) is generated. The A/D converter continues A/D conversion and stores the conversion values to the CR registers CR0 to CR3 until the setting of the A/D Channel Mode register is changed.

In the scan mode, Analog inputs AN0-AN3 (ANI2=0) or AN4-AN7 (ANI2=1) can be selected by the A/D Channel Mode register. When bit 3 (ANI2) of the A/D Channel Mode register is set to 0, the analog input is selected in the order of AN0 → AN1 → AN2 → AN3 → AN0 →, and A/D conversion value of each input is stored to CR registers in the order of CR0 → CR1 → CR2 → CR3 → CR0 →. When ANI2 of the A/D Channel Mode register is set to 1, the analog input is selected in the order of AN4 → AN5 → AN6 → AN7 → AN4 →, and A/D conversion value of each input is stored to CR registers in the order of CR0 → CR1 → CR2 → CR3 → CR0 →. In the same way as the select mode, an internal interrupt (INTAD) is generated when the conversion values are stored to all four CR registers. Above operation is repeated

the setting of the A/D Channel Mode register is changed. An internal interrupt is inhibited by setting bit 0 (MKAD) of the interrupt mask register (MKH) to 1.



Note: Connect a capacitor to analog input and reference voltage input pins to prevent malfunction due to noise.

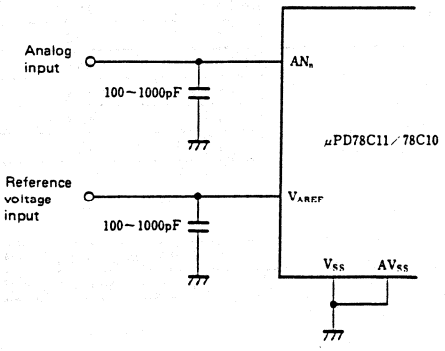


Fig. 3-28 A/D Converter Block Diagram

(1) A/D Channel Mode register (ANM)

This is an 8-bit register which controls operation modes of the A/D converter.

Bit 0 (MS) of the A/D Channel Mode register specifies the operation mode. Bits 1, 2, and 3 (ANI0, ANI1, ANI2) specify A/D conversion inputs. Bit 4 (FR) controls the operation of the A/D converter according to the variation of the oscillation frequency.

Operation mode specification can be written to the A/D Channel Mode register and the contents of the A/D Channel Mode register can be also read out. Therefore, the analog input which was the cause of the A/D interrupt generation can be identified.

This register is cleared to 00H when the $\overline{\text{RESET}}$ signal is input or in the hardware STOP mode.

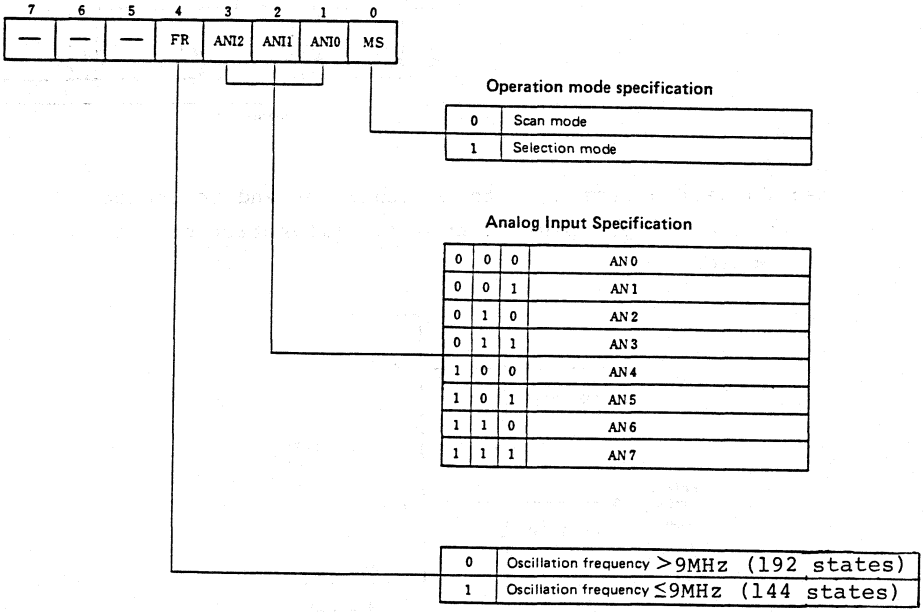


Fig. 3-29 A/D Channel Mode Register Format

3.10 Zero-cross Detection Circuit

INT1 and $\overline{\text{INT2}}$/TI (shared with PC3) pins can be used for zero-cross detection when specified by the zero-cross mode register.

The zero-cross detection circuit comprises self-biased high gain amplifiers. Its input is biased to the switching point and a small change in input is detected and converted to digital change.

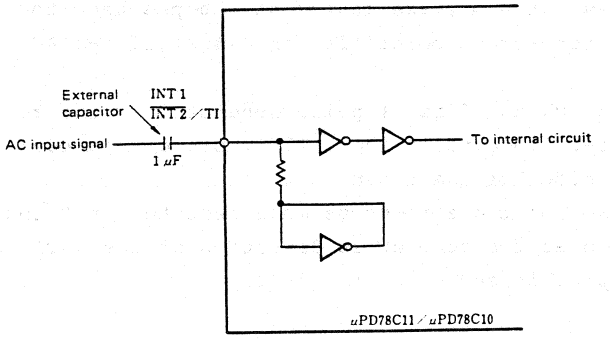


Fig. 3-30 Zero-cross Detection Circuit

The zero-cross detection circuit detects the negative-to-positive and positive-to-negative voltage change of the AC signal input through the external capacitor and generates digital pulses that change from 0 to 1 or vice versa at each zero-cross point.

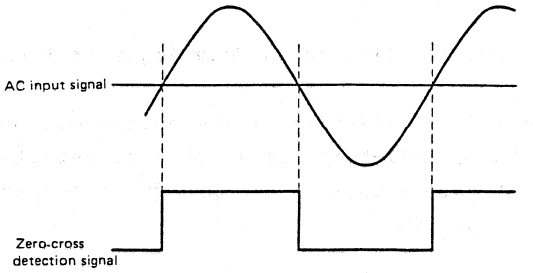


Fig. 3-31 Zero-cross Detection Signal

The digital pulse generated by the zero-cross detection circuit of the INT1 pin is sent to the interrupt control circuit and sets the INTF1 interrupt request flag at the negative-to-positive zero-cross point (rising edge) of the AC signal. Then the pulse starts the interrupt process if the INT1 interrupt has been enabled. The digital pulse generated by the zero-cross detection circuit of the $\overline{\text{INT2}}$/TI pin is sent to the interrupt control circuit and performs the same operation as described for INT1 pin. However,the process starts at the positive-to-negative zero-cross point (falling edge) of the AC signal.

Additionally, the digital pulse generated by the zero-cross detection circuit for $\overline{\text{INT2}}$/TI pin can be used as an input clock of the timer.

The format for the zero-cross mode register that controls the self-bias for zero-cross detection of the INT1 and $\overline{\text{INT2}}$/TI pins is shown in Fig. 3-32.

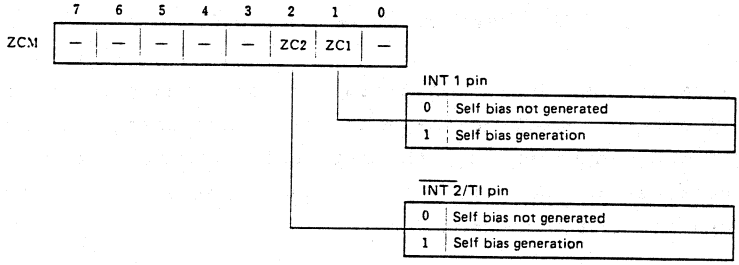


Fig. 3-32 Zero-cross Mode Register Format

When the ZC1 and ZC2 bits of the zero-cross mode register are set to 0, self-bias for zero-cross detection of each pin is not generated and each pin functions as normal digital input pin.

When the ZC1 and ZC2 bits are set to 1, self-bias is generated and the zero-cross of AC signal can be detected by connecting a capacitor to each pin. Each pin whose bit ZC1 or ZC2 bit is set to 1 can be directly driven without connecting external capacitor and functions as digital input pin. In this case, however, some input load current is required and a suitable external output driver circuit must be considered. Therefore, when each pin is used only for interrupt input, timer input, or port I/O, set the ZC1 and ZC2 bits of the zero-cross mode register to 0. When the $\overline{\text{RESET}}$ signal is input, both ZC1 and ZC2 bits are set to 1 and self-bias is generated.

Note: For the operating theory of the zero-cross detection circuit, in the zero-cross detection circuit, power supply current flows constantly even in the standby state (HALT, software/hardware STOP mode) unlike the other CMOS circuits. Therefore, the μPD78C11/78C10 draws a slightly greater current when the zero-cross detection circuit is activated (when self-bias is generated; ZCx=1) compared to when it is not activated. Note that the software/hardware STOP mode is considerably affected by this.

4. INTERRUPT CONTROL FUNCTIONS

There are three external and eight internal interrupt sources. These 11 interrupt sources are divided into six groups with six priority levels and six interrupt address. The priority and address of each interrupt source are as shown in the table below.

Priority	Interrupt address	Interrupt request	Internal/ external
1	4	$\overline{\text{NMI}}$ (Falling edge)	External
2	8	INTT0 (Coincidence signal from TIMER 0)	Internal
		INTT1 (Coincidence signal from TIMER 1)	
3	16	INT1 (Rising edge)	External
		$\overline{\text{INT2}}$ (Falling edge)	
4	24	INTE0 (Coincidence signal from timer/event counter)	Internal
		INTE1 (Coincidence signal from timer/event counter)	
5	32	INTEIN (Falling edge of CI or TO)	Internal
		INTAD (A/D converter interrupt)	
6	40	INTSR (Serial receive interrupt)	Internal
		INTST (Serial transmit interrupt)	

4.1 Interrupt Control Circuit

The interrupt control circuit consists of the Request register, Mask register, Priority Control, Test Control, Interrupt Enable F/F, and Test Flag register (refer to Fig. 4-1).

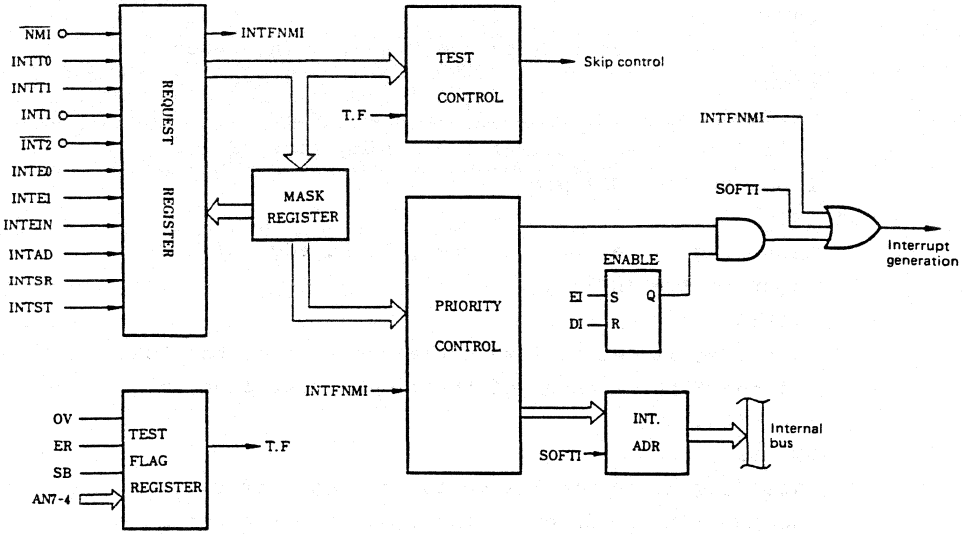


Fig. 4-1 Interrupt Control Circuit Block Diagram

(1) Request register

The request register consists of 11 types of interupt request flags each set by an interrupt request signal. The interrupt request flags are reset when an interrup request is accepted or when a skip instruction (SKIT or SKNIT) is executed. When the $\overline{\text{RESET}}$ signal is input, all flags are reset. These 11 types of interrupt flags are:

° INTFNMI

This flag is set at the falling edge of input to the $\overline{\text{NMI}}$ pin. Unlike other interrupt request flags, this flag cannot be tested by a skip instruction.

° INTFT0

This flag is set to 1 by the TIMER0 coincidence signal.

° INTFT1
This flag is set to 1 by the TIMER1 coincidence signal.

° INTF1
This flag is set to 1 at the rising edge of the input to the INT1 pin.

° INTF2
This flag is set to 1 at the falling edge of the input to the $\overline{\text{INT2}}$ pin.

° INTFE0
This flag is set to 1 when the contents of the ECNT and the ETM0 register of the timer/event counter coincide.

° INTFE1
This flag is set to 1 when the contents of the ECNT and the ETM1 register of the timer/event counter coincide.

° ITFEIN
This flag is set to 1 at the falling edge of the timer output (TO) or CI input of the timer/event counter.

° INTFAD
This flag is set to 1 when the A/D conversion results are transferred to four registers CR0 to CR3.

° INTFSR
This flag is set to 1 when the receive buffer register of the serial interface becomes full.

° INTFST
This flag is set to 1 when the transmit buffer register of the serial interface becomes full.

(2) Mask register

Except for the nonmaskable interrupt ($\overline{\text{NMI}}$), a 10-bit mask register is provided corresponding to each interrupt source. Each bit can be set to 1 or 0 independently by an instruction. Each interrupt source is masked (inhibited) when the corresponding bit in

the mask register becomes 1 and acknowledged when the corresponding bit in the mask register becomes 0. When the $\overline{\text{RESET}}$ signal is input, all bits in the mask register are set to 1 and all interrupt requests except the nonmaskable interrupts are masked. All bits in the mask register are also set to 1 in the hardware STOP mode.

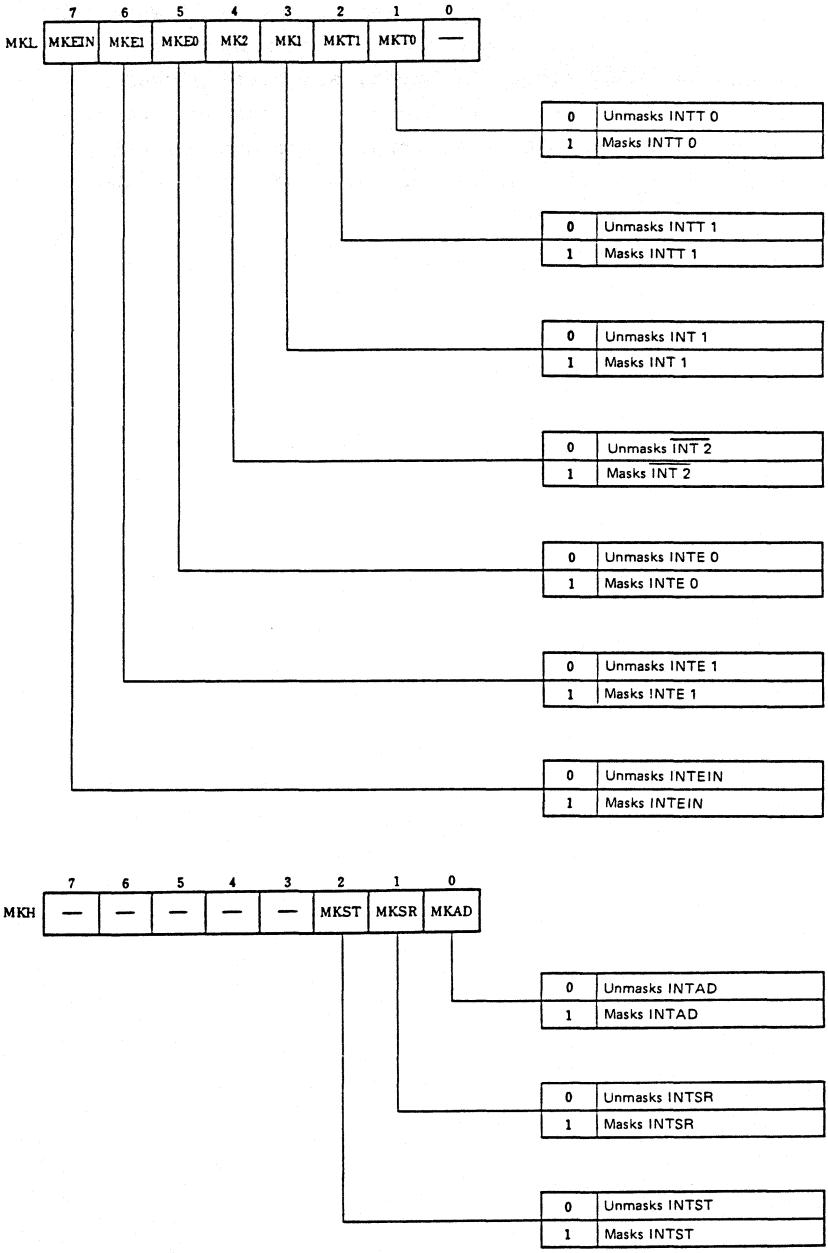


Fig. 4-2 Mask Register (MKL, MKH) Format

(3) Priority Control Circuit

This circuit controls the previously mentioned six levels of priority. When two or more interrupt request flags are simultaneouly set, the interrupt having the highest priority is accepted.

(4) Test Control Circuit

This circuit functions when executing a skip instruction (SKIT, SKNIT) to test an interrupt request flag (except INTFNMI) corresponding to each interrupt source, state of the $\overline{\text{NMI}}$ pin, and test flag.

(5) Interrupt Enable F/F (IE F/F)

This is a flip-flop set by the EI instruction and reset by the DI instruction. If an interrupt is accepted, this will be reset.

Also, inputting the $\overline{\text{RESET}}$ signal resets this flip-flop. When it is set, interrupts are to be enabled. If it is reset, interrupts are to be disabled.

(6) Test Flag register

The test flag register consists of seven types of test flags that are tested or reset by executing an skip instruction (SKIT, SKNIT).

- OV

  This flag is set to 1 when the ECNT of the timer/event counter overflows.

- ER

  This flag is set to 1 when a parity, framing, or overrun error occurs during serial data operation.

- SB

  This flag is set to 1 when the V_{DD} voltage goes up from below the rated low level to over the rated high level.

- AN7-AN4

  These flags are set to 1 at the falling edge of AN7-AN4 pins.

4.2 Nonmaskable Interrupt operations

Regardless of the status of the EI/DI, a nonmaskable interrupt is accepted in the following sequence when the interrupt request flag (INTFNMI) is set at the falling edge input to the $\overline{\text{NMI}}$ pin (refer to Fig. 4-3).

- The INTFNMI flag is checked at the final timing of each instruction. If the INTFNMI flag is set, the nonmaskable interrupt is accepted and the INTFNMI flag is then reset.
- When the nonmaskable interrupt is accepted, the IE F/F is reset and all interrupts except the nonmaskable interrupt and the SOFTI instruction will be inhibited (DI state).
- The PSW, PC upper byte, and PC lower byte will be saved to the stack memory in that order.
- Jumps to the interrupt address (0004H)

These interrupt operations are performed automatically in 16 states. The interrupt request flag (INTFNMI) cannot be tested by a skip instruction, however, the status of the $\overline{\text{NMI}}$ pin can be tested by a skip instruction (SKIT NMI, SKNIT NMI). Therefore, noises of relatively long duration can be eliminated by testing the status of the $\overline{\text{NMI}}$ pin several times in the nonmaskable interrupt service routine. Testing by skip instruction has no effect on the status of the $\overline{\text{NMI}}$ pin.

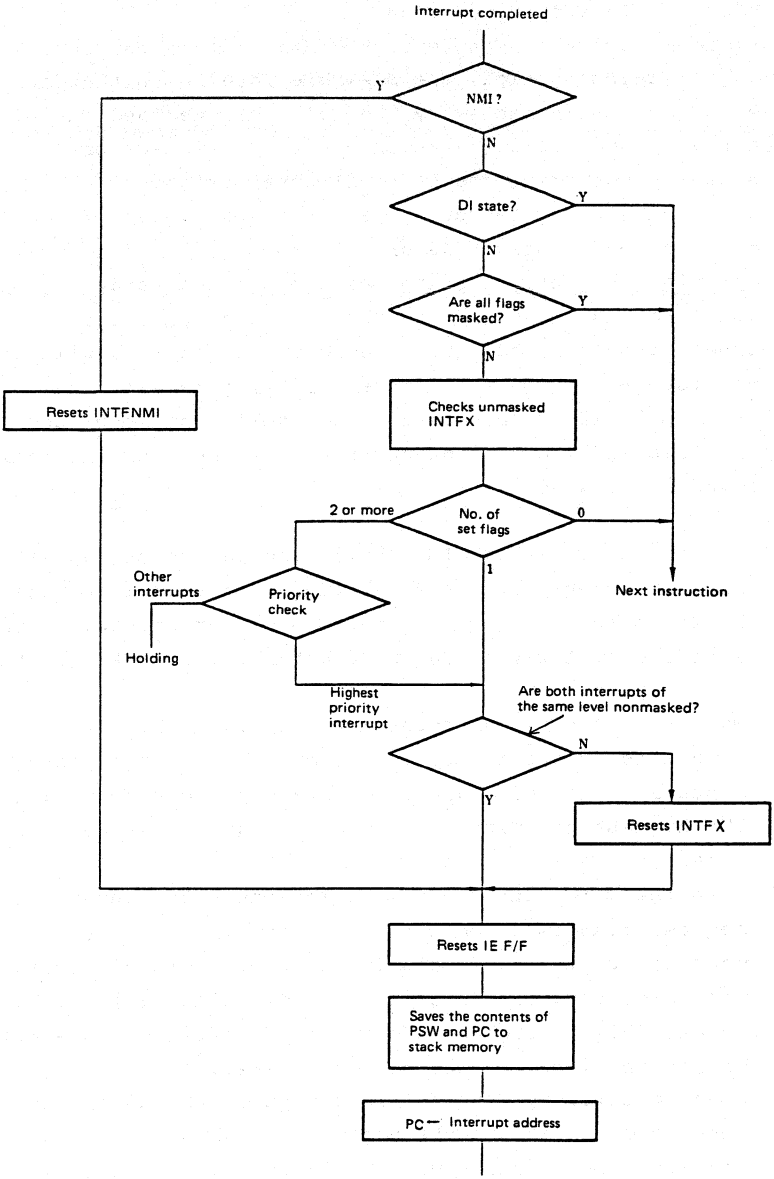


Fig. 4-3 Interrupt Operating Procedure

4.3 Maskable Interrupt Function

All interrupt requests (except nonmaskable interrupts and SOFTI instruction) are maskable interrupts that can be independently masked by the mask register and can be enabled or disabled (in which the IE F/F is set or reset) by executing the EI or DI instruction.
For an external interrupt, the interrupt request flag is set when the signal is determined to be a correct interrupt signal by checking the duration of the active level input. For an internal interrupt, if the interrupt request is generated, the interrupt request flag will be immediately set. Once the interrupt flag is set, the interrupt will be processed in the following sequence regardless of whether interrupt was external or internal (refer to Fig. 4-3).

- If the status is EI (IE F/F=1), the interrupt request flag is checked at the final timing of each instruction. If it is found to be set, the process enters the interrupt cycle. However, an interrupt request masked by the mask register will not be checked.
- When two or more interrupt request flags are set at the same time, the priority levels of the competing interrupts are checked. Then, the interrupt request having the highest priority will be accepted and the remaining interrupt will be held.
- When the interrupt request is accepted, the interrupt request flag is automatically reset. If two interrupt requests of the same priority have been masked by the mask register, the interrupt request flag will not be reset. This is because the two types of interrupts will be distinguished by the software later.
- When an interrupt request is accepted, the IE F/F is reset and all interrupts except nonmaskable interrupts and SOFTI instruction will be inhibited (DI state).
- The PSW, PC upper byte, then PC lower byte will be saved to the stack memory in that order.
- The routine jumps to the interrupt address.

These interrupt operations are performed automatically in 16 states. The interrupt request being held will be accepted when the interrupt operation is enabled by executing the IE instruction and if no other interrupt request of higher priority has been generated.

For the maskable interrupt (except INTEIN), two interrupt requests have the same priority and the same interrupt addresses. Any of releasing the masking of both interrupt requests, releasing the masking of one of two interrupt requests, or masking both interrupt requests can be selected by setting the mask register.

(1) When both interrupt requests are unmasked
Set both bits corresponding to the two types of interrupt requests in the mask register to 0. In this case, the logical sum of these two interrupt request flags becomes an interrupt request.
Although an interrupt request generated by setting one or both interrupt request flags of the same priority is accepted and is jumped to the interrupt address according to the interrupt sequence, the interrupt request flag will not be reset.
Therefore, which interrupt request is generated can be determined and the interrupt request flag can be reset by executing the skip instruction at the beginning of the interrupt service routine to test the interrupt request flag.

(2) When one of two interrupt request is unmasked
When there are two types of interrupt requests with the same priority, set the bit corresponding to the interrupt request to be unmasked to 0 and set another bit to 1. In this case, setting the unmasked interrupt request flag generates an interrupt request and the interrupt request flag will be automatically reset when the interrupt request is accepted according to the interrupt sequence.

When the masked interrupt request flag is set, that interrupt request will be held. The pending interrupt will be accepted if the interrupt operation is enabled when the interrupt is unmasked and if no other interrupt request with higher priority has been generated.

(3) When both interrupt requests are masked
Set both bits corresponding to the two types of interrupt requests in the mask register to 1. In this case, even though the interrupt request flag is set, the interrupt request will not be accepted, but held. The pending interrupt will be acknowledged if the interrupt operation is enabled when the interrupt request is unmasked and if no other interrupt request with higher priority has been generated.

4.4 Interrupt Operation by SOFTI Instruction

When the SOFTI instruction is executed, the program jumps unconditionally to the interrupt address (0060H). The SOFTI instruction interrupt is neither affected by, nor does it affect, the IE F/F.
The interrupt caused by the SOFTI instruction is processed in the following sequence.

- Bytes are saved to the stack memory in the order of PSW, PC upper byte, and PC lower byte.
- The program jumps to the interrupt address (0060H).

Note: The SOFTI instruction will not be skipped but executed even though the skip condition is satisfied by an instruction (arthmetic, logical operation, increment/decrement, shift, skip, or RETS instruction) that appear immediately before the SOFTI instruction. Being set to 1, the SK flag of the PSW will be saved to the stack area by executing the SOFTI instruction. Therefore, when returned from the SOFTI process routine, the SK flag of the PSW

remains set and the instruction immediately after the SOFTI instruction will be skipped.
In the SOFTI instruction of the μPD78C11/78C10, the contents of the address to be saved to the stack memory is the starting address of the next instruction. This point differs from that of the μCOM-87.

Note: To eliminate noise signals on external interrupt lines, 14 states are required by the μPD78C10/C11.

5. STANDBY FUNCTION

Three standby modes are provided for the μPD78C11/78C10 to reduce power consumption during program wait period: HALT mode, software STOP mode, and hardware STOP mode.

5.1 HALT mode

When the HALT instruction is executed, the CPU enters the HALT mode at any time unless the unmasked interrupt request flag is set. In the HALT mode, the CPU clock is stopped and program execution is halted. However, the contents of all registers and the internal RAM will be retained. Even in the HALT mode, the timer, timer/event counter, serial interface, A/D converter, and interrupt control circuit operate normally.

In the HALT mode, the status of the output pins of the μPD78C11/78C10 will be as shown in Table 5-1.

Table 5-1 Status of Output Pins

Output pin	Single chip*	External expansion
PA7-0	Data holding	Data holding
PB7-0	Data holding	Data holding
PC7-0	Data holding	Data holding
PD7-0	Data holding	High inpedance
PF7-0	Data holding	Next address holding** Data holding***
$\overline{WR}$, $\overline{RD}$	High level	High level
ALE	High level	High level

* When using μPD78C11

** Pin to output address

*** Pin to output port data

Note: Because an interrupt request flag is used for releasing the HALT mode, if even one interrupt request flag for unmasked interrupt is set, the CPU will not enter the HALT mode even if the HALT instruction is executed. To set the HALT mode in

the place where an interrupt request flag might have been set (pending interrupt), the pending interrupt must be processed first, or the interrupt request flag must be reset by executing a skip instruction, or all interrupts that are not used for releasing the HALT mode must be masked.

5.2 Software STOP Mode

When the STOP instruction is executed, the CPU enters the STOP mode unless the unmasked external interrupt request flag is set. In the software STOP mode, all clocks are stopped. In the software STOP mode, program execution is stopped and the contents of the internal RAM is retained (timer UPCOUNTER is cleared to 00H), and only the $\overline{\text{NMI}}$ and $\overline{\text{RESET}}$ signals remain effective. However, all other functions are stopped.

In the same way as in the HALT mode, in the software STOP mode, the status of output pins of the μPD78C11/78C10 will be as shown in Table 5-1.

Note:

1. The internal interrupt should be masked before executing the STOP instruction to prevent erroneous operation caused by an internal interrupt generated during the oscillation stabilizing period when the software STOP mode is released.
2. When the software STOP mode is released by setting the interrupt request flag for nonmaskable interrupt, TIMER1 coincidence signal is used to start the CPU operation so that the oscillation stabilizing time can be obtained. For this, before executing the STOP instruction, taking into consideration the stabilizing time, the number of counts should be set to TIMER REG and the Timer Mode register should be set for timer operation mode.

5.3 Hardware STOP Mode

The CPU enters the hardware STOP mode whenever the $\overline{\text{STOP}}$ signal goes from high to low. In the hardware STOP mode, all clocks are stopped. When the CPU enters the hardware STOP mode, program execution is stopped and the contents of the RAM is retained. Then, only the $\overline{\text{STOP}}$ signal that is to be used for releasing the hardware STOP mode remains effective. However, all other functions are stopped and become stop condition.

During the hardware STOP mode, all output pins of the μPD78C11/78C10 become high impedance state.

5.4 Low Power-Supply Voltage Data Retain Mode

Low power-supply voltage data retain mode can be set by simply lowering V_{DD} voltage down to 2.0V after the software/hardware STOP mode is set and the data in the RAM can be retained with lower power consumption than that in the software/hardware STOP mode.

Note: The software/hardware STOP mode should not be released in the low power supply voltage data retain mode. V_{DD} voltage must be raised to the normal V_{DD} voltage before releasing the software/hardware STOP mode.

5.5 Releasing HALT Mode

(1) Release by $\overline{\text{RESET}}$ signal

The HALT mode is released when the $\overline{\text{RESET}}$ signal goes from high to low during the HALT mode and the CPU enters the reset state. When the $\overline{\text{RESET}}$ signal returns to high, the CPU starts executing the program from address 0. Even when the $\overline{\text{RESET}}$ signal is input, the contents of the RAM are retained. However, the contents of other registers become undefined.

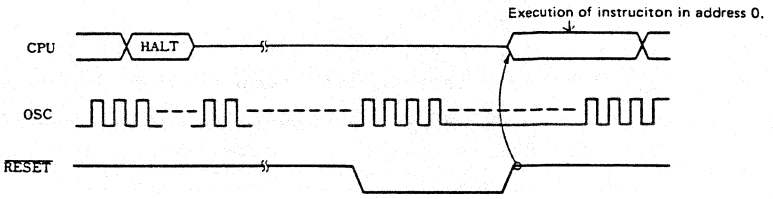


Fig. 5-1 HALT Mode Release Timing
(at $\overline{\text{RESET}}$ signal input)

(2) Release by interrupt request flag

The HALT mode is released when one or more interrupt request flags are set by the generation of the non-maskable interrupt ($\overline{\text{NMI}}$) or unmasked interrupts of those ten maskable interrupts (INTT0, INTT1, INT1, $\overline{\text{INT2}}$, INTE0, INTE1, INTEIN, INTAD, INTST, and INTSR). When the HALT mode is released by a nonmaskable interrupt, the program jumps to the interrupt address (004H) without executing the instruction placed after the HLT instruction regardless of whether the status is EI or DI.

When the HALT mode is released by a maskable interrupt, the operation following the release of the HALT mode differs depending on whether the status is EI or DI.

(a) When the status is EI

Jumps to the respective interrupt address without executing the instruction placed after the HLT instruction.

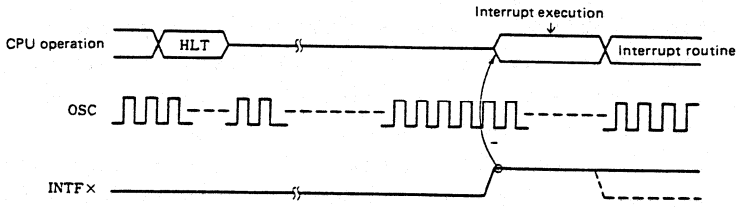


Fig. 5-2 HALT Mode Release Timing (in EI state)

(b) When the status is DI

The execution starts from the instruction placed after the HLT instruction (jump to the interrupt address is not done). The interrupt request flag used for releasing the HALT mode remains set this time, therefore, a skip instruction should be executed to reset the interrupt request flag as necessary.

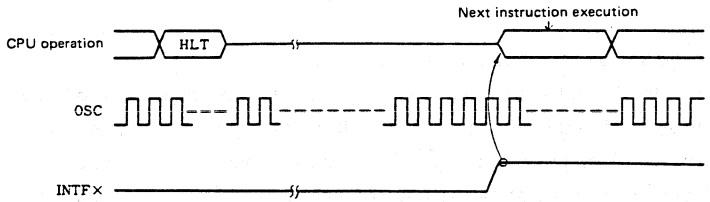


Fig. 5-3 HALT Mode Release Timing (in DI state)

5.6 Releasing Software STOP Mode

(1) Release by $\overline{\text{RESET}}$ signal

The software STOP mode is released when the $\overline{\text{RESET}}$ signal goes from high to low during the software STOP mode and enters the reset state. The clock oscillation is started simultaneously. When the $\overline{\text{RESET}}$ signal is set to high after the oscillation is stabilized, the CPU starts executing the program from address 0. The clock oscillation starts immediately after the $\overline{\text{RESET}}$ signal goes from high to low. However, oscillation stabilizing time is required, namely, the duration until the oscillation stabilizes. Therefore, the low level width of the $\overline{\text{RESET}}$ signal must be longer than the oscillation stabilizing time.

Even when the $\overline{\text{RESET}}$ signal is input, the contents of the RAM are retained, however, the contents of other registers become undefined.

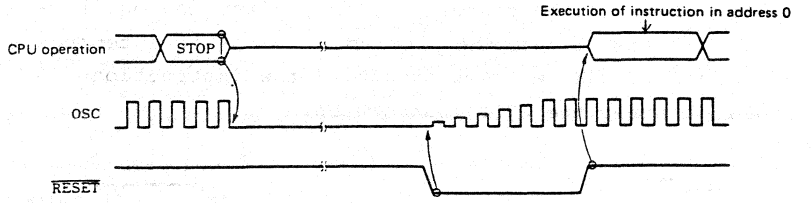


Fig. 5-4 Software STOP Mode Release Timing (at $\overline{\text{RESET}}$ signal input)

When the software STOP mode is released by the $\overline{\text{RESET}}$ signal, the program execution starts from address 0 in the same way as normal power-on reset. Thus, the SB (standby) flag distinguishes this from normal power-on reset. When the V_{DD} voltage crosses the rated voltage going from low to high level, the SB flag is set to 1 and execution of a skip instruction resets the SB flag to 0. Therefore, testing the SB flag by executing a skip instruction after the $\overline{\text{RESET}}$ input provides a means to distinguish between recovery that has taken place after power on and the releasing the software STOP mode.

(2) Release by interrupt request flag

When the nonmaskable interrupt request flag is set during the software STOP mode, the software STOP mode is released and the clock oscillation starts simultaneously. When the clock oscillation starts, the timer UPCOUNTER starts countup from 00H according to the setting made before executing the STOP instruction. The CPU function is started by the coincidence signal (wait time for which the oscillation stabilizing time is considered) from the UPCOUNTER of TIMER1. However, in this case, the interrupt request flag is not set by the coincidence signal from the UPCOUNTER. Additionally, the Timer Mode register of the UPCOUNTER is set to FFH and the timer stops operating.

Regardless of the status of EI/DI, after the oscillation is stabilized, the program jumps to interrupt address (0004H) without executing the instruction placed after the STOP instruction.

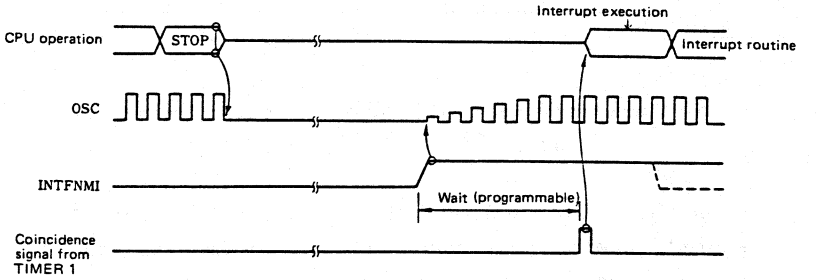


Fig. 5-5 Software STOP Mode Release Timing

5.7 Releasing hardware STOP mode

When the $\overline{\text{STOP}}$ signal goes from low to high during the hardware STOP mode, the hardware STOP mode is released and the clock oscillation is simultaneouly started. Then, after the wait time (approximately 65ms at 12MHz) that takes into account the oscillation stabilizing time has elapsed, the CPU starts executing the program from address 0 (refer to Fig. 5-7).

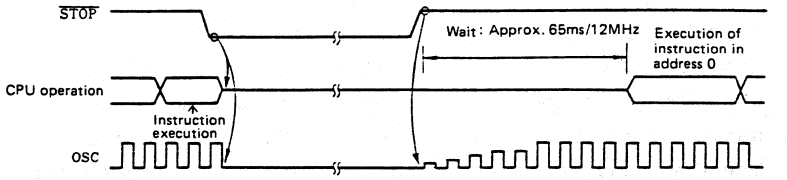


Fig. 5-7 Releasing Hardware STOP Mode

The hardware STOP mode is not released even if the $\overline{\text{RESET}}$ signal goes from high to low. If the $\overline{\text{STOP}}$ signal goes from low to high when the $\overline{\text{RESET}}$ signal is at low, the hardware STOP mode is released and the clock starts operating. If the $\overline{\text{RESET}}$ signal returns to high from low without oscillation stabilization time, the CPU starts executing the program from address 0 (refer to Fig. 5-8). Even if the $\overline{\text{RESET}}$ signal goes from high to low immediately after the hardware STOP mode is released (when the $\overline{\text{STOP}}$ signal goes from low to high), the program execution starts when the RESET signal goes from low to high (refer to Fig. 5-9). Accordingly RESET signal must be returned to high level considering oscillation stabilizing time.

Therefore, even if the hardware STOP mode is released by the $\overline{\text{RESET}}$ signal input, the contents of the RAM are retained. However, the contents of other registers become undefined.

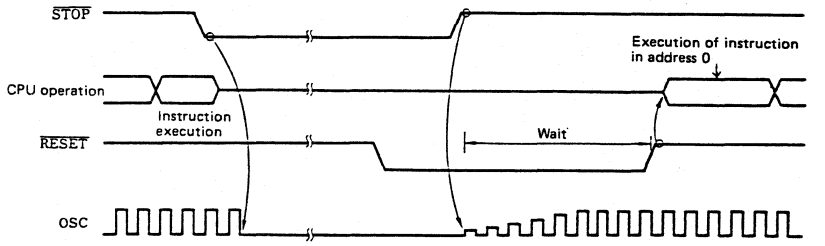


Fig. 5-8 Hardware STOP Mode Release Timing

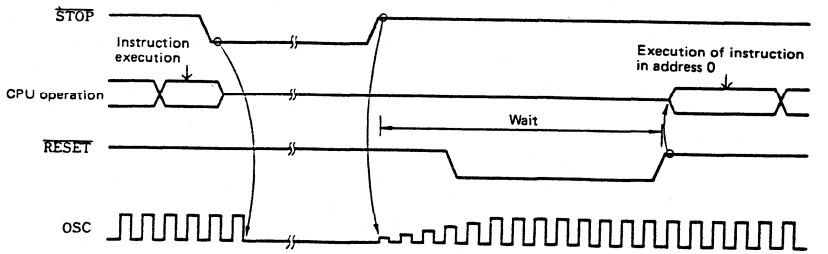


Fig. 5-9 Hardware STOP Mpde Release Timing

In the same way as when the software STOP mode is released by the $\overline{\text{RESET}}$ signal, when the hardware STOP mode is released, testing the SB flag by executing the skip instruction provides a means to distinguish between recovery that has taken place after power on and releasing the hardware STOP mode.

STANDBY CONDITIONS μPD78C10/C11

PARAMETER		HALT MODE	SOFTWARE STOP MODE	HARDWARE STOP MODE
OSCILLATION / SYSTEM CLOCK		YES	NO	NO
CPU CLOCK		NO	NO	NO
INTERNAL RAM BACK-UP		YES	YES	YES
REGISTER		YES	YES	NO
TIMER, SERIAL, EVENT COUNTER, INTERRUPT } OPERATION		YES	NO ($\overline{NMI}$, $\overline{RESET}$ ARE EFFECTIVE)	NO
SINGLE-CHIP	PA, PB, PC	DATA RETENTION	DATA RETENTION	HIGH IMPEDANCE
	PD	DATA RETENTION	DATA RETENTION	HIGH IMPEDANCE
	PF	DATA RETENTION	DATA RETENTION	HIGH IMPEDANCE
	$\overline{WR}$, $\overline{RD}$, ALE	HIGH LEVEL	HIGH LEVEL	HIGH IMPEDANCE
USE EXTERNAL MEMORY	PA, PB, PC	DATA RETENTION	DATA RETENTION	HIGH IMPEDANCE
	PD	HIGH IMPEDANCE	HIGH IMPEDANCE	HIGH IMPEDANCE
	PF	HOLD NEXT ADDRESS (1) HOLD NEXT DATA (2)	HOLD NEXT ADDRESS HOLD NEXT DATA	HIGH IMPEDANCE
	$\overline{WR}$, $\overline{RD}$, ALE	HIGH LEVEL	HIGH LEVEL	HIGH IMPEDANCE

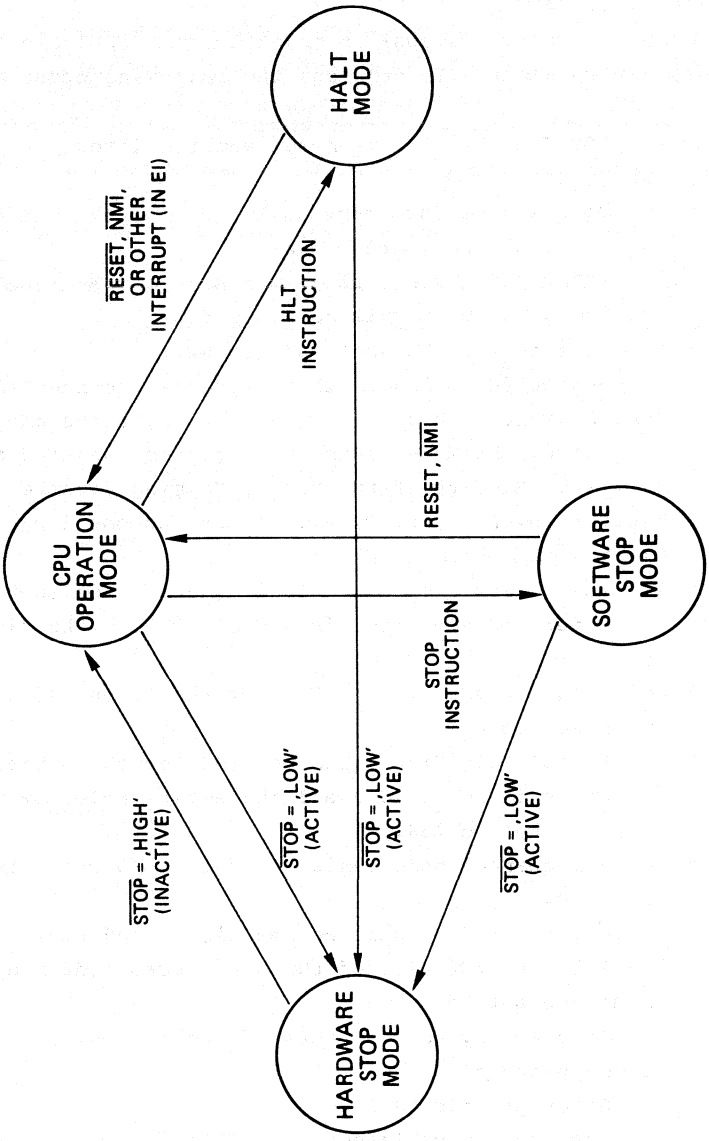
NOTE 1. ADDRESS OUTPUT PINS
2. PORT DATA OUTPUT PINS

STANDBY CONDITIONS μPD78C10/C11 (continued)

PARAMETER	HALT MODE	STOP MODE	
		BY INSTRUCTION	BY STOP (INPUT)
MEMORY MAPPING REGISTER (MM)	HOLD	HOLD	BIT0, 1, 2 : 0 BIT3 : HOLD
PROGRAM COUNTER (PC)	HOLD	HOLD	0000H
STACK POINTER (SP)	HOLD	HOLD	UNKNOWN
GENERAL REGISTER	HOLD	HOLD	UNKNOWN
PROGRAM STATUS WORD (PSW)	HOLD	HOLD	00H
TIMER UPCOUNTER	RUN	00H	00H
TIMER MODE REGISTER (TMM)	HOLD	*1	FFH
TM0, TM1	HOLD	HOLD	FFH
INTERRUPT CONTROL CIRCUIT	RUN	STOP	STOP
PENDING INTERRUPTS (INTFX)	HOLD	RESET	RESET
INTERRUPT MASK REGISTER	HOLD	HOLD	FFH
NMI INPUT	ACTIVE	ACTIVE	INACTIVE
INT1, INT2	ACTIVE	INACTIVE	INACTIVE
TIMER/EVENT COUNTER CIRCUIT	RUN	STOP	STOP
MODE REGISTERS (EOM, ETMM)	HOLD	HOLD	00H
ETM0, ETM1	RUN	HOLD	UNKNOWN
ECNT	RUN	UNKNOWN	0000H
SERIAL INTERFACE CIRCUIT	RUN	STOP	STOP
MODE REGISTERS (SMH, SML)	HOLD	HOLD	SMH : 00H SML : 48H
RxB, TxB	RUN	UNKNOWN	UNKNOWN
SERIAL REGISTERS	RUN	UNKNOWN	FFH
A/D CONVERTER CIRCUIT	RUN	STOP	STOP
MODE REGISTER (ANM)	HOLD	HOLD	00H
CR0, CR1, CR2, CR3	ACTIVE	UNKNOWN	UNKNOWN
STANDBY FLAG (SB)	HOLD	HOLD	HOLD
STOP INPUT	ACTIVE	ACTIVE	ACTIVE
ZERO CROSS MODE REG. (ZCM)	HOLD	HOLD	SET (1)
TEST FLAGS (EXCEPT SB)	ACTIVE	RESET (0)	RESET (0)
RESET INPUT	ACTIVE	ACTIVE	ACTIVE

NOTE *1: DURING STOP MODE : 011000xxB
AFTER RELEASE : 11111111B

STATE TRANSITION DIAGRAM μPD78C10/C11



6. RESET OPERATION

When a low level signal is input to the $\overline{\text{RESET}}$ pin, the system reset is effected, and the following occur.

- INTERRUPT ENABLE F/F is reset and the interrupt is inhibited.
- All bits in the interrupt mask register are set to 1 and all interrupts are masked.
- The interrupt request flag is reset to 0 and the all pending interrupts will be cleared.
- All bits of the PSW are reset to 0.
- Address 0000H is loaded to the program counter (PC).
- Mode A, Mode B, Mode C, and Mode F registers are set to FFH. Also, the Mode Control C register, and the MM0, MM1, and MM2 bits of the Memory Mapping register are reset to 0. A, B, C, D, and F ports become input ports (output high impedance).
- All test flags except the SB flag are reset to 0.
- The timer mode register is set to FFH and the timer F/F is reset.
- Mode registers (ETMM, EOM) of the timer/event counter are reset to 0.
- The Serial Mode High register (SMH) of the serial interface is reset to 0 and the Serial Mode Low register (SML) is set to 48H.
- The A/D channel mode register of the A/D converter is set to 0.
- $\overline{\text{WR}}$, $\overline{\text{RD}}$, and ALE signals become high impedance.
- The ZC1, and ZC2 bits of the Zero-cross Mode register (ZCM) are set to 1.
- The data memory and contents of the following registers become undefined.
  - Stack pointer (SP)
  - Expanded accumulators (EA, EA'), accumulators (A, A')
  - General-purpose registers (B, C, D, E, H, L, B', C', D', E', H', L')
  - Output latches of each port
  - TIMER REG0 and REG1 (TM0, TM1)

- TIMER/EVENT COUNTER REG0 and REG1 (ETM0, ETM1)
- RAE bit of the MEMORY MAPPING register
- The SB flag of the test flag

° Pins PD7-0, and PF7-0 of the uPD78C10 become high impedance output.

When the $\overline{\text{RESET}}$ input becomes high, the reset state is released, and the program execution starts from address 0000H; however, the contents of each register should be initialized or reinitialized in the program as necessary.

7. INSTRUCTION SET

7.1 Operand Expression Format/Description method

Expression format	Description method
r r1 r2	V, A, B, C, D, E, H, L EAH, EAL, B, C, D, E, H, L A, B, C
sr sr1 sr2 sr3 sr4	PA, PB, PC, PD, PF, MKH, MKL, ANM, SMH, SML, EOM, ETMM, TMM, MM, MCC, MA, MB, MC, MF, TXB, TM0, TM1, ZCM PA, PB, PC, PD, PF, MKH, MKL, ANM, SMH, EOM, TMM, RXB, CR0, CR1, CR2, CR3 PA, PB, PC, PD, PF, MKH, MKL, ANM, SMH, EOM, TMM ETM0, ETM1 ECNT, ECPT
rp rp1 rp2 rp3	SP, B, D, H V, B, D, H, EA SP, B, D, H, EA B, D, H
rpa rpa1 rpa2 rpa3	B, D, H, D+, H+, D−, H− B, D, H B, D, H, D+, H+, D−, H−, D+byte, H+A, H+B, H+EA, H+byte D, H, D++, H++, D+byte, H+A, H+B, H+EA, H+byte
wa	8 bit immediate data
word byte bit	16bit immediate data 8 bit immediate data 3 bit immediate data
f	CY, HC, Z
irf	NMI*, FT0, FT1, F1, F2, FE0, FE1, FEIN, FAD, FSR, FST, ER, OV, AN4, AN5, AN6, AN7, SB

* NMI can be also described as FNMI.

Note

1. sr～sr4(special register)

PA	: PORT A	ETMM	: TIMER/ EVENT COUNTER MODE
PB	: PORT B	EOM	: TIMER/ EVENT COUNTER OUTPUT MODE
PC	: PORT C	ANM	: A/D CHANNEL MODE
PD	: PORT D	CR0	: A/D CONVERSION
PF	: PORT F	1	RESULT 0−3
MA	: MODE A	CR3	
MB	: MODE B	TXB	: Tx BUFFER
MC	: MODE C	RXB	: Rx BUFFER
MCC	: MODE CONTROL C	SMH	: SERIAL MODE High
MF	: MODE F	SML	: SERIAL MODE Low
MM	: MEMORY MAPPING	MKH	: MASK High
TM0	: TIMER REG0	MKL	: MASK Low
TM1	: TIMER REG1	ZCM	: ZERO CROSS MODE
TMM	: TIMER MODE		
ETM0	: TIMER/ EVENT COUNTER REG0		
ETM1	: TIMER/ EVENT COUNTER REG1		
ECNT	: TIMER/ EVENT COUNTER UPCOUNTER		
ECPT	: TIMER/ EVENT COUNTER CAPTURE		

2. rp～rp3(register pair)

SP	: STACK POINTER
B	: BC
D	: DE
H	: HL
V	: VA
EA	: EXTENDED ACCUMULATOR

3. rpa～rpa3(rp addressing)

B	: (BC)
D	: (DE)
H	: (HL)
D+	: (DE)+
H+	: (HL)+
D−	: (DE)−
H−	: (HL)−
D++	: (DE)++
H++	: (HL)++
D+byte	: (DE+byte)
H+A	: (HL+A)
H+B	: (HL+B)
H+EA	: (HL+EA)
H+byte	: (HL+byte)

4. f(flag)

CY	: CARRY
HC	: HALF CARRY
Z	: ZERO

5. irf (interrupt flag)

NMI	: NMI INPUT
FT0	: INTFT0
FT1	: INTFT1
F1	: INTF1
F2	: INTF2
FE0	: INTFE0
FE1	: INTFE1
FEIN	: INTFEIN
FAD	: INTFAD
FSR	: INTFSR
FST	: INTFST
ER	: ERROR
OV	: OVERFLOW
AN4	: ANALOG INPUT4−7
1	
AN7	
SB	: STANDBY

7.2 Instruction Code Description

r

R_2	R_1	R_0	reg
0	0	0	V
0	0	1	A
0	1	0	B
0	1	1	C
1	0	0	D
1	0	1	E
1	1	0	H
1	1	1	L

r1

T_2	T_1	T_0	reg
0	0	0	EAH
0	0	1	EAL
0	1	0	B
0	1	1	C
1	0	0	D
1	0	1	E
1	1	0	H
1	1	1	L

rpa

A_3	A_2	A_1	A_0	addressing
0	0	0	0	——
0	0	0	1	(BC)
0	0	1	0	(DE)
0	0	1	1	(HL)
0	1	0	0	(DE)+
0	1	0	1	(HL)+
0	1	1	0	(DE)-
0	1	1	1	(HL)-
1	0	1	1	(DE+byte)
1	1	0	0	(HL+A)
1	1	0	1	(HL+B)
1	1	1	0	(HL+EA)
1	1	1	1	(HL+byte)

sr

S_5	S_4	S_3	S_2	S_1	S_0	Special-reg
0	0	0	0	0	0	PA
0	0	0	0	0	1	PB
0	0	0	0	1	0	PC
0	0	0	0	1	1	PD
0	0	0	1	0	1	PF
0	0	0	1	1	0	MKH
0	0	0	1	1	1	MKL
0	0	1	0	0	0	ANM
0	0	1	0	0	1	SMH
0	0	1	0	1	0	SML
0	0	1	0	1	1	EOM
0	0	1	1	0	0	ETMM
0	0	1	1	0	1	TMM
0	1	0	0	0	0	MM
0	1	0	0	0	1	MCC
0	1	0	0	1	0	MA
0	1	0	0	1	1	MB
0	1	0	1	0	0	MC
0	1	0	1	1	1	MF
0	1	1	0	0	0	TXB
0	1	1	0	0	1	RXB
0	1	1	0	1	0	TM0
0	1	1	0	1	1	TM1
1	0	0	0	0	0	CR0
1	0	0	0	0	1	CR1
1	0	0	0	1	0	CR2
1	0	0	0	1	1	CR3
1	0	1	0	0	0	ZCM

rpa3

C_3	C_2	C_1	C_0	addressing
0	0	1	0	(DE)
0	0	1	1	(HL)
0	1	0	0	(DE)++
0	1	0	1	(HL)++
1	0	1	1	(DE+byte)
1	1	0	0	(HL+A)
1	1	0	1	(HL+B)
1	1	1	0	(HL+EA)
1	1	1	1	(HL+byte)

irf

I_4	I_3	I_2	I_1	I_0	INTF
0	0	0	0	0	NMI
0	0	0	0	1	FT0
0	0	0	1	0	FT1
0	0	0	1	1	F1
0	0	1	0	0	F2
0	0	1	0	1	FE0
0	0	1	1	0	FE1
0	0	1	1	1	FEIN
0	1	0	0	0	FAD
0	1	0	0	1	FSR
0	1	0	1	0	FST
0	1	0	1	1	ER
0	1	1	0	0	OV
1	0	0	0	0	AN4
1	0	0	0	1	AN5
1	0	0	1	0	AN6
1	0	0	1	1	AN7
1	0	1	0	0	SB

sr3

U_0	Special-reg
0	ETM0
1	ETM1

sr4

V_0	Special-reg
0	ECNT
1	ECPT

rp

P_2	P_1	P_0	reg-pair
0	0	0	SP
0	0	1	BC
0	1	0	DE
0	1	1	HL
1	0	0	EA

rp1

Q_2	Q_1	Q_0	reg-pair
0	0	0	VA
0	0	1	BC
0	1	0	DE
0	1	1	HL
1	0	0	EA

f

F_2	F_1	F_0	Flag
0	0	0	——
0	1	0	CY
0	1	1	HC
1	0	0	Z

7.3 Instruction Execution Time

In the following table, 1 state consists of 3 clock cycles. So, when the 12MHz clock is used, 1 state becomes 250ns (=3x1/12μs). Execution time of the 4-state instruction, the shortest among instructions, becomes 1μs.

Note: For deeper explanations on the instruction set please refer to μPD7810/7811 Product Description.

Instruction group	Mnemonic		Operand	Instruction code B1	Instruction code B2	Instruction code B3	Instruction code B4	State	Operation	Skip condition
8-bit data transfer	MOV		r1, A	0 0 0 1 1 T_2 T_1 T_0				4	r1←A	
			A, r1	0 0 0 0 1 T_2 T_1 T_0				4	A←r1	
		*	sr, A	0 1 0 0 1 1 0 1	1 1 S_5 S_4 S_3 S_2 S_1 S_0			10	sr←A	
		*	A, sr1	0 1 0 0 1 1 0 0	1 1 S_5 S_4 S_3 S_2 S_1 S_0			10	A←sr1	
			r, word	0 1 1 1 0 0 0 0	0 1 1 0 1 R_2 R_1 R_0	Low Adrs	High Adrs	17	r←(word)	
			word, r	0 1 1 1 0 0 0 0	0 1 1 1 1 R_2 R_1 R_0	Low Adrs	High Adrs	17	(word)←r	
	MVI	*	r, byte	0 1 1 0 1 R_2 R_1 R_0	← Data →			7	r←byte	
			sr2, byte	0 1 1 0 0 1 0 0	S_1 0 0 0 0 S_2 S_1 S_0	Data		14	sr2←byte	
	MVIW	*	wa, byte	0 1 1 1 0 0 0 1	← Offset →	Data		13	(V. wa)←byte	
	MVIX	*	rpa1, byte	0 1 0 0 1 0 A_1 A_0	← Data →			10	(rpa1)←byte	
	STAW	*	wa	0 1 1 0 0 0 1 1	← Offset →			10	(V. wa)←A	
	LDAW	*	wa	0 0 0 0 0 0 0 1	← Offset →			10	A←(V. wa)	
	STAX	*	rpa2	A_3 0 1 1 1 A_2 A_1 A_0	Data(注1)			7/13	(rpa2)←A	
	LDAX	*	rpa2	A_3 0 1 0 1 A_2 A_1 A_0	Data(注1)			7/13	A←(rpa2)	
	EXX			0 0 0 1 0 0 0 1				4	{B←→B', C←→C', D←→D' E←→E', H←→H', L←→L'	
	EXA			0 0 0 1 0 0 0 0				4	V,A←→V',A', EA←→EA'	
	EXH			0 1 0 1 0 0 0 0				4	H,L←→H',L'	
	BLOCK			0 0 1 1 0 0 0 1				13 (C+1)	(DE)'←(HL)', C←C−1 End if borrow	
16-bit data transfer	DMOV		rp3, EA	1 0 1 1 0 1 P_1 P_0				4	$rp3_L$←EAL, $rp3_H$←EAH	
			EA, rp3	1 0 1 0 0 1 P_1 P_0				4	EAL←$rp3_L$, EAH←$rp3_H$	

Instruction group	Mnemonic	Operand	Instruction code B1	Instruction code B2	Instruction code B3	Instruction code B4	State	Operation	Skip condition
16-bit data transfer	DMOV	sr3, EA	0 1 0 0 0 1 0 0 0	1 1 0 1 0 0 1 U_0			14	sr3←EA	
		EA, sr4	↓ ↓	1 1 0 0 0 0 0 V_0			14	EA←sr4	
	SBCD	word	0 1 1 1 0 0 0 0	0 0 0 1 1 1 1 0	Low Adrs	High Adrs	20	(word)←C, (word+1)←B	
	SDED	word	↓ ↓	0 0 1 0 1 1 1 0	↓	↓	20	(word)←E, (word+1)←D	
	SHLD	word	↓ ↓	0 0 1 1 1 1 1 0	↓	↓	20	(word)←L, (word+1)←H	
	SSPD	word	↓ ↓	0 0 0 0 1 1 1 0	↓	↓	20	(word)←SP_L, (word+1)←SP_H	
	STEAX	rpa3	0 1 0 0 0 1 0 0 0	1 0 0 1 $C_3C_2C_1C_0$	Data(注2)		14/20	(rpa3)←EAL, (rpa3+1)←EAH	
	LBCD	word	0 1 1 1 0 0 0 0	0 0 0 1 1 1 1 1	Low Adrs	High Adrs	20	C←(word), B←(word+1)	
	LDED	word	↓ ↓	0 0 1 0 1 1 1 1	↓	↓	20	E←(word), D←(word+1)	
	LHLD	word	↓ ↓	0 0 1 1 1 1 1 1	↓	↓	20	L←(word), H←(word+1)	
	LSPD	word	↓ ↓	0 0 0 0 1 1 1 1	↓	↓	20	SP_L←(word), SP_H←(word+1)	
	LDEAX	rpa3	0 1 0 0 0 1 0 0 0	1 0 0 0 $C_3C_2C_1C_0$	Data(注2)		14/20	EAL←(rpa3), EAH←(rpa3+1)	
	PUSH	rp1	1 0 1 1 0 $Q_2Q_1Q_0$				13	(SP−1)←$rp1_H$, (SP−2)←$rp1_L$ SP←SP−2	
	POP	rp1	1 0 1 0 0 $Q_2Q_1Q_0$				10	$rp1_L$←(SP), $rp1_H$←(SP+1) SP←SP+2	
	LXI *	rp2, word	0 $P_2P_1P_0$ 0 1 0 0	←— Low Byte —→	High Byte		10	rp2←word	
	TABLE		0 1 0 0 0 1 0 0 0	1 0 1 0 1 0 0 0			17	C←(PC+3+A) B←(PC+3+A+1)	
8-bit arithmetic operation (register)	ADD	A, r	0 1 1 0 0 0 0 0	1 1 0 0 0 $R_2R_1R_0$			8	A←A+r	
		r, A	↓ ↓	0 1 0 0			8	r←r+A	
	ADC	A, r	↓ ↓	1 1 0 1			8	A←A+r+CY	
		r, A	↓ ↓	0 1 0 1			8	r←r+A+CY	

Instruction group	Mnemonic	Operand	Instruction code B1	Instruction code B2	Instruction code B3	Instruction code B4	State	Operation	Skip condition
8-bit arithmetic operation (register)	ADDNC	A, r	01100000	10100$R_2R_1R_0$			8	A←A+r	No Carry
		r, A		0010			8	r←r+A	No Carry
	SUB	A, r		1110			8	A←A−r	
		r, A		0110			8	r←r−A	
	SBB	A, r		1111			8	A←A−r−CY	
		r, A		0111			8	r←r−A−CY	
	SUBNB	A, r		1011			8	A←A−r	No Borrow
		r, A		0011			8	r←r−A	No Borrow
	ANA	A, r		10001$R_2R_1R_0$			8	A←A∧r	
		r, A		0000			8	r←r∧A	
	ORA	A, r		1001			8	A←A∨r	
		r, A		0001			8	r←r∨A	
	XRA	A, r		10010$R_2R_1R_0$			8	A←A∀r	
		r, A		0001			8	r←r∀A	
	GTA	A, r		10101$R_2R_1R_0$			8	A−r−1	No Borrow
		r, A		0010			8	r−A−1	No Borrow
	LTA	A, r		1011			8	A−r	Borrow
		r, A		0011			8	r−A	Borrow
	NEA	A, r		1110			8	A−r	No Zero
		r, A		0110			8	r−A	No Zero

Instruction group	Mnemonic	Operand	Instruction code B1	Instruction code B2	Instruction code B3	Instruction code B4	State	Operation	Skip condition
8-bit arithmetic operation (register)	EQA	A, r	0 1 1 0 0 0 0 0	1 1 1 1 1 $R_2R_1R_0$			8	A−r	Zero
		r, A		0 1 1 1			8	r−A	Zero
	ONA	A, r		1 1 0 0			8	A∧r	No Zero
	OFFA	A, r		1 1 0 1			8	A∧r	Zero
8-bit arithmetic operation (memory)	ADDX	rpa	0 1 1 1 0 0 0 0	1 1 0 0 0 $A_2A_1A_0$			11	A←A+(rpa)	
	ADCX	rpa		1 1 0 1			11	A←A+(rpa)+CY	
	ADDNCX	rpa		1 0 1 0			11	A←A+(rpa)	No Carry
	SUBX	rpa		1 1 1 0			11	A←A−(rpa)	
	SBBX	rpa		1 1 1 1			11	A←A−(rpa)−CY	
	SUBNBX	rpa		1 0 1 1			11	A←A−(rpa)	No Borrow
	ANAX	rpa		1 0 0 0 1 $A_2A_1A_0$			11	A←A∧(rpa)	
	ORAX	rpa		1 0 0 1			11	A←A∨(rpa)	
	XRAX	rpa		1 0 0 1 0 $A_2A_1A_0$			11	A←A∀(rpa)	
	GTAX	rpa		1 0 1 0 1 $A_2A_1A_0$			11	A−(rpa)−1	No Borrow
	LTAX	rpa		1 0 1 1			11	A−(rpa)	Borrow
	NEAX	rpa		1 1 1 0			11	A−(rpa)	No Zero
	EQAX	rpa		1 1 1 1			11	A−(rpa)	Zero
	ONAX	rpa		1 1 0 0			11	A∧(rpa)	No Zero
	OFFAX	rpa		1 1 0 1			11	A∧(rpa)	Zero

Instruction group	Mnemonic	Operand	Instruction code B1	Instruction code B2	Instruction code B3	Instruction code B4	State	Operation	Skip condition
Arithmetic operation of immediate data	ADI *	A, byte	0 1 0 0 0 1 1 0	← Data →			7	A←A+byte	
	ADI	r, byte	0 1 1 1 0 1 0 0	0 1 0 0 0 $R_2R_1R_0$	Data		11	r←r+byte	
		sr2, byte	0 1 1 0	S_3 1 0 0 0 $S_2S_1S_0$			20	sr2←sr2+byte	
	ACI *	A, byte	0 1 0 1 0 1 1 0	← Data →			7	A←A+byte+CY	
	ACI	r, byte	0 1 1 1 0 1 0 0	0 1 0 1 0 $R_2R_1R_0$	Data		11	r←r+byte+CY	
		sr2, byte	0 1 1 0	S_3 1 0 1 0 $S_2S_1S_0$			20	sr2←sr2+byte+CY	
	ADINC *	A, byte	0 0 1 0 0 1 1 0	← Data →			7	A←A+byte	No Carry
	ADINC	r, byte	0 1 1 1 0 1 0 0	0 0 1 0 0 $R_2R_1R_0$	Data		11	r←r+byte	No Carry
		sr2, byte	0 1 1 0	S_3 0 1 0 0 $S_2S_1S_0$			20	sr2←sr2+byte	No Carry
	SUI *	A, byte	0 1 1 0 0 1 1 0	← Data →			7	A←A−byte	
	SUI	r, byte	0 1 1 1 0 1 0 0	0 1 1 0 0 $R_2R_1R_0$	Data		11	r←r−byte	
		sr2, byte	0 1 1 0	S_3 1 1 0 0 $S_2S_1S_0$			20	sr2←sr2−byte	
	SBI *	A, byte	0 1 1 1 0 1 1 0	← Data →			7	A←A−byte−CY	
	SBI	r, byte	0 1 1 1 0 1 0 0	0 1 1 1 0 $R_2R_1R_0$	Data		11	r←r−byte−CY	
		sr2, byte	0 1 1 0	S_3 1 1 1 0 $S_2S_1S_0$			20	sr2←sr2−byte−CY	
	SUINB *	A, byte	0 0 1 1 0 1 1 0	← Data →			7	A←A−byte	No Borrow
	SUINB	r, byte	0 1 1 1 0 1 0 0	0 0 1 1 0 $R_2R_1R_0$	Data		11	r←r−byte	No Borrow
		sr2, byte	0 1 1 0	S_3 0 1 1 0 $S_2S_1S_0$			20	sr2←sr2−byte	No Borrow
	ANI *	A, byte	0 0 0 0 0 1 1 1	← Data →			7	A←A∧byte	
	ANI	r, byte	0 1 1 1 0 1 0 0	0 0 0 0 1 $R_2R_1R_0$	Data		11	r←r∧byte	

Instruction group	Mnemonic	Operand	Instruction code B1	Instruction code B2	Instruction code B3	Instruction code B4	State	Operation	Skip condition
Arithmetic operation of immediate data	ANI	sr2,byte	0 1 1 0 0 1 0 0	$S_3$0 0 0 1 $S_2S_1S_0$	Data		20	sr2←sr2∧byte	
	ORI *	A,byte	0 0 0 1 0 1 1 1	←Data→			7	A←A∨byte	
		r,byte	0 1 1 1 0 1 0 0	0 0 0 1 1 $R_2R_1R_0$	Data		11	r←r∨byte	
		sr2,byte	0 1 1 0 ↓	$S_3$0 0 1 1 $S_2S_1S_0$	↓		20	sr2←sr2∨byte	
	XRI *	A,byte	0 0 0 1 0 1 1 0	←Data→			7	A←A∀byte	
		r,byte	0 1 1 1 0 1 0 0	0 0 0 1 0 $R_2R_1R_0$	Data		11	r←r∀byte	
		sr2,byte	0 1 1 0 ↓	$S_3$0 0 1 0 $S_2S_1S_0$	↓		20	sr2←sr2∀byte	
	GTI *	A,byte	0 0 1 0 0 1 1 1	←Data→			7	A−byte−1	No Borrow
		r,byte	0 1 1 1 0 1 0 0	0 0 1 0 1 $R_2R_1R_0$	Data		11	r−byte−1	No Borrow
		sr2,byte	0 1 1 0 ↓	$S_3$0 1 0 1 $S_2S_1S_0$	↓		14	sr2−byte−1	No Borrow
	LTI *	A,byte	0 0 1 1 0 1 1 1	←Data→			7	A−byte	Borrow
		r,byte	0 1 1 1 0 1 0 0	0 0 1 1 1 $R_2R_1R_0$	Data		11	r−byte	Borrow
		sr2,byte	0 1 1 0 ↓	$S_3$0 1 1 1 $S_2S_1S_0$	↓		14	sr2−byte	Borrow
	NEI *	A,byte	0 1 1 0 0 1 1 1	←Data→			7	A−byte	No Zero
		r,byte	0 1 1 1 0 1 0 0	0 1 1 0 1 $R_2R_1R_0$	Data		11	r−byte	No Zero
		sr2,byte	0 1 1 0 ↓	$S_3$1 1 0 1 $S_2S_1S_0$	↓		14	sr2−byte	No Zero
	EQI *	A,byte	0 1 1 1 0 1 1 1	←Data→			7	A−byte	Zero
		r,byte	0 1 1 1 0 1 0 0	0 1 1 1 1 $R_2R_1R_0$	Data		11	r−byte	Zero
		sr2,byte	0 1 1 0 ↓	$S_3$1 1 1 1 $S_2S_1S_0$	↓		14	sr2−byte	Zero

Instruction group	Mnemonic	Operand	Instruction code B1	B2	B3	B4	State	Operation	Operation Skip
Arithmetic operation of immediate data	ONI *	A,byte	0 1 0 0 0 1 1 1	← Data →			7	A∧byte	No Zero
		r,byte	0 1 1 1 0 1 0 0	0 1 0 0 1 $R_2R_1R_0$	Data		11	r∧byte	No Zero
		sr2,byte	0 1 1 0	S_3 1 0 0 1 $S_2S_1S_0$			14	sr2∧byte	No Zero
	OFFI *	A,byte	0 1 0 1 0 1 1 1	← Data →			7	A∧byte	Zero
		r,byte	0 1 1 1 0 1 0 0	0 1 0 1 1 $R_2R_1R_0$	Data		11	r∧byte	Zero
		sr2,byte	0 1 1 0	S_3 1 0 1 1 $S_2S_1S_0$			14	sr2∧byte	Zero
Arithmetic operation of working register	ADDW	wa	0 1 1 1 0 1 0 0	1 1 0 0 0 0 0 0	offset		14	A←A+(V.wa)	
	ADCW	wa		1 1 0 1			14	A←A+(V.wa)+CY	
	ADDNCW	wa		1 0 1 0			14	A←A+(V.wa)	No Carry
	SUBW	wa		1 1 1 0			14	A←A−(V.wa)	
	SBBW	wa		1 1 1 1			14	A←A−(V.wa)−CY	
	SUBNBW	wa		1 0 1 1			14	A←A−(V.wa)	No Borrow
	ANAW	wa		1 0 0 0 1 0 0 0			14	A←A∧(V.wa)	
	ORAW	wa		1 0 0 1			14	A←A∨(V.wa)	
	XRAW	wa		1 0 0 1 0 0 0 0			14	A←A∀(V.wa)	
	GTAW	wa		1 0 1 0 1 0 0 0			14	A−(V.wa)−1	No Borrow
	LTAW	wa		1 0 1 1			14	A−(V.wa)	Borrow
	NEAW	wa		1 1 1 0			14	A−(V.wa)	No Zero
	EQAW	wa		1 1 1 1			14	A−(V.wa)	Zero
	ONAW	wa		1 1 0 0			14	A∧(V.wa)	No Zero

Instruction group	Mnemonic	Operand	Instruction code B1	Instruction code B2	Instruction code B3	Instruction code B4	State	Operation	Skip condition
Arithmetic operation of working register	OFFAW	wa	01110100	11011000	Offset		14	A∧(V.wa)	Zero
	ANIW *	wa,byte	00000101	← Offset →	Data		19	(V.wa)←(V.wa)∧byte	
	ORIW *	wa,byte	0001				19	(V.wa)←(V.wa)∨byte	
	GTIW *	wa,byte	0010				13	(V.wa)−byte−1	No Borrow
	LTIW *	wa,byte	0011				13	(V.wa)−byte	Borrow
	NEIW *	wa,byte	0110				13	(V.wa)−byte	No Zero
	EQIW *	wa,byte	0111				13	(V.wa)−byte	Zero
	ONIW *	wa,byte	0100				13	(V.wa)∧byte	No Zero
	OFFIW *	wa,byte	0101				13	(V.wa)∧byte	Zero
16-bit arithmetic operation	EADD	EA, r2	01110000	010000R_iR_0			11	EA←EA+r2	
	DADD	EA, rp3	0100	110001P_iP_0			11	EA←EA+rp3	
	DADC	EA, rp3		1101			11	EA←EA+rp3+CY	
	DADDNC	EA, rp3		1010			11	EA←EA+rp3	No Carry
	ESUB	EA, r2	0000	011000R_iR_0			11	EA←EA−r2	
	DSUB	EA, rp3	0100	111001P_iP_0			11	EA←EA−rp3	
	DSBB	EA, rp3		1111			11	EA←EA−rp3−CY	
	DSUBNB	EA, rp3		1011			11	EA←EA−rp3	No Borrow
	DAN	EA, rp3		100011P_iP_0			11	EA←EA∧rp3	
	DOR	EA, rp3		1001			11	EA←EA∨rp3	
	DXR	EA, rp3		100101P_iP_0			11	EA←EA∀rp3	

Multiplication/division →

Instruction group	Mnemonic	Operand	Instruction code B1	Instruction code B2	Instruction code B3	Instruction code B4	State	Operation	Skip condition
16-bit arithmetic operation	DGT	EA, rp3	0 1 1 1 0 1 0 0	1 0 1 0 1 1 P_1 P_0			11	EA − rp3 − 1	No Borrow
	DLT	EA, rp3		1 0 1 1			11	EA − rp3	Borrow
	DNE	EA, rp3		1 1 1 0			11	EA − rp3	No Zero
	DEQ	EA, rp3		1 1 1 1			11	EA − rp3	Zero
	DON	EA, rp3		1 1 0 0			11	EA ∧ rp3	No Zero
	DOFF	EA, rp3		1 1 0 1			11	EA ∧ rp3	Zero
	MUL	r2	0 1 0 0 1 0 0 0	0 0 1 0 1 1 R_1 R_0			32	EA ← A × r2	
	DIV	r2		0 0 1 1			59	EA ← EA ÷ r2, r2 ← 余り	
Decrement/Increment	INR	r2	0 1 0 0 0 0 R_1 R_0				4	r2 ← r2 + 1	Carry
	INRW *	wa	0 0 1 0 0 0 0 0	← Offset →			16	(V.wa) ← (V.wa) + 1	Carry
	INX	rp	0 0 P_1 P_0 0 0 1 0				7	rp ← rp + 1	
		EA	1 0 1 0 1 0 0 0				7	EA ← EA + 1	
	DCR	r2	0 1 0 1 0 0 R_1 R_0				4	r2 ← r2 − 1	Borrow
	DCRW *	wa	0 0 1 1 0 0 0 0	← Offset →			16	(V.wa) ← (V.wa) − 1	Borrow
	DCX	rp	0 0 P_1 P_0 0 0 1 1				7	rp ← rp − 1	
		EA	1 0 1 0 1 0 0 1				7	EA ← EA − 1	
Other arithmetic operation	DAA		0 1 1 0 0 0 0 1				4	Decimal Adjust Accumulator	
	STC		0 1 0 0 1 0 0 0	0 0 1 0 1 0 1 1			8	CY ← 1	
	CLC			0 0 1 0 1 0 1 0			8	CY ← 0	
	NEGA			0 0 1 1 1 0 1 0			8	A ← $\overline{A}$ + 1	

Instruction group	Mnemonic	Operand	Instruction code B1	Instruction code B2	Instruction code B3	Instruction code B4	State	Operation	Skip condition
Rotation/shift	RLD		01001000	00111000			17	Rotate Left Digit	
	RRD			1001			17	Rotate Right Digit	
	RLL	r2		01 R_1R_0			8	$r2_{m+1} \leftarrow r2_m, r2_0 \leftarrow CY, CY \leftarrow r2_7$	
	RLR	r2		00 R_1R_0			8	$r2_{m-1} \leftarrow r2_m, r2_7 \leftarrow CY, CY \leftarrow r2_0$	
	SLL	r2		001001 R_1R_0			8	$r2_{m+1} \leftarrow r2_m, r2_0 \leftarrow 0, CY \leftarrow r2_7$	
	SLR	r2		00 R_1R_0			8	$r2_{m-1} \leftarrow r2_m, r2_7 \leftarrow 0, CY \leftarrow r2_0$	
	SLLC	r2		000001 R_1R_0			8	$r2_{m+1} \leftarrow r2_m, r2_0 \leftarrow 0, CY \leftarrow r2_7$	Carry
	SLRC	r2		00 R_1R_0			8	$r2_{m-1} \leftarrow r2_m, r2_7 \leftarrow 0, CY \leftarrow r2_0$	Carry
	DRLL	EA		10110100			8	$EA_{n+1} \leftarrow EA_n, EA_0 \leftarrow CY, CY \leftarrow EA_{15}$	
	DRLR	EA		0000			8	$EA_{n-1} \leftarrow EA_n, EA_{15} \leftarrow CY, CY \leftarrow EA_0$	
	DSLL	EA		10100100			8	$EA_{n+1} \leftarrow EA_n, EA_0 \leftarrow 0, CY \leftarrow EA_{15}$	
	DSLR	EA		0000			8	$EA_{n-1} \leftarrow EA_n, EA_{15} \leftarrow 0, CY \leftarrow EA_0$	
Jump	JMP *	word	01010100	← Low Adrs →	High Adrs		10	PC←word	
	JB		00100001				4	$PC_H \leftarrow B$, $PC_L \leftarrow C$	
	JR	word	11 ←jdisp1 →				10	PC←PC+1+jdisp1	
	JRE *	word	01001111 ←	jdisp →			10	PC←PC+2+jdisp	
	JEA		01001000	00101000			8	PC←EA	
Call	CALL *	word	01000000	← Low Adrs →	High Adrs		16	$(SP-1) \leftarrow (PC+3)_H, (SP-2) \leftarrow (PC+3)_L$ PC←word, SP←SP−2	
	CALB		01001000	00101001			17	$(SP-1) \leftarrow (PC+2)_H, (SP-2) \leftarrow (PC+2)_L$ $PC_H \leftarrow B, PC_L \leftarrow C$, SP←SP−2	
	CALF *	word	01111 ←	fa →			13	$(SP-1) \leftarrow (PC+2)_H, (SP-2) \leftarrow (PC+2)_L$ $PC_{15-11} \leftarrow 00001, PC_{10-0} \leftarrow fa$, SP←SP−2	

Instruction group	Mnemonic	Operand	Instruction code B1	B2	B3	B4	State	Operation	Skip condition
Call	CALT	word	1 0 0 ← ta →				16	$(SP-1)\leftarrow(PC+1)_H, (SP-2)\leftarrow(PC+1)_L$ $PC_L\leftarrow(128+2ta), PC_H\leftarrow(129+2ta), SP\leftarrow SP-2$	
	SOFTI		0 1 1 1 0 0 1 0				16	$(SP-1)\leftarrow PSW, (SP-2)\leftarrow(PC+1)_H, (SP-3)$ $\leftarrow(PC+1)_L, PC\leftarrow 0060H, SP\leftarrow SP-3$	
Return	RET		1 0 1 1 1 0 0 0				10	$PC_L\leftarrow(SP), PC_H\leftarrow(SP+1)$ $SP\leftarrow SP+2$	
	RETS		1 0 0 1				10	$PC_L\leftarrow(SP), PC_H\leftarrow(SP+1)SP\leftarrow SP+2$ $PC\leftarrow PC+n$	Uncon-ditional
	RETI		0 1 1 0 0 0 1 0				13	$PC_L\leftarrow(SP), PC_H\leftarrow(SP+1),$ $PSW\leftarrow(SP+2), SP\leftarrow SP+3$	
Skip	BIT *	bit, wa	0 1 0 1 1 $B_2B_1B_0$	← Offset →			10	Skip if (V.wa) bit = 1	(V.wa)bit = 1
	SK	f	0 1 0 0 1 0 0 0	0 0 0 0 1 $F_2F_1F_0$			8	Skip if f = 1	f = 1
	SKN	f		0 0 0 1			8	Skip if f = 0	f = 0
	SKIT	irf		0 1 0 $I_4I_3I_2I_1I_0$			8	Skip if irf = 1, then reset irf	irf = 1
	SKNIT	irf		0 1 1 $I_4I_3I_2I_1I_0$			8	Skip if irf = 0 Reset irf, if irf = 1	irf = 0
CPU operation	NOP		0 0 0 0 0 0 0 0				4	No Operation	
	EI		1 0 1 0 1 0 1 0				4	Enable Interrupt	
	DI		1 0 1 1 1 0 1 0				4	Disable Interrupt	
	HLT		0 1 0 0 1 0 0 0	0 0 1 1 1 0 1 1			12	Set Halt Mode	
	STOP		0 1 0 0 1 0 0 0	1 0 1 1 1 0 1 1			12	Set Stop Mode	

Note 1. B2(Data) is applied for rpa2 = D + byte or H + byte.

2. B3(Data) is applied for rpa3 = D + byte or H + byte.

3. In item "states", the right of slant is applied when rpa2 or rpa3 is D + byte, H + A, H + B, H + EA, or H + byte.

4. When each instruction is skipped, idle states is different from execution states (See following data).

1-byte instruction	:	4-state
2-byte " (with *)	:	7-state
2-byte "	:	8-state
3-byte " (with *)	:	10-state
3-byte "	:	11-state
4-byte "	:	14-state

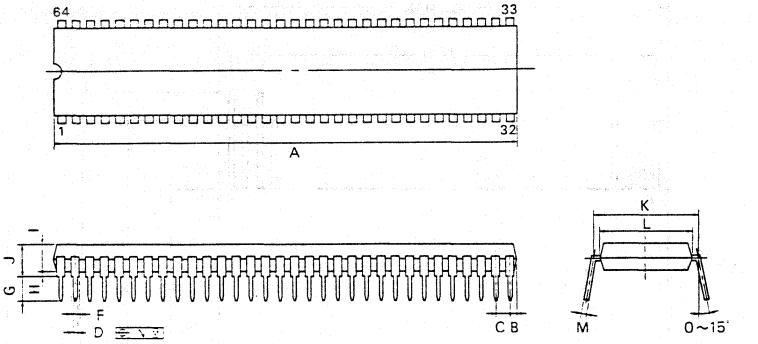
8. LIST OF MODE REGISTERS

Name of mode register		Read/ write	Function
MA	MODE A	W	Specifies input/output of Port A in bit units
MB	MODE B	W	Specifies input/output of Port B in bit units
MCC	MODE CONTROL C	W	Specifies port/control mode of Port C in bit units
MC	MODE C	W	Specifies input/output of Port C set in the port mode in bit units
MM	MEMORY MAPPING	W	Specifies port/expansion mode of Ports D and F
MF	MODE F	W	Specifies input/output of Port F set in the port mode in bit units
TMM	Timer mode	R/W	Specifies operation mode of the timer
ETMM	Timer/Event Counter Mode	W	Specifies operation mode of the Timer Event Counter
EOM	Timer/Event Counter Output Mode	R/W	Controls output level of CO0 and CO1
SML	Serial Mode	W	Specifies operation mode of the serial interface
SMH		R/W	
ANM	A/D Channel Mode	R/W	Specifies operation mode of the A/D converter
ZCM	Zero-cross Mode	W	Specifies operation mode of the zero-cross detection circuit

9. DIFFERENCE BETWEEN μPD78C11 AND μPD7811

Item \ Product	μPD78C11	μPD7811
No. of instructions	159 (STOP instruction was added.)	158
No. of special registers	28 (ZCM register was added.)	27
Standby function	HALT MODE, software STOP mode, hardware STOP mode. In addition, in the software/hardware STOP mode, the internal RAM data (256 bytes) are retained at the power supply voltage as low as 2.0V.	32 bytes of the 256-byte internal RAM data are retained at power supply voltage as low as 3.2V.
Control of zero-cross detection circuit's self-bias	Available by setting the ZCM register	Not available
No. of states of the HLT instruction	12	11
Device construction	CMOS	NMOS
Power consumption Operating Standby	T.B.D. T.B.D.	750mW TYP. 4.8mW TYP.
Pin configuration	V_{DD} : Pin 64 $\overline{STOP}$: Pin 63	V_{CC}: Pin 64 V_{DD}: Pin 63

64PIN PLASTIC SHRINK DIP (750 mil)



P64C-70-750A.C

NOTES

1) Each lead centerline is located within 0.17 mm (0.007 inch) of its true position (T.P.) at maximum material condition.

2) Item "K" to center of leads when formed parallel.

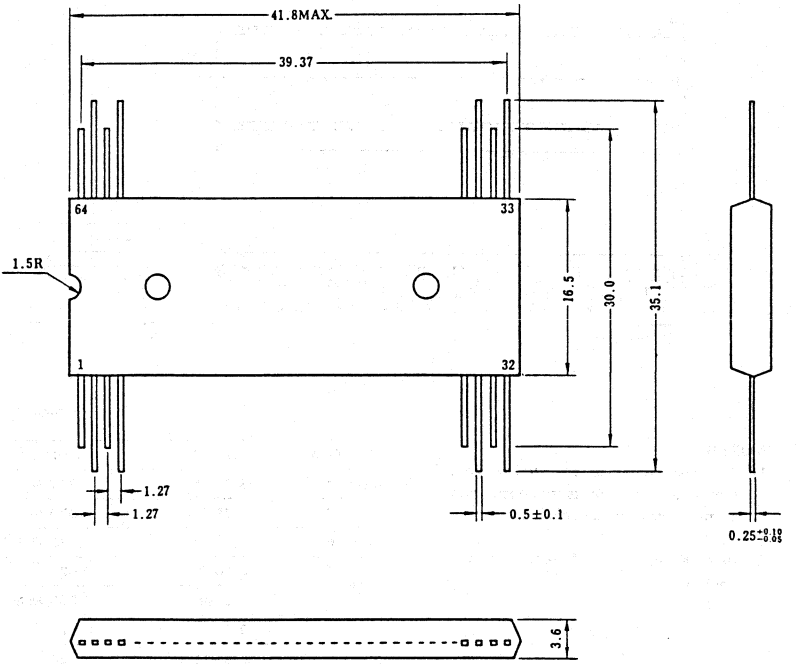
ITEM	MILLIMETERS	INCHES
A	58.68 MAX.	2.311 MAX.
B	1.78 MAX.	0.070 MAX.
C	1.778 (T.P.)	0.070 (T.P.)
D	$0.50^{-0.10}$	$0.020^{-0.004}_{-0.005}$
F	0.9 MIN.	0.035 MIN.
G	$3.2^{-0.3}$	$0.126^{-0.012}$
H	0.51 MIN.	0.020 MIN.
I	4.31 MAX.	0.170 MAX.
J	5.08 MAX.	0.200 MAX.
K	19.05 (T.P.)	0.750 (T.P.)
L	17.0	0.669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

When order this package, please specify as follows:

μPD78C11CW-XXX

μPD78C10CW

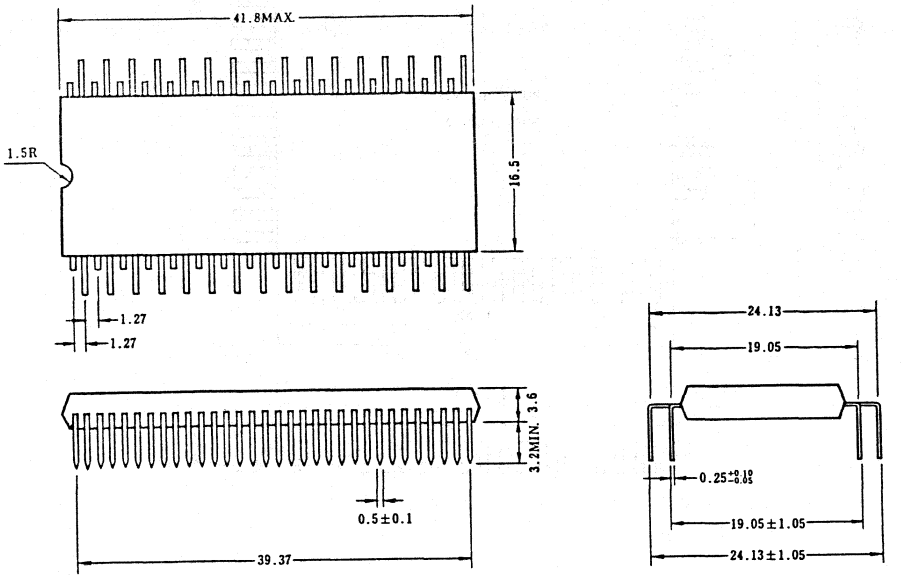
Dimensions of 64-pin Plastic Flat Package for μPD78C11G (Unit: mm)



When order with this package, please specify as follows:

uPD78C11G-xxx-37

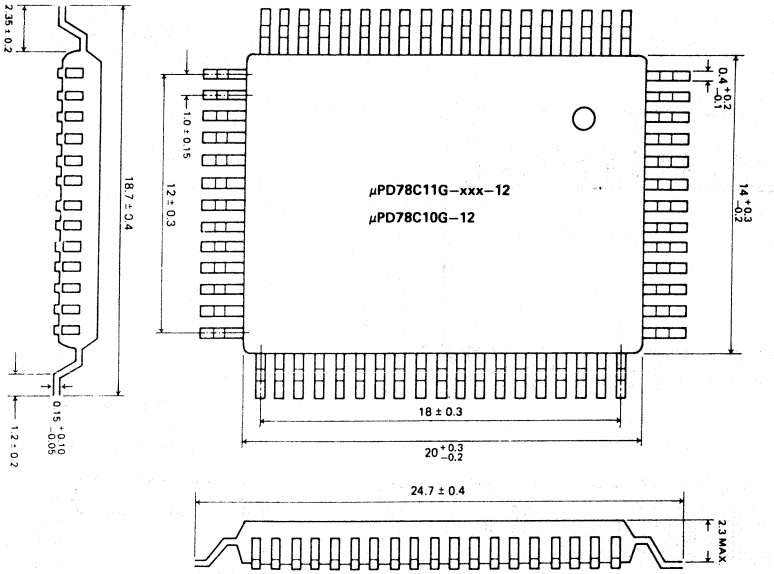
Dimensions of 64-pin Plastic QUIP for μPD78C10G/μPD78C11G (Unit: mm)



When order with this package, please specify as follows:

uPD78C10G
uPD78C11G-xxx-36

PACKAGE DIMENSIONS 64-PIN FLAT PACK μPD78C10/11



When order with this package, please specify as follows:

μPD78C11G-XXX-12
μPD78C10G-12

NEC Electronics (Europe) GmbH

EUROPEAN DISTRIBUTORS

AUSTRIA
A & D
ABRAHAMCZIK & DEMEL
GES. MBH. & CO KG
EICHENSTRASSE 58-64/1
1120 WIEN
TEL.: (222) 857661
TLX.: 134273

BELGIUM
CN ROOD
DE JAMBLINNE DE MEUXPLEIN 37
1040 BRUSSEL
TEL.: (02) 7352135
TLX.: 22846

MALCHUS ELECTRONICS PVBA.
PLANTIN EN MORETUSLEI 172
2000 ANTWERPEN
TEL.: (032) 353256
TLX.: 33637

DENMARK
MER-EL A/S
VED KLAEDEBO 18
2970 HOERSHOLM
TEL.: (2) 571000
TLX.: 37360

FINLAND
OY FERRADO A/B
P.O. BOX 54
VALIMONTIE 1
00380 HELSINKI 38
TEL.: (0) 550002
TLX.: 122214

FRANCE
TEKELEC
RUE CARLE VERNET
CITE DES BRUYERES
92310 SEVRES
TEL.: (1) 45 34 75 35

EALING
BATIMENT AUVIDULIS
AVENUE D'OCEANIE
Z.A. D'ORSAY COURTABŒUF
BP 90
91943 LES ULIS CEDEX
TEL.: (1) 69 28 01 31

CELTI
Z.I. DE COURTABŒUF
9, AVENUE DU QUEBEC
91940 LES ULIS
TEL.: (1) 64 46 09 09

ASAP
RUE DE TROIS PEUPLES
78190 MONTIGNY LE BRETONNEUX
TEL.: (1) 30 43 82 33
TLX.: 698887

ASAP
MONSIEUR LEGRIS
42, RUE HENRI MATISSE
59930 LA CHAPELLE D'ARMENTIERES
TEL.: 20 35 11 10

CCI
5, RUE MARCELIN BERTHELOT
BP 92
92164 ANTONY
TEL.: (1) 46 66 21 82
TLX.: 203881

CCI
5, RUE BATAILLE
69008 LYON
TEL.: 78 74 44 56

DIM INTER
65 – 67, RUE DES CITES
93300 AUBERVILLIERS
TEL.: (1) 48 34 93 70
TLX.: 230524

DIM INTER (VILLEURBANNE)
101, RUE DEDIEU
69100 VILLEURBANNE
TEL.: 78 68 32 29

DIM INTER (COLMAR)
27, RUE KLEBER
68000 COLMAR
TEL.: 89 41 15 43

GEDIS
352, AVENUE G. CLEMENCEAU
92000 NANTERRE
TEL.: (1) 42 04 04 04

GEDIS (ALPES)
21, RUE DES GLAISONS
38400 ST. MARTIN D'HERES
TEL.: 75 51 23 32

GEDIS (AIX)
MERCURE C
Z.I. D'AIX EN PROVENCE
13763 LES MILLES CEDEX
TEL.: 42 60 01 77

CEDIS (TOURS)
1, RUE DU DANEMARK
37100 TOURS
TEL.: 47 41 76 46

SERTRONIQUE (MANS)
60, RUE SAGEBIEN
CEDEX 43
72040 LE MANS
TEL.: 43 84 24 60
TLX.: 720019

SERTRONIQUE (LILLE)
20, RUE CABANIS
BP 35
59007 LILLE CEDEX
TEL.: 20 47 70 70

GERMANY
GLYN GMBH
SCHÖNE AUSSICHT 30
6272 NIEDERNHAUSEN
TEL.: (0 61 27) 80 77
TLX.: 4 186 911

MICROSCAN GMBH
ÜBERSEERING 31
2000 HAMBURG 60
TEL.: (0 40) 6 32 00 30
TLX.: 2 13 288

REIN ELEKTRONIK GMBH
LÖTSCHERWEG 66
4054 NETTETAL 1
TEL.: (0 21 53) 73 31 11
TLX.: 8 54 251

ULTRATRONIK GMBH
MÜNCHENER STRASSE 6
8031 SEEFELD
TEL.: (0 81 52) 70 90
TLX.: 5 26 459

H3W ELEKTRONIK VERTRIEB GMBH
NEUMARKTER STRASSE 75
8000 MÜNCHEN 80
TEL.: (0 89) 4 31 32 60
TLX.: 5 214 514

NIPTRON GMBH
LUITPOLTSTRASSE 52
8300 LANDSHUT
TEL.: (08 71) 6 90 48
TLX.: 5 8 443

SYSTEM ELEKTRONIK VERTRIEB GMBH
HEESFELD 4
3300 BRAUNSCHWEIG
TEL.: (05 31) 31 40 95
TLX.: 9 52 351

GLEICHMANN + CO ELECTRONICS GMBH
INDUSTRIESTRASSE 16
7513 STUTENSEE 3
TEL.: (0 72 49) 70 01
TLX.: 7 825 602

ITALY
MELCHIONI S.P.A.
VIA COLETTA, 37
20135 MILANO
TEL.: (02) 57941

CLAITRON S.P.A.
VIA GALLARATE, 211
20151 MILANO
TEL.: (02) 3010091

ADELSY S.R.L.
VIA DEL FONDITORE, 5
LOCALITA ROVERI
40127 BOLOGNA
TEL.: (051) 532119

PANTRONIC S.R.L.
VIA MATTIA BATTISTINI, 212/a
00167 ROMA
TEL.: (06) 6273909

NETHERLANDS
INNOCIRCUIT
MALCHUS ELECTRONICA
ADVIESGROEP
MALCHUS B.V.
FOKKERSTRAAT 511-513
3125 BD SCHIEDAM
TEL.: (010) 373777
TLX.: 21598

CN ROOD
CORT V.D. LINDENSTRAAT 11-13
2288 EV RIJSWIJK
TEL.: (070) 996360
TLX.: 31238

NORWAY
JACOB HATTELAND ELECTRONIC
P.B. 25
5578 NEDRE VATS
TEL.: (47) 63300
TLX.: 42850

PORTUGAL
AMPEREL S.A.
AV. FONTES PEREIRA DE MELO 47, 4D
1000 LISBOA
TEL.: (1) 53 26 98
TLX.: 18588

SPAIN
LOBER S.A.
MONTE ESQUINZA 28
MADRID 4
TEL.: (1) 442 11 00
TLX.: 49533

COMELTA S.A.
EMILIO MUNOZ 41, NAVE 1-1-2
MADRID 17
TEL.: (1) 754 30 01
TLX.: 42007

AMITRON S.A.
AVENIDA DE VALLADOLID 47 A
28008 MADRID
TEL.: (1) 247 93 13
TLX.: 45550

SWEDEN
TH'S ELEKTRONIK
BOX 3027
16303 SPAANGA
TEL.: (0) 8362970
TLX.: 11145

NORDQVIST & BERG
BOX 9145
AARSTAAENGS VAEGEN 19
10272 STOCKHOLM
TEL.: (0) 8690400
TLX.: 10407

SWITZERLAND
MEMOTEC AG
GASWERKSTRASSE 32
4901 LANGENTHAL
TEL.: (01) 8201545
TLX.: 55547

TURKEY
BURC ELEKTRONIK
VE MAKINA
SANAYI VE TICARET A.S.
REFIK SAYDAM CAD.
NO. 89 ARSLAN HAN
KAT. 2 – 7
SISHANE/ISTANBUL
TEL.: (1) 144 81 82 + 149 57 88 / 89
TLX.: 25883

UNITED KINGDOM
ANZAC COMPONENTS LTD
BURNHAM LANE
SLOUGH SL1 6LN
ENGLAND
TEL.: (06286) 4701

FARNEHL ELECTRONIC
COMPONENTS LTD
CANAL ROAD
LEEDS LS12 2TU
ENGLAND
TEL.: (0532) 636311

STC MULTI COMPONENTS
EDINBURGH WAY
HARLOW
CM20 2DF
ENGLAND
TEL.: (0279) 442971

DIALOGUE DISTRIBUTION LTD
WATCHMOOR ROAD
CAMBERLER
SURREY GU15 3AQ
ENGLAND
TEL.: (0276) 688001

IMPULSE ELECTRONICS LTD
HAMMOND HOUSE
CATERHAM
SURREY CR3 6XG
TEL.: (0883) 46433

VSI ELECTRONICS LTD
ROYDOMBURY INDUSTRIAL PARK
HORSECROFT ROAD 9
HARLOW, 5
ESSEX CM19 5BYQM
TEL.: (0279) 29666

NEC OFFICES

NEC Electronics (Europe) GmbH, Oberrather Str. 4, 4000 Düsseldorf 30, W. Germany,
Tel. (0211) 650301, Telex 858996-0

NEC Electronics (Germany) GmbH, Oberrather Str. 4, 4000 Düsseldorf 30,
Tel. (0211) 650302, Telex 858996-0
- Hindenburgstr. 28/29, 3000 Hannover 1, Tel. (0511) 881013–16, Telex 9230109
- Arabellastr. 17, 8000 München 81, Tel. (089) 4160020, Telex 522971
- Heilbronner Str. 314, 7000 Stuttgart 30, Tel. (0711) 890910, Telex 7252220

NEC Electronics (BNL) – Boschdijk 187a, NL-5612 HB Eindhoven, Tel. (040) 445845,
Telex 51923

NEC Electronics (Scandinavia) – Box 4039, S-18304 Täby, Tel. (08) 7328200,
Telex 13839

NEC Electronics (France) S.A., 9, rue Paul Dautier, B.P. 187,
F-78142 Velizy Villacoublay Cedex, Tél. (1) 39469617, Télex 699499

NEC Electronics Italiana S.R.L., Via Cardano 3, I-20124 Milano, Tel. (02) 6709108,
Telex 315355

NEC Electronics (UK) Ltd., Block 3 Carfin Industrial Estate, Motherwell M L1 4UL,
Scotland, Tel. (0698) 732221, Telex 777565
- Birmingham Office, 9th Floor, Swan Office Centre, 1508 Coventry Road, Yardley,
Birmingham B 25 8 VL, Tel. (021) 7081500, Telex 333014
- Reading Office, Reading Central Building, 30 Garrad Street, Reading Berks,
RG1 1NR, Tel. (0734) 596551, Telex 847998
- Dublin Office, 34/35 South William Street, Dublin 2, Ireland, Tel. (0001) 710200