NEC

PRODUCT DESCRIPTION

μPD 78C06

μPD 78C06A

NEC ELECTRONICS (EUROPE) GMBH

PRODUCT DESCRIPTION

µPD 78C06

µPD 78C06A

8/84 V1.2

Table of Contents

uPD78C06/78C06A *

SINGLE-CHIP 8-BIT MICROCOMPUTER

GENERAL DESCRIPTION

The NEC uPD78C06/78C06A is a general purpose single-chip microcomputer. The uPD78C06/78C06A is fabricated with CMOS technology.

This contains the functional blocks of program memory, data memory, ALU, I/O ports, on-chip timer serial I/O, and internal clock generator. It can extend external memory capacity (ROM, RAM) up to 60K bytes.

uPD78C06A is a high speed version of uPD78C06, but only ϕ out function of uPD78C06A is different from uPD 78C06. uPD78C06As ϕ out is 1/8 of crystal frequency, but uPD78C06s is 1/4 of crystal frequency.

Therefore, the uPD78C06A is 33% facter in instruction - execution than the uPD78C06.

As the uPD78C06/78C06A is a CMOS device and provided with the standby function, it is suitable for application of hand-held computers requiring low power consumption. It is also suitable for application in control field because of its rich input/output functions, timer function and bit operation function.

(NOTE)

* 78C06/06A means not only 78C06/06A but also 78C05/05A (Evaluation chip) except the cases where specifically mentioned its difference.

FEATURES Single-chip Microcomputer (uCOM-87LC)

powerful 101 Instructions

Transfer 17

Operation 60

Rotation 4

Jump 4

Call/Return ... 6

Port output ... 2

Others 8

Instruction Cycle Time : 6 us (on-chip ROM) (at 4MHz;uPD78C06)

4 us (") (at 6MHz;uPD78C06A)

4 us (external memory & on-chip RAM)

(at 4MHz;uPD78C06)

2.76 us (") (at 6MHz;uPD78C06A)

Program Memory (ROM) : 4096W x 8

Data Memory (RAM) : 128W x 8

Direct Addressing Capability up to 60kB External Memory

Powerful Addressing Modes Capability

Register, register indirect, auto-increment, auto-decrement, working register, immediate, direct, relative

Multi-level Stack

Vectored Interrupts (External : 2, Internal : 1)

On-chip 8-bit Timer with 4-bit Prescaler

46 I/O lines

8-bit output port (PA0-7)

8-bit input/output port (PB0-7)

8-bit input port (PC0-PC5)

8-bit data bus (DB0-7)

Serial I/O Ports

Stand-by Capability (STOP/HALT mode)

Fully Bus Compatible with 8080A

On-chip Clock Generator

Single Supply, CMOS Technology

Clock output (ϕout):

uPD78C06 - fosc/4 (= internal system clock)

uPD78C06A - fosc/8 (= half of internal system clock)

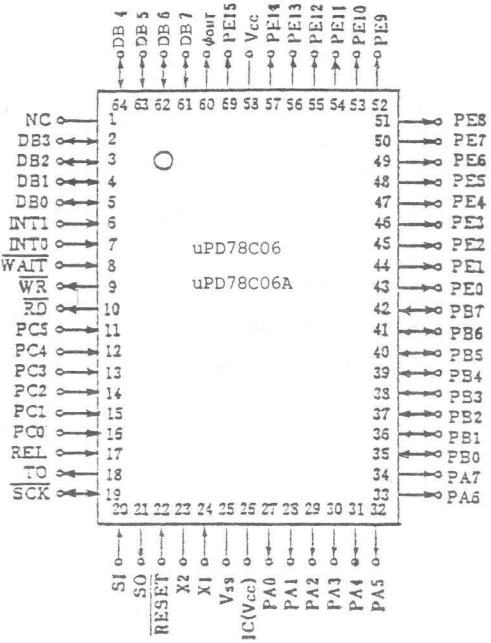
Low Power Consumption

64 pin Plastic Flat Package (uPD78C06/uPD78C06A)

64 pin Plastic QUIP Package (uPD78C06A)

PIN CONFIGURATION

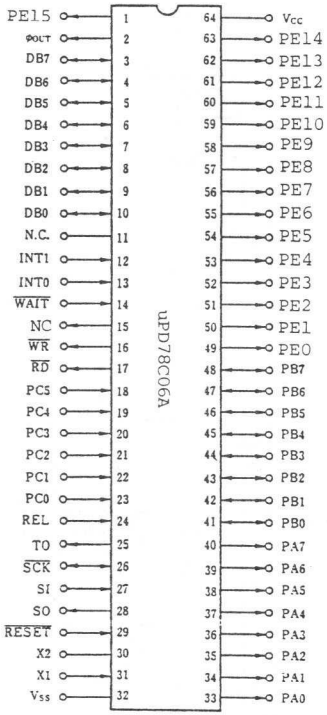
(TOP VIEW)



Pin Names

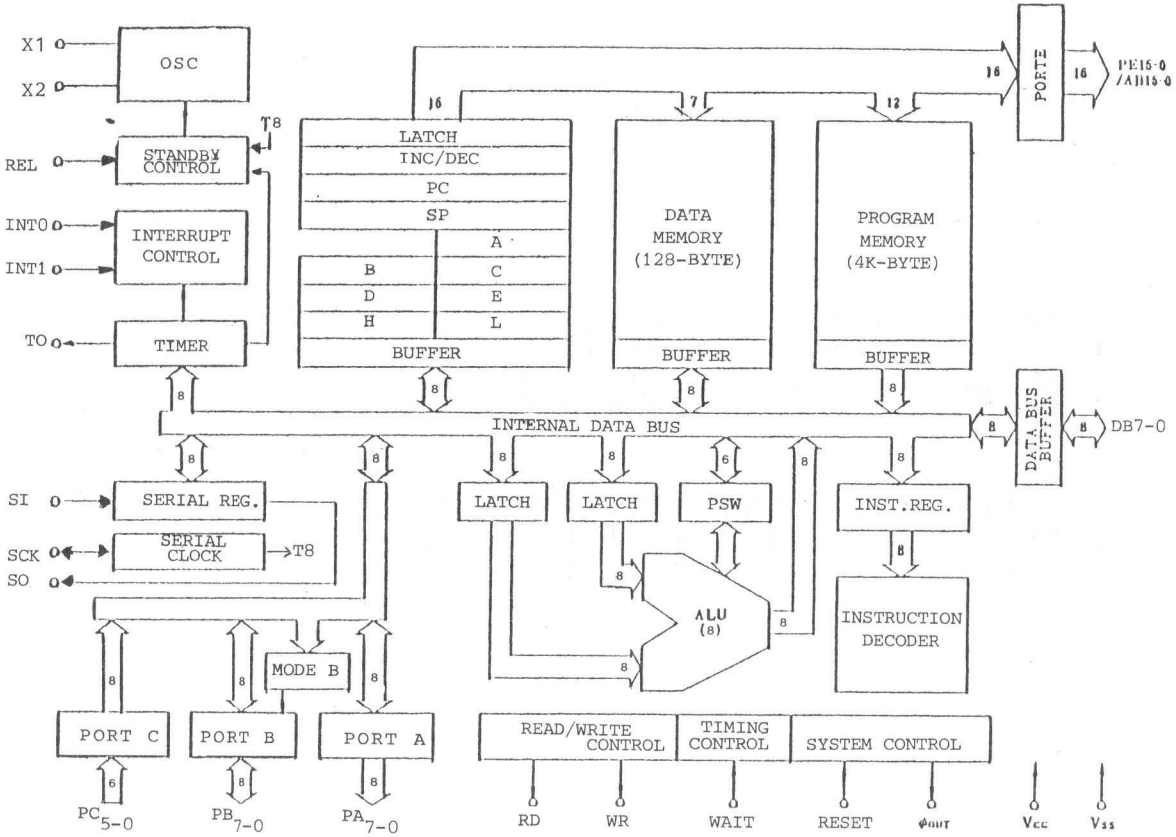
PA_{7-0}, PB_{7-0}, PC_{5-0}, PE_{15-0}	: I/O Ports
DB_{7-0}	: Data Bus
$\overline{WAIT}$	: Wait Request
INT0, INT1	: Interrupt Request
X_2, X_1	: Xtal
SCK	: Serial Clock Input/Output
SI	: Serial Input
SO	: Serial Output
$\overline{RESET}$	: Reset
$\overline{RD}$	: Read Strobe
$\overline{WR}$	: Write Strobe
ϕ out	: Clock Output

(Top View)



NC: Non-connection

uPD78C06/uPD78C06A Block Diagram



1. PIN DESCRIPTION

1.1 PA_{7-0} (PortA) ... Output

This is a 8-bit output port, and it has latch capability. With Move instructions, the data can be transferred between latch buffers and accumulator. Besides, the latched contents on the buffers can be freely handled with Logic instructions. Once the data is written on the buffers, it is kept on the buffers until another Port A handling instructions will be executed or a reset signal will be issued. Output level is TTL compatible.

When the $\overline{RESET}$ signal is entered, the latch contents are cleared and the low level signal is output.

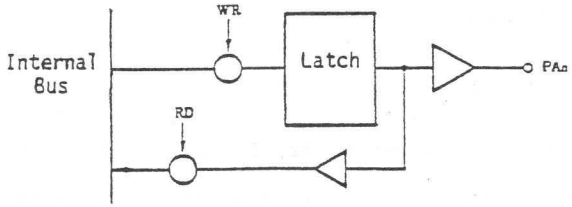


Fig. 1-1 The Port A Structure

1.2 PB_{7-0} (PortB) ... 3-state Input/Output

This is a 8-bit input/output port, and its output has latch capability. Each line of the Port B can be independently manipulated by the MODE B Register, and either of an input or output port can be programmed at that time. In case of being set as input port lines, or during reset, the port lines become a high impedance state. Input level is CMOS compatible and output level is TTL compatible.

The output latch contents are cleared (0) with the $\overline{RESET}$ signal.

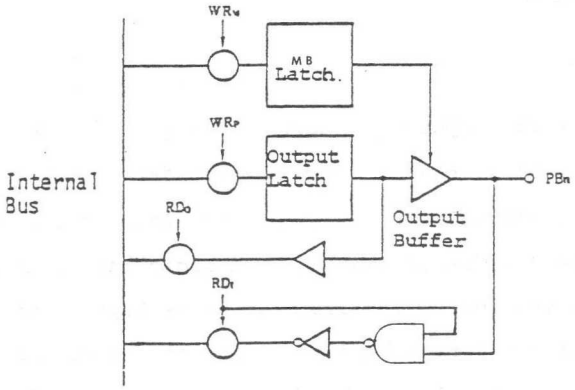


Fig. 1-2a The Port B Structure

(a) A bit of the Port B is defined as an output port (MODE Bn=0) In this case, an output latch contents can be output from PBn. And with move instructions, the data can be transferred between output latch buffers of the port-lines and an accumulator.

(continued)

The latched contents can be freely handled with Logic instruction. Once the data is written on the output, the latch will maintains the data until another Port B handling instriction will be executed or a reset will be issued.

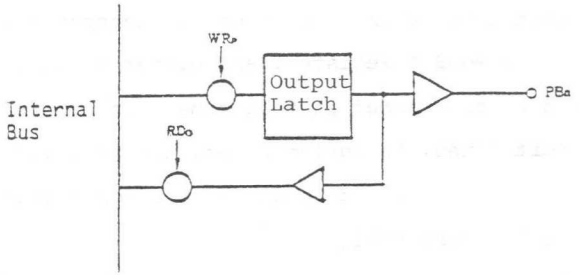


Fig. 1-2b The lines of the Port B used as output ports

(b) A bit of the Port B is defined as an input port (MODE·Bn=1)

The content of the PB lines can be loaded onto the accumulator with a move instruction. Also a write operation into output latch of the port lines are possible, too, i.e., by a move instruction, the data on the accumulator can be stored into all the output latchs of the Port B, no matter whether a port line is set as input or output port. However, the contents of the output latch buffers of the port lines used as input port cannot be loaded onto the accumulator, by a move instruction. Besides, since the output latch buffers of the input ports are high impedance state, the contents of the buffers do not appear on the port lines. But, if the port lines then are switched over from input mode to output mode, the contents stored in the above mentioned output latchs can appear on the port lines, and can be loaded into the accumulator.

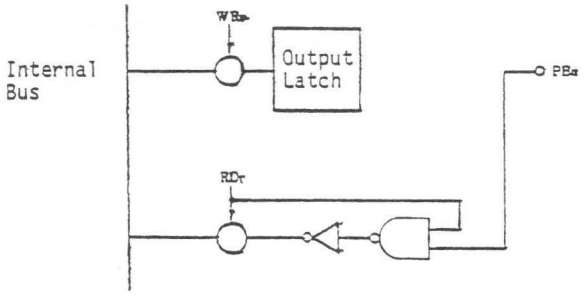


Fig. 1-2c The lines of the Port B used as input ports

(continued)

However, an actual instruction execution is done per 8-bit data. If a Port B read instruction (MOV A,PB) is executed, the contents of input lines set as input ports and the ones of output latches set as output ports are loaded into the accumulator. If a Port B write instruction (MOV PB,A, etc.) is executed, the contents of the output latch buffers set as input ports are replaced with another data by the instruction execution, but input lines are not affected by it.

1.3 PC_{5-0} (Port C) ... Input

This is a 6-bit input port with pull-up resistors. Input data to this port can be tested by test instruction, and also moved to least significant 6-bit of accumulator and higher 2 bit of accumulator are loaded with "0". Input level is CMOS compatible. This port is fit for key-input port.

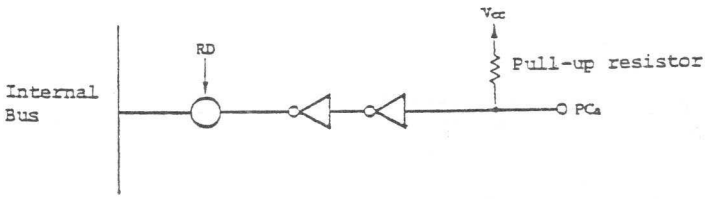


Fig. 1-3 The Port C Structure

1.4 $\overline{WR}$ (Write Strobe) ... Output

This is used as a strobe signal for a write operation for an external memory or I/O. This is at the high level in inactive condition.

1.5 $\overline{RD}$ (Read Strobe) ... Output

This is used as a strobe signal for a read operation for an external memory or I/O. This is at the high level in inactive condition.

1.6 TO (Timer Out) ... Output

The square wave is output from this line. Its cycle time is half of a count time of the internal timer. It goes on low level after reset.

1.7 DB_{7-0} (Data Bus) ... 3-state Input/Output

This is an 8-bit bi-directional data bus. The data move between an external memory or I/O, and accumulator is done through this data bus. During an input, HALT, STOP mode and RESET, the output of the data bus goes an high impedance state. Input/Output level are TTL compatible.

1.8 PE_{15-0} (Port E) ... Output

This is a 16-bit address bus/output port. There are two ways to use these lines as follows: Output level is TTL compatible.

(a) 16-bit Output Port Mode (Only applicable to 78C06/06A)

If users do not want to extend any external memory on the system, they can use the full 16-bit lines of the Port E as general purpose output ports; by a PEX instruction, the contents of the B register and C Register appear on PE_{15-8} and PE_{7-0}, respectively. By an execution of a PEX instruction, this mode can be set, automatically. $\overline{RESET}$ signal also sets the port to this mode (different to μPD78C05/C05A). By $\overline{RESET}$ signal, the output latch buffers are cleared, and its output level goes to low.

(continued)

(b) 16-bit Address Bus Mode

If users want to attach external memories up to 60K bytes to the uCOM-87LC (uPD78C06), they can attain their intention by using all these lines as 16-bit address bus. This mode can be set by a PER instruction. In this mode, the data of the internal address bus are always placed on PE_{15-0} disregarding internal or external memory access. Address bus data are retained in halt-mode and are forced to low in stop-mode.

1.9 $\overline{WAIT}$ (Wait Request) ... Input

If users use rather slow speed external memories or I/O devices in the systems, they can extend a READ/WRITE timing to meet this slow device, by providing a low level signal to this line. The $\overline{WAIT}$ signal is checked at the end of T_2. If it is low, it goes to a wait state (Tw), and it stays in the state until the $\overline{WAIT}$ goes high. Pull-up register is built-in.

1.10 INT0, INT1 (Interrupt Request) ... Input

These are Interrupt Request Input lines. INT0 is a level-sensitive, INT1 is a rising-edge sensitive, respectively. The interrupt priority among the interrupts is shown below.

INT 0 > INT T > INT 1

Here, INTT is internal interrupt.

(a) INT 0

It is a level-sensitive interrupt input line which is high level active.

(b) INT 1

It is a rising-edge sensitive interrupt line, and it becomes valid when INT 1 input goes low to high. Subsequently, if users want to perform the next INT 1 interrupt on this line after an interrupt on this line is accepted, they must take it into consideration that INT 1 input should be maintained at low state a little while, and then it should go high. Unless, it does not enable the next INT 1 interrupt.

In order to avoid a possible mis-operation due to noise signals, both of two interrupt lines are sampled with internal clocks of lus cycle (at 4MHz operation) or 0.67us (at 6MHz operation) in the uPD78C06 or uPD78C06A so that noise signals less than lus and 0.67us are not accepted. Therefore, an interrupt request signal must have an active (high level) time of more than 2us and 1.33us in the uPD78C06 and uPD78C06A, respectively with respect to both INT 0 and INT 1.

1.11 X_1, X_2 (Xtal)

These are connected to the crystal for the internal clock generator circuit. The X_1 is also used as the external clock input, instead of the crystal. Input level of X_1 is CMOS compatible. System clock is a quarter of the crystal frequency or external clock frequency.

1.12 ϕ out (Clock Output) ... Output

In the uPD78C06, uPD78C05 and uPD78C05A 1/4 of crystal oscillation frequency or external clock frequency is output, that is,

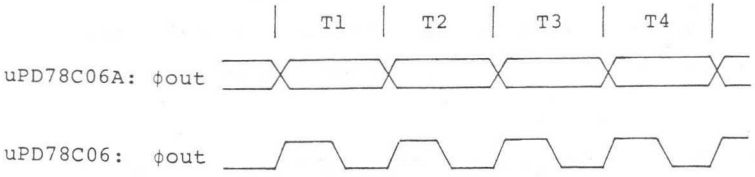
a clock of 1MHz at 4MHz operation.

In the uPD78C06A, a clock of 1/2 of the system clock frequency (1/8 of crystal oscillation frequency or xl external clock frequency), that is, a clock of 750kHz at 6MHz operation is output.

The clock output (φout) continues in the hold mode, but it is fixed to the high level in the stop mode.

The clock output (φout) can be optionally used for synchronization with an external device and other purposes.

In the uPD78C06A, since the clock output (φout) frequency is 1/2 of the system clock frequency, high and low levels are not determined for each basic clock cycle (T1 - T4).



1.13 $\overline{SCK}$ (Serial Clock) ... Input/Output

This is used as control clocks for serial input/output data. The serial clock generated in the internal circuit is placed out in internal serial clock mode, and the external clock is input to this line in external serial clock mode. At the rising edge of $\overline{SCK}$, the data on a SI line is loaded to the Serial Register (S/P), and at the falling edge of $\overline{SCK}$, the contents of the Serial Register appear on a SO line with a bit-order from MSB to LSB. Internal serial clock frequency is a half of system clock frequency (fosc/8).

1.14 SI (Serial Input) ... Input

This is a serial output port, and the data on the SI is loaded to the Serial Register at the rising edge of $\overline{SCK}$. The MSB is start bit.

1.15 SO (Serial Output) ... Output

This is a serial output port, and the data on the Serial Register appears on the SO. The MSB is start bit.

1.16 REL (Release STOP mode) ... Input

This is an input to release the STOP mode of stand-by function. STOP mode is released by raising the REL input high, then clock generator which has been stopped will restart.
During REL input is high, the bit 3 of Stand-by Control Register (SC3) is set to one, and it is reset to zero after **REL signal** returns low. Pull-down resistor is built in.

1.17 $\overline{RESET}$ (Reset) ... Input

When a low level signal over 8us (at 4MHz operation) and 5.34us (at 6MHz operation) is entered in the uPD78C06 and uPD78C06A, respectively, the system is reset and the uPD78C06/uPD78C06A goes to the following conditions;

- All interrupt mask register bits are set, and the all interrupt sources are masked.
- An interrupt enable flag is reset, and all interrupts are disabled.
- All interrupt request flags are reset, accordingly all pending interrupts are reset, too.

- MODE·B register is set to FFH, and the Port B lines become an input mode.
- The bit 6 of Serial Mode Register (SM6) is set, and the serial clock is in external mode.
- The bit 3 of Stand-by Control Register (SC3) is reset, other bits are set. Both the HALT and STOP mode are released.
- PSW are all reset to zero.
- PC is loaded with 0000H.
- The Prescaler and Upcounter in timer circuits are cleared.
- All TIMER·REG bits are set to FFH.
- All Timer Mode Register (TTM) bits are set. As a result, TO goes to low and the timer circuit becomes the mode of adding PRESCALER 0.
- All Port A outputs go to low.
- Port E becomes port mode, and its output level becomes low.
- Data bus (DB7·DB0) goes to high impedance.
- SO output goes to low.
- $\overline{WR}$, $\overline{RD}$ output go to low.
- Other internal registers and RAM, etc. in CPU are not specified.

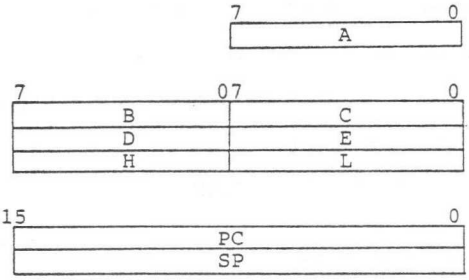
In order to avoid a possible mis-operation due to noise signals, this line does not accept level changes less than 4us and 2.67us by sampling at every 4us (at 4MHz operation) in the uPD78C06 and 2.67us in the uPD78C06A, respectively. Accordingly, a low level over 8us and 5.34us are required in the uPD78C06 and uPD78C06A, respectively to accept $\overline{RESET}$ asynchronously and exactly. Following $\overline{RESET}$ goes to high, program starts from location 0000H.

2. INTERNAL BLOCK FUNCTIONS

2.1 Registers

This mainly consists of the seven 8-bit registers and two 16-bit registers as below.

Fig. 2-1 Register Configuration



(a) General Purpose Registers (B, C, D, E, H, L)

In addition to a role of auxiliary registers for the accumulator, these registers also have a Data Pointer capability if they are used as pair-register (BC, DE, HL). There is an auto-increment/decrement addressing mode capability for the pair-registers of DE, HL, and it can contribute to increase the program efficiency.

(c) Accumulator (A)

Since the uPD78C06/06A has an accumulator based architecture, all the data operation are done through the accumulator.

(d) Program Counter (PC)

This is a 16-bit register to maintain the address information of a program step which should be executed, next. According to a number of bytes needed for an instruction which is going to be fetched, it is automatically incremented, usually. However, in case of executing a branch instruction, an immediate data or content of a register appears on the PC. If a reset signal is issued, or in stop mode the PC is reset to 0000H.

(e) Stack Pointer (SP)

The Stack Pointer is a 16-bit register and is used to maintain a top of the address information of the stack area (Last-In-First-Out Style). The content of the SP is decremented if a CALL or PUSH instruction is executed or interrupt happens, and it is incremented if an RETURN or POP instruction is executed.

2.2 Arithmetic Logic Unit (ALU)

This is used to perform a arithmetic and logic operation such as binary addition/subtraction, decimal adjust, logic and compare operation, and rotation or digit-rotation etc.

2.3 Memory

The uPD78C06/78C06A can directly address the memory up to 64k bytes. Except on-chip ROM (0-4095) and RAM (65,408-65535), any memory location can be used as cither of RAM or ROM, freely. The memory map of the uPD78C06/78C06A is shown on the next page. In the internal ROM area, the Reset/Stop mode Restart Address, Interrupt Start Address, Call Table etc. are included. External memory and on-chip RAM area can be used as data memory (RAM), program memory (ROM), and/or working registers, freely.

(a) Interrupt Start Address

Interrupt Source	Starting Address
INT0	4(0004H)
INTT	8(0008H)
INT1	16(0010H)

Since each interval among interrupt addresses is not the same, it is necessary to put an adequate interrupt service program for pre-treatment of data to interrupts.

(b) Call Address Table

This is used to store the call-address of each one-byte Call instruction over location 128-225 (CALT), up to 64 call-addresses.

(c) Internal Program Memory Area (On-Chip ROM)
(Only applicable to 78C06/06A)

There is on-chip 4k bytes ROM area over location 0-4095 on the memory space. When users write the program onto the internal ROM area, they should take care of the built-in Reset Address, Interrupt Start Address and Call-Table on the ROM area. Users can directly address the memory area over location 2048-4095 with 2-byte Call instruction (CALF).

(d) Internal Data Memory Area (On-Chip RAM)

There is on-chip 128 bytes RAM area over location 65408-65535.

(e) External Extended Memory Area

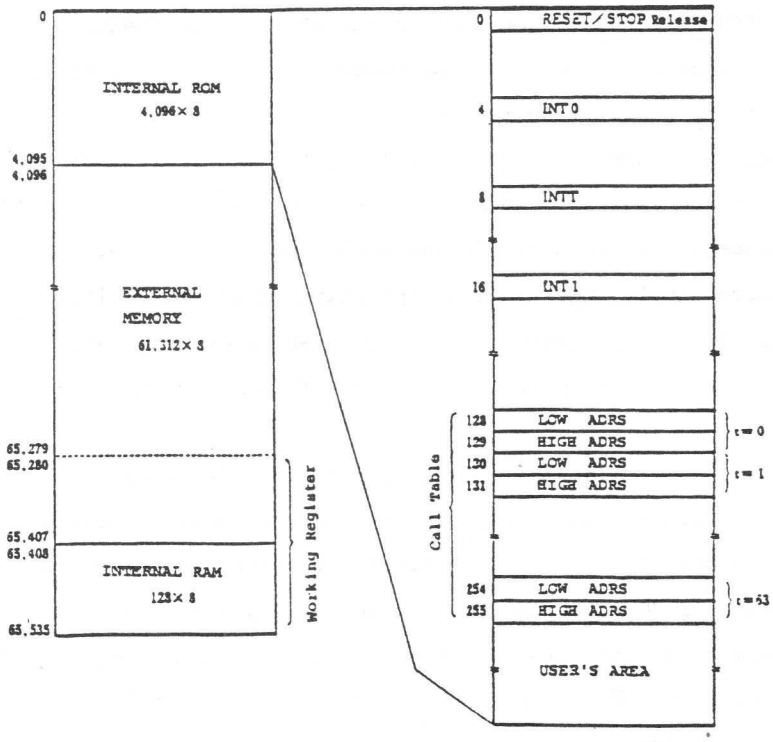
Users can extend data and/or program memory area by using address bus (PE_{15-0}), data bus (DB_{7-0}), and RD, WR Signals up to external 60k bytes memory area (4096-65407).

$\overline{RD}/\overline{WR}$ signals are not output in program area 0-4096 and RAM-area 65408-65535.

(f) Working Register Area

On the location 65280 to 65535, the working registers which contain the 256 bytes area can be located.

Fig. 2-2 The Memory Map



2.4 Serial Interface

Serial interface section (see Fig. 2-3) consists of Serial Input (SI) line, Serial Output (SO) line, Serial Clock ($\overline{SCK}$) input/output line, an 8-bit Serial Register (S/P), an octal counter, a R-S flip-flop used for transfer control, and some gates. The serial clock becomes the internal clock mode fixed to the frequency of 1/2 of the system clock (500kHz at 4MHz operation in the PD78C06 and 750kHz at 6MHz operation in the PD78C06A, respectively) when the bit 6 (SM6) of the serial mode register is set to Ø .
On the other hand, when SM6 is 1, the external clock mode is set and external clocks of DC - 500kHz and DC - 750kHz are entered in the uPD78C06 and uPD78C06A, respectively. Accordingly, the transfer operation in internal clock mode performed synchronously with constant frequency, and in external clock mode it is performed synchronously with variable

A transmitting data is set to serial register by MOV S, A instruction, then the octal counter is reset and serial transfer is triggered by SIO instruction. At every falling edge of $\overline{SCK}$, the contents of serial register are shifted, and shift-out data are placed to SO line with starting bit of MSB. While the $\overline{SCK}$ is low the data on SI line is loaded in continuously, and then latched to serial register at the rising edge of the $\overline{SCK}$. Like this both the input and output of serial data are performed by same $\overline{SCK}$.

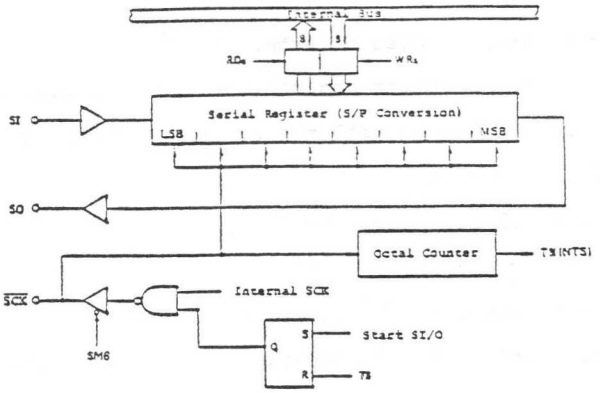


Fig.2-3 The Block Diagram of Serial Ports

After occuring eight $\overline{SCK}$ pulses and completing 8-bit serial data transfer the carry T8 is generated from the octal counter and it sets the interrupt request flag (INTFS). But uPD78C06/78C06A has no serial interrupt, then INTFS is checked by test instruction (SKNIT FS). In internal $\overline{SCK}$ mode, as T8 signal resets the control flip-flop, the following transfer is disabled until next SIO instruction will be given.

Accordingly, the data transfer should be restarted by SIO instruction with the next conditions. In case of data reception, after receiving the data from serial register by MOV A,S instruction, and in case of data transmission, after setting the data to serial register by MOV S,A instruction, data transfer must be done by new SIO instruction.

T8 is also generated in case of external $\overline{SCK}$ mode, however, it has no effect to control the external $\overline{SCK}$, so that it is necessary to control the number of $\overline{SCK}$ by the external $\overline{SCK}$ source side.

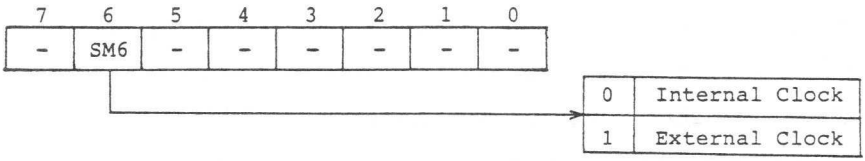
In case of external $\overline{SCK}$ mode, the trigger by SIO instruction are not required basically, but to avoid a mis-operation by noise on $\overline{SCK}$ line, the data transfer should be restarted by SIO instruction which resets the octal counter, after setting or receiving the data to /from serial register as in internal $\overline{SCK}$, after completing 8-bit transfer.

$\overline{RESET}$ input cause the SO to low level and the $\overline{SCK}$ is set to external clock mode.

2.5 Serial Mode Register (SM)

This is an 1-bit register used to specify the serial clock source (internal or external) as the SCK, SM is set to 1 by RESET input, then the external clock is selected as serial clock source, and cleared to 0 by move instruction (MOV SM,A ; A=X0XXXXXX), then the internal clock is selected. SM (SM6) is referenced to bit 6 of accumulator. See following format.

Fig. 2-4 Serial Mode Register Format



2.6 Timer

This is an programmable 8-bit interval timer with prescaler. It consists of TIMER REG (8-bit), PRESCALER0 (4-bit), PRESCALER1 (3-bit), UPCOUNTER (8-bit), COMPARATOR (8-bit), and TIMER F/F.

Count time and TO output are controlled by Timer Mode Register (TMM). Resolution and count time shown in the following table can be selected according to the count data set in the bit 2 (TMM2) of the timer mode register and TIMER·REG.

Count time (us)

Set value of TIMER·REG (HEX)	TMM2 = 0		TMM2 = 1	
	uPD78C06	uPD78C06A	uPD78C06	uPD78C06A
00	8	5.33	128	85.3
01	16	10.66	256	170.6
02	24	15.99	384	255.9
03	32	21.32	512	341.2
≈	≈	≈	≈	≈
FD	2,032	1353.82	32,384	21,666.2
FE	2,040	1359.15	32,640	21,751.5
FF	2,048	1364.48	32,768	21,836.8

Note: It is assumed that the uPD78C06 is operated at 4MHz and the uPD78C06A is operated at 6MHz, respectively.

At first set the count value to the TIMER REG by MOV TM, A instruction, then initialize the PRESCALER0, 1, TIMER F/F, UPCOUNTER and start timer by STM instruction.

UPCOUNTER is counted up at every 8us (TMM2 = 0) or 128us (TMM2 = 1) in the uPD78C06 (at 4MHz operation) and 5.33us (TMM2 = 0) or 85.3us (TMM2 = 1) in the uPD78C06A (at 6MHZ operation).

COMPARATOR always compares the contents of UPCOUNTER with TIMER REG, and it generates match signal (internal interrupt ; INTT) when they are matched. The match signal clears the content of UPCOUNTER, and restarts the countup. Accordingly, this timer operates as the interval timer which generates repetitive interrupt with the interval of count time specified by count value of TIMER REG. When a timer interrupt is generated in HALT mode, the HALT mode is released.

Note that the timer interrupt is disabled by setting the bit 1 of interrupt mask register (MK1) to 1. The content of TIMER F/F whose state is changed by every match signal from COMPARATOR is placed on the TO line, so that the TO signal

becomes the square wave with its half cycle time equivalent to count time.

This output is suitable for driving a piezo buzzer, etc.

When the set value of TIMER REG is assumed to be 00H, a coincidence signal (internal interrupt:INTT) is generated after the first set time elapses and the contents of UPCOUNTER are cleared and the TO output is changed from low to high level. However, unlike the case of the set value other than 00H, the count operation is not repeated thereafter, the coincidence signal (internal interrupt:INTT) is not generated and UPCOUNTER is not cleared. This state continues until any one of the following conditions takes place. (TO output is held high.)

1 $\overline{\text{RESET}}$ input becomes low level.

2 STOP mode is set.

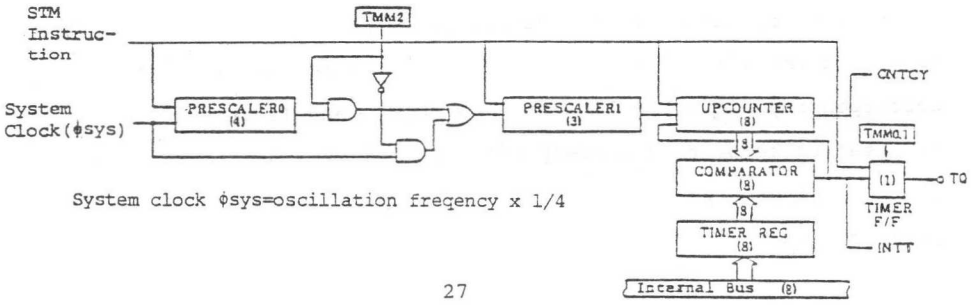
3 STM (Start Timer) instruction is executed.

Therefore, it should be noted that when 00H is set, the timer does not work as an interval timer.

After $\overline{\text{RESET}}$ input or STOP mode is set, the operation is the same as the normal operation.

If STM instruction is executed to go out of this state without changing the set value of TIMER REG (00H is held), the TO output returns to low level immediately after execution of the STM instruction and it is held again at high level after the set time is over.

Fig. 2-5 Timer



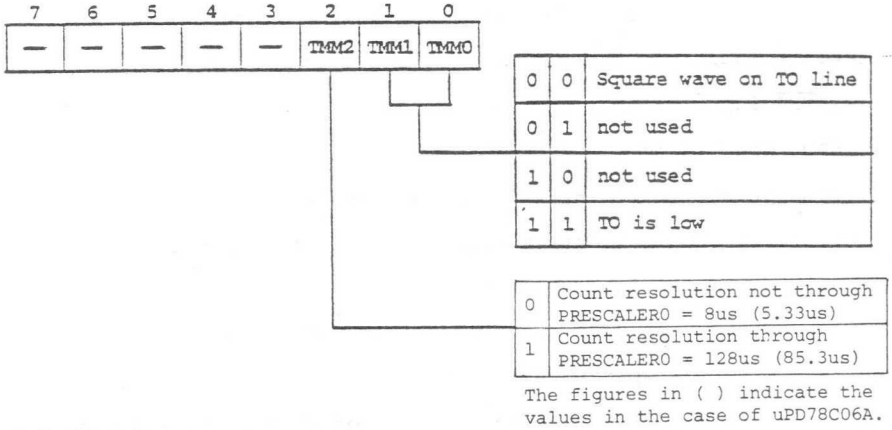
System clock φsys=oscillation freqency x 1/4

2.7 Timer Mode Register (TMM)

This is a 3-bit register used to control the timer operation. Its contents are transferred to/from the accumulator by move instruction. It is referenced to bit 0-2 of accumulator.

Bit 0 and 1 of timer mode register (TMM0,1) enable or disable the square wave output of TO, and bit 2 (TMM2) controls the PRESCALERO.

Fig. 2-6 Timer Mode Register Format



The figures in ( ) indicate the values in the case of uPD78C06A.

2.8 MODE·B

This is an 8-bit Register used for programming the input/output modes of the Port B. With Move instructions (MOV MB, A), programmer can determine the contents of the MODE·B freely. They can program each bit line as either of input or output mode, individually.

If one of the bits of MODE B register is set to one, then corresponding bit of PB line becomes input, and if it is reset to zero, then corresponding bit of PB line becomes output. All MODE B register bits are set to one by $\overline{\text{RESET}}$ input.

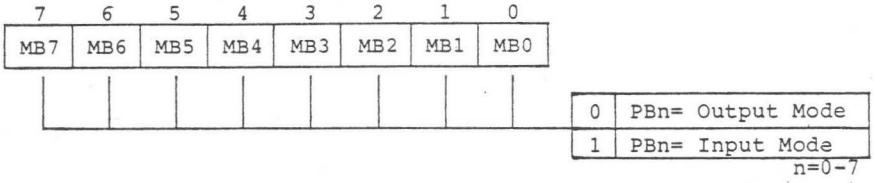


Fig. 2-7 Mode·B Register Format

2.9 Program Status Word (PSW)

It contains the six flags which are set or reset by the results of instruction execution. Two of these flags (Z, CY) can be tested with instructions. The contents of the PSW are automatically saved onto the Stack at interrupt occurrence (External Interrupt, Internal Interrupt), and are retrieved by RETI instruction. All the contents are reset to 0 by the RESET input or STOP mode.

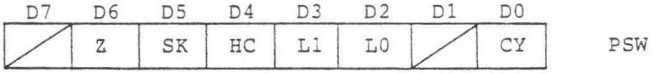


Fig. 2-8 PSW Format

(a) Z (Zero)

If an result of execution is 0, it is set to 1, otherwise it is reset to 0. By instruction, it can be tested.

(b) SK (Skip)

If the Skip condition is just complete, it is set to 1, otherwise it is reset to 0.

(c) HC (Half Carry)

If a Carry from the Bit 3 of the accumulator occurs as a result of operation, it is set to 1, otherwise it is reset to 0.

(d) L1

If a string of MVI A, byte instructions is executed, it is set to 1, otherwise it is reset to 0.

(e) L0

If string of MVI L, byte or LXI H, word instructions is executed, it is set to 1, otherwise it is reset to 0.

(f) CY (Carry)

If a Carry from the Bit 7 of the ALU occurs as a operation result, it is set to 1, otherwise it is reset to 0. By instructions, it can be tested.

(g) The Correlation of flags with instruction executions

By execution of 18 kinds of ALU instruction, rotation instructions and Carry manipulation instructions, the flags of the uPD78C06/78C06A are affected as shown in the following table.

Table 2-1 Flag Operation

Operation					D6	D5	D4	D3	D2	D0
reg. memory		immediate		skip	Z	SK	HC	L1	L0	CY
ADD ADC SUB SBB	ADDX ADCX SUBX SBBX	ADI ACI SUI SBI			↕	0	↕	0	0	↕
ANA ORA XRA	ANAX ORAX XRAX	ANI ORI XRI	ANIW ORIW		↕	0	●	0	0	●
ADDNC SUBNB GTA LTA	ADDNCX SUBNBX GTAX LTAX	ADINC SUINB GTI LTI	GTIW LTIW		↕	↕	↕	0	0	↕
	ONAX OFFAX	ONI OFFI	ONIW OFFIW		↕	↕	●	0	0	●
NEA EQA	NEAX EQAX	NEI EQI	NEIW EQIW		↕	↕	↕	0	0	↕
INR DCR	INRW DCRW				↕	↕	↕	0	0	●
DAA					↕	0	↕	0	0	↕
RLL, RLR					●	0	●	0	0	↕
RLD, RRD					●	0	●	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	●
				SKNC SKNZ SKNIT	●	↕	●	0	0	●
				RETS	●	1	●	0	0	●
All other instructions					●	0	●	0	0	●

↕ ····· Flag Affected (Set or Reset)
1 ····· Flag Set
0 ····· Flag Reset
● ····· Flag Not Affected

2.10 Stand-by Control Register (SC)

The is a 5-bit register used to control the stand-by function, i.e. STOP or HALT mode. Bit 0-2,4 of Stand-by Control Register (SC0-2,4) can be set or reset by loading it with the contents of accumulator with MOV SC, A instruction. Bit 1 and 3 (SC1,3) can be moved to the corresponding bits accumulator with MOV A, SC instruction, and other bits of accumulator (A0, A2, A4-7) will be undefined.

$\overline{\text{RESET}}$ resets SC3, and sets other bits (SC0-2,4).

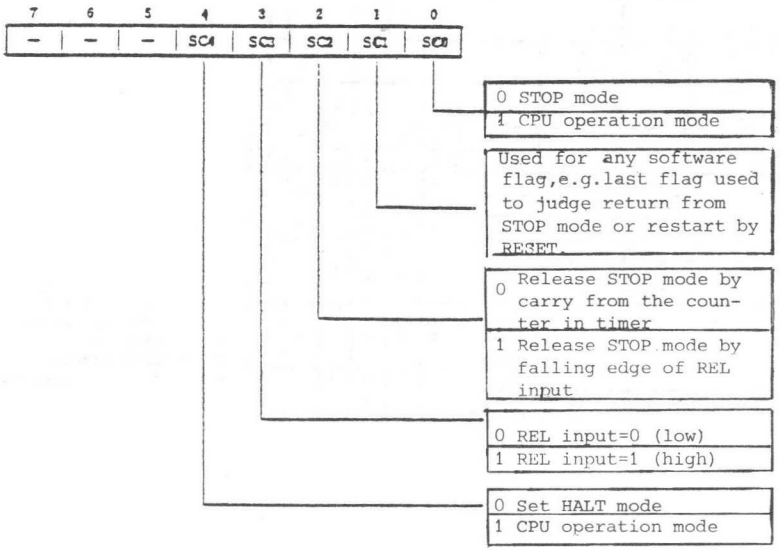


Fig. 2-9 Stand-by Control Register Format

Note that SC3 is not a control bit, but status bit to reflect a state of REL input directly. Accordingly, it will be possible to check the off-chattering of STOP mode release key signals, etc. by checking the content of SC3.

2.11 Interrupt Control Block

There are two external interrupts and one internal interrupt shown at the following, and all these are vectored interrupts.

	Interrupt Source	Starting Address	Priority
External	INT 0 (Level)	4	1
	INT 1 (Rising Edge)	16	3
Internal	INT T (Match on timer comparator)	8	2

The interrupt Control Block contains INTERRUPT REQUEST Register, MASK Register, PRIORITY CONTROL, TEST CONTROL, INTERRUPT ENABLE F/F etc.

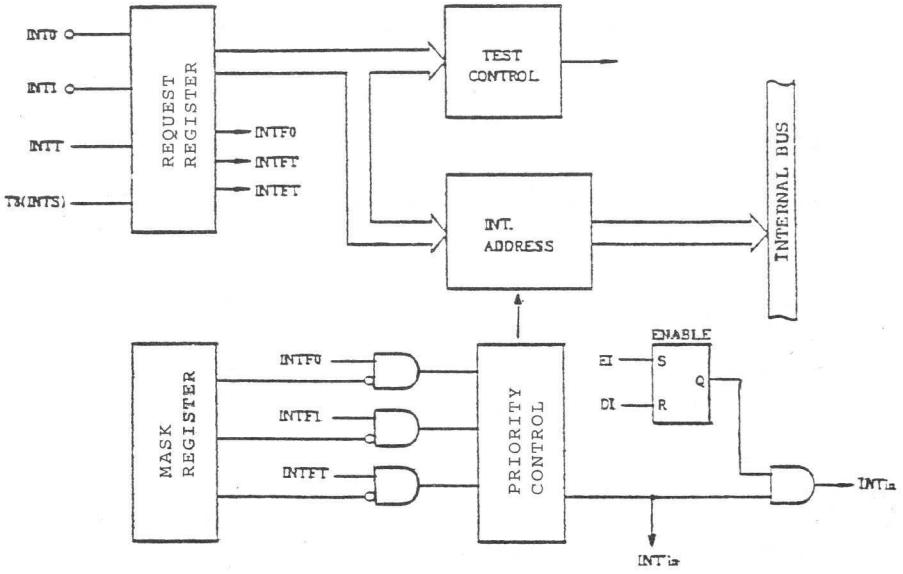


Fig. 2-10 Interrupt Control Block

(a) INTERRUPT REQUEST Register

It contains 3 kinds of interrupt request flags which are set to 1 by each interrupt, individually. By a system reset or STOP mode, all the flags are reset.

· INTF0

This is a flag set by an external level interrupt (INT0). If the line receives a high level signal, this flag is set to 1. If it receives a low level signal, the flag is reset.

· INTF1

By the rising edge of an input signal to the INT1, the flag is set to 1.

· INTFT

This flag is set to 1 by match signal from the comparator in Timer.

· INTFS

If a Serial Register completes the reception of 8-bit data through a SI line or if it completes the transmission of the data through a SO line, the flag is set to 1. However no interrupt is driven by this, and INTFS is checked by on SKNIT instruction.

(b) MASK Register

It contains 3 bits mask flags corresponding to each interrupt. By instructions, each flag can be set or reset, freely. If a MASK bit is 1, the corresponding bit is masked.

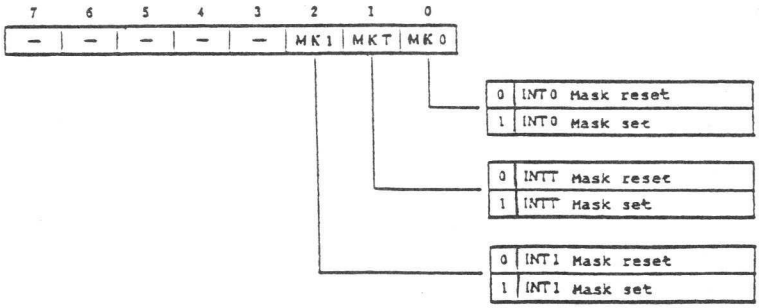


Fig. 2-11 Mask Register Format

(c) PRIORITY CONTROL Circuit

This is the circuit used for controlling the interrupt priority among the interrupts mentioned in the above. If more than two interrupts occur in the system at the same timing, the uPD78C06/uPD78C06A accepts an interrupt which has a higher (or highest) priority than others.

INT 0 > INT T > INT 1

(d) TEST CONTROL Circuit

This circuit operates if instructions are executed which check a status of interrupt request flags (INTF0, INTF1, INTFT) or a test flag showing serial tranfer completion (INTFS).

(e) INTERRUPT ENABLE F/F

It is set by EI instruction, and is reset by DI instruction. Once any one of interrupts is accepted, it is reset. If this F/F is set to 1, it means 'Interrupt Enable'. If it is reset to 0, it means 'Interrupt Disable'. By a RESET input or STOP mode, it is also reset.

3. INTERRUPT OPERATION

There are two external interrupts (INT0 and INT1) and one internal interrupt (INTT). All of them are vectored interrupts and interrupt addresses are as shown below.

In the data concerning interrupt signal sampling time used in the following description, the figures put in ( ) indicate the values at 6MHz operation of uPD78C06A and those not put in ( ) the values at 4MHz operation of uPD78C06.

Interrupt source			Interrupt address	
			Decimal	Hexadecimal
External	INT0	Detection of high level	4	0004
	INT1	Detection of leading edge	16	0010
Internal	INTT	Coincidence of timer comparator	8	0008

3.1 Priority level

When several interrupt requests not masked by the mask register occur at a time in EI state, several interrupt request flags (INTFx) are set. In such cases, the internal priority level control circuit selects the one of highest priority among those shown below and accepts that request only and reserves the others.

INT0 > INTT > INT1

The flag of the accepted interrupt request is reset, but those of the others remain in being set until these interrupt requests are accepted.

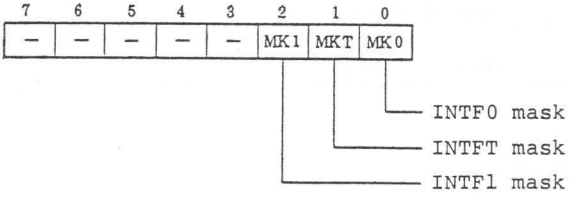
3.2 Masking

All interrupt requests are masked (disabled) by DI instruction (resetting interrupt enable F/F) and permitted by EI instruction (setting interrupt enable F/F).

Even in EI state, masking is possible for each interrupt according to the mask register contents.

When the corresponding bit of the mask register is set (1), that interrupt is masked and it is permitted by resetting the bit (0).

Fig. 3-1 Mask Register Function



The mask function is given strictly to the interrupt request flag (INTFx) which has been set by being recognized as a normal interrupt. Whether an external interrupt is permitted depends upon the external interrupt detect circuit having a sampling pulse of 1us (0.67us) cycle.

3.3 Sampling of External Interrupt

Either INT0 or INT1 is detected in either active or inactive level by a sampling pulse of 1us cycle (0.67us)

in order to prevent a malfunction due to a noise signal. (See Fig. 3-2.)

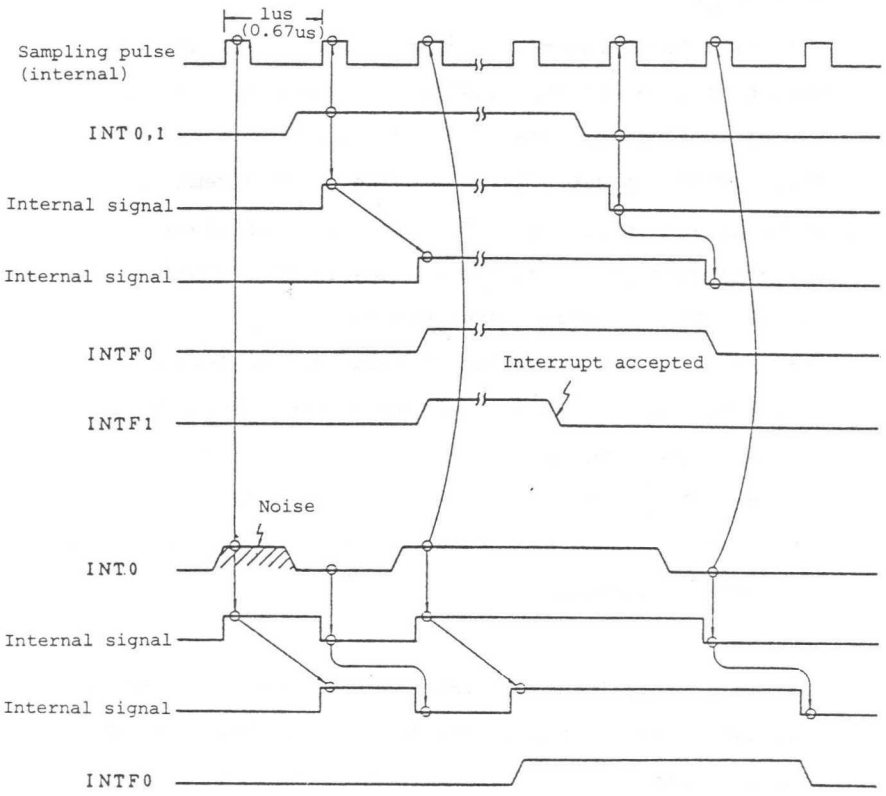
(1) INT0 input

This is a level interrupt input of high level active. When high level of INT0 signal with the sampling pulse of lus (0.67us) or more is detected, INT0 request flag (INTF0) is set after one pulse. INTF0 remains in being set as long as INT0 signal is high level. When the sampling pulse detects low level of INT0 signal, INTF0 is reset after one pulse.
Therefore, as INTF0 remains in being set as long as INT0 signal is high level even after interrupt service routine takes place and INT0 signal is high level, reset INT0 signal by informing the INT0 signal source that the signal has been accepted by using the output port in the interrupt processing routine.

(2) INTl input

This is an edge interrupt input of leading edge active. When INTl signal changes from low to high level and the sampling pulse detects that high level of lus (0.67us) is being held, INTl request flag (INTFl) is set.
When it is confirmed at the end of instruction that INTFl is being set, and if no other interrupt request flag with higher priority level is set, it jumps to the interrupt address corresponding to INTl interrupt request.
When an interrupt is accepted, INTFl is automatically reset.
A new INTl interrupt request is detected when INTl signal goes once to low and then high level, and INTFl is set.

Fig. 3-2 Sampling of Interrupt



As seen from the above description, as INT0 and INT1 are sampled by a clock of lus (0.67us) cycle, a noise signal of less than lus (0.67us) is eliminated.
An interrupt is completely accepted with a high level input of 2us (1.33us) or more.

4. Interrupt Procedure

Each interrupt request is processed by the following procedure.

1) Interrupt request flags are checked at T1 timing of first machine cycle of every instruction, and when it is set interrupt sequence will start. However the masked interrupt requests are not checked.

2) If more than two interrupts are set at same time, then their priorities will be checked and the highest priority interrupt request is acknowledged, and others are pended.

(INT 0 > INT T > INT 1)

3) Reset interrupt enable flip-flop, then all interrupts are disabled.

4) Reset the interrupt request flag which is acknowledged, except for level-activated interrupt (INT0).

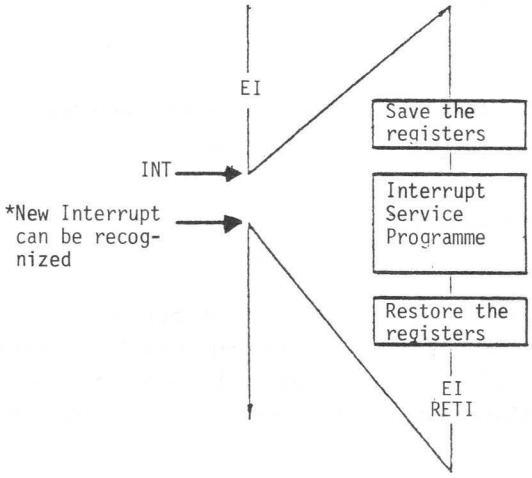
5) Save the PSW, upper byte of PC, lower byte of PC onto the stack in this sequence.

For the whole interrupt procedure (without sampling-test for external interrupts) the uPD78C06/78C06A requires 4 internal clock cycles (4μ sec/2,68μ sec); of course the processor has to finish the actual instruction before going to interrupt sub-routine.

6) Jump to each interrupt starting address.

After execution of interrupt service program, it should be performed to return to the address where the interrupt was acknowledged. First, the programme restores registers or flags and interrupt enable flag is set by EI instruction.

Then the lower byte of PC, the upper byte of PC and PSW are restored in this sequence by RETI instruction. To avoid a stack overflow another interrupt will be accepted after one instruction has been executed subsequent to EI instruction. This means a new interrupt can be recognized after completing an execution of RETI instruction following EI instruction and, as a result, the stack is restored completely.



*: If an interrupt is acknowledged at location N, programme returns to location N after completing interrupt procedure.

Fig. 4-1 Interrupt Sequence

5. Stand-by operation

Stand-by function is used to decrease the power consumption in stand-by condition, and there are two types of it, HALT mode and STOP mode. It is specified to HALT mode by resetting the bit 4 of Stand-by Control Register(SC4), or STOP mode by resetting the bit 0 (SC0). HALT mode can be released by one of the external interrupt (INT0, 1), timer interrupt (INTT), carry form the serial clock counter (T8), and RESET signal. When HALT mode is released by an interrupt, program jumps to corresponding interrupt starting address in EI condition, or steps to the instruction following the HALT mode setting instruction (MOV SC,A) in DI condition.

Yet in HALT mode the masking function is active, so that programmer can choose an interrupt source for release use.

When HALT mode is released by T8, it returns the program control to an instruction of the main routine which is placed immediately after the instruction activating the HALT mode.

When HALT mode is released by $\overline{\text{RESET}}$, normal reset operation will be performed and program jumps to location 0.

STOP mode can be released by REL or $\overline{\text{RESET}}$ signal, and there are two ways of releasing (shown following) by REL signal according to the content of bit 2 of Stand-by Control Register (SC2).

1) In case of SC2=0

Start the oscillator and timer by rising edge of REL signal, and start to provide the internal clock after four carries (i.e. after 1,024 counts) from UPCOUNTER in timer, then program will start at location 0.

2) In case of SC2=1

Start the oscillator by rising edge of REL signal and inhibit to provide the internal clock during REL signal raised to high, and restart to provide the internal clock after REL signal returns to low, then program will start at location 0.

When STOP mode is released by RESET, normal reset operation will be performed, so that the oscillator will start at the falling edge of RESET, and program will start at location 0 by next rising edge. Accordingly, $\overline{\text{RESET}}$ should be held at low level sufficient time for oscillator to become stable.

Table 5-1 HALT Mode and STOP Mode

Parameter	HALT mode	STOP mode
Oscillator	Run	Stop
Internal System Clock	Stop	
Timer	Run	
TIMER · REG	Hold	Set
UPCOUNTER, PRECALER 0, 1	Run	Cleared
Serial Interface		Run *1
Serial Clock	Hold	Hold
Interrupt Control Circuit	Run	Stop
Interrupt Enable Flag	Hold	Reset
INT0, INT1 Input	Active	Inactive
INTT		—
T8 (INTFS)		—
MASK Register	Hold	Set
Pending Interrupts (INTFX)		Reset
REL Input	Inactive	Active
RESET Input	Active	
On-Chip RAM	Hold	Hold
Output Latch in Port A, B, E		
Program Counter (PC)		Cleared
Stack Pointer (SP)		Unkown
General Registers (A, B, C, D, E, H, L)		
Program Status Word (PSW)		Reset
MODE B·Register		Hold
Stand-by Control Register (SCI-SC3)		
Stand-by Control Register (SC4)		Set
Timer Mode Register (TMM0,1)		Hold
Timer Mode Register (TMM2)		Set
Serial Mode Register (SM)		Hold
Data Bus (DB0-7)	High-Z	High-Z
RD, WR Output	High	High

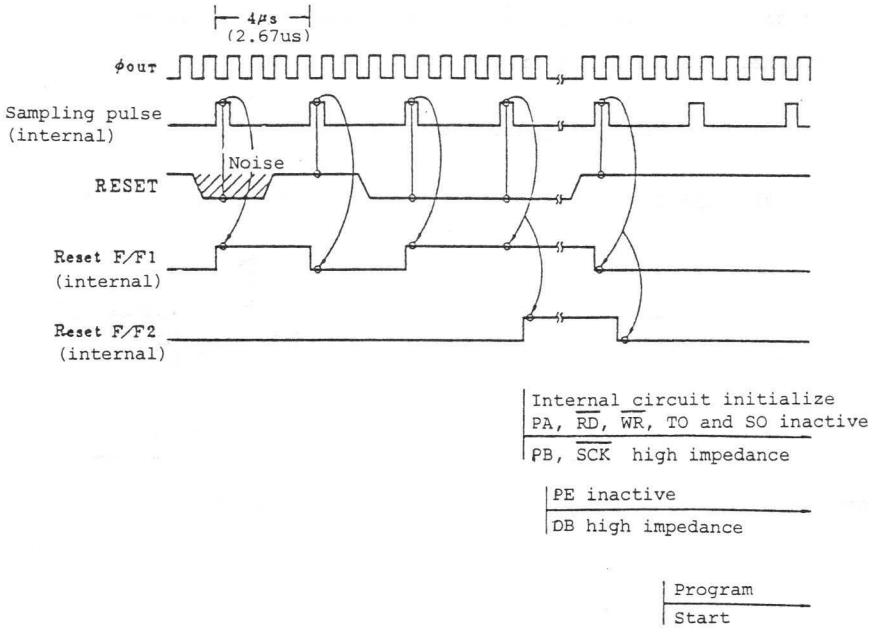
Note 1. Serial clock counter are running and T8 is generated, however, there are no effect by it.

Note 2. If PE is used as address bus data are retained in halt-mode-bus and forced low by stop-mode.

6. RESET OPERATION

When two sampling pulses continuously sample low level of $\overline{\text{RESET}}$ signal, the $\overline{\text{RESET}}$ signal is judged correct. The sampling signal is of 4us cycle (at 4MHz operation) in the uPD78C06 and 2.67us cycle in the uPD78C06A, respectively. Therefore, the $\overline{\text{RESET}}$ signal needs to be kept low for 8us and 5.34us, respectively in order to be recognized as a correct $\overline{\text{RESET}}$ signal in the asynchronous case. When the $\overline{\text{RESET}}$ signal is recognized (the internal Reset F/F2 is set), the program counter (PC) is cleared and the other internal circuits are initialized. Outputs of PA7 ∿ PA0, TO and SO become low level. $\overline{\text{RD}}$ and $\overline{\text{WR}}$ outputs become high level (respective inactive level). PB7 ∿ PB0 and $\overline{\text{SCK}}$ of input/output line become high impedance (input mode). With a delay of one clock, PE15 ∿ PE0 become low level and DB7 ∿ DB0 become high impedance. When high level of $\overline{\text{RESET}}$ signal is sampled, the internal Reset F/F1 is immediately reset and after one clock, the internal Reset F/F2 is reset. After further two clocks, the program starts (from address 0). However, since the address 0 is an internal ROM, neither address nor $\overline{\text{RD}}$ signal is output (uPD78C05/C05A behave in different way). The $\overline{\text{RESET}}$ signal has the priority over all control signals including interrupt.

Fig. 6-1 Reset Timing

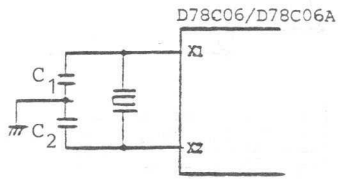


7. Clock Driver Circuit

You may drive the clock input (X1, X2) of the uPD78C06/78C06A with a crystal, or an external clock source. The driving frequency must be four times the desired internal system clock frequency.

Fig. 7-1 Clock Driver Circuit

a) Crystal

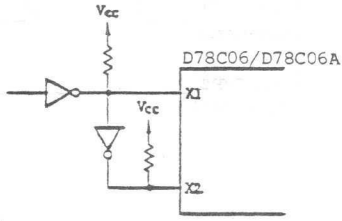


Input frequency: ∿4MHz (uPD78C06),
∿6MHz (uPD78C06A)

$C_{1/2}$ + stray capacity < 20pF

$C_1 \approx C_2$

b) External Source



Input frequency: ∿4MHz (uPD78C06),
∿6MHz (uPD78C06A)

Note:

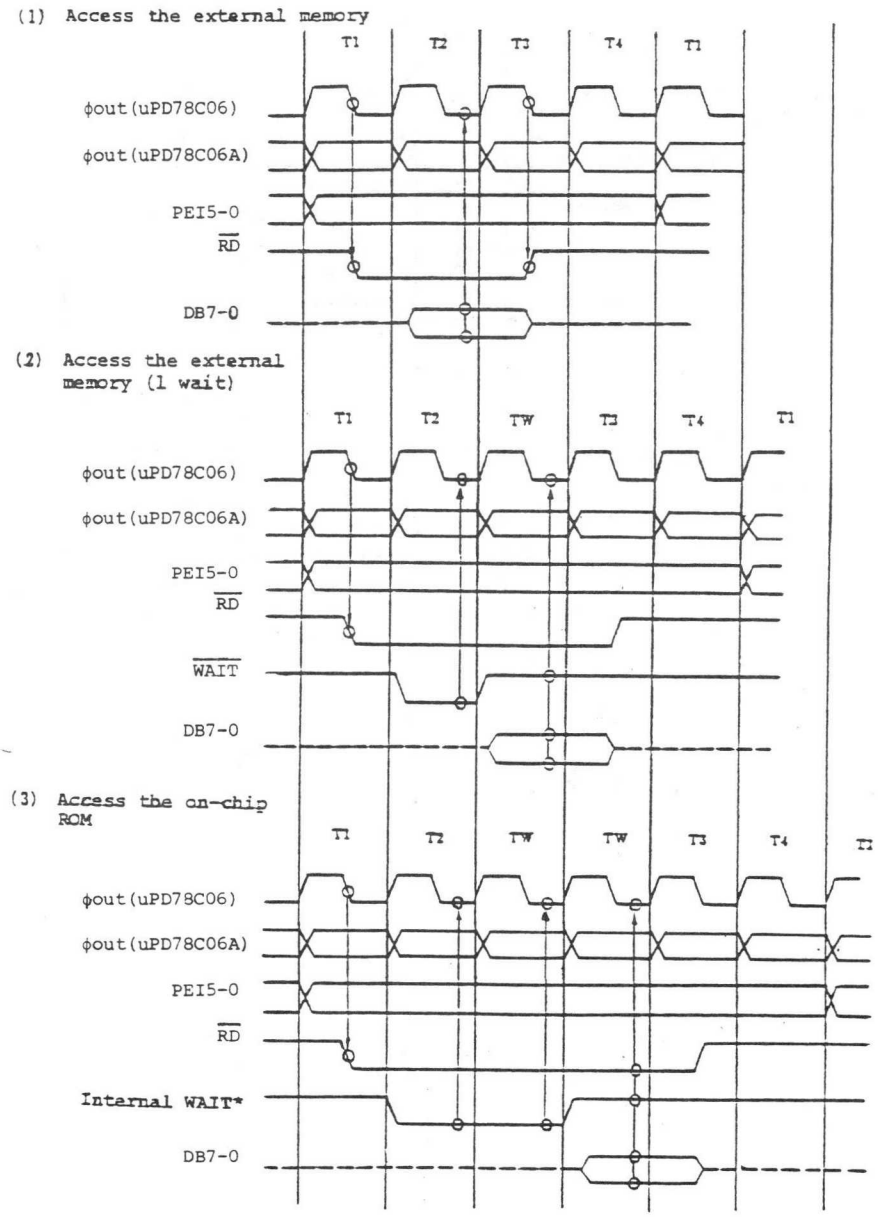
a) efficient reduction of power consumption (entering stop mode) only possible with internal clock.

b) ceramic resonators can be applied, too. Please inform you at your NEC-office.

8. Bus Timing

8.1 OP Code fetch

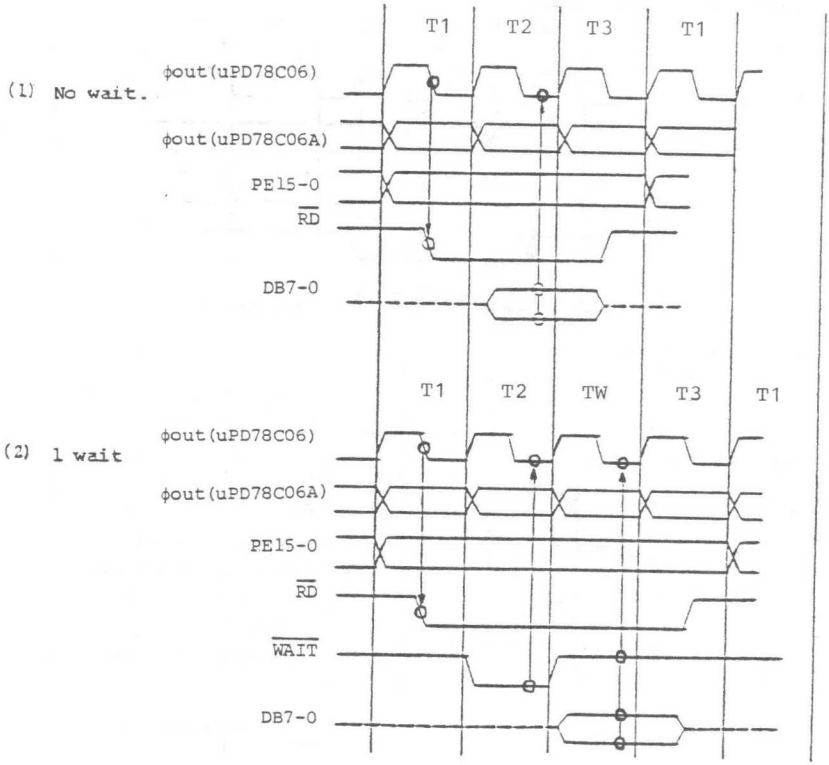
Fig. 8-1 OP code fetch (1st or 2nd) timing



* Two wait cycles are automatically inserted when the on-chip ROM access.

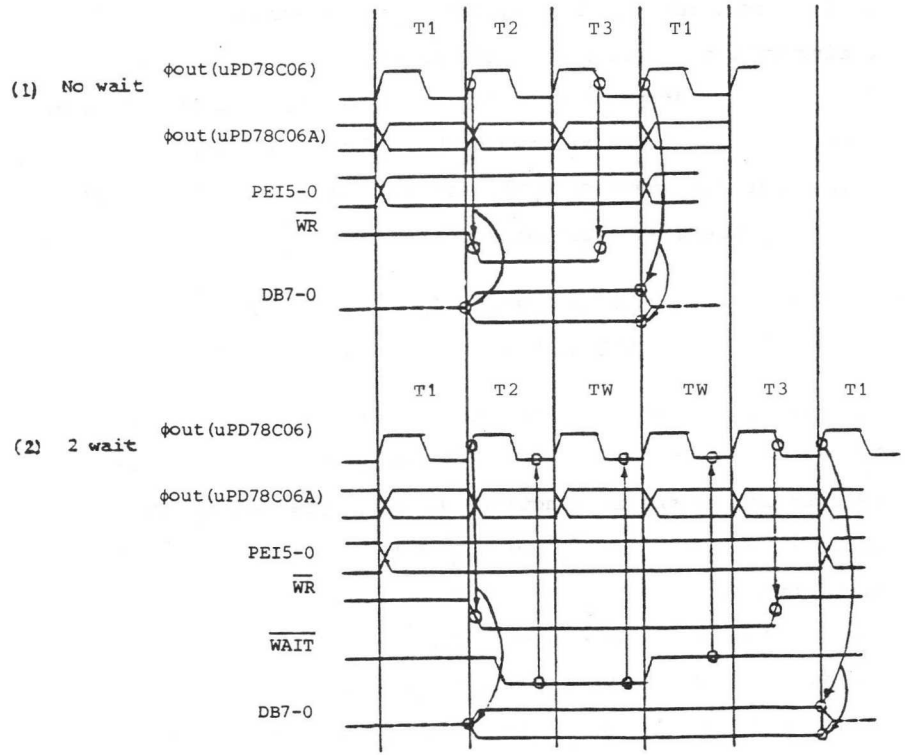
8.2 Memory read

Fig. 8-2 Memory read timing



8.3 Memory write

Fig. 8-3 Memory write timing



9. String of three 'special' Instructions

When more than one certain instruction among these three occur in sequence, only the first instruction encountered will be executed. The remainder of the instructions in the string will be ignored and just be replaced with idle clocks ; they require same number of clocks as usual execution of these instructions. There are two groups of the instructions which have the above special feature, and these two groups are independent of each other.

Group A:	MVI A, byte	(L1 Flag)
Group B:	MVI L, byte : LXI H. word	(L0 Flag).

If there is a string of the instructions (MVI A, byte) , belonging to the group A, the L0 flag is set to 1. If there is a string of the instructions (MVI L, byte and/or LXI H, word) belonging to the group B, the L0 flag is set to 1. An interrupt is not disabled during the execution of a string of these instructions. Since the L flags are saved to the stack during an interrupt operation, the uPD78C06/78C06A can resume to execute the peculiar feature after returning from the interrupt procedure.

```
Start ──→ MVI A, 0          ;A ← 0
          MVI A, 1          ;NOP(11CLOCKS), L1=1
          MVI A, 2          ;NOP(11CLOCKS), L1=1
          MVI L, 0AH        ;L ← 0AH
          MVI L, 0BH        ;NOP(11CLOCKS), L0=1
Interrupt→MVI L, 0CH        ;NOP(11CLOCKS), L0=1
          LXI H, 00H        ;NOP(16CLOCKS), L0=1
```

10. uCOM-87LC Instruction Set

10.1 Symbols/Description on Operand

Symbols	Descriptions
r r1 r2	A, B, C, D, E, H, L B, C, D, E, H, L A, B, C
sr sr1 sr2	PA PB MK MB TM S TMM SM SC PA PB PC MK S TMM SC PA PB PC MK
rp rp1	SP, B, D, H V, B, D, H
rpa wa	B, D, H, D+, H+, D−, H− 8 bit immediate data
word byte	16bit 8 bit
if	F0, F1, FT, FS,
f	CY, Z

(Notes)

1. At sr~sr2, the symbols of 'PA', 'PB', etc. stand for the following, respectively:

   PA=PORTA, PB=PORTB, PC=PORTC, MK=MASK·reg, MB=MODE·B, TM=TIMER·REG, S=SERIAL I/O, TMM=TIMER MODE REG, SM=SERIAL MODE REG, SC=STANDBY CONTROL REG

2. At rp~rp1, the 'SP', 'B', etc. stand for the following, respectively:

   SP=STACK POINTER, B=BC, D=DE, H=HL, V=FFH·A

3. At rpa, the 'B' 'D', etc. stand for the following, respectively:

   B=(BC), D=(DE), H=(HL), D+=$(DE)^{+}$, H+=$(HL)^{+}$,
   D-=$(DE)^{-}$, H-=$(HL)^{-}$

4. At if, the 'F0', 'F1', etc. stand for the following, respectively:

   F0=INTF0, F1=INTF1, FT=INTFT, FS=INTFS

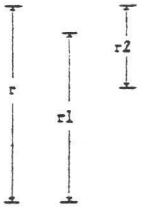
5. At f, the 'CY', 'Z', stand for the following, respectively.

   CY=CARRY, Z=ZERO

10.2 The description of the symbols on Operation Codes is as follows:

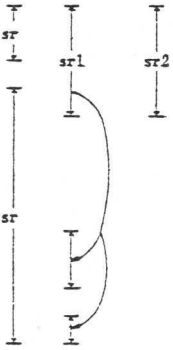
r

R_2	R_1	R_0	reg
0	0	0	
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



sr

S_3	S_2	S_1	S_0	special reg
0	0	0	0	PORT A
0	0	0	1	PORT B
0	0	1	0	PORT C
0	0	1	1	MASK
0	1	0	0	MODE · B
0	1	0	1	—
0	1	1	0	TIMER · REG
0	1	1	1	—
1	0	0	0	SERIAL · I/O
1	0	0	1	TIMER MODE REG.
1	0	1	0	SERIAL MODE REG.
1	0	1	1	STANDBY CONTROL REG.

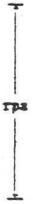


rp

P_1	P_0	reg·pair
0	0	SP
0	1	BC
1	0	DE
1	1	HL

rpa

A_2	A_1	A_0	addressing
0	0	0	—
0	0	1	(BC)
0	1	0	(DE)
0	1	1	(HL)
1	0	0	(DE)$^+$
1	0	1	(HL)$^+$
1	1	0	(DE)$^-$
1	1	1	(HL)$^-$



rpl

Q_1	Q_0	reg·pair
0	0	FFHLA
0	1	BC
1	0	DE
1	1	HL

if

I_2	I_1	I_0	INTF
0	0	0	INTF 0
0	0	1	INTFT
0	1	0	INTF 1
0	1	1	—
1	0	0	INTFS

f

F_2	F_1	F_0	flag
0	1	0	CY
1	0	0	Z

	Mnemonics	Operand	Op Codes B1	B2	B3	B4	Clock Cycle	Functions	Skip Condition
8-Bit Data Move Instructions	MOV	r1, A	0 0 0 1 1 $R_2R_1R_0$				6	r1←A	
	MOV	A, r1	0 0 0 0 1 $R_2R_1R_0$				6	A←r1	
	MOV	sr, A	0 1 0 0 1 1 0 1	1 1 0 0 $S_2S_1S_0$			14	sr←A	
	MOV	A, sr1	0 1 0 0 1 1 0 0	1 1 0 0 $S_2S_1S_0$			14	A←sr1	
	MOV	r, word	0 1 1 1 0 0 0 0	0 1 1 0 1 $R_2R_1R_0$	Low Adrs	High Adrs	25	r←(word)	
	MOV	word, r	0 1 1 1 0 0 0 0	0 1 1 1 1 $R_2R_1R_0$	Low Adrs	High Adrs	25	(word)←r	
	MVI	r, byte	0 1 1 0 1 $R_2R_1R_0$	← Data →			11	r←byte	
	STAW	wa	0 0 1 1 1 0 0 0	← Offset →			14	(FFH. wa)←A	
	LDAW	wa	0 0 1 0 1 0 0 0	← Offset →			14	A←(FFH. wa)	
	STAX	rpa	0 0 1 1 1 $A_2A_1A_0$				9	(rpa)←A	
	LDAX	rpa	0 0 1 0 1 $A_2A_1A_0$				9	A←(rpa)	
	SDCD	word	0 1 1 1 0 0 0 0	0 0 0 1 1 1 1 0	Low Adrs	High Adrs	28	(word)←C, (word+1)←B	
	SDED	word		0 0 1 0 1 1 1 0			28	(word)←E, (word+1)←D	
	SHLD	word		0 0 1 1 1 1 1 0			28	(word)←L, (word+1)←H	
	SSPD	word		0 0 0 0 1 1 1 0			28	(word)←SP'_L (word+1)←SP_H	

	Mnemonics	Operand	Op Codes B1	Op Codes B2	Op Codes B3	Op Codes B4	Clock Cycle	Functions	Skip Condition
16-Bit Data Move Instructions	LBCD	word	01110000	00011111	Low Adrs	High Adrs	28	C←(word), B←(word+1)	
	LDED	word		00101111			28	E←(word), D←(word+1)	
	LHLD	word		00111111			28	L←(word), H←(word+1)	
	LSPD	word		00001111			28	SP_L←(word), SP_H←(word+1)	
	PUSH	rp1	01001000	00$Q_1Q_0$1110			21	(SP−1)←rp1H, (SP−2)←rp1L	
	POP	rp1	01001000	00$Q_1Q_0$1111			18	rp1L←(SP), rp1H←(SP+1) SP←SP+2	
	LXI	rp, word	00$P_1P_0$0100	——Low Byte——	High Byte		16	rp←word	
Arithmetic Instructions	ADD	A, r	01100000	11000$R_2R_1R_0$			12	A←A+r	
	ADDX	rpa	0111	11000$A_2A_1A_0$			15	A←A+(rpa)	
	ADC	A, r	0110	11010$R_2R_1R_0$			12	A←A+r+CY	
	ADCX	rpa	0111	11010$A_2A_1A_0$			15	A←A+(rpa)+CY	
	SUB	A, r	0110	11100$R_2R_1R_0$			12	A←A−r	
	SUBX	rpa	0111	11100$A_2A_1A_0$			15	A←A−(rpa)	
	SBB	A, r	0110	11110$R_2R_1R_0$			12	A←A−r−CY	
	SBBX	rpa	0111	11110$A_2A_1A_0$			15	A←A−(rpa)−CY	
	ADDNC	A, r	0110	10100$R_2R_1R_0$			12	A←A+r	No Carry
	ADDNCX	rpa	0111	10100$A_2A_1A_0$			15	A←A+(rpa)	No Carry
	SUBNB	A, r	0110	10110$R_2R_1R_0$			12	A←A−r	No Borrow
	SUBNBX	rpa	0111	10110$A_2A_1A_0$			15	A←A−(rpa)	No Borrow

	Mnemonics	Operand	Op Codes				Clock Cycle	Functions	Skip Condition
			B1	B2	B3	B4			
Logic Instructions	ANA	A, r	01100000	10001 $R_2R_1R_0$			12	A←A∧r	
	ANAX	rpa	0111	10001 $A_2A_1A_0$			15	A←A∧(rpa)	
	ORA	A, r	0110	10011 $R_2R_1R_0$			12	A←A∨r	
	ORAX	rpa	0111	10011 $A_2A_1A_0$			15	A←A∨(rpa)	
	XRA	A, r	0110	10010 $R_2R_1R_0$			12	A←A∨r	
	XRAX	rpa	0111	10010 $A_2A_1A_0$			15	A←A∨(rpa)	
	GTA	A, r	0110	10101 $R_2R_1R_0$			12	A−r−1	No Borrow
	GTAX	rpa	0111	10101 $A_2A_1A_0$			15	A−(rpa)−1	No Borrow
	LTA	A, r	0110	10111 $R_2R_1R_0$			12	A−r	Borrow
	LTAX	rpa	0111	10111 $A_2A_1A_0$			15	A−(rpa)	Borrow
	ONAX	rpa	0111	11001 $A_2A_1A_0$			15	A∧(rpa)	No Zero
	OFFAX	rpa	0111	11011 $A_2A_1A_0$			15	A∧(rpa)	Zero
	NEA	A, r	0110	11101 $R_2R_1R_0$			12	A−r	No Zero
	NEAX	rpa	0111	11101 $A_2A_1A_0$			15	A−(rpa)	No Zero
	EQA	A, r	0110	11111 $R_2R_1R_0$			12	A−r	Zero
	EQAX	rpa	0111	11111 $A_2A_1A_0$			15	A−(rpa)	Zero

	Mnemonics	Operand	Op Codes B1	B2	B3	B4	Clock Cycle	Functions	Skip Condition
Immediate Operation Instructions (Accumulator)	XRI	A, byte	00010110	Data			11	A←A∨byte	
	ADINC	A, byte	0010				11	A←A+byte	No Carry
	SUINB	A, byte	0011				11	A←A−byte	No Borrow
	ADI	A, byte	0100				11	A←A+byte	
	ACI	A, byte	0101				11	A←A+byte+CY	
	SUI	A, byte	0110				11	A←A−byte	
	SBI	A, byte	0111				11	A←A−byte−CY	
	ANI	A, byte	00000111				11	A←A∧byte	
	ORI	A, byte	0001				11	A←A∨byte	
	GTI	A, byte	0010				11	A−byte−1	No Borrow
	LTI	A, byte	0011				11	A−byte	Borrow
	ONI	A, byte	0100				11	A∧byte	No Zero
	OFFI	A, byte	0101				11	A∧byte	Zero
	NEI	A, byte	0110				11	A−byte	No Zero
	EQI	A, byte	0111				11	A−byte	Zero
(Special Register)	ANI	sr2, byte	01100100	100010SrSs	Data		23	sr2←sr2∧byte	
	ORI	sr2, byte		1001			23	sr2←sr2∨byte	
	ONI	sr2, byte		1100			20	sr2∧byte	No Zero
	OFFI	sr2, byte		1101			20	sr2∧byte	Zero

	Mnemonics	Operand	Op Codes B1	B2	B3	B4	Clock Cycle	Functions	Skip Condition
Working Register Operation Instructions	ANIW *	wa, byte	00000101	Offset	Data		22	(FFH.wa)←(FFH.wa)∧byte	
	ORIW	wa, byte	0001				22	(FFH.wa)←(FFH.wa)∨byte	
	GTIW	wa, byte	0010				19	(FFH.wa)−byte−1	No Borrow
	LTIW	wa, byte	0011				19	(FFH.wa)−byte	Borrow
	ONIW	wa, byte	0100				19	(FFH.wa)∧byte	No Zero
	OFFIW	wa, byte	0101				19	(FFH.wa)∧byte	Zero
	NEIW	wa, byte	0110				19	(FFH wa)−byte	No Zero
	EQIW	wa, byte	0111				19	(FFH wa)−byte	Zero
Increment/Decrement Instructions	INR	r2	010000R₁R₀				6	r2←r2+1	Carry
	INRW	wa	00100000	Offset			17	(FFH.wa)←(FFH.wa)+1	Carry
	DCR	r2	010100R₁R₀				6	r2←r2−1	Borrow
	DCRW	wa	00110000	Offset			17	(FFH.wa)←(FFH.wa)−1	Borrow
	INX	rp	00P₁P₀0010				9	rp←rp+1	
	DCX	rp	00P₁P₀0010				9	rp←rp−1	
Other Operation Instructions	DAA		01100001				6	Decimal Adjust Accumulator	
	STC		01001000	00101011			12	CY←1	
	CLC		01001000	00101010			12	CY←0	

* Working register commands are indicated by an appended 'W'

	Mnemonics	Operand	Op Codes B1	B2	B3	B4	Clock Cycle	Functions	Skip Condition
Rotation Instructions	RLD		01001000	00111000			21	$A_{3-0} \leftarrow (HL)_{7-4} \leftarrow (HL)_{3-0}$ $(HL)_{3-0} \leftarrow A_{3-0}$	
	RRD			1001			21	$A_{7-4} \leftarrow (HL)_{3-0} \leftarrow (HL)_{7-4}$ $A_{3-0} \leftarrow (HL)_{3-0}$	
	RLL	A		0000			12	$A_{m+1} \leftarrow A_m$, $A_0 \leftarrow CY$, $CY \leftarrow A_7$	
	RLR	A		0001			12	$A_{m-1} \leftarrow A_m$, $A_7 \leftarrow CY$, $CY \leftarrow A_0$	
Jump Instructions	JMP	word	01010100	← Low Adrs →	High Adrs		16	PC←word	
	JB		01111011				6	$PC_H \leftarrow B$ $PC_L \leftarrow C$	
	JR	word	11 ← jdisp →				12	$PC \leftarrow PC+1+jdisp$ Max. $^{+31}_{-32}$ bytes	
	JRE	word	01001111 ←	jdisp →			17	$PC \leftarrow PC+2+jdisp$ Max. $^{+255}_{-256}$ bytes	
Call Instructions	CALL	word	01000100	← Low Adrs →	High Adrs		22	$(SP-1) \leftarrow (PC+3)_H$, $(SP-2) \leftarrow (PC+3)_L$, PC←word	
	CALF	word	01111 ←	fa →			17	$(SP-1) \leftarrow (PS+2)_M$, $(SP-2) \leftarrow (PC+2)_L$ $PC15-11 \leftarrow 00001$, $PC10-0 \leftarrow fa$ *1	
	CALT	word	10 ← fa →				21	$(SP-1) \leftarrow (PC+1)_H$, $(SP-2) \leftarrow (PC+1)_L$ $PC_L \leftarrow (128+2fa)$, $PC_H \leftarrow (129+2fa)$ *2	
Return Instructions	RET		00001000				12	$PC_L \leftarrow (SP)$, $PC_H \leftarrow (SP+1)$ $SP \leftarrow SP+2$	
	RETS		00011000				12+n	$PC_L \leftarrow (SP)$, $PC_H \leftarrow (SP+1)$, $SP \leftarrow SP+2$ $PC \leftarrow PC+n$	unconditional skip
	RETI		01100010				16	$PC_L \leftarrow (SP)$, $PC_H \leftarrow (SP+1)$ $PSW \leftarrow (SP+2)$, $SP \leftarrow SP+3$	

*1 Call to 'fixed area': 800H - FFFH

*2 Call 'Table': new PC-contents is taken from 80H to FFH

	Mnemonics	Operand	Op Codes B1	B2	B3	B4	Clock Cycle	Functions	Skip Condition
Skip Instructions	SKN	CY	01001000	00011010			12	Skip if No Carry	CY = 0
	SKN	Z		00011100			12	Skip if No Zero	Z = 0
	SKNIT	if		00010i₁i₀i₈			12	Skip if No INTx Reset INTx if INTx=1	f = 0
CPU Control Instructions	NOP		00000000				6	No Operation	
	EI		01001000	00100000			12	Enable Interrupt	
	DI		01001000	00100100			12	Disable Interrupt	
Register Control Instr.	SIO		00001001				6	Start (Trigger) Serial I/O	
	STM		00011001				6	Start Timer	
Input/Output Instr.	PEX		01001000	00101101			15	PE_{15-8}—D, PE_{7-0}—C	
	PER		01001000	00111100			12	PortE AD Mode	

(Note) The clock cycles shown here are indicated as if the program (B1-B4) is located in the on-chip ROM, and the other data are located in the on-chip RAM or external memory requiring on wait. If the programs were located in the on-chip RAM or external memory, then the clock cycles are shortened by two clock cycles per one-byte fetch.

Ex. PER instruction (2-byte instruction)

In case of the on-chip ROM access : 12 clock cycles

In case of the on-chip RAM or external memory access : 12-(2x2)=8 clock cycles

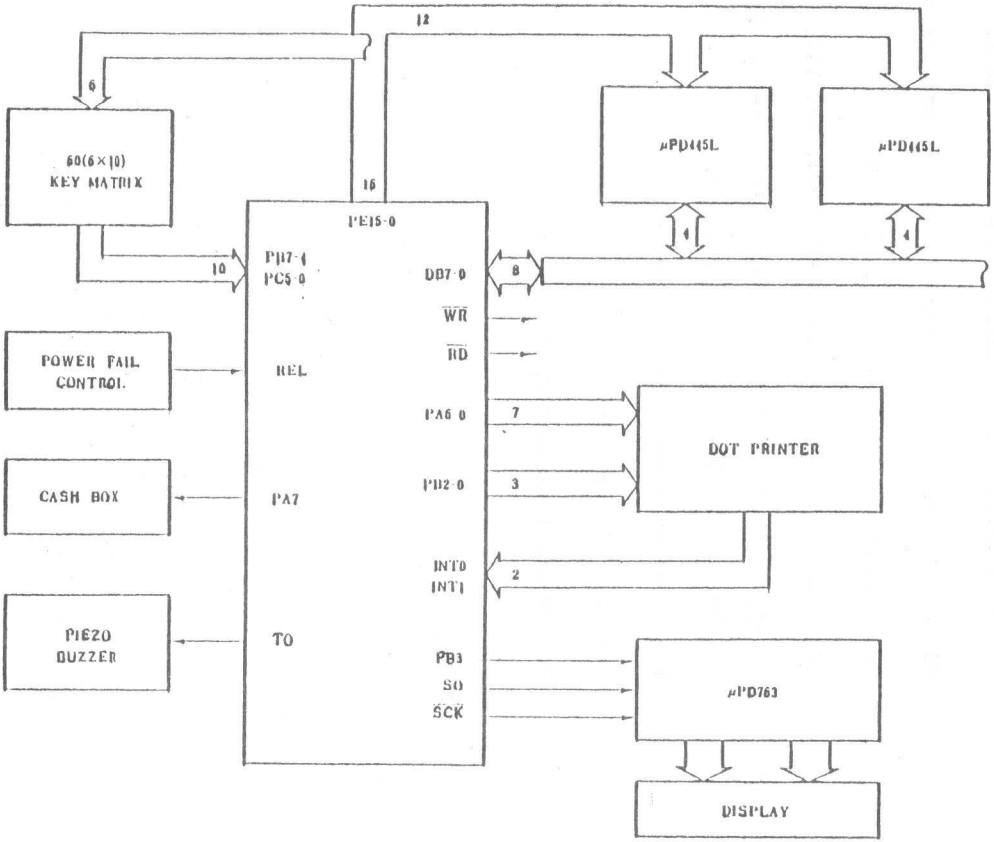
11. System Configuration Example

An ECR system as the system configuration example of the uPD78C06 is shown in Figure 11-1.

The configuration of ECR peripherals are following.

- 60 (6 x 10) key Matrix (Control with PE, PB, and PC)
- External Memory 1k x 8 (uPD445L x 2)
- Dot Printer (Control with PA, PB, INT0, and INT1)
- 7-Segment Display (Control with uPD763 - Serial Input Display Controller - PB3, SO, and SCK)
- Piezo Buzzer (Drive with PA output)
- Cash-Box (Drive with PA output)
- Power-fail (released by REL input)

Fig. 11-1 ECR Application Example



Appendix A Differences between uPD78C06/78C06A and uPD78C05/78C05A

Parameter		uPD78C06/uPD78C06A	uPD78C05/uPD78C05A
4K-byte built-in ROM		Yes	No
Internal WAIT of built-in ROM		2 WAIT cycle	No
Port E (Address bus)	After reset	Port mode	Address bus mode
	Latch function	Yes	No
	PEX instruction	PE15 - 8 ← B and PE7 - 0 ← C are executed and latched at M3T1 timing. Output is unchanged until the next PEX or PER instruction is executed.	AB15 - 8 ← B and AB7 - 0 ← C are executed only at M3T1 timing and the contents of the internal address bus are output at the other timing.
$\overline{RD}/\overline{WR}$ signal		Output against the address space of 1000H - FF7FH (4096 - 65407).	Output against the address space of 0000H - FF7FH (0 - 65407).
M1 output		No	Yes
Pin connection		Difference	
Package		64 pin Flat (uPD78C06/uPD78C06A) 64 pin QUIP (uPD78C06A)	64 pin QUIP

Appendix B Differences between uPD78C06A and uPD78C06

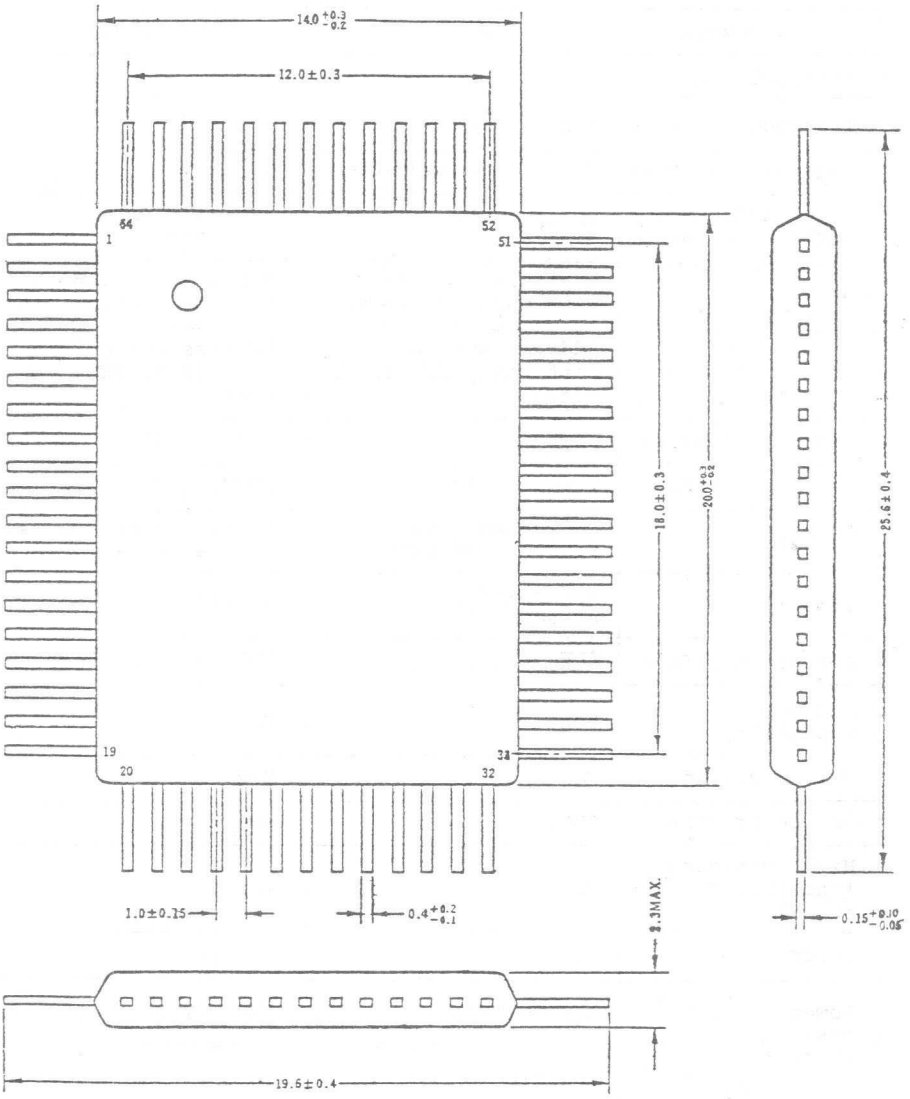
Parameter	uPD78C06A	uPD78C06
Instruction cycle	2.67us (External memory or internal RAM), 4us (Internal ROM)	4us (External memory or internal RAM), 6us (Internal ROM)
Crystal oscillation frequency or xl external clock frequency	6MHz MAX	4MHz MAX
Clock output (φout)	fosc/8	fosc/4
Max. serial clock frequency	750 kHz	500 kHz

Appendix C Differences between uPD78C06 and uPD7801

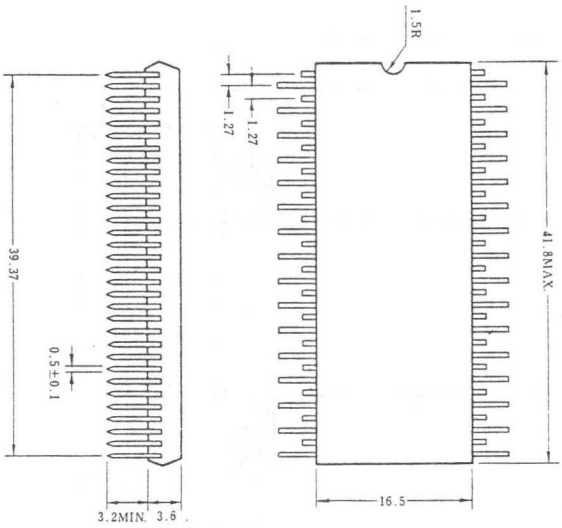
Parameter		uPD78C06	uPD7801
Instructions		101	140
Instruction cycle		6us (4us)*/4MHz	2us/4MHz
General registers		7(no V,ALT register)	16
Upper address for working register		Fixed (FFH)	Variable (by V register)
Port C		6 bit input port with pull-up register	8 bit input/output port or control line
Port E		Address bus/output port (only PEX or PER mode)	Address bus/output port (PEX, PEN, PER mode)
Interrupt source	Internal	INTT	INTT, INTS
	External	INT0, INT1	INT0, INT1, INT2
TO output		Dedicated line Square wave output	Timer out or PC4 use Single pulse output
Timer		8 bit (+4 bit prescaler)	12 bit
Serial interface		$\overline{SCK}$, SI, SO	$\overline{SCK}$, SI, SO, $\overline{SCS}$, SAK
Stand-by capability		Yes	No
Hold function		No	Yes
Device process		CMOS	NMOS
High impedance functions of $\overline{RD}$, $\overline{WR}$, PE		No	Yes
Power supply		5V ± 10%	5V ± 10%
Power consumption	At operation	17.5mW TYP.	550mW TYP.
	At stand-by	5uW TYP.	-

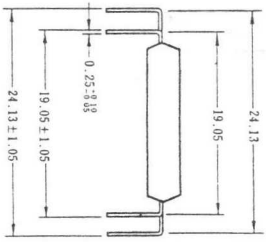
*: When an external memory is accessed.

64-Pin Flat Package Outline (Unit:mm)... uPD78C06/uPD78C06A

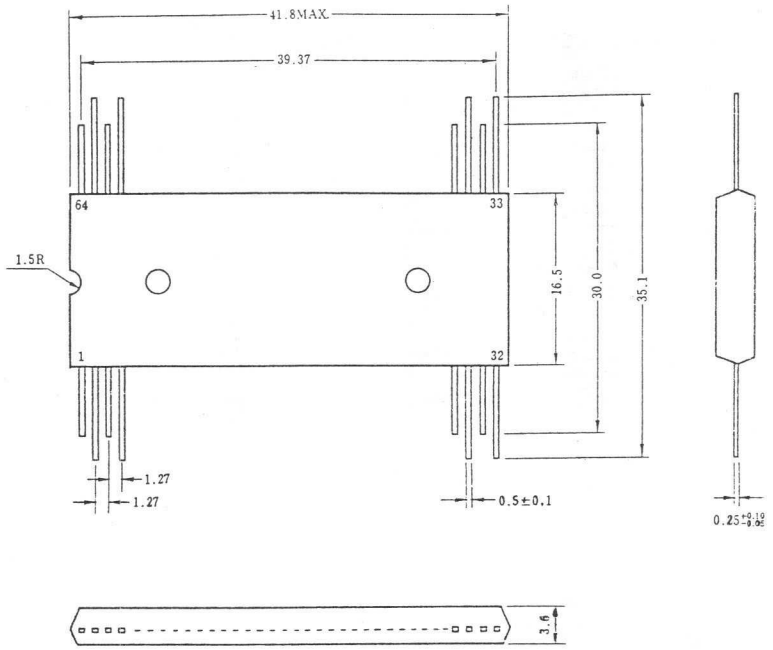


64-Pin Plastic QUIP Package Outline (Unit:mm) ... μPD78C06AGxxx-36
μPD78C05AG





64-Pin Plastic QUIP Outline (Unit:mm) ... uPD78C06AG$_{xxx-37}$



NEC OFFICES

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

NEC Electronics (Germany) GmbH, Kanzlerstr. 6, 4000 Düsseldorf 30, Tel. (0211) 650302, Telex 858996-0
- Hindenburgstr. 28/29, 3000 Hannover 1, Tel. (0511) 881013–16, Telex 9230109
- Bayerstr. 21, 8000 München 2, Tel. (089) 551150, Telex 522971
- Heilbronner Str. 314, 7000 Stuttgart 30, Tel. (0711) 890910, Telex 7252220
- NEC Electronics (BNL) – 33 Alard du Hamelstraat, 5622 CC Eindhoven, Tel. (040) 445845, Telex 51923
- NEC Electronics (Scandinavia) – Box 4039, S-18304 Täby, Tel. (08) 756 72 45, Telex 13839

NEC Electronics (France) S.A., Tour Chenonceaux, 204, Rond Point du Pont de Sèvres, F-92516 Boulogne Billancourt, Tel. (01) 609 90 04, Telex 203544

NEC Electronics Italiana S.R.L., Via Cardano 3, I-20124 Milano, Tel. (02) 67 09108, Telex 315355

NEC Electronics (UK) Ltd., Block 3 Carfin Industrial, Motherwell M L1 4UL, Schottland, Tel. (0698) 732221, Telex 777565