

# MOS DIGITAL INTEGRATED CIRCUIT

## $\mu$ PD1709CT

### SINGLE-CHIP MICROCOMPUTER INCORPORATING LED DRIVER and PLL FREQUENCY SYNTHESIZER

The  $\mu$ PD1709CT is a 4-bit CMOS microcomputer for digital tuning, containing PLL and controller in one chip and allowing direct LED connection.

CPU is capable of 4-bit parallel addition/subtraction (AD, SU instructions), logical operation (EXL instruction), plural bit tests (TMT instruction), set/reset of carry F/F (STC instruction) as well as it has the interrupt- and timer functions.

Contained in its slim 28-pin DIP (dual in-line package) are the input/output ports, which are controllable by powerful input/output instructions (IN, OUT instructions), serial interfaces (serial I/O and shift CLOCK), 4-bit A/D converter and frequency/duty variable pulse port (CGP: clock generator port).

The A/D converter can be used, when connected to the TV set, as an S-shaped curve input terminal for AFT (auto-fine tuning). By adding LPF (low pass filter) CGP can be also used as a simplified D/A converter to control the volume.

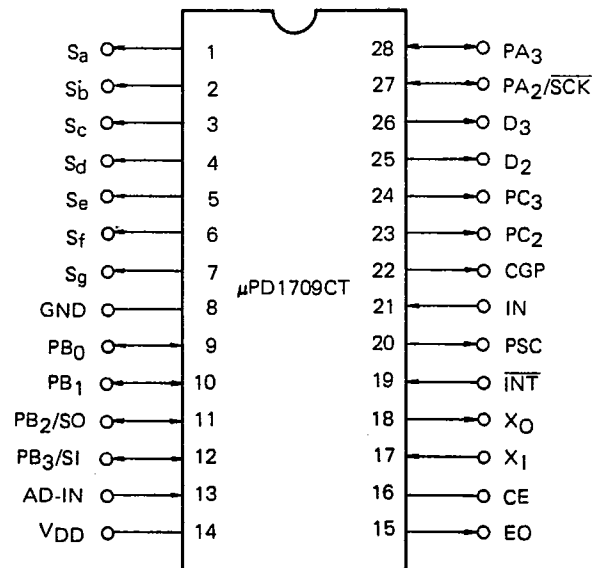
#### FEATURES:

- 4-bit microcomputer for digital tuning
- PLL and controller are incorporated
- Single 5 V  $\pm$  10 % power supply
- Low current consumption CMOS
- Easy backup of data memory (RAM) by CE terminal
- Program memory (ROM): 16 bits  $\times$  1,526 steps
- Data memory (RAM) : 4 bits  $\times$  64 words
- A powerful set of 82 different kinds of instructions (all by one-word instruction)
- Command execute time: 33.3  $\mu$ s (with 4.5 MHz crystal oscillator) ,
- An ample set of addition/subtraction instructions (12 add instructions, 12 subtract instructions)
- Powerful composite judge instructions (TMT, TMF)
- Storage-to-storage data transfer with same low address
- Indirect transfer between registers (MVRD, MVRS instructions)
- Sixteen powerful general registers (on RAM space)
- Three stack levels
- LED direct drive (segments or less)
- Segment ( $S_a$  to  $S_g$ ) open drain output (12 V, 40 mA max.)
- Clock stop by instruction (supply current: 10  $\mu$ A or less)
- Powerful 10 I/O ports ( $PA_3$ ,  $PA_2$ : 1-bit input/output,  $PB_0$  to  $PB_3$ : group input/output,  $PC_3$ ,  $PC_2$ ,  $D_3$ ,  $D_2$ : output ports)
- Serial interface incorporated ( $PA_2$  = shift clock,  $PB_3$  = serial input,  $PB_2$  = serial output; 8-bit input/output, SIO instruction)
- 4-bit A/D converter incorporated ( $V_{ref} = V_{DD}$ , sequential comparison method by program; TADT, TADF instructions)

- CGP incorporated (clock generator port; output of 64 divided-frequencies based on 180 kHz and 18 kHz; duty on 2.69 kHz variable into 64 steps)
- Power input/output instructions (IN, OUT)
- Test on status of input and output ports (TPT, TPF instructions)
- Edge trigger interrupt function (by  $\overline{\text{INT}}$  terminal)
- Timer F/F incorporated (easy setting of clock function at 125 ms-interval)
- Interval pulse output (internal output) incorporated (pulse at every 5 ms (200 Hz, duty 60 %); test by TIP instruction)
- Test on locked condition of PLL (TUL instruction)
- Transfer of data of dividing ratio and reference frequency to PLL by one instruction (PLL instruction)
- PLA (programmable logic array) for segment incorporated (user program)
- High-precision AFT (auto fine tuning) over high reference frequency available by pulse swallowing system
- Direct connection to  $\mu\text{PB562AC}$  (two-modulus prescaler; 1 GHz)
- Seven different reference frequencies selectable by program (1, 5, 6.25, 9, 10, 12.5 & 25 kHz)
- Hardware support; EVAKIT-1700 + EV-1709 (debug tool)  
SE-1700 + EV-1709 (evaluation board on PROM base)
- Software support; cross assembler under CP/MTM and MP/MTM

Note: CP/MTM and MP/MTM are the trademarks of Digital research Co., Ltd.

### TERMINAL CONNECTION (Top View)





## PIN DESCRIPTION

| SYMBOL   | PIN NAME             | FUNCTION  | OUTPUT FORMAT   |
|--|----------------------|---|-----------------|
| S <sub>a</sub> to S <sub>g</sub>   | Segment Output       | <p>Display segment signal output terminal.</p> <p>The data of data memory (RAM) at a given address is loaded by the SEG instruction on the segment PLA (programmable logic array) and output to these pins via PLA. 32 different patterns can be designated by the segment PLA (see Segment PLA).</p> <p>These pins can be also used as the key return signal source of key matrix.</p> <p>The output format of N-ch open drain output requires the pull-up resistance.</p>   | N-ch open drain |
| GND  | Ground               | Ground pin of the device.   | —               |
| PA <sub>3</sub><br>PA <sub>2</sub> /SCK  | Port A               | Two-bit I/O ports. Each port can be designated for one-bit input or output by the content of two higher-order bits of the data memory (RAM) at the address 1FH. Both ports can be used as the serial interfaces by the SIO instruction. Then the PA <sub>2</sub> pin functions as the SCK (shift clock) pin. Similarly the PB <sub>3</sub> pin functions as the SI (serial input) pin and the PB <sub>2</sub> pin as the SO (serial output) pin (Note 1 & 2).   | CMOS push-pull  |
| PB <sub>3</sub> /SI<br>PB <sub>2</sub> /SO<br>PB <sub>1</sub><br>PB <sub>0</sub> | Port B               | 4-bit I/O ports. Each port can be designated as the input port by 4 bits each by executing the input instruction (IN) or as the output port by executing the output instruction (OUT, SPB or RPB instruction). When the SIO instruction is executed, the PB <sub>2</sub> terminal can be as well used as the SO (serial output) pin or the PB <sub>3</sub> pin as the SI (serial input) pin (Note 1 & 2).   | CMOS push-pull  |
| PC <sub>3</sub><br>PC <sub>2</sub>   | Port C               | Two-bit output ports. As both are N-ch open drain output, an external pull-up resistance is required.   | N-ch open drain |
| D <sub>3</sub><br>D <sub>2</sub>   | Digit Outputs        | <p>Two-bit output ports. Either port can be used as the displayed digit signal output pin. Unlike other pins (PA, PB &amp; PC), these two pins are controlled by the DIG instruction, and the content of any register designated by the DIG instruction is output. Note that there are no PLA (programmable logic array) provided.</p> <p>Both ports can be designated as active, if CE pin=low, when ordering the mask. An external pull-up resistor is required as both outputs are the N-ch open drain outputs (Note 1).</p> | N-ch open drain |
| CGP  | Clock Generator Port | <p>CGP (clock generator port) or one-bit output port (PG<sub>2</sub>) can be selected by the program. As CGP two modes, VDP (variable duty pulse) function or SG (signal generator) function can be chosen by the program.</p> <p>The VDP function is to continuously output the 2.69 kHz pulses and outputs the frequency divided into 64 steps (duty 50 %) after varying it by the program (Note 1 &amp; 3).</p>  | CMOS push-pull  |
| AD-IN  | Analog-Digital Input | <p>A/D (analog/digital) converter input pin.</p> <p>A 4-bit A/D converter is incorporated to sequentially compare data by program under the reference voltage V<sub>DD</sub> (5 V ± 10 %).</p>  | Input           |
| $\overline{\text{INT}}$  | Interrupt            | Input pin of interrupt request signal, which is issued at the fall edge of signal applied to this pin. When the request for interrupt is accepted, the program flow unconditionally jump to Address 1H.   | Input           |
| X <sub>1</sub><br>X <sub>0</sub>   | X'tal                | Crystal oscillator connection pin to which a 4.5 MHz crystal oscillator is connected. Adjust the oscillation frequency (4.5 MHz) by the X <sub>0</sub> pin.   | CMOS            |

| SYMBOL | PIN NAME                       | FUNCTION   | OUTPUT FORMAT            |
|--------|--------------------------------|--|--------------------------|
| VDD    | Power Supply                   | Device power pin, which supplies the voltage of $5\text{ V} \pm 10\%$ while the device is in operation. The voltage can be lowered to $2.5\text{ V}$ when to hold the internal data memory (RAM) (as the CKSTP instruction is executed). If the voltage of $0$ to $4.5\text{ V}$ is supplied to this pin, the device is reset and the program starts from Address $0\text{H}$ (see Timer F/F).   | —                        |
| CE     | Chip Enable                    | Device select signal input pin. Set this pin to the high level to regularly operate the device or to the low level when the device is not to be used. While this pin is held under the low level, PLL, segment output and digit output are inhibited, provided that the input below $134\ \mu\text{s}$ cannot be accepted. If the CKSTP instruction is executed while this instruction is used by the program and the CE pin is under the low level, the internal clock generator and CPU stop operating and the memory can be maintained under a low consumptive current ( $10\ \mu\text{A}$ or less).<br>By changing the CE pin from the low to the high level, the device is reset and the program starts from Address $0\text{H}$ (see Timer F/F). And the I/O ports (PA, PB) turn to the input ports. | Input                    |
| IN     | Local Oscillation Signal Input | The local oscillation output of tuner, which is divided by the prescaler $\mu\text{PB562AC}$ is input here. The $\mu\text{PB562AC}$ is a two-modulus prescaler of $1\text{ GHz}$ input.  | Input Internal self-bias |
| PSC    | Pulse Swallowing Control       | Output pin of switching signal of frequency dividing ratio to the two-modulus prescaler $\mu\text{PB562AC}$ . It is connected direct to the PSC pin of $\mu\text{PB562AC}$ ; the dividing ratio can be switched to $1/128$ and $1/136$ (or $1/64$ and $1/68$ ).  | CMOS push-pull           |
| EO     | Error Output                   | PLL error output pin. The high level is output from this pin when the divided oscillation frequency exceeds the reference frequency or the low level is output if it is lower. And the floating takes place when both are coincident. This output is applied to the varactor diode of the tuner via the low pass filter.   | CMOS three-state         |

Note 1:  $\text{PB}_0$  corresponds to the lowest-order bit of a register or an operand data and  $\text{PB}_3$  to the highest-order bit under the port operation instructions (I/O instructions, test port instruction, set/reset instructions). Same goes to PA, PC, Digit ( $\text{D}_3$ ,  $\text{D}_2$ ) and CGP ( $\text{PG}_2$ ).

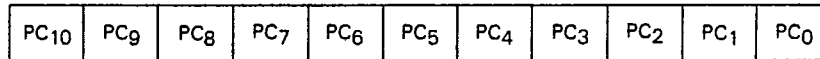
- All I/O ports (PA & PB) turn to the input mode when the device is reset ( $V_{\text{DD}} = \text{low to high}$ , CE = low to high) or when the CKSTP instruction is executed.
- All pins turn to high impedance (CGP to the low level) when the device is reset ( $V_{\text{DD}} = \text{low to high}$ ) or when the CKSTP instruction is executed. All pins remain same without turning to the high impedance when CE changes from the low to the high level, except when  $V_{\text{DD}}$  changes from the low to the high level and if CE = low to high after the CKSTP instruction is executed.

## INTERNAL BLOCK FUNCTION

### 1. CPU

#### 1.1 PROGRAM COUNTER

The program counter consisting of a 11-bit binary counter addresses the program memory (ROM) or program.



11 bits

The counter generally adds up one by one whenever a instruction is executed, and the address designated by the operand is loaded on the counter when any of the jump instruction and subroutine call instruction is executed. When the skip instruction (ADS, TMT or RTS) is executed, the address of a instruction next to the skip instruction is designated, regardless of the content of skip condition. If the condition is to be skipped, the instruction next to the skip instruction is regarded as NOP (no operation) – after executing NOP, the address of next instruction is designated. The address 1 is unconditionally loaded when the request for interrupt is accepted.

**Note:** The program counter of each of  $\mu$ PD1701C, 1703C, 1704C and 1705C (a series having the ROM capacity less than 1K steps) is composed of ten bits.

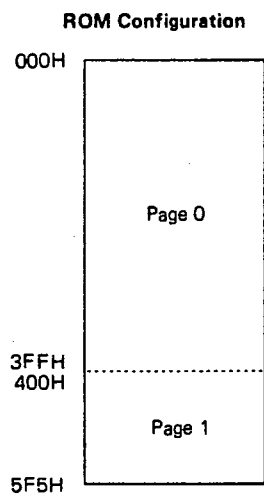
#### 1.2 STACK REGISTER

The stack register consisting of 3 x 12 bits stores the return address (11 bits), which is the value adding 1 to the counting of program counter when a subroutine call instruction is executed or when the request for interrupt is accepted, as well as the result of judgement (1 bit) if any instruction having the skip function has been executed upon the acceptance of an interrupt request. The content of stack register is loaded on the program counter by the execution of a return instruction (RT or RTS), and the original program flow is restored.

The stack register is used both for the subroutine call and the interrupt so that two remaining levels of the stack registers are applicable to the subroutine call if one level is used for the interrupt.

#### 1.3 PROGRAM MEMORY (ROM)

The program memory (ROM) consisting of 16 bits by 1,526 steps stores the programs within the address range from 000H to 5F5H.



The page concept applies to this ROM of  $\mu$ PD1709CT; Page 0 ranges from the ROM addresses 000H to 3FFH and Page 1 from 400H to 5F5H.

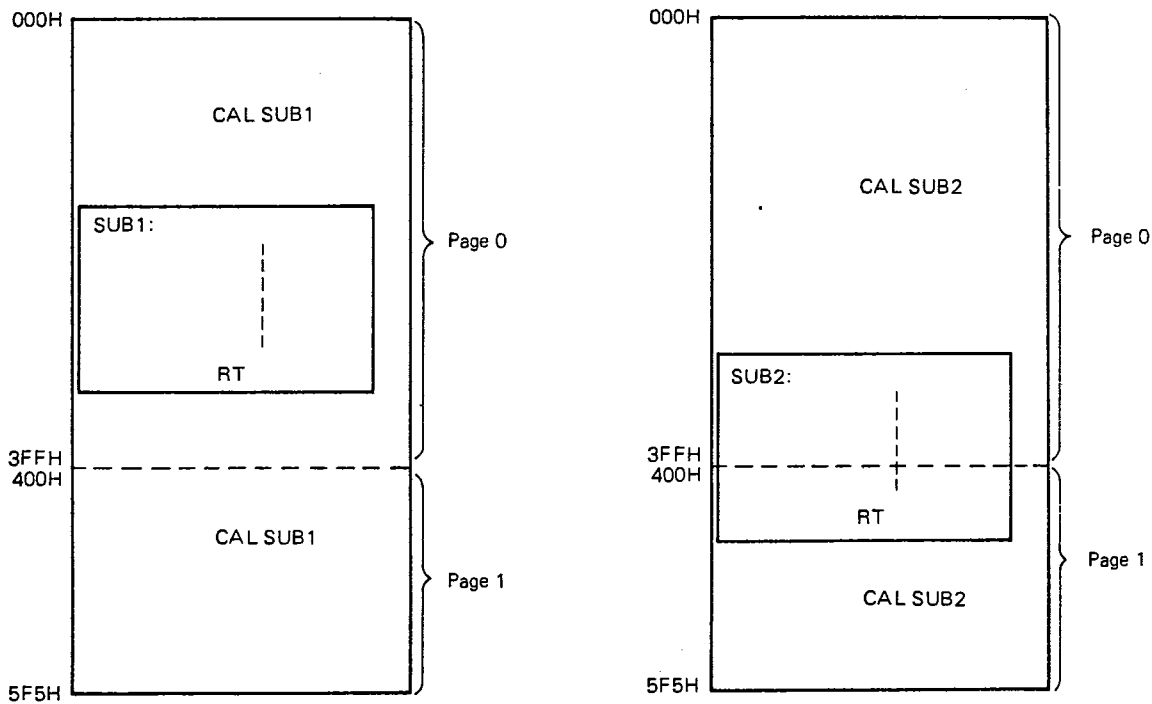
The head address of a subroutine should be placed within Page 0 when a program is created; no subroutine having its head address within Page 1 can be called either from Page 0 or Page 1 (see Notes to Use of CAL instruction). The JMP instruction, which is described in the assembler, can use any address between 000H to 5F5H in the same description (JMP ADDR) without being conscious of the page, provided that the patch correction needs be made for debug as the operation code of JMP instruction differs between Page 0 and Page 1 (see Note to Use of JMP instruction).

The following must be noted in the use of CAL and JMP instructions as the page concept (discrimination between Page 0 and Page 1) applies to ROM of  $\mu$ PD1709CT.

**Notes to Use of CAL Instruction**

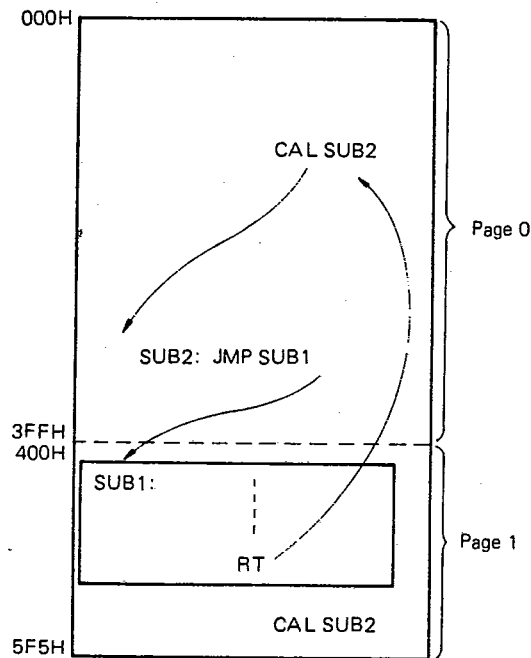
When the CALL instruction is used, its call address or the head address of subroutine must be set within Page 0 (000H to 3FFH); the subroutine whose head address is set within Page 1 (400H to 5F5H) cannot be called, provided that the return address (RT or RTS instruction) can be placed within Page 1.

[Example 1] Head Address of Subroutine in Page 0



The return address (RT or RTS instruction) can be placed either in Page 0 or Page 1 so far as the head address of subroutine is held within Page 0; the CAL instruction can be used then without being conscious of the page concept. The following technique is however effective if the placing of head address of any subroutine within Page 0 is prevented by programming:

It is to call the actual subroutine (SUB1) by means of the JMP instruction, which is set within Page 0.

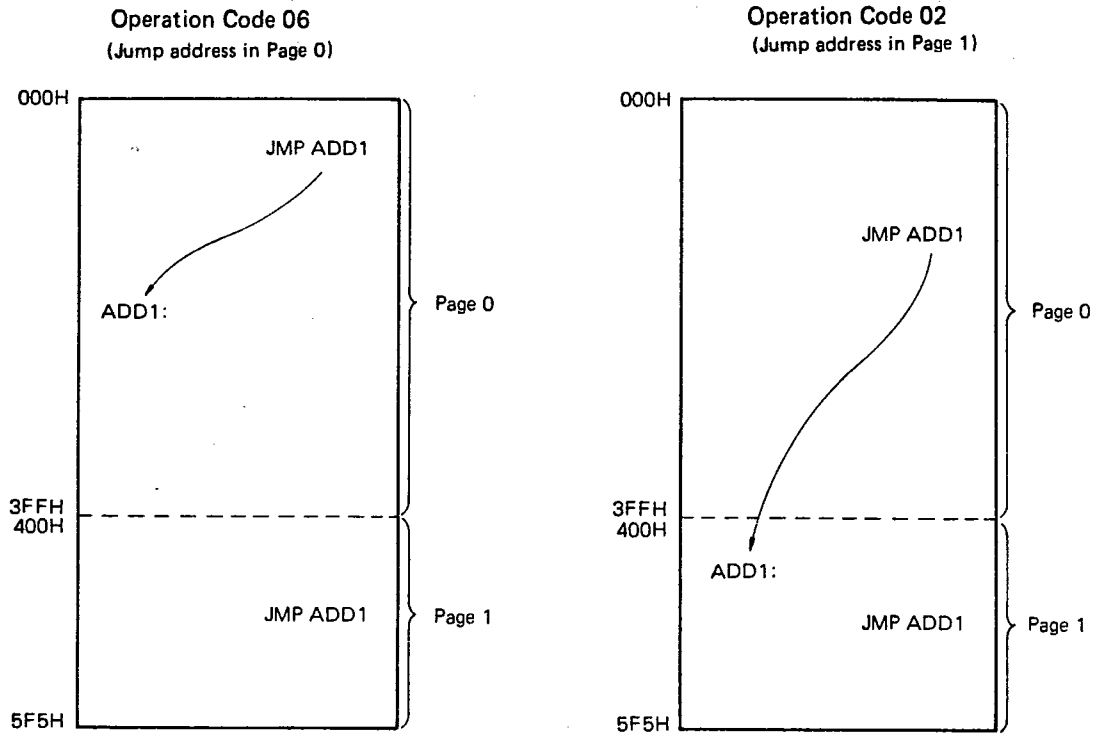


#### Notes to Use of JMP Instruction

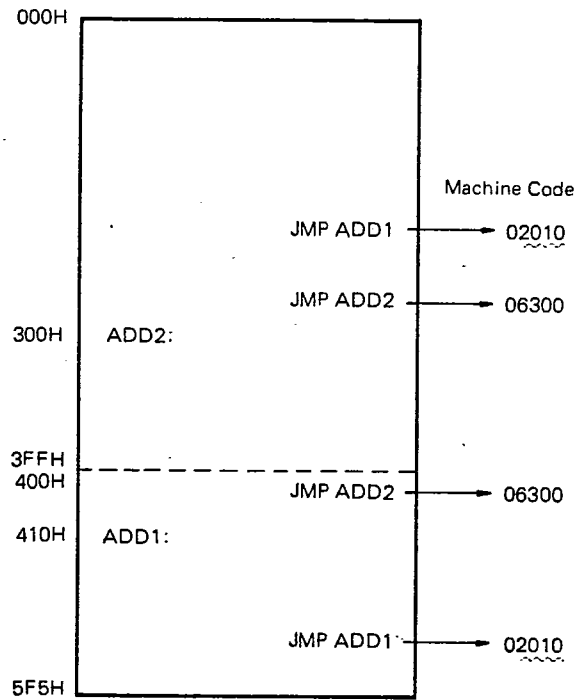
The JMP instruction can be used in the same description between the ROM addresses 000H and 5F5H without being conscious of the page concept insofar as it is described in the assembler. Note that the operation code of JMP instruction differs between Page 0 (000H to 3FFH) and Page 1 (400H to 5F5H). The operation code of JMP instruction within Page 0 is "06" or "02" if it is within Page 1.

The assembler of  $\mu$ PD1700 Series can automatically refer to the jump address and convert the operation codes.





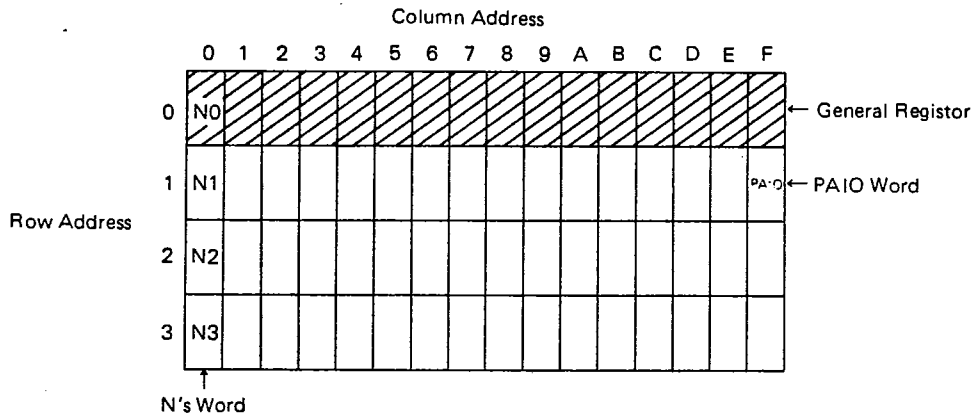
The programmer is however required to convert the operation codes "06" and "02" when the patch correction takes place in debugs. The address also needs to be converted if the jump address of JMP instruction is beyond 400H (operation code 02); the address is then increased one by one from 400H, which is then considered 000H. Thus the address 5F5H turns to 1F5H.



To effect the patch correction of JMP 400H described in the assembler, for instance, input 02000 and turn JMP 000H to 06000.

**1.4 DATA MEMORY (RAM)**

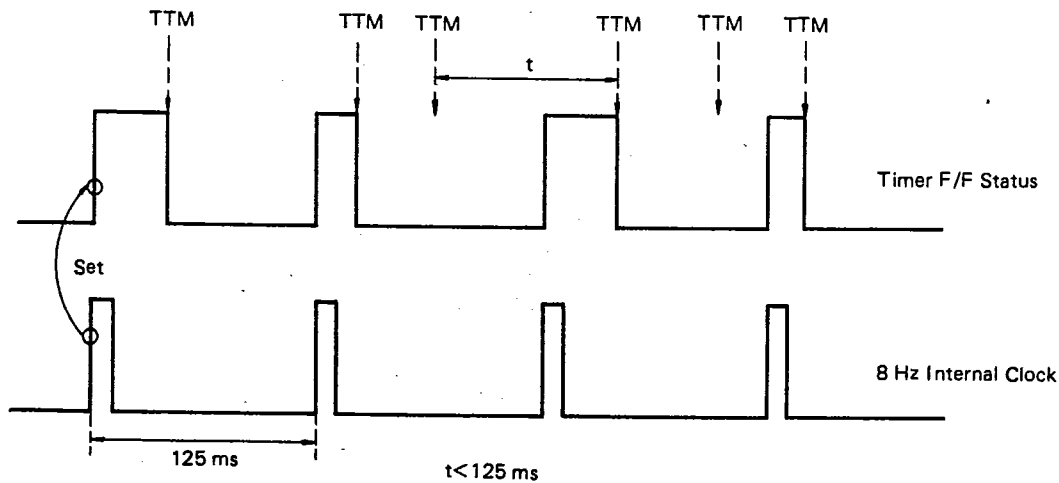
The RAM consisting of 4 bits x 64 words is generally used to store data. Its general register ranges from the address 00H to 0FH, which can be used for the operation and transfer between the memory or which can be used as an ordinary memory. All information required for PLL control (i.e. divided frequencies and reference frequency) can be set via RAM. A total of 17 bits, 4 bits x 4 words (N's words) at the address 00H, 10H, 20H and 30H and the highest-order bit ( $N_F$  bit) of a given general register, are allocated to the setting of divided frequencies; similarly another total of N's words and a given word (4 bits) of RAM except the one containing the  $N_F$  bit are allocated as the bits to set the reference frequency (control words). These bits are then transferred to the PLL register with a PLL instruction. The address 1FH is called PAIO word, two highest-order bits of which are used to designate the port A ( $PA_3, PA_2$ ) for input or output.



**Fig. 1 RAM Configuration**

**1.5 TIMER F/F**

The timer F/F is set with the 8 Hz (125 ms) signal or reset with the test timer (TTM) instruction. As it is automatically set at the intervals of 125 ms each, this timer F/F can be used to count the clock (8 counts equal to one second) or the mute time. As it can be reset only by the execution of TTM instruction, this instruction must be executed within the period of 125 ms; otherwise its counting errs and makes it unable to control the accurate time.



**Fig. 2 TTM Instruction Execution Timing**

The timer F/F can be used to judge the detection of power failure; it is reset when the power is turned on ( $V_{DD}$ =low to high) and it is set when the CKSTP instruction is executed or CE=low to high. Fig. 3 shows the status transition of timer F/F set/reset.

As apparent from the status transition diagram, the program starts from the address 0 after the power is turned on, while the timer F/F is reset, regardless of the status of CE pin; the timer F/F is not reset until the TTM instruction is executed (timer F/F set unable status). Once the TTM instruction is executed, the timer F/F becomes enable to set and it can be set at the intervals of 125 ms each.

If the CE pin changes from the low to the high level while the power is held on ( $V_{DD}$ =high), the program flow jumps to the address 0 immediately when the timer F/F is set; thus the program starts from the address 0 with the timer F/F being set.

Thus the content of timer F/F differs between the time when the power failure is recovered ( $V_{DD}$ =low to high) and the time without power failure ( $V_{DD}$ =high, CE=low to high) or when it is restored from the back-up condition. By testing the content of timer F/F at this moment (to execute the TTM instruction), it is possible to judge if it is returned from the power failure or not.

The TTM instruction must be executed within 125 ms after the program started from the address 0, and it is judged to be the power failure if the execution resulted in 0 (false) or not the power failure (to be backed up) if resulted in 1 (true).

Care must be taken by the programming to the restoration from non-power failure ( $V_{DD}$ =high, CE=low to high) if the program has the clock function and it must be continuously operated even when CE=low (without using the CKSTP instruction). In other words the clock must be updated after the TTM instruction is executed to detect the power failure (and resulted in true), because the program jumps to the address 0 as soon as the timer F/F is set. Otherwise the clock delays by 125 ms whenever the CE pin changes from the low to the high level.

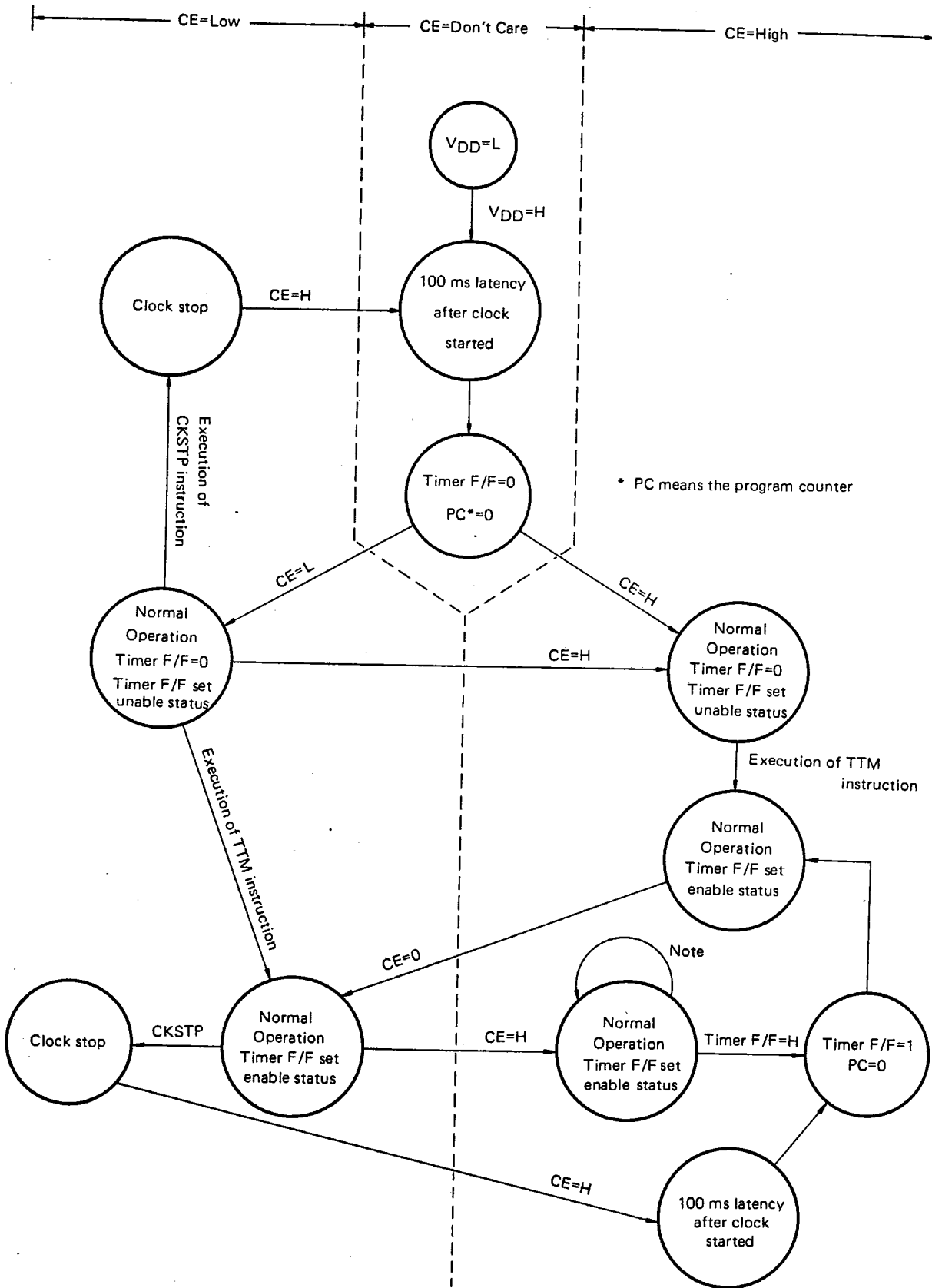


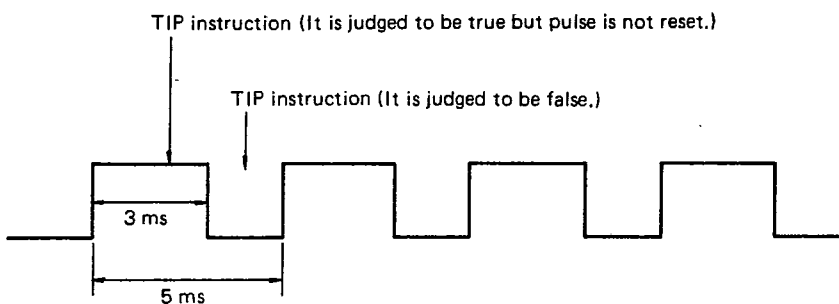
Fig. 3 CPU Status Change by CE Pin

- Note 1:** It is unable to come out of this loop if the TTM instruction is executed simultaneously with the setting of timer F/F. It is then allowed to come out with the next setting of timer F/F (125 ms later) and the program jumps to the address 0 after turning the timer F/F to "1". It must be noted that the execution of TTM instruction is cyclic and that it is permanent unable to clear to the address 0 if the instruction execution happens to coincide with the cycle of timer F/F setting (125 ms each).
- 2:** The timer F/F of  $\mu$ PD1709CT can be set and the program starts from the address 0 when CE changes from the low to the high level after executing the CKSTP instruction while the program starts after the timer F/F of  $\mu$ PD1701C,  $\mu$ PD1703C,  $\mu$ PD1704C or  $\mu$ PD1701G is reset. Note the content of timer F/F differs, when the CKSTP instruction is executed, between  $\mu$ PD1709CT and  $\mu$ PD1701C,  $\mu$ PD1703C,  $\mu$ PD1704C or  $\mu$ PD1701G.

**1.6 INTERVAL PULSE**

The interval pulse is the 60 %-duty pulse, which is output every 5 ms, and it can be tested by the TIP instruction.

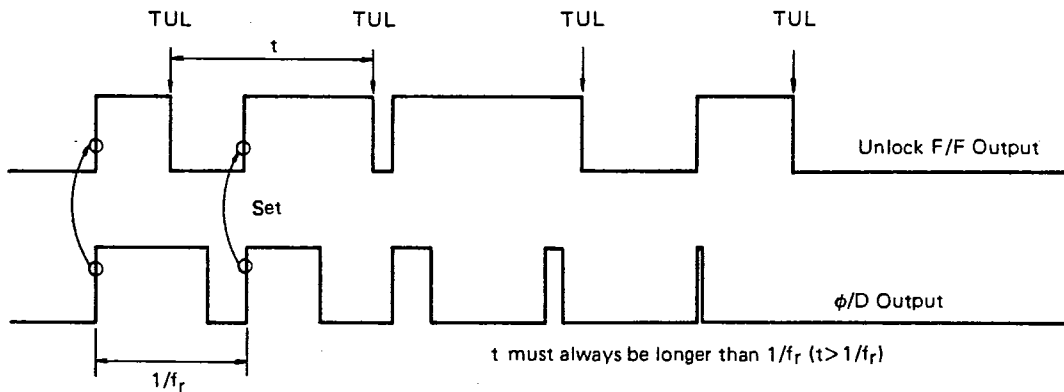
The pulse output cannot be reset even by the execution of TIP instruction as no flipflop (F/F) is provided. It is possible to make out the timer of exact multiplex of 5 ms when the TIP instruction is constantly executed to obtain the edges of interval pulses.



**Fig. 4 Interval Pulse Timing**

**1.7 UNLOCK F/F**

The pulses are output from the phase detector ( $\phi$ -DET) at the period of reference frequency ( $f_r$ ) when the PLL system is not locked or when  $f_r$  is not coincident with the divided output frequency of VCO. The unlock F/F is reset by this pulse or it is reset by the execution of TUL instruction. Accordingly, the period of executing the TUL instruction should be longer than the  $f_r$  period; otherwise the PLL system is considered to be locked even if it is not, and a false operation would be performed. The first TUL instruction should be executed after certain period longer than the  $f_r$  period following the execution of PLL instruction.



**Fig. 5 TUL Instruction Execution Timing**

### 1.8 CARRY F/F

The carry F/F is set if the execution of any operation instruction resulted in a carry or a borrow or it is reset if neither carry nor borrow is resulted. The content of this carry F/F remains unchanged unless an operation instruction is executed; it can be further set or reset directly by the carry F/F set/reset instructions (STC & RSC instructions) or by the status word operation instructions (SS & RS instructions).

**Note:** Even if the interrupt is accepted, the content of carry F/F is not automatically saved.

### 1.9 BANK F/F

The BANK F/F is used for addressing of port groups, which is achieved with two bits of the instruction operand and the content of this BANK F/F. BANK0 can be designated by the reset of BANK F/F or BANK1 if it is set, both designation to be made by BANK0, BANK1, SS and RS instructions.

Out of the  $\mu$ PD1700 Series the devices having the data memory (RAM) of 64 words or more (e.g.  $\mu$ PD1704C,  $\mu$ PD1706G,  $\mu$ PD1707G,  $\mu$ PD1708G) provides BANK0 and BANK1 by 64 words each, which can be likewise designated by the BANK F/F.

The RAM of  $\mu$ PD1709CT has 64 words and it is designated as BANK0; therefore it is not allowed to access to RAM with BANK1 (after accessing to the port group of BANK1), and it must be designated as BANK0 in advance (by executing the BANK0 instruction). When the power is turned on ( $V_{DD}$ =low to high) and  $CE$ =low to high, or when the device is reset, the content of BANK F/F is reset and BANK0 is automatically designated.

**Note:** Even if the interrupt is accepted, the content of BANK F/F is not automatically saved.

### 1.10 INT F/F & INTE F/F

The INT F/F is unconditionally set with the fall edge of signal to be applied to the  $\overline{INT}$  pin. If the internal INTE F/F is then enable, the interrupt request can be accepted or it cannot if the INTE F/F is disable. As far as the INT F/F is being set, the interrupt request can be accepted after it is turned to be enable by executing the DI instruction on the program. Or INTE F/F can be made disable by the execution of DI instruction.

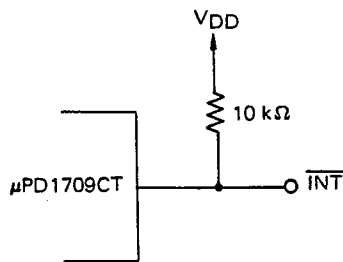
The INT F/F can be reset when the instruction that makes the INTE F/F disable is executed under the DI status (e.g. DI or RS instruction).

If any interrupt request is accepted, the INT F/F is automatically turned to the DI status and the program flow jumps to the address 1 (interrupt process routine). And the address of instruction to be executed next to the instruction where the interrupt request is made is stored in the stack (or the jump address when executing the JMP instruction). If the instruction being executed has the skip function when an interrupt request was made, the result of judgement of the skip condition is also stored in the stack. The content of stack is restored and so is the program flow when the RT instruction is executed by the interrupt process routine. The contents of carry F/F and bank F/F must be saved by the program if they are likely to be destroyed by the interrupt processing as they are not saved into the stack.

One level of the stack is used for the interrupt processing so that the stack levels should be placed under the control.

The  $\overline{INT}$  pin can be used as the general input port (by the TITT & TITF instructions), but it must be noted that it turns to true when the low level is input or turns to false by the high level input as the  $\overline{INT}$  pin is active low when it is used for the above purpose.

**Note:** The high level should always be input before accessing to the  $\overline{\text{INT}}$  pin from the program. If it is held under the low level, the low level input is read internally to be high unless the high level input is once made. After the high level input, the  $\overline{\text{INT}}$  can be operated regularly so that it must be pulled up on the applied circuit via the resistor.



### 1.11 STATUS WORD

The status word is to split the internal status of device, which must be made known for the program execution of which must be designated without fail, into four bits each, and thereby allows to test, set or reset the status by the program. Two different kinds of the status words are available, status word 1 and 2, both of which are connected any of the following pins or F/F input/output:

#### 1) Status Word 1 (write only word)

Operation instructions: SS, RS

| #3 | #2          | #1           | #0          |
|----|-------------|--------------|-------------|
| 0  | BANK<br>F/F | Carry<br>F/F | INTE<br>F/F |

The status word 1 can be set or reset by the SS, RS or EI instruction.

#### 2) Status Word 2 (read only word)

Operation instructions: TST, TSF

| #3 | #2          | #1        | #0                             |
|----|-------------|-----------|--------------------------------|
| 0  | BANK<br>F/F | CE<br>Pin | $\overline{\text{INT}}$<br>Pin |

The content of status word 2 can be judged by the TST, TSF or SBK0 instruction.

## 2. PLL

### 2.1 REFERENCE FREQUENCY GENERATOR

Seven different kinds of the reference frequencies, viz., 1 kHz, 5 kHz, 6.25 kHz, 9 kHz, 10 kHz, 12.5 kHz and 25 kHz, can be generated by dividing the external crystal (4.5 MHz). Any kind of the reference frequency can be chosen by the program (data of the control word).

### 2.2 PHASE DETECTOR

This circuit detects the difference of phases between the reference frequency ( $f_r$ ) and the VCO output, which is divided by the programmable divider. The output of this circuit is input into the internal charge pump and the following pulses are then output to the EO<sub>1</sub> and EO<sub>2</sub> pins:

- 1)  $f_r > f_{OSC}/N$  : Low level
- 2)  $f_r < f_{OSC}/N$  : High level
- 3)  $f_r = f_{OSC}/N$  : Floating

where  $f_{OSC}$  represents the VCO oscillation frequency and  $N$  the dividing ratio of programmable divider.

### 2.3 PROGRAMMABLE DIVIDER

The programmable divider is a binary-down counter consisting of a swallow counter and a programmable counter. The swallow counter is a 5-bit presettable down counter, and the contents of NR0 (4-bits) and N<sub>F</sub> register (1-bit) out of  $N$  registers can be preset at the period of reference frequency.

The programmable counter consists of 12 bits into which the contents of NR1 through NR3 out of the  $N$  registers are preset and counted down simultaneously with the swallow counter.

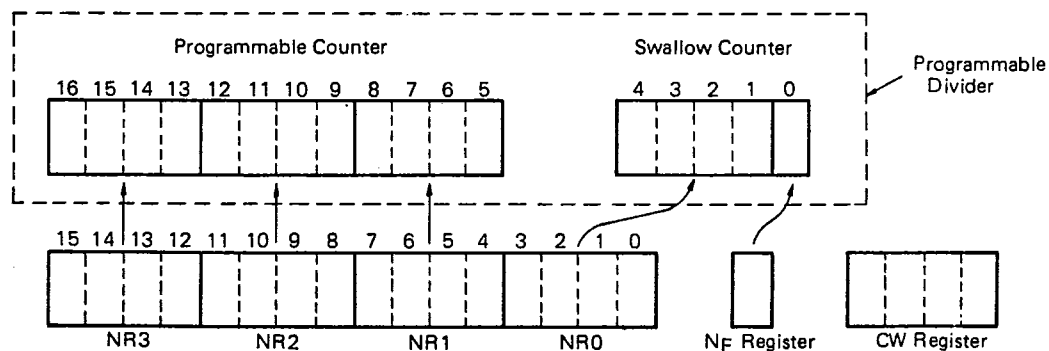


Fig. 6 Programmable Divider Configuration

The data of swallow counter (5 bits) is output to the prescaler from the PSC pin.

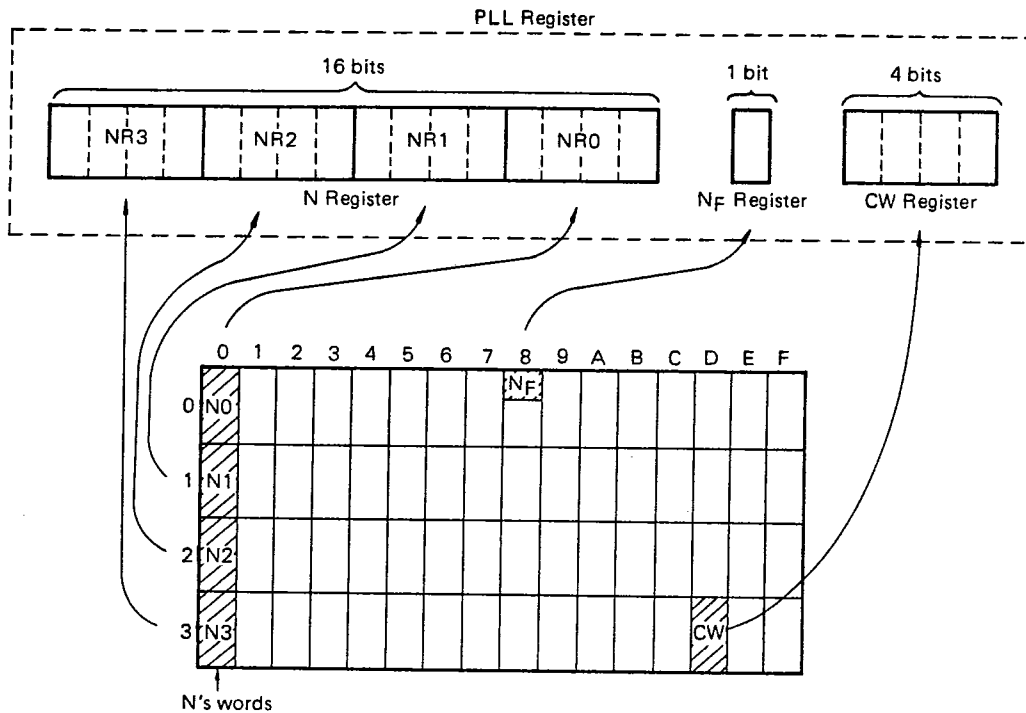


**2.4 PLL REGISTER**

The following two information are required to control the  $\mu$ PD1709CT PLL.

- 1) Frequency dividing ratio (N)
- 2) Reference frequency ( $f_r$ )

The above information are stored in the PLL register; it consists of the N registers (16 bits) that set the frequency dividing ratio, NF register (1 bit) and control word registers (4 bits) that set the reference frequency, each corresponding to the N's word, NF bit and control words (CW) of the data memory (RAM). All the contents are transferred to the memory simultaneously with one PLL instruction. The N's words are assigned to the addresses of RAM, 00H, 10H, 20H and 30H, the NF bit to the highest-order bit of a given general register and CW to any RAM area excepting one word containing the N's word and NF bit.



**Fig. 7 Operation under Execution of PLL Instruction**

The data codes of control words are as shown below and seven kinds of the reference frequencies can be thereby selected:

**Table 1 Control Word Codes**

| #3 | #2 | #1 | #0 | $f_r$ (kHz) |
|----|----|----|----|-------------|
| 1  | 0  | 0  | 0  | 1           |
| 1  | 0  | 0  | 1  | 6.25        |
| 1  | 0  | 1  | 0  | 5           |
| 1  | 0  | 1  | 1  | 9           |
| 1  | 1  | 0  | 0  | 10          |
| 1  | 1  | 0  | 1  | 12.5        |
| 1  | 1  | 1  | 0  | 25          |
| 0  | 1  | 1  | 1  | DISABLE PLL |

**2.5 SETUP OF PLL INFORMATION**

The PLL information (frequency dividing ratio and reference frequency) is set by the program. How to set the divided frequency of program divider is shown below:

$$N = \frac{f_p + f_{IF}}{P \times f_r} \quad (f_{OSC} = f_p + f_{IF})$$

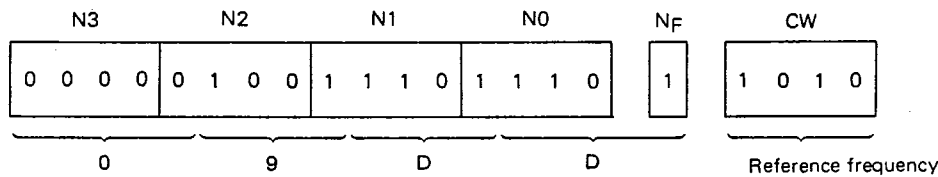
- f<sub>p</sub> : picture carrier frequency
- f<sub>IF</sub> : intermediate frequency
- P : frequency dividing ratio of prescaler
- f<sub>r</sub> : reference frequency
- N : frequency dividing ratio of programmable divider
- f<sub>OSC</sub> : local oscillation frequency

If the prescaler μPB562AC is used, its frequency dividing ratio P should be based on 8.

Ex. 1 : US TV Band 02ch:

(f<sub>p</sub> = 55.25 MHz, f<sub>IF</sub> = 45.75 MHz, P=8, f<sub>r</sub>=5 kHz)

$$N = \frac{(55.25 + 45.75) \times 10^3}{8 \times 5} = 2525 = 09DDH \text{ (H means a hexadecimal)}$$



The above example is the case where the NF bit, as the lowest-order bit, is changed one by one; then the VCO oscillation frequency (f<sub>OSC</sub> = f<sub>p</sub> + f<sub>IF</sub>) changes by 40 kHz (P × f<sub>r</sub>) each. In other words the minimum variable frequency is the product of the prescaler's dividing ratio and the reference frequency.

When the above N0 bit #0 is changed (by 1 each), f<sub>OSC</sub> changes by 80 kHz each, or by 160 kHz each if Bit #1 is changed (by changing N0 by 2 each) or by 320 kHz each if Bit #2 is changed (by changing N0 by 4 each).

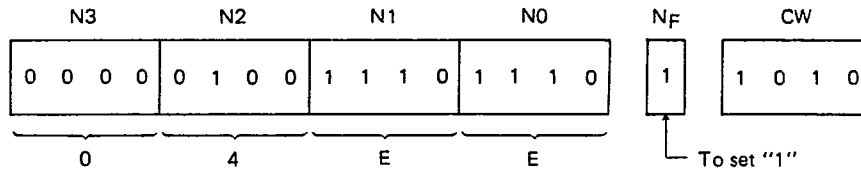
While the N value is determined in the above example by dividing the bits by four bits each starting with the N<sub>F</sub> bit as the lowest-order bit, the programming can be more readily understood if the PLL information is set by four bits each from N0. In this case the N<sub>F</sub> bit is assumed to be fixed and the bits are changed from N0 by applying the following formula:

$$N = \frac{(f_p + f_{IF}) - (P \times f_r) N_F}{2 \times (P \times f_r)}$$

The term  $(P \times f_r) N_F$  turns to  $(P \times f_r)$  if the  $N_F$  bit is "1" or it disappears when the bit is set to "0". Another term  $2 \times (P \times f_r)$  means a change from  $N_0$  by 80 kHz. The following can be found by applying the values of Example 1 to the above formula:

$$N = \frac{(55.25 + 45.75) \times 10^3 - 40}{2 \times 8 \times 5} = 1262$$

$$= 04EEH$$



As proven above, the content of programmable divider turns equal to Example 1. For the US TV bands, the above formula is always applicable when the IF frequency  $f_{IF}=45.75$  MHz and the  $N_F$  bit is set to "1".

### 3. PORT

The  $\mu$ PD1709CT provides as its I/O ports the Port A (PA<sub>3</sub>, PA<sub>2</sub>) and the Port B (PB<sub>3</sub> to PB<sub>0</sub>) and as the output ports the Port C (PC<sub>3</sub>, PC<sub>2</sub>) and Digits (D<sub>3</sub>, D<sub>2</sub>).

It also has the internal ports of Port E (PE<sub>3</sub> to PE<sub>0</sub>), Port F (PF<sub>3</sub> to PF<sub>0</sub>), Port G (PG<sub>3</sub> to PG<sub>0</sub>) and Port H (PH<sub>3</sub> to PH<sub>0</sub>).

Eight bits of Port E and F are used as the data buffer for access to the serial I/O or A-D converter and another 8 bits of Port G and H as the data buffer to set the duty or the divided frequency when accessing to CGP (clock generator port).

These ports are addressed by the direct addressing of two bits of the operand of a instruction and by the bank F/F, as shown below. The digit pins can be however accessed only by the DIG instruction.

Table 2 Port Addresses

| Direct Add. |    | BANK F/F |        |
|-------------|----|----------|--------|
| #1          | #0 | BANK 0   | BANK 1 |
| 0           | 0  | PA       | PE     |
| 0           | 1  | PB       | PF     |
| 1           | 0  | PC       | PG     |
| 1           | 1  | —        | PH     |

To access to the internal ports of Port E, F, G or H, the bank F/F must be set to BANK1, but it is not allowed to access to RAM while holding BANK1. Thus the bank F/F must be restored to BANK0 after the port group of BANK1 is accessed.

Example:

```

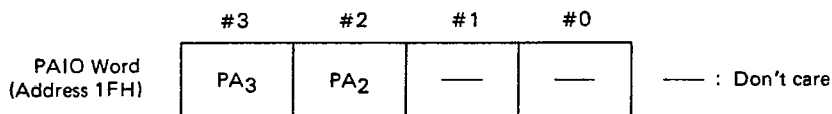
MVI    0AH, 1111B ; To set "F" to Address 0AH of the register
MVI    0BH, 1101B ; To set "D" to Address 0BH of the register
FLOOP:
BANK1
OUT    3, 0AH    ; To output the content of 0AH to Port H
OUT    2, 0BH    ; To output the content of 0BH to Port G
BANK0      ; To restore to BANK0 after the internal port access
CAL    WT1SEC    ; To call the subroutine to wait for one second
SI     0BH, 0100B ;
SIS    0AH, 0    ;
JMP    FLOOP

```

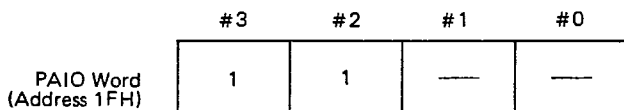
The above is the case of accessing to CGP and outputting for one second each 64 divided parts of 18 kHz reference frequency, starting with the lower frequency.

### 3.1 PORT A

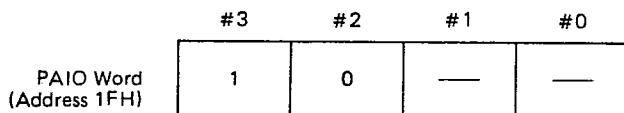
The Port A (PA<sub>3</sub>, PA<sub>2</sub>) can be set by one bit each for input or output, which is set according to the content (2 higher-order bits) of the address 1FH within the data memory (RAM) called the PAIO word. To select the input port, "0" must be set to the corresponding bit of PAIO word or "1" needs be set if it is chosen as the output port. The contents of both Bits #1 and #0 of the PAIO word are ignored.



Example 1: To set PA<sub>3</sub> & PA<sub>2</sub> as Output Ports



Example 2: To set PA<sub>3</sub> as Output Port & PA<sub>2</sub> as Input Port



As shown above, an input or output instruction must be executed after Port A is set to the PAIO word. The input or output mode once set can be held on unless the content of PAIO word (data at 1FH) is changed later. Port A, however, automatically works under the input mode when the power is turned on (V<sub>DD</sub>=low to high) or when the CKSTP instruction is executed or CE=low to high.

It must be noted that the content of PAIO word does not coincide with the input or output status of Port A; the port then continues the input mode until the content of PAIO word is set.

As the serial I/O the Port A's PA<sub>2</sub> can be used or it works as the shift clock (SCK) pin by the SIO instruction. In the latter case the bit #2 of PAIO word or the bit #2 of Port A must be set to the input mode (see Sec. 5. Serial I/O).

### 3.2 PORT B

The Port B (PB<sub>3</sub> to PB<sub>0</sub>) can be set by four bits each for input or output. All its four bits are set to the input port by executing an input instruction (IN instruction) or to the output port when an output instruction (OUT, SPB, RPB instructions) is executed. The input or output mode can be readily switched by the execution of input or output instruction.

When the OUT instruction is executed, the data as designated by the OUT instruction is output from Port B and it turns to the output mode. When the SPB or RPB instruction is executed, the content of data latch of Port B is output to the bit other than the one designated by the instruction's operand, and the port turns to the output mode. The content of data latch of Port B remains unchanged unless any of the OUT, SPB and RPB instructions is executed.

When PB<sub>3</sub> or PB<sub>2</sub> pin is used as serial I/O, it works as SI (Serial Input), SO (Serial Output) respective. (see Sec. 5. Serial I/O)

The Port B turns automatically to the input mode when the power is turned on (V<sub>DD</sub>=low to high), the CKSTP instruction is executed and CE=low to high; it keeps the input mode until an output instruction (OUT, SPB or RPB instruction) is executed.

The content of data latch of Port B remains unchanged even when the CKSTP instruction is executed or when CE=low to high (the content is not definite when the power is turned on ( $V_{DD}$ =low to high)). The following instructions can be used to output the current content of data latch:

SPB 1,0000B or RPB 1,0000B

It is to execute the instruction that does not set or reset any bit of Port B.

### 3.3 PORT C

The Port C ( $PC_3$ ,  $PC_2$ ) is the N-ch open drain type output port, which is generally accessed by the output instruction (OUT, SPB or RPB instruction). When the input instruction (IN instruction) is executed, the data currently being output is read into the designated register, but its content remains unchanged even if the IN instruction is executed.

When "1" is output under the execution of an output instruction, the high level (pull-up potential) is output or the low level (GND potential) if "0" is output. The Port C stays under the same status (high-impedance status) as "1" or the pull-up potential is output when the power is turned on ( $V_{DD}$ =low to high) or the CKSTP instruction is executed. The content of internal output data latch does not change from the original status (it is however indefinite when the power is turned on ( $V_{DD}$ =low to high)). The following instructions can be used to output the content of output data latch:

SPB 2,0000B or RPB 2,0000B

It means to execute the instruction that does not set or reset any bit of Port C.

If CE=low to high, the original status is held rather changing to the high impedance status.

### 3.4 DIGIT

The digit ( $D_3$ ,  $D_2$ ) is the N-ch open drain type output port, which is generally used as the display digit signal output pin. This port is accessible by the DIG instruction unlike other ports (Port A, Port B or Port C). The content of two higher-order bits of a given register as designated by the DIG instruction is output while two lower-order bits of the same register are ignored.

This pin is not provided with the digit PLA (programmable logic array), which is available for other device of  $\mu$ PD1700 Series, and the content of any designated register can be output to the output data latch circuit.

Example:

```
MVI 0EH, 1000B
DIG 0EH
```

The above example sets "8" (1000B) by the MVI instruction to the register address 0EH. The  $D_3$  pin turns to the high impedance status (to output the pull-up potential) after the DIG instruction is executed while the low level is output to the  $D_2$  pin. The output status of digit pin, when the CE pin turns to the low level, can be designated by the MTDIG pseudo instruction of assembler upon the order of mask, as shown below:

```
MTDIG 01B
```

Bit corresponding to  $D_3$  pin  
 Bit corresponding to  $D_2$  pin  
 (B means the binary)

In the above example the  $D_3$  pin is automatically set to the low level and the  $D_2$  to the high impedance status when the CE pin turns to the low level.

#### 4. CGP

The clock generator port (CGP) has both the VDP (variable duty pulse) generating function and the SG (signal generator) function, both of which are controlled by the internal ports, Port G and Port H. Except that both are the internal ports, both ports are identical to other ports and all instructions associated with the ports can be executed. If the IN instruction is executed against Port G or H, all data currently set on Port G or H are read into the designated register.

CGP has four modes each of which is designated by the Bit #1 and #0 of Port G, which are called the control bits (CB).

**Table 3 Code & Function of Control Bit (CB)**

| Port H    |    |    |    | Port G |    |    |    |
|-----------|----|----|----|--------|----|----|----|
| #3        | #2 | #1 | #0 | #3     | #2 | #1 | #0 |
| MSB       |    |    |    | LSB    |    |    |    |
| DATA BITS |    |    |    |        |    | CB |    |

| CB (Control Bits) |    | Function  |
|-------------------|----|---|
| #1                | #0 |   |
| 0                 | 0  | PG #2 through-mode:<br>to output the content of Port G Bit #2 to the CGP pin  |
| 1                 | 0  | VDP mode:<br>to vary and output 64 steps of duty over the frequency 2.69 kHz. |
| 0                 | 1  | SG mode 0:<br>to output 64 divided frequencies of 18 kHz                      |
| 1                 | 1  | SG Mode 1:<br>to output 64 divided frequencies of 180 kHz                     |

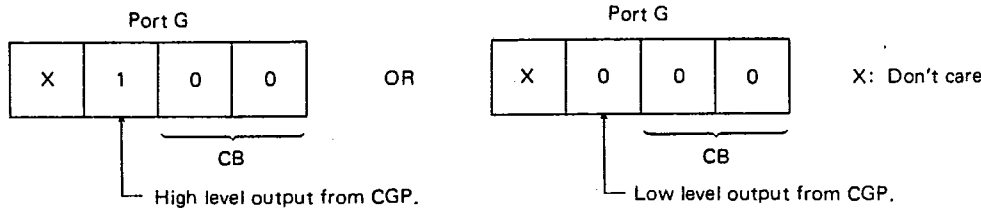
Six bits from #3 of Port H to #2 of Port G are called the data bits by which to set the data of duty for the VDP mode designated or the divided frequency for the SG mode designated. The MSB of data bits correspond to the bit #3 of Port H and LSB to the bit #2 of Port G.

The CGP pin is unconditionally set to the low level (initial low status) when the power is turned on ( $V_{DD}$ =low to high) or when the CKSTP instruction is executed, but the contents of data latches of both Port G and H remain unchanged (the contents are however not definite when the power is turned on ( $V_{DD}$ =low to high)).

The CGP pin turns from the initial low status to the active at the time of data setting to Port G and the pulses (or high/low level) are output. To make the CGP pin active without changing the content of data latch, the instruction (SPB 2,0) that does not set any bit of Port G or another instruction (RPB2,0) that does not reset any bit must be used. Even if the instruction that control Port H only is executed, the CGP pin cannot be changed from the initial low status to the active nor can it be turned to the low level but it remains same if the CE pin is changed from the low to the high level of vice versa. To turn the CGP pin to the low level when CE-high to low, it requires to test the level of CE pin (by TCET or TCEF instruction) and set it under the PG #2 through mode, depending on the test result, thus executing the instruction to turn it to the low level. In other words it is enough to output X000B (X denotes Don't care) to Port G.

**4.1 PG #2 THROUGH MODE**

The data of Bit #2 of Port G is output to the CGP pin when CB=00B. When PG #2=1 then the high level is output or the low level if PG #2=0; and it can be used as one-bit output port.



Then the content of bit #3 of Port G is ignored and so is the content of Port H. It is therefore not required to output any data to port H under the PG #2 through-mode.

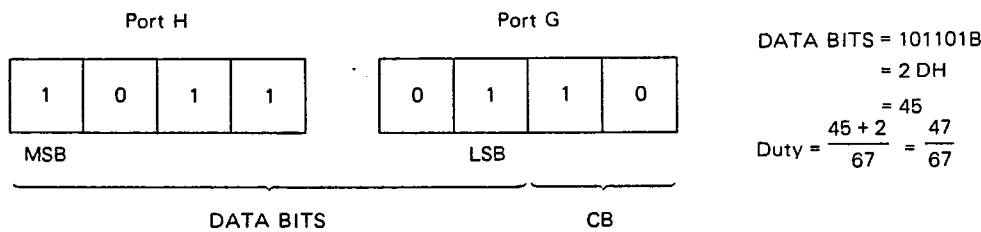
**4.2 VDP MODE**

The duty pulses as designated by the data, which are set to the data bits (PH<sub>3</sub> to PD<sub>0</sub>, PG<sub>3</sub>, PG<sub>2</sub>), are continuously output to the CGP pin when CB=10B. The frequency of output pulse is 2.69 kHz. The following relation is established between the data set to the data bits and the duty of pulse to be output:

$$\text{Duty} = \frac{\text{Time of high level}}{\text{Period}} = \frac{(\text{data bits}) + 2}{67}$$

Hence the duty can be varied into 64 steps from 2/67 to 65/67. The duty can be varied fully into 66 steps, including the low or high output, when the above relation is used together with the afore-mentioned PG #2 through-mode.

Example:



As shown above, the bit #3 of Port H turns to MSB of the data bits and the bit #2 of Port G to LSB.

The pulses from the CGP pin are output at the time when 10B is set to CB, which is two lower-order bits of Port G. No pulses are, for instance, output with the data output to Port H when CB=00B, because it only sets the data to the data latch circuit of Port H.

To change the initial low status to the VDP mode, it is necessary to output data first to Port H then to Port G. Otherwise the output from CGP pin is operated with the previous data to Port H until new data is output to it, and there would be certain period without any required duty level being output. It is however allowed to output data only to Port H or output data from Port H first if the device is operating under the VDP mode and no data to Port G are changed.



```

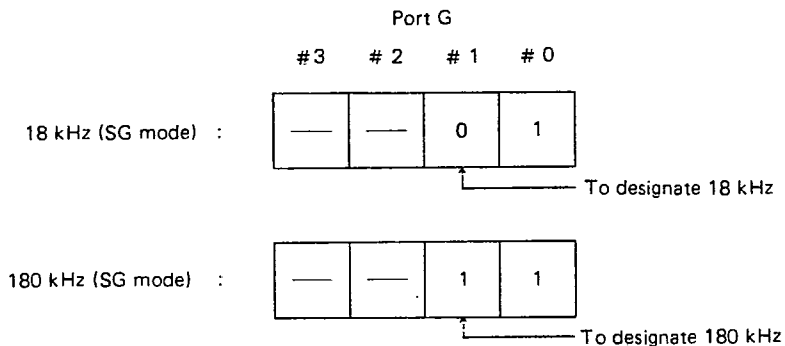
MVI    0AH, 1011B ; To set Port H data to the register 0AH
MVI    0BH, 0110B ; To set Port G data to the register 0BH
BANK1
OUT    3,0AH      ; To output data to Port H
OUT    2,0BH      ; To output data to Port G; pulse of duty 64/67 is output
BANK0
    
```

**4.3 SG MODE**

The CGP pin turns to the SG (signal generator) mode when changing the bit #0 (of Port G) of the control bits (CB) to "1". The SG mode is to output from the CGP pin the frequency pulse (duty=50 %) as designated by the data bits and the frequencies divided into 128 steps can be output under this mode. The following relation is established between the data, which is set to the data bits, and the pulse frequency (f<sub>OUT</sub>) to be output:

$$f_{OUT} = \frac{f_B}{2(2 + (DATA BITS))}$$

where f<sub>B</sub> (base frequency) is the reference frequency to be output to the CGP pin, which can be chosen from two kinds, 18 kHz or 180 kHz, depending on the value of CB bit #1 (bit #1 of Port G):



Similar to the VDP mode, the pulses from the CGP pin are output when data is output to Port G. Table 4 below shows the relation of frequencies to be output the CGP pin and the data bits:

Table 4 Table of SG Mode Output Frequencies

| DATA BITS |        |    | OUTPUT FREQUENCY |             | DATA BITS |        |    | OUTPUT FREQUENCY |             |
|-----------|--------|----|------------------|-------------|-----------|--------|----|------------------|-------------|
| DEC       | BINARY |    | MODE0 (Hz)       | MODE1 (kHz) | DEC       | BINARY |    | MODE0 (Hz)       | MODE1 (kHz) |
|           | PH     | PG |                  |             |           | PH     | PG |                  |             |
| 0         | 0      | 0  | 4500.000         | 45.0000     | 32        | 1      | 0  | 264.706          | 2.6471      |
| 1         | 0      | 0  | 3000.000         | 30.0000     | 33        | 1      | 0  | 257.143          | 2.5714      |
| 2         | 0      | 0  | 2250.000         | 22.5000     | 34        | 1      | 0  | 250.000          | 2.5000      |
| 3         | 0      | 0  | 1800.000         | 18.0000     | 35        | 1      | 0  | 243.243          | 2.4324      |
| 4         | 0      | 0  | 1500.000         | 15.0000     | 36        | 1      | 0  | 236.842          | 2.3684      |
| 5         | 0      | 0  | 1285.710         | 12.8571     | 37        | 1      | 0  | 230.769          | 2.3077      |
| 6         | 0      | 0  | 1125.000         | 11.2500     | 38        | 1      | 0  | 225.000          | 2.2500      |
| 7         | 0      | 0  | 1000.000         | 10.0000     | 39        | 1      | 0  | 219.512          | 2.1951      |
| 8         | 0      | 0  | 900.000          | 9.0000      | 40        | 1      | 0  | 214.286          | 2.1429      |
| 9         | 0      | 0  | 818.182          | 8.1818      | 41        | 1      | 0  | 209.302          | 2.0930      |
| 10        | 0      | 0  | 750.000          | 7.5000      | 42        | 1      | 0  | 204.545          | 2.0455      |
| 11        | 0      | 0  | 692.308          | 6.9231      | 43        | 1      | 0  | 200.000          | 2.0000      |
| 12        | 0      | 0  | 642.857          | 6.4286      | 44        | 1      | 0  | 195.652          | 1.9565      |
| 13        | 0      | 0  | 600.000          | 6.0000      | 45        | 1      | 0  | 191.489          | 1.9149      |
| 14        | 0      | 0  | 562.500          | 5.6250      | 46        | 1      | 0  | 187.500          | 1.8750      |
| 15        | 0      | 0  | 529.412          | 5.2941      | 47        | 1      | 0  | 183.673          | 1.8367      |
| 16        | 0      | 1  | 500.000          | 5.0000      | 48        | 1      | 1  | 180.000          | 1.8000      |
| 17        | 0      | 1  | 473.684          | 4.7368      | 49        | 1      | 1  | 176.471          | 1.7647      |
| 18        | 0      | 1  | 450.000          | 4.5000      | 50        | 1      | 1  | 173.077          | 1.7308      |
| 19        | 0      | 1  | 428.571          | 4.2857      | 51        | 1      | 1  | 169.811          | 1.6981      |
| 20        | 0      | 1  | 409.091          | 4.0909      | 52        | 1      | 1  | 166.667          | 1.6667      |
| 21        | 0      | 1  | 391.304          | 3.9130      | 53        | 1      | 1  | 163.636          | 1.6364      |
| 22        | 0      | 1  | 375.000          | 3.7500      | 54        | 1      | 1  | 160.714          | 1.6071      |
| 23        | 0      | 1  | 360.000          | 3.6000      | 55        | 1      | 1  | 157.895          | 1.5789      |
| 24        | 0      | 1  | 346.154          | 3.4615      | 56        | 1      | 1  | 155.172          | 1.5517      |
| 25        | 0      | 1  | 333.333          | 3.3333      | 57        | 1      | 1  | 152.542          | 1.5254      |
| 26        | 0      | 1  | 321.429          | 3.2143      | 58        | 1      | 1  | 150.000          | 1.5000      |
| 27        | 0      | 1  | 310.345          | 3.1034      | 59        | 1      | 1  | 147.541          | 1.4754      |
| 28        | 0      | 1  | 300.000          | 3.0000      | 60        | 1      | 1  | 145.161          | 1.4516      |
| 29        | 0      | 1  | 290.323          | 2.9032      | 61        | 1      | 1  | 142.857          | 1.4286      |
| 30        | 0      | 1  | 281.250          | 2.8125      | 62        | 1      | 1  | 140.625          | 1.4063      |
| 31        | 0      | 1  | 272.727          | 2.7273      | 63        | 1      | 1  | 138.462          | 1.3846      |

### 5. SERIAL I/O

The serial I/O is an 8-bit  $\mu$ COM standard serial I/O that transfers and receives data synchronously with the internal or external clock. The following three pins are provided for the serial I/O:

- SI (concurrent with PB<sub>3</sub>): serial data input pin
- SO (concurrent with PB<sub>2</sub>): serial data output pin
- $\overline{\text{SCK}}$  (concurrent with PA<sub>2</sub>): shift clock I/O pin (active-low)

All the three pins that are jointly used with other ports (PA<sub>2</sub>, PB<sub>3</sub> & PB<sub>2</sub>), cannot be used as the ports (PA<sub>2</sub>, PB<sub>3</sub>, PB<sub>2</sub>) at the timing as they are used as the serial I/O.

An 8-bit presettable shift register is used as the data buffer of this serial I/O; its 4 higher-order bits are assigned to Port F of the internal port and 4 lower-order bits to Port E. Hence data are written in or read out of the presettable shift register by the port operation instruction (OUT, SPB, RPB, IN instructions) that access to Port E and F. Four lower-order bits (Port E) of the presettable shift register are also used as the data latch under the A/D conversion so that it is not allowed to perform the serial I/O operation and the A/D conversion simultaneously. It is therefore necessary during the A/D conversion to reset SMR1 (bit #1) of the shift mode register, which is described later, and to inhibit the input of shift clocks into the presettable shift register.

The serial I/O consists of SMR (shift mode register), PSR (presettable shift register), SCC (shift clock counter) and SCG (shift clock generator), the function of each being described below:

#### 5.1 SMR (SHIFT MODE REGISTER)

SMR is a register consisting of three bits, SMR3, 1 & 0, that determines the mode of serial I/O. As the SIO instruction is executed, the immediate data of its operand is written into SMR, which however does not provide any bit corresponding to the bit #2 of the operand. The bit #2 of immediate data of the operand of the SIO instruction becomes therefore 'don't care'.

As the SIO instruction (SIO b<sub>3</sub>b<sub>2</sub>b<sub>1</sub>b<sub>0</sub>B) is executed, the following are set to SMR to start the respective mode operations:

Table 5 SMR Bit Functions

The diagram shows the SIO instruction bits b<sub>3</sub>, b<sub>2</sub>, b<sub>1</sub>, b<sub>0</sub>, and B. Bit b<sub>3</sub> is connected to SMR0. Bit b<sub>1</sub> is connected to SMR1. Bit b<sub>0</sub> is connected to SMR3. Bit B is also connected to SMR3.

| Symbol | Name  | Operation  |
|--------|---|--|
| SMR0   | SO Enable bit                               | "0": PB <sub>2</sub> /SO pin function as PB <sub>2</sub><br>"1": the above functions as SO   |
| SMR1   | Shift Enable bit                            | "0": no shift takes place<br>"1": shift takes place  |
| SMR3   | Internal $\overline{\text{SCK}}$ Enable bit | "0": to allow to input external clocks from PA <sub>2</sub> / $\overline{\text{SCK}}$ pin<br>"1": to output internal clocks to PA <sub>2</sub> / $\overline{\text{SCK}}$ pin |

All bits of SMR are reset to "0" when the power is turned on (V<sub>DD</sub>=low to high) or when the internal clock is stopped by the execution of CKSTP instruction.

##### (1) SMR0 (SO Enable bit)

The PB<sub>2</sub>/SO pin function as the serial data output pin or the SO pin when "1" is set to SMR0 (SIO XXX1B is executed). Then the PB<sub>2</sub>/SO pin automatically switches to the output mode when the SIO XXXX1B instruction is executed, even if Port B is under the input mode, and it starts to work as the SO pin. The I/O modes of other ports (PB<sub>3</sub>, PB<sub>1</sub>, PB<sub>0</sub>) of Port B however remain unchanged.

**Note:** To work the PB<sub>0</sub>/SO pin of  $\mu$ PD1707G (its SO pin is jointly used with PB<sub>0</sub>, and Port B is the output port) or to execute the SIO XXX1B instruction, "1" must always be set to PB<sub>0</sub> before executing the above instruction, unlike  $\mu$ PD1709CT.

The data of pre-settable shift registers (PE, PF) are shifted from MSB sequentially and output from the PB<sub>2</sub>/SO pin when the SIO XX11B instruction is executed. The data of PB<sub>2</sub>/SO pin is output synchronously with the fall edge of clock to be output (or input) from PA<sub>2</sub>/ $\overline{\text{SCK}}$ . The pre-settable shift register is shifted with the rise edge of clock and simultaneously reads data from the PB<sub>3</sub>/SI pin or the content output from PB<sub>3</sub> if the PB<sub>3</sub>/SI pin works as the PB<sub>3</sub> output pin.

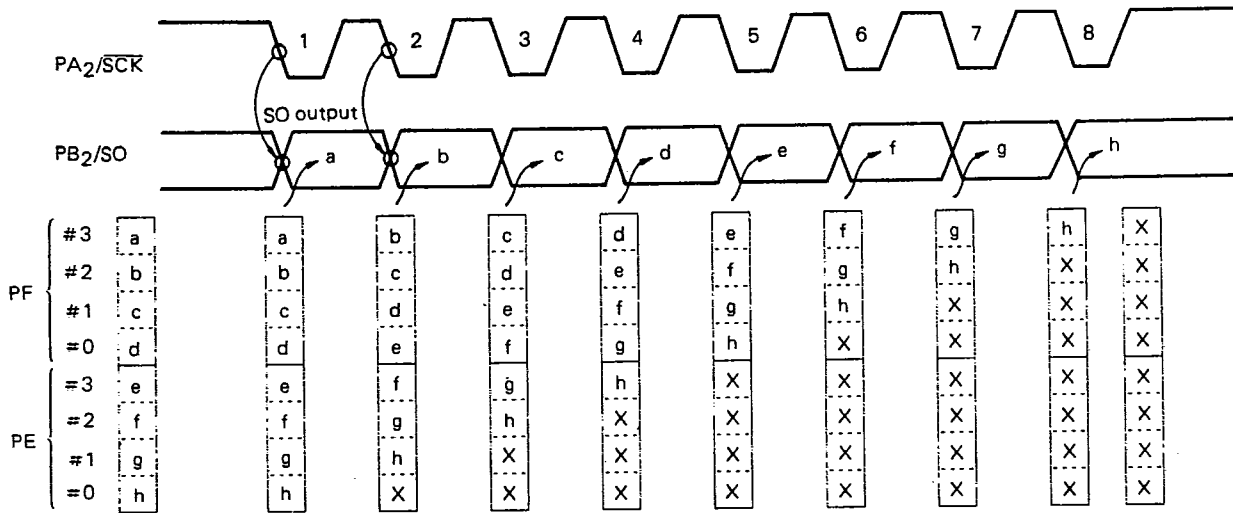


Fig. 8 SO Operation

To use the PB<sub>2</sub>/SO pin as PB<sub>2</sub>, SMR0 must be set to "0" (to execute SIO XXX0B). The mode of PB<sub>2</sub> cannot be switched to the input mode from the current mode under which it is held while "1" is set to SMR0 (output mode), even if an input instruction is executed against Port B. Hence the content output to PB<sub>2</sub> is read in by the input instruction.

By setting SMR0 to "0" after the SO operation, the I/O mode of PB<sub>2</sub> pin is automatically switched to the same mode as other ports of Port B (PB<sub>3</sub>, PB<sub>1</sub>, PB<sub>0</sub>). If the port is under the output mode, the output of PB<sub>2</sub>/SO pin is the content that has been output to the data latch of Port B before the SO mode or the content of data latch of Port B.

(2) SMR1 (Shift Enable bit)

The content of PB<sub>3</sub>/SI pin is sequentially shifted toward MSB (bit #3 of Port F) and read in from LSB (bit #0 of Port E) of the pre-settable shift register, synchronous with the clock to be output from (or input to) the PA<sub>2</sub>/ $\overline{\text{SCK}}$  pin when "1" is set to SMR1. The read timing is the rise edge of clock.

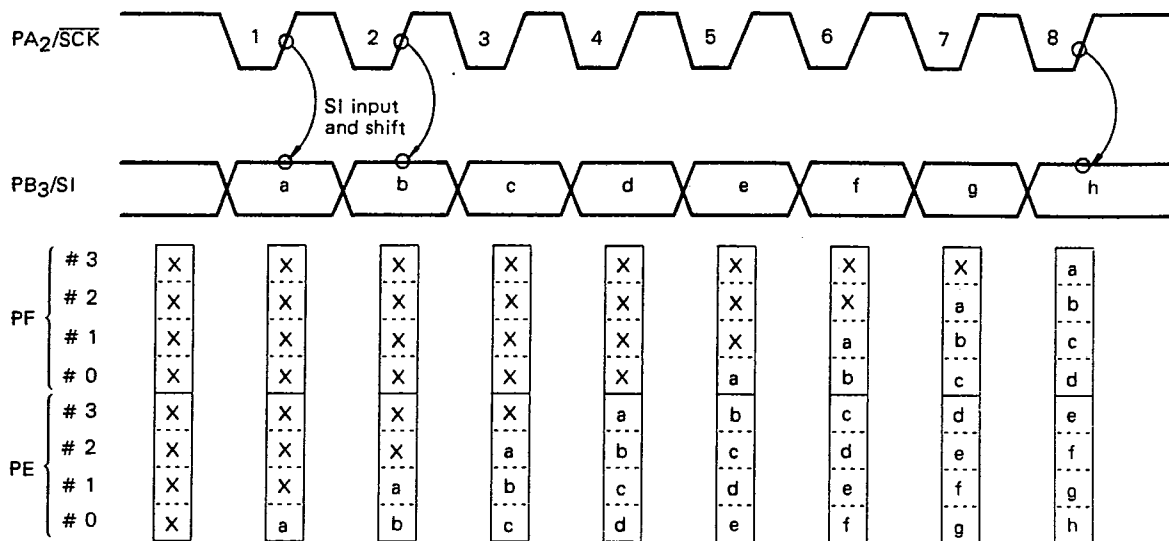


Fig. 9 SI Operation

To input the serial data from the PB<sub>3</sub>/SI pin, Port B must be designated with the input mode. If it is set to the output mode, the output of PB<sub>3</sub> is sequentially shifted and read into the presettable shift registers (PE & PF). In other words, the PB<sub>3</sub>/SI pin functions as PB<sub>3</sub> and all the contents of presettable shift register are written by "1" if "1" is output from PB<sub>3</sub> or all "0" if "0" is output.

It is necessary for the A/D conversion to set SMR1 to "0", thereby inhibiting the input of shift clocks of the presettable shift register.

When "1" is set to SMR1 and to SMR0 (SIO XX11B is executed), both the PB<sub>3</sub>/SI and PB<sub>2</sub>/SO pins work as the SI and SO pins. The data to be output from the SO pin are output at the fall edge of clocks to be output from (or input to) the PA<sub>2</sub>/SCK pin while the data input to the SI pin are read in at the rise edge of same clock. The serial I/O data are thus output first from the SO pin, then shifted and simultaneously read in by the SI pin.

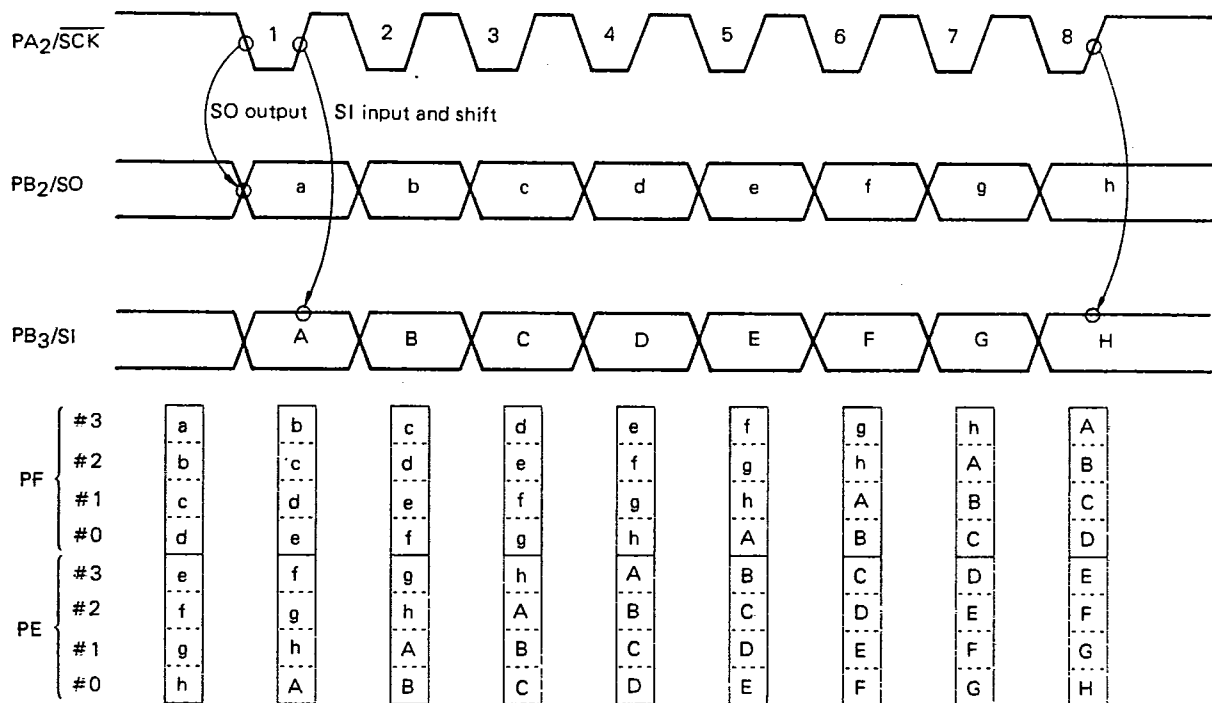


Fig. 10 SIO Operation

As shown above, the presettable shift registers (PF & PE) are sequentially shifted to read and write data. Thus data can be simultaneously transmitted and received when the SI & SO pins of  $\mu$ COM standard are connected to those of  $\mu$ PD1709CT.

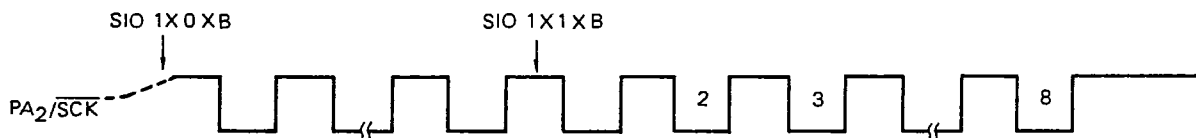
A typical application may be found in the transfer of serial data to external shift registers. It starts with the execution of SIO 1X11B instruction, then synchronizing with the clock output from the PA<sub>2</sub>/ $\overline{\text{SCK}}$  pin the data of presettable shift registers (PF & PE) are sequentially shifted and output from the PB<sub>2</sub>/SO. pin. The presettable shift registers are however not shifted if the content of SMR1 is "0", and instead the last data during the execution of SIO instruction is output from the SO pin. SMR1 must always be set to "1" when the serial I/O is used.

### (3) SMR3 (Internal $\overline{\text{SCK}}$ Enable bit)

SMR3 is the bit to select whether the internal or external clock is used as the shift clock. The internal clocks (15 kHz, duty 50 %) is output from the PA<sub>2</sub>/ $\overline{\text{SCK}}$  pin when SMR3 is set with "1" or it is allowed to output the external clocks into the PA<sub>2</sub>/ $\overline{\text{SCK}}$  pin by setting "0". To use the PA<sub>2</sub>/ $\overline{\text{SCK}}$  pin for input or output of  $\overline{\text{SCK}}$ , or to execute the SIO instruction, PA<sub>2</sub> must be first set to the input mode (to set "0" to bit #2 of PAIO word). Otherwise the device is damaged if it is used as the  $\overline{\text{SCK}}$  pin by executing the SIO instruction while keeping PA<sub>2</sub> under the output mode ("1" to bit #2 of PAIO word).

When "1" is set to SMR3 by the SIO instruction, the internal clock is immediately generated and output from the PA<sub>2</sub>/ $\overline{\text{SCK}}$  pin. A set of 8 clocks are output and the clock automatically stopped when "1" is set to SMR1 before or simultaneously when "1" is set to SMR3.

The internal clock to be output is synchronized with the machine cycle, and one clock is equal to the time where two instructions are executed (single instruction execute time: 33.3  $\mu$ s). In other words, the clock stops after 16 instructions from the execution of SIO 1X1XB instruction. But the internal clock is continuously output when the SIO 1X0XB instruction is executed.



By setting SMR3 to "0", the external clock can be input into the PA<sub>2</sub>/ $\overline{\text{SCK}}$  pin; the input clock frequency ranges from DC to 200 kHz. Synchronizing with the external clocks, the presettable shift register performs the shift, but it cannot be automatically stopped under this mode even if 8 external clocks are input and the shift is carried for 8 times. The presettable shift registers continue the shift with the next clock input. Therefore it is necessary to stop the clock every 8 clock inputs and resume it after the data processing (read or write) of the presettable shift registers has been made. Whether the shift takes place 8 times or not can be tested, after the execution of SIO instruction, by the TSET (Test Shift End, then skip if True) or TSEF (Test Shift End, then skip if False) instruction, regardless of the mode of internal or external clock. The test turns true under the external clock mode only if the shift has been made 8 x (2n+1) times (n=a positive integer over 0); otherwise it turns all false.

The output of internal clock continues even if the CE pin changes from the low level to the high level, and it is stopped only under any of the following conditions:

- 1) If "0" is set to SMR3 (external clock mode); but the PA<sub>2</sub>/ $\overline{\text{SCK}}$  pin turns to high impedance (input mode).
- 2) If the shift has been made 8 times, provided that "1" is set to SMR1.
- 3) If the CKSTP instruction is executed; all bits of SMR are automatically cleared to "0".

**SMR SUMMARY**

| SMR0 | PB status before execution of SIO | Status of PB <sub>2</sub> /SO pin  |
|------|-----------------------------------|--|
| 0    | Input mode                        | PB <sub>2</sub> input port   |
| 0    | Output mode                       | PB <sub>2</sub> output port  |
| 1    | Input mode                        | Operate as SO (The output mode is automatically set to the PB <sub>2</sub> /SO pin by the execution of SIO instruction, and the content of pre-settable shift register is output synchronously with the internal or external clock.) |
| 1    | Output mode                       |  |

| SMR1 | PB status before execution of SIO | Status of PB <sub>3</sub> /SI pin   |
|------|-----------------------------------|---|
| 0    | Input mode                        | PB <sub>3</sub> input port  |
| 0    | Output mode                       | PB <sub>3</sub> output port   |
| 1    | Input mode                        | Operate as SI (Data input to SI pin is input to pre-settable shift register.)                           |
| 1    | Output mode                       | PB <sub>3</sub> output port (Data output from PB <sub>3</sub> is input to pre-settable shift register.) |

| SMR3 | Content of PAIO <sub>2</sub> | Status of PA <sub>2</sub> /SCK pin   |
|------|------------------------------|--|
| 0    | 0                            | PA <sub>2</sub> input port or external clock allowed to input (External clock can be monitored by testing PA <sub>2</sub> .) |
| 0    | 1                            | PA <sub>2</sub> output port  |
| 1    | 0                            | Internal clock output (Internal clock can be monitored by testing PA <sub>2</sub> .)   |
| 1    | 1                            | Inhibit (Device may be destroyed.)   |

| SMR |    |    |    | PB status before execution of SIO | PAIO <sub>2</sub> status before execution of SIO | Operation                               | PA <sub>2</sub> /SCK  | PB <sub>3</sub> /SI | PB <sub>2</sub> /SO                |                 |
|-----|----|----|----|-----------------------------------|--|---|-----------------------|---------------------|------------------------------------|-----------------|
| #3  | #2 | #1 | #0 |                                   |  |   |                       |                     |                                    |                 |
| 1   | X  | 1  | 0  | Input mode                        | 0 (Input mode)                                   | SI operation                            | Internal clock output | SCK output          | SI                                 | PB <sub>2</sub> |
| 0   |    |    |    |                                   |  |   | External clock input  |                     |                                    |                 |
| 1   | X  | 1  | 1  | Output mode                       | 0 (Input mode)                                   | SO operation                            | Internal clock output | SCK output          | PB <sub>3</sub> (Output port)      | SO              |
| 0   |    |    |    |                                   |  |   | External clock input  |                     |                                    |                 |
| 1   | X  | 1  | 1  | Input mode                        | 0 (Input mode)                                   | SIO operation                           | Internal clock output | SCK output          | SI or PB <sub>3</sub> (Input port) | SO              |
| 0   |    |    |    |                                   |  |   | External clock input  |                     |                                    |                 |
| 1   | X  | 0  | 0  | X                                 | 0 (Input mode)                                   | Continuous clock output (no shifts)     | SCK output            | PB <sub>3</sub>     | PB <sub>2</sub>                    |                 |
| 0   | X  | 0  | 0  | X                                 | X  | SI function not used (as used as ports) | PA <sub>2</sub>       | PB <sub>3</sub>     | PB <sub>2</sub>                    |                 |

X : Don't care

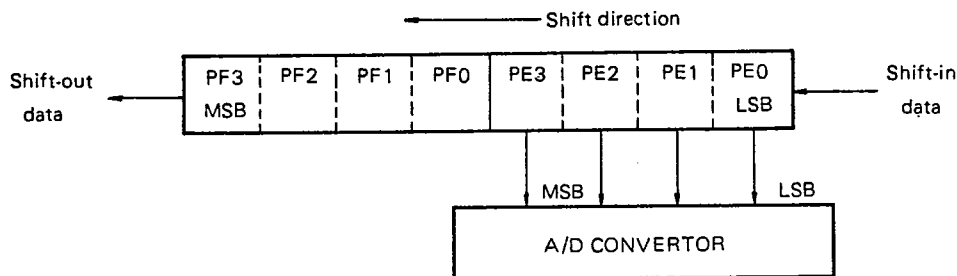
\* This status remains when the power is turned on (V<sub>DD</sub>=low to high) or when PB started again after the operation clock was stopped by the execution of CKSTP instruction.

## 5.2 PSR (PRESETTABLE SHIFT REGISTER)

PSR consists of an 8-bit shift register; its four higher-order bits are assigned to Port F of the internal port and four lower-order bits to Port E. The data setting to or read from PSR is performed with the port operation instruction that accesses to Port E and F. Data are set by the OUT or SPB/RPB instruction, or read by the IN or TPT/TPF instruction.

As the shift clock that shifts PSR, either the internal or external clock can be chosen by SMR3; but the clock that is input to or output from the  $PA_2/\overline{SCK}$  pin can shift PSR only when "1" is set to SMR1. If it is set with "0", the input of shift clock into PSR is inhibited. Four lower-order bits of PSR (Port E) are further used as the data latch of A/D converter, and "0" must be set therefore to SMR1 to inhibit the shift during the A/D conversion.

The shift is carried out in the ascending order from the lower-order bit; the shift-out data is output at the fall of shift clock or the PSR data is shifted at the rise and the shift-in data is read in.



Note: Data can be set to PSR only when SMR1=0 or SMR1=1 and the  $PA_2/\overline{SCK}$  pin is on the high level; otherwise no data can be properly set to PSR.

## 5.3 SCC (SHIFT CLOCK COUNTER)

The SCC is a binary counter consisting of four bits, and it counts the shift clocks to be input into PSR when "1" is set to SMR1 or when PSR can be shifted. Then it outputs "1" to the judge input of CPU when the counter counts up 8 (1000B) or it outputs "0" otherwise.

The judge output can be tested either by the TSET (Test Shift End, skip if True) instruction or the TSEF (Test Shift End, skip if False) instruction. The test turns true when the counter counts 8 or it otherwise turns false.

The content of SCC is cleared to "0" under any of the following conditions:

- 1) When the power is turned on ( $V_{DD}$ =low to high),
- 2) When the CKSTP instruction is executed and the internal clock stops, or
- 3) When the SIO instruction is executed.

While the internal clocks are used as the shift clock, the internal clock is stopped by the afore-mentioned judge output so that only 8 bits of data can be transferred or received. Unless SCC is cleared, the execution of TSET/TSEF instruction results in the True condition. The shift clock input is not inhibited, if the external clocks are used, even if 8 clocks are input. It is therefore necessary to stop the external clocks when 8 clocks are input.

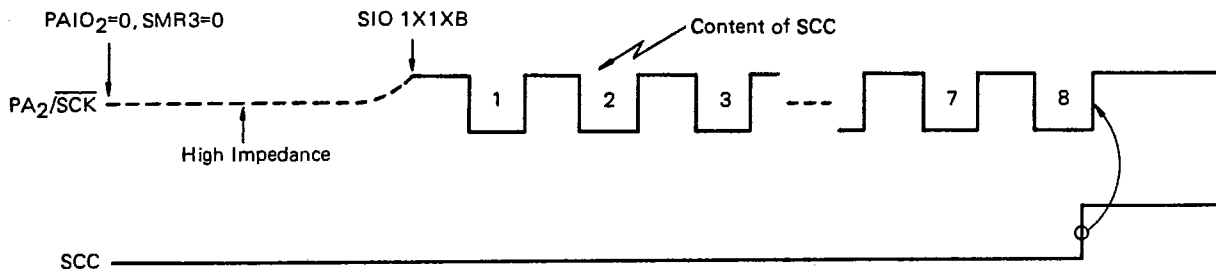
The judge output turns to be True only when  $8 \times (2n+1)$  external clocks were input ( $n$  = a positive integer other than 0); otherwise it turns to be False. Hence it is when the content of Bit #3 of SCC changes from "0" to "1" that SCC outputs "1" to the judge input.



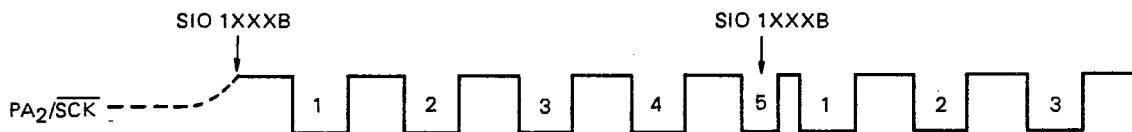
**5.4 SCG (SHIFT CLOCK GENERATOR)**

The SCG is a clock generator that produces the internal clocks of 15 kHz/50 % duty, which are output from the PA<sub>2</sub>/SCK pin when "1" is set to SMR3. The clock is synchronized with the machine cycle and each clock is equivalent to the time consumed to execute two instructions (execute time: 33.3 μs).

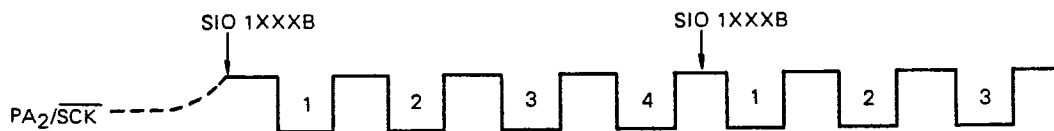
The SCG always output the clocks ranging from the high level to the active low level when "1" is set to SMR3, and it stops the clock generation with the high level output when the True judge output is made by SCC.



If "0" is set to SMR1 (to inhibit the shift) or when the SIO 1X0XB instruction is executed, the output of clocks continues by 8 clocks each without being automatically stopped. The clock output starts from the high level whenever the SIO instruction is executed. If the SIO 1XXXB instruction is executed again after the same instruction was once executed, the duty of clock deviates from 50 %, depending on the timing of execution.



To prevent such a deviation, the PA<sub>2</sub>/SCK pin must be turned to PA<sub>2</sub> and tested (by executing the TPT/TPF instruction), and the SIO 1XXXB instruction must be executed by a instruction that falls an even numbered instruction after PA<sub>2</sub> is judged high. In other words it is enough to execute the SIO 1XXXB instruction once again when the clock becomes the high level. Simply speaking, the SIO instruction needs be executed again at an even execution timing after the 1st execution of the SIO instruction.



The PA<sub>2</sub>/SCK pin is held on a high impedance until after the SIO instruction is executed, for PA<sub>2</sub> is set under the input mode before executing the instruction. Therefore it is required to connect the pull-up resistor to the PA<sub>2</sub>/SCK pin when the serial I/O is used.

5.5 EXTENDED APPLICATION OF SERIAL I/O

(1) Transfer of Data over 8 bits with Internal Clock

The continuous transfers of 8-bit or larger data with the internal clocks can be performed by setting the data of 4 bits each to PSR at a given timing during the shift and then executing the SIO instruction again. This action is detailed below with an example of 12-bit data transfer.

An 8-bit data is first set to PSR and the SIO 1X11B instruction is executed. The internal clock (PSR shift clock), which is output from the PA<sub>2</sub>/SCK pin, is synchronous with the machine cycle, as mentioned before. The content of PSR is therefore shifted by four bits at the 7th instruction after the execution of SIO instruction. Four remaining bits are set at the 7th instruction to four lower-order bits of PSR, viz., Port E by the OUT instruction, and the SIO 1X11B instruction is executed at the 8th instruction. Thus another 8-bit data can be transferred without suspending the clock output, as the content of SCC is cleared by the execution of SIO instruction.

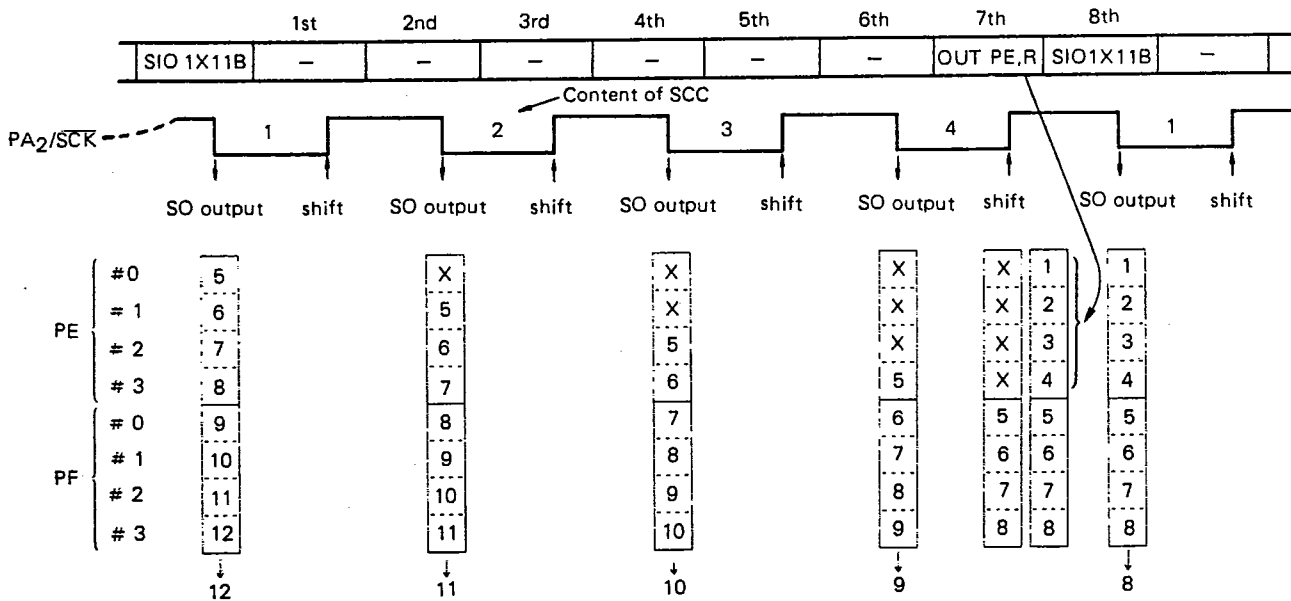


Fig. 11. Example of 12-bit Data Transfer

To similarly transfer the n-bit data ( $8 < n \leq 12$ ), the OUT instruction must be executed at the  $(n-8) \times 2 - 1$  instruction after the 1st SIO instruction execution, then the same instruction is to be executed again at the next instruction or  $(n-8) \times 2$  instruction. The transfer of data of 12 or more bits can be likewise achieved by the repeated executions of both the OUT and SIO instructions.

(2) Reception of Data over 8 bits with Internal Clock

The afore-mentioned transfer process can be applied to the continuous reception of data more than 8 bits with the internal clock, provided that the execution timing of IN instruction differs from that of the OUT instruction. The following example is the reception of data more than 12 bits.

The Port B must be set to the input mode (by executing the IN instruction) before the 1st execution of SIO 1X1XB instruction. Then the SIO instruction is again executed to clear the SCC content at the 8th instruction after the 1st execution or when a 4-bit serial data has been input to PSR. Then the IN instruction is executed against Port E (four-lower-order bits of PSR) at the next instruction or 9th instruction after the SIO instruction, thereby storing the 4-bit data into RAM. When the contents of both Port E and Port F are then stored when the shift is stopped, the reception of 12-bit serial data can be completed.

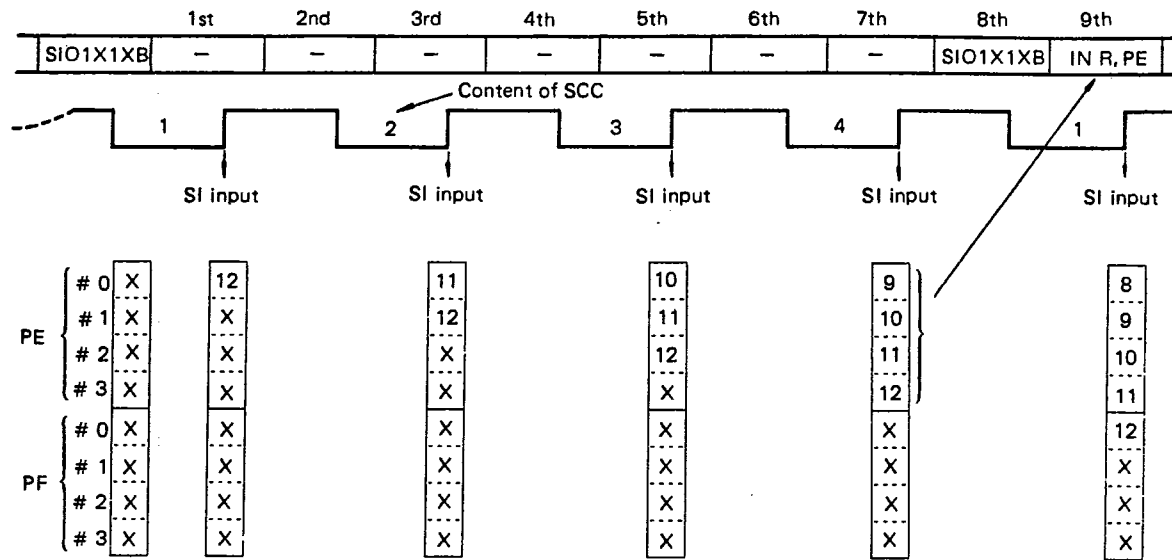


Fig. 12 Reception of 12-bit Data

To receive the n-bit data ( $8 < n \leq 12$ ), the SIO instruction is generally executed at the  $(n-8) \times 2$  instruction after the 1st SIO instruction execution, and the IN instruction is executed against Port E at the next instruction or  $(n-8) \times 2 + 1$ . The data more than 12 bits can be similarly received by the repeated executions of both the SIO and IN instructions.

**Note:** As apparent from the above two examples, it is not allowed to transfer and receive data more than 8 bits simultaneously. While the OUT instruction must be executed in the process of transfer at the  $(n-8) \times 2 - 1$  instruction, the IN instruction must be executed in the process of data reception at the  $(n-8) \times 2 + 1$  instruction. If data are simultaneously transferred and received, therefore, the 4-bit data initially stored is erased by the execution of OUT instruction. Then the data to be read out by the IN instruction is the data that is set by the OUT instruction rather than the shifted-in data.

## 6. A/D CONVERTER

The  $\mu$ PD1709CT has a built-in 4-bit A/D converter of the sequential comparison by program. The analog input is input from the A/D IN pin; when it is used for TV reception, the S-shaped curve that is output by VIF (video intermediate frequency) can be used as the judge signal of an optimum receiving frequency under the AFT mode (auto fine-tuning). The A/D IN pin can be also used as the 1-bit input port the threshold level of which can be set by the program.

### 6.1 PRINCIPLE OF OPERATION

The A/D converter of  $\mu$ PD1709CT consists of a 4-bit D/A converter of resistance-string type and a comparator. Data can be set to the D/A converter via the internal Port E. Hence the comparison data is set by the execution of an output instruction (OUT, SPB or RPB instruction), and the comparison data can be read out by the execution of an input instruction (IN, TPT or TPF instruction).

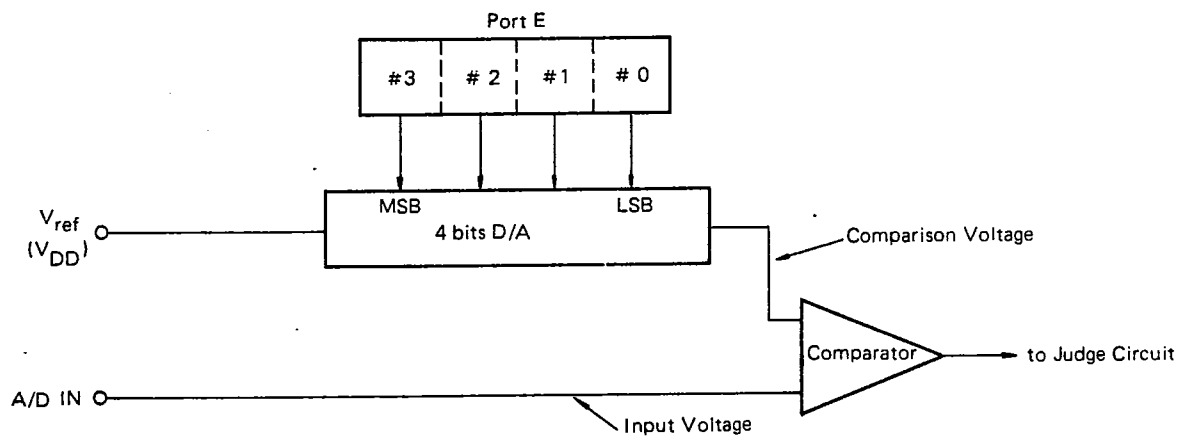


Fig. 13 A/D Converter Configuration

The Port E is used as the data latch for serial I/O so that SMR1 must be set to "0" (shift inhibit mode) during the A/D conversion.

The D/A converter generates as the comparison voltage 16 different voltage (divided from the reference voltage V<sub>ref</sub> (V<sub>DD</sub>)), depending on the data preset to Port E. This comparison voltage is input into the comparator together with the analog voltage (input voltage), which is input into the A/D IN pin, and both are compared. The compared result can be obtained by executing either the TADT (Test A/D comparator, then skip if True) instruction or the TADF (Test A/D comparator, then skip if False) instruction. The following relation is set up between the input voltage and the comparison voltage:

Input voltage > comparison voltage, True  
 Input voltage ≤ comparison voltage, False

### 6.2 STRUCTURE OF D/A CONVERTER

The D/A converter of  $\mu$ PD1709CT is of the resistance-string type that serially connects 16 resistors between  $V_{ref}$  ( $V_{DD}$ ) and GND and selects the voltage at each connecting point. It must be noted that the values of serially-connected resistors differ from each other and that the resistor closest to GND has the resistance of  $0.5/16$  while the one closest to  $V_{ref}$  has  $1.5/16$ . Fig. 14 shows the structure of D/A converter.

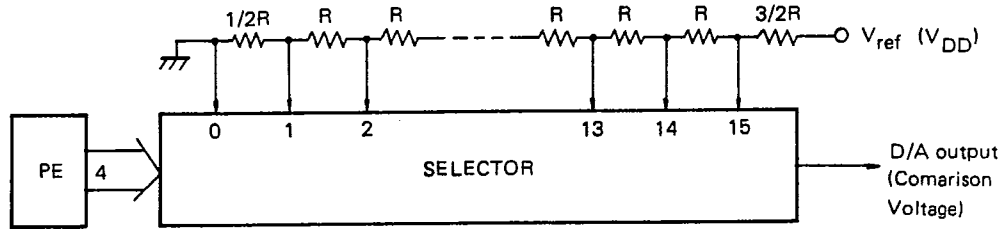


Fig. 14 Structure of D/A Converter

The D/A converter of this structure outputs the GND level when the value 0000B is set to Port E or outputs the comparison voltage of  $0.5/16 \times V_{ref}$  when the value 0001B is set. Similarly the comparison voltage  $V_{OUT}$  can be expressed by the following formula when the value  $n$  (decimal) is set:

$$V_{out} = V_{ref} \times \frac{n - 0.5}{16} \quad (15 \geq n \geq 1)$$

Table 6 Input Data – Comparison Voltage

| Input Data (Port E) |        | Comparison Voltage |                |
|---------------------|--------|--------------------|----------------|
| DEC.                | Binary | $\times V_{ref}$   | $V_{ref} = 5V$ |
| 0                   | 0000   | 0                  | 0 (V)          |
| 1                   | 0001   | $0.5/16$           | 0.15625        |
| 2                   | 0010   | $1.5/16$           | 0.46875        |
| 3                   | 0011   | $2.5/16$           | 0.78125        |
| 4                   | 0100   | $3.5/16$           | 1.09375        |
| 5                   | 0101   | $4.5/16$           | 1.40625        |
| 6                   | 0110   | $5.5/16$           | 1.71875        |
| 7                   | 0111   | $6.5/16$           | 2.03125        |
| 8                   | 1000   | $7.5/16$           | 2.34375        |
| 9                   | 1001   | $8.5/16$           | 2.65625        |
| 10                  | 1010   | $9.5/16$           | 2.96875        |
| 11                  | 1011   | $10.5/16$          | 3.28125        |
| 12                  | 1100   | $11.5/16$          | 3.59375        |
| 13                  | 1101   | $12.5/16$          | 3.90625        |
| 14                  | 1110   | $13.5/16$          | 4.21875        |
| 15                  | 1111   | $14.5/16$          | 4.53125        |

As the A/D conversion data the A/D converter of sequential comparison type generally outputs the value  $n$ , if the following relation is established between the input voltage  $V_{in}$  and the comparison voltage  $V_n$  against the data  $n$ , which is set to D/A:

$$V_n < V_{in} \leq V_{n+1}$$

Assuming now  $V_{ref}=5.0$  V and  $V_{ref}/16$  or  $0.3125$  V is input into  $V_{in}$ ; then the output voltages (comparison voltages) of D/A converter,  $V_0$ ,  $V_1$  and  $V_2$  when  $n=0, 1$  and  $2$  are:

$$V_0 = 0 \text{ V}$$

$$V_1 = 0.15625 \text{ V}$$

$$V_2 = 0.46875 \text{ V}$$

so that the comparison data of A/D converter should be  $n=1$ . If, therefore, the analog voltage  $V_{in}$  to be input is in the relation of  $0.15625 \text{ V} < V_{in} \leq 0.46875 \text{ V}$ , the voltage of  $0.3125 \text{ V}$  is all input, or  $0 \text{ V}$  is considered to have been input if  $V_0 < V_{in} \leq V_1$ . In this term the input voltage of  $V_{in} = V_{ref}/16$  (1 LBS) is called the least decomposition voltage.

As apparent from the above, the input of  $0.3125 \text{ V}$  is considered to be made if the input voltage is held in the relation of  $0.15625 \text{ V} < V_{in} \leq 0.46875 \text{ V}$ . There would be the least read error of  $\pm 0.15625 \text{ V}$  against the actual input voltage. This kind of error is known the quantized error of A/D converter, which is expressed by  $\pm 1/2$  LSB or by  $\pm 1/16 \times 2 = \pm 3.125 \%$  as it is a 4-bit A/D converter. The largest read error is  $\pm 1$  LSB.

The measurable range of input voltage  $V_{in}$  of the A/D converter is:

$$0 \text{ V} \leq V_{in} \leq V_{15} (V_{ref} \times 14.5/16)$$

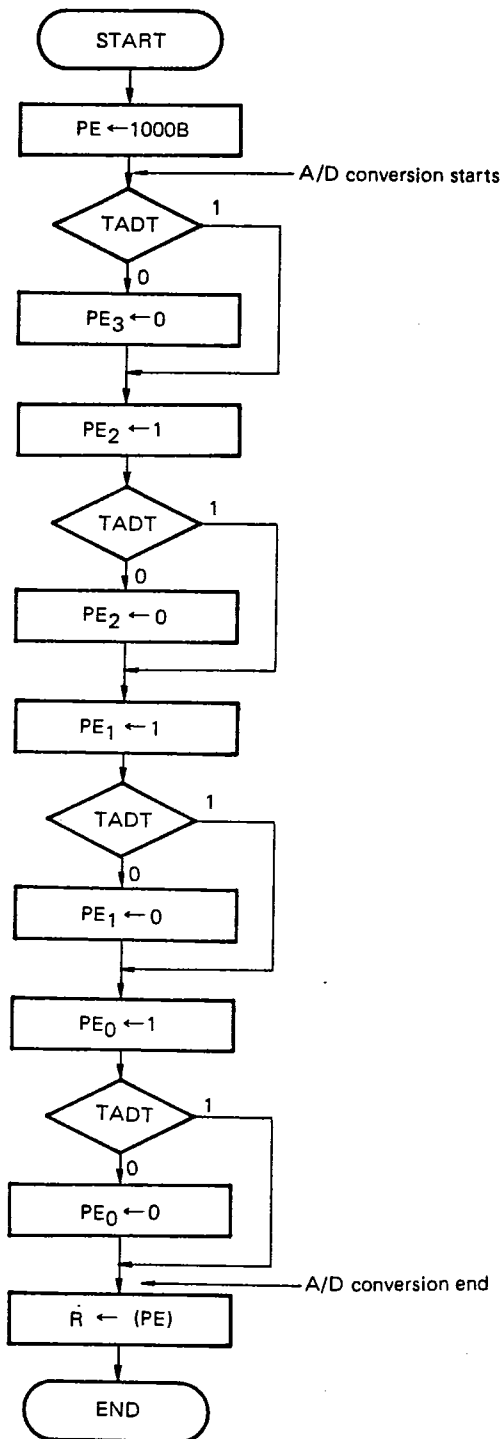
If  $V_{in}$  is within the range of  $V_{15} < V_{in} \leq V_{ref}$  ( $V_{DD}$ ), it is considered as an over-range.

The TADT or TADF instruction must be executed for A/D conversion, as mentioned above. When the comparison voltage of  $V_{13}$  is output against the input of analog voltage of  $V_{in} = V_{14}$  ( $V_{ref} \times 13.5/16$ ), the input voltage is larger than the comparison voltage; hence the data True is output to the Judge circuit. Or the data False is output if  $V_{14}$  is output as the input voltage is equal to or smaller than the comparison voltage.

6.3 TYPICAL A/D CONVERSION PROGRAM

Binary Search Method

Typical Flow Chart



Typical Coding

START :

```

MVI R,1000B
BANK1
OUT 0,R
TADT
RPB 0,1000B
SPB 0,0100B
TADT
RPB 0,0110B
SPB 0,0010B
TADT
RPB 0,0010B
SPB 0,0001B
TADT
RPB 0,0001B
IN R,0
BANK0
    
```

END :

A/D conversion time: 366.3  $\mu$ s  
 No. of total steps: 16 steps

## 7. PLA (PROGRAMMABLE LOGIC ARRAY)

The  $\mu$ PD1709CT has the segment PLA to be used by the user programs. Generally the display patterns, key return signal source patterns for dynamic display are programmed, and the segment PLA can generate a total of 32 different kinds of patterns, 16 kinds by two.

### 7.1 SEGMENT PLA CONFIGURATION

The segment PLA consists of a 5-bit segment latch circuit and a 7-bit output PLA into which the output of the segment latch circuit is input and which corresponds to the segment pins (Sa to Sg).

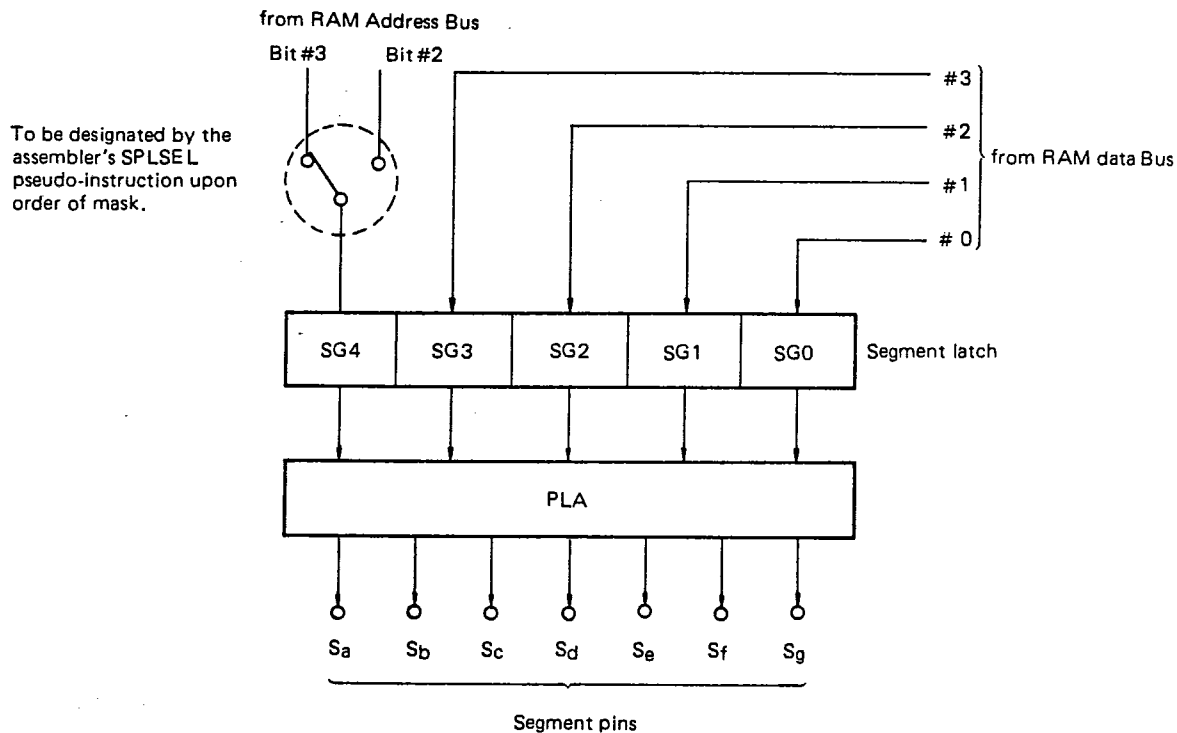


Fig. 15 Segment PLA Configuration

The content of data memory (RAM) designated by the operand of SEG instruction is latched to four lower-order bits (SG0 to SG3) of the segment latch circuit. If, for instance, the SEG 1,05H instruction is executed for the content of RAM as shown below, the content of RAM which is addressed by 1 (to designate the row address of RAM) and the content of register 05H (to designate the column address), that is, the address 16H content "F" is latched.

|             |   | Column address |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|-------------|---|----------------|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
|             |   | 0              | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8 | 9 | A | B | C | D | E | F |
| Row address | 0 |                |   |   |   |   | 6 |   |   |   |   |   |   |   |   |   |   |
|             | 1 |                |   |   |   |   |   | F |   |   |   |   |   |   |   |   |   |
|             | 2 |                |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |
|             | 3 |                |   |   |   |   |   |   |   |   |   |   |   |   |   |   |   |



To the highest-order bit SG4 of the segment latch circuit the content of either Bit #3 or #2 of the column address of RAM as designated by the SEG instruction is latched. It must be designated upon ordering of the mask which bit content should be latched (see 7.3, PLA Program). If Bit #3 is connected, "0" is latched to SG4 when RAM over the column addresses, 00H to 07H, is designated by the SEG instruction or "1" is latched if 08H to 0FH are designated. When Bit #2 is connected, "0" is latched against 00H to 03H as well as 08H to 0BH or "1" against 04H to 07H as well as 0CH to 0FH.

The segment patterns of 32 different kinds are divided into two groups of 16 different patterns, depending on the data to be latched to SG4. Different segment patterns can be therefore output from the segment pin so far as different column addresses of RAM are designated by the SEG instruction, even if the data of RAM are same. Sixteen different patterns to be generated when "0" is set to SG4 are called the "pattern group 0" and those when "1" is set are called the "pattern group 1".

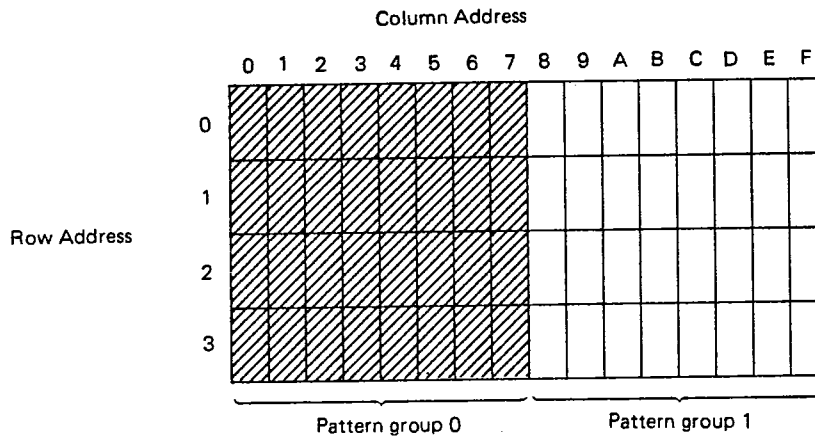


Fig. 16 Division of Pattern Groups with SG4 set to Bit #3

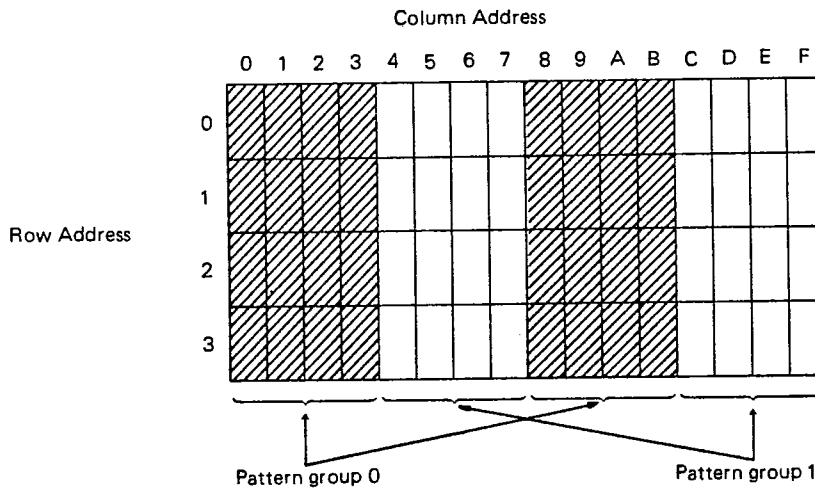


Fig. 17 Division of Pattern Groups with SG4 set to Bit #2

The division of pattern groups is to be determined in the creation of program, considering the RAM or program (ROM) efficiency. The designation of SG4 input to Bit #3 or Bit #2 is made by SPLSEL, which is described as:

SPLSEL 3 or SPLSEL 2

(see 7.3 TYPICAL PLA PROGRAM)

### 7.2 SEGMENT PLA PATTERNS

The patterns of Pattern Groups 0 and 1 are shown in Tables 7 and 8 below.

Table 7 Patterns of Pattern Group 0

| Segment Latch |     |     |     |     | Segment Output (Note) |   |   |   |   |   |   | Output Pattern      |
|---------------|-----|-----|-----|-----|-----------------------|---|---|---|---|---|---|---------------------|
| SG4           | SG3 | SG2 | SG1 | SG0 | g                     | f | e | d | c | b | a |                     |
| 0             | 0   | 0   | 0   | 0   | 1                     | 0 | 0 | 0 | 0 | 0 | 0 |                     |
|               | 0   | 0   | 0   | 1   | 1                     | 1 | 1 | 1 | 0 | 0 | 1 |                     |
|               | 0   | 0   | 1   | 0   | 0                     | 1 | 0 | 0 | 1 | 0 | 0 |                     |
|               | 0   | 0   | 1   | 1   | 0                     | 1 | 1 | 0 | 0 | 0 | 0 |                     |
|               | 0   | 1   | 0   | 0   | 0                     | 0 | 0 | 1 | 1 | 0 | 1 |                     |
|               | 0   | 1   | 0   | 1   | 0                     | 0 | 0 | 1 | 0 | 0 | 1 |                     |
|               | 0   | 1   | 1   | 0   | 0                     | 0 | 0 | 0 | 0 | 1 | 0 |                     |
|               | 0   | 1   | 1   | 1   | 1                     | 1 | 0 | 1 | 1 | 0 | 0 |                     |
|               | 1   | 0   | 0   | 0   | 0                     | 0 | 0 | 0 | 0 | 0 | 0 |                     |
|               | 1   | 0   | 0   | 1   | 0                     | 0 | 0 | 1 | 0 | 0 | 0 |                     |
|               | 1   | 0   | 1   | 0   | 0                     | 1 | 1 | 1 | 1 | 1 | 1 | BLANK (Display off) |
|               | 1   | 0   | 1   | 1   | 0                     | 0 | 0 | 0 | 0 | 1 | 1 |                     |
|               | 1   | 1   | 0   | 0   | 0                     | 0 | 1 | 1 | 1 | 1 | 1 |                     |
|               | 1   | 1   | 0   | 1   | 0                     | 1 | 1 | 1 | 0 | 1 | 1 |                     |
|               | 1   | 1   | 1   | 0   | 0                     | 0 | 0 | 1 | 1 | 1 | 0 |                     |
| 1             | 1   | 1   | 1   | 0   | 0                     | 1 | 0 | 0 | 0 | 1 |   |                     |

Note: The segment output is set active low; to make it active high, "0" must be replaced by "1" or vice versa. The segment output of  $\mu$ PD 1709CT is generally set active low as it is of the N-ch open drain format.

Table 8 Patterns of Pattern Group 1

| Segment Latch |     |     |     |     | Segment Output |   |   |   |   |   |   | Output Pattern |
|---------------|-----|-----|-----|-----|----------------|---|---|---|---|---|---|----------------|
| SG4           | SG3 | SG2 | SG1 | SG0 | g              | f | e | d | c | b | a |                |
|               | 0   | 0   | 0   | 0   | 0              | 0 | 0 | 0 | 0 | 0 | 0 | Key all ON     |
|               | 0   | 0   | 0   | 1   | 1              | 1 | 1 | 1 | 1 | 1 | 0 | Seg a (Key)    |
|               | 0   | 0   | 1   | 0   | 1              | 1 | 1 | 1 | 1 | 0 | 1 | Seg b (Key)    |
|               | 0   | 0   | 1   | 1   | 1              | 1 | 1 | 1 | 0 | 1 | 1 | Seg c (Key)    |
|               | 0   | 1   | 0   | 0   | 1              | 1 | 1 | 0 | 1 | 1 | 1 | Seg d (Key)    |
|               | 0   | 1   | 0   | 1   | 1              | 1 | 0 | 1 | 1 | 1 | 1 | Seg e (Key)    |
|               | 0   | 1   | 1   | 0   | 1              | 0 | 1 | 1 | 1 | 1 | 1 | Seg f (Key)    |
|               | 0   | 1   | 1   | 1   | 0              | 1 | 1 | 1 | 1 | 1 | 1 | Seg g (Key)    |
| 1             | 1   | 0   | 0   | 0   | 1              | 1 | 1 | 1 | 1 | 1 | 1 | Key all OFF    |
|               | 1   | 0   | 0   | 1   | 1              | 1 | 1 | 1 | 1 | 1 | 1 | No use         |
|               | 1   | 0   | 1   | 0   | 1              | 1 | 1 | 1 | 1 | 1 | 1 | No use         |
|               | 1   | 0   | 1   | 1   | 1              | 1 | 1 | 1 | 1 | 1 | 1 | No use         |
|               | 1   | 1   | 0   | 0   | 1              | 1 | 1 | 1 | 1 | 1 | 1 | No use         |
|               | 1   | 1   | 0   | 1   | 1              | 1 | 1 | 1 | 1 | 1 | 1 | No use         |
|               | 1   | 1   | 1   | 0   | 1              | 1 | 1 | 1 | 1 | 1 | 1 | No use         |
|               | 1   | 1   | 1   | 1   | 1              | 1 | 1 | 1 | 1 | 1 | 1 | No use         |

The following are output by the execution of SEG instruction when any of the above display patterns is used, provided that the SG4 input of segment latch circuit is designated with Bit #3 of the column address.

|             |   | Column Address  |   |   |   |   |   |   |   |                 |   |   |   |   |   |   |   |
|-------------|---|-----------------|---|---|---|---|---|---|---|-----------------|---|---|---|---|---|---|---|
|             |   | 0               | 1 | 2 | 3 | 4 | 5 | 6 | 7 | 8               | 9 | A | B | C | D | E | F |
| Row Address | 0 |                 |   |   | 4 | A |   |   |   |                 |   |   |   |   |   |   |   |
|             | 1 |                 |   |   |   | 3 |   |   |   |                 |   |   |   |   |   |   |   |
|             | 2 |                 |   |   |   |   |   |   |   |                 |   | 3 |   |   |   |   |   |
|             | 3 |                 |   |   |   |   |   |   |   |                 |   |   |   |   |   |   |   |
|             |   | Pattern group 0 |   |   |   |   |   |   |   | Pattern group 1 |   |   |   |   |   |   |   |

(1) SEG 1, 03H . . . The content (0011B) of RAM address 14H is latched.

Output Pattern 

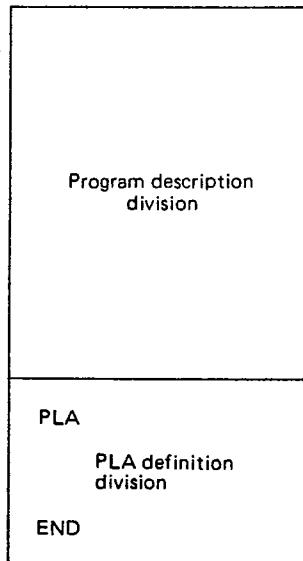
(2) SEG 2, 04H . . . The content (0011B) of RAM address 2AH is latched.

Output Pattern 

As shown above any different pattern can be output as far as the content of SG4 is different, even if the data to be set to four lower-order bits of the segment circuit are same.

### 7.3 TYPICAL PLA PROGRAM

The PLA should be defined for any device of  $\mu$ PD1700 Series. No tapes that do not define the PLA can be accepted when they are ordered. The PLA definition is described with the following items at the end of each assembler source program. All items should be stated without omitting any of them.



#### 1. PLA Pseudo-Instruction

This description indicates the start of PLA definition division simultaneously with the end of program description division.

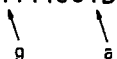
**2. SPLSEL (Segment PLA Select) Pseudo-Instruction**

This description selects the division of RAM addresses by which to create the segment pattern groups 0 and 1; either of the following two descriptions can be used:

SPLSEL 3 or SPLSEL 2


**3. DSP (Define Segment PLA) Pseudo-Instruction**

This instruction defines 32 different patterns of the segment PLA, starting with 16 patterns of Pattern Group 0. The definition may be described as follows; the 1st bit to be described corresponds to the segment g, as shown below, and others to f, e, d, c, b and a.

DSP 1111001B  


**4. MTDIG Pseudo-Instruction**

This instruction defines the digit pins (D<sub>3</sub>, D<sub>2</sub>) to be active when the CE pin turns to the low level, as shown below:

MTDIG 11B  


The bit to be first described corresponds to the D<sub>3</sub> pin and the next bit to the D<sub>2</sub> pin. When "1" is set to either of the digit pins of μPD1709CT, which is the N-ch open drain output, it is held at a high impedance while the CE pin is on the low level.

**5. END**

This description expresses the end of source program simultaneously with the end of PLA definition division. No programs can be assembled without this description.

- Note 1:** Out of the μPD1700 Series any device having the digit PLA (i.e. μPD1701C, 1703C, 1704C, 1705C, 1707G & 1710G) must be defined with the digit PLA by the DDP (define digit PLA) pseudo instruction.
- 2:** "PLA" must be described first and "END" at the last of PLA definition, but "SPLSEL", "DSP", "DDP" & "MTDIG" can be described between the above two descriptions in any order.

## Typical PLA Programs

```

: * * * *  PLA DEFINITION  * * * *
PLA
:
SPLSEL 3          : RAM ADDR BIT = 3
:
:***  SEGMENT PATTERN 0  ***
:
:           gfedcba
DSP          1000000B   : 0
DSP          1111001B   : 1
DSP          0100100B   : 2
DSP          0110000B   : 3
DSP          0011001B   : 4
DSP          0010010B   : 5
DSP          0000010B   : 6
DSP          1011000B   : 7
DSP          0000000B   : 8
DSP          0010000B   : 9
DSP          1111111B   : BLANK
DSP          0000110B   : E
DSP          0111111B   : SEG G ON
DSP          1110111B   : SEG D ON
DSP          0011100B   : SEG A, B, F, G ON
DSP          0100011B   : SEG C, D, E, G ON
:
:***  SEGMENT PATTERN 1  ***
:
DSP          0000000B   : SEG ALL ON, KEY RETERN
DSP          1111110B   : SEG A ON
DSP          1111101B   : SEG B ON
DSP          1111011B   : SEG C ON
DSP          1110111B   : SEG D ON
DSP          1101111B   : SEG E ON
DSP          1011111B   : SEG F ON
DSP          0111111B   : SEG G ON
DSP          1111111B   : SEG ALL OFF
DSP          1111111B   : NO USE
DSP          1111111B   : "
DSP          1111111B   : "
DSP          1111111B   : "
DSP          1111111B   : "
DSP          1111111B   : "
DSP          1111111B   : "
DSP          1111111B   : "
:
MTDIG          11B          : ALL DIG OFF WHEN CE=LOW
:
END

```

μPDI709CT INSTRUCTION SET

μPDI709CT Instruction Set Table

| b15 b14         |   | 0 0  | 0 1       | 1 0                          | 1 1       |
|-----------------|---|--|-----------|------------------------------|-----------|
| b13 b12 b11 b10 |   | 0  | 1         | 2                            | 3         |
| 0 0 0 0         | 0 | NOP<br>SIO N                               |           | DIG r                        | ST M, r   |
| 0 0 0 1         | 1 | SPB P, N<br>SS N1<br>BANK1<br>EI<br>STC    | ORI M, I  | SEG D11, r                   | MVRS M, r |
| 0 0 1 0         | 2 | JMP ADDR<br>(page 1)                       | MVI M, I  | OUT P, r                     | IN r, P   |
| 0 0 1 1         | 3 | RPB P, N<br>RS N1<br>BANK0<br>DI<br>RSC    | ANI M, I  | CKSTP                        | MVRD r, M |
| 0 1 0 0         | 4 | RT   | AI M, I   | MVSR M1, M2                  | AD r, M   |
| 0 1 0 1         | 5 | RTS  | SI M, I   | EXL r, M                     | SU r, M   |
| 0 1 1 0         | 6 | JMP ADDR<br>(page 0)                       | AIC M, I  | LD r, M                      | AC r, M   |
| 0 1 1 1         | 7 | CAL ADDR<br>(page 0)                       | SIB M, I  |                              | SB r, M   |
| 1 0 0 0         | 8 | SBK0 P, N<br>TPF N2<br>TSF<br>TCEF<br>TITF | AIN M, I  | TSET<br>TSEF<br>TADT<br>TADF | ADN r, M  |
| 1 0 0 1         | 9 | SBK1 P, N<br>TPT N2<br>TST<br>TCET<br>TITT | SIN M, I  | TTM<br>TIP                   | SUN r, M  |
| 1 0 1 0         | A | TNF M, N                                   | AICN M, I | TUL                          | ACN r, M  |
| 1 0 1 1         | B | TMT M, N                                   | SIBN M, I | PLL M, r                     | SBN r, M  |
| 1 1 0 0         | C | SLTI M, I                                  | AIS M, I  | SLT r, M                     | ADS r, M  |
| 1 1 0 1         | D | SGEI M, I                                  | SIS M, I  | SGE r, M                     | SUS r, M  |
| 1 1 1 0         | E | SEQI M, I                                  | AICS M, I | SEQ r, M                     | ACS r, M  |
| 1 1 1 1         | F | SNEI M, I                                  | SIBS M, I | SNE r, M                     | SBS r, M  |

List of PD1709CT Instructions

NOTE : D<sub>H</sub> : Data memory address high (row address) 2 bits  
 D<sub>L</sub> : Data memory address low (column address) 4 bits  
 R<sub>n</sub> : Register number 4 bits  
 I : Immediate data 4 bits  
 N : Bit position 4 bits  
 ADDR : Program memory address 10 bits  
 — : All "1"  
 r : General register  
 One of addresses 00-0FH of BANK0

M : Data memory address  
 One of 00-3FH of BANK0  
 P : Port, 0 ≤ P ≤ 3  
 N<sub>1</sub> : Bit position of status word 1 0 ≤ N<sub>1</sub> ≤ 7  
 N<sub>2</sub> : Bit position of status word 2 0 ≤ N<sub>2</sub> ≤ 7  
 ( ) : Contents of register or memory  
 c : Carry  
 b : Borrow  
 ( )<sub>n</sub> : Contents on bit n of register or memory  
 | : Decoded value of contents of register or memories

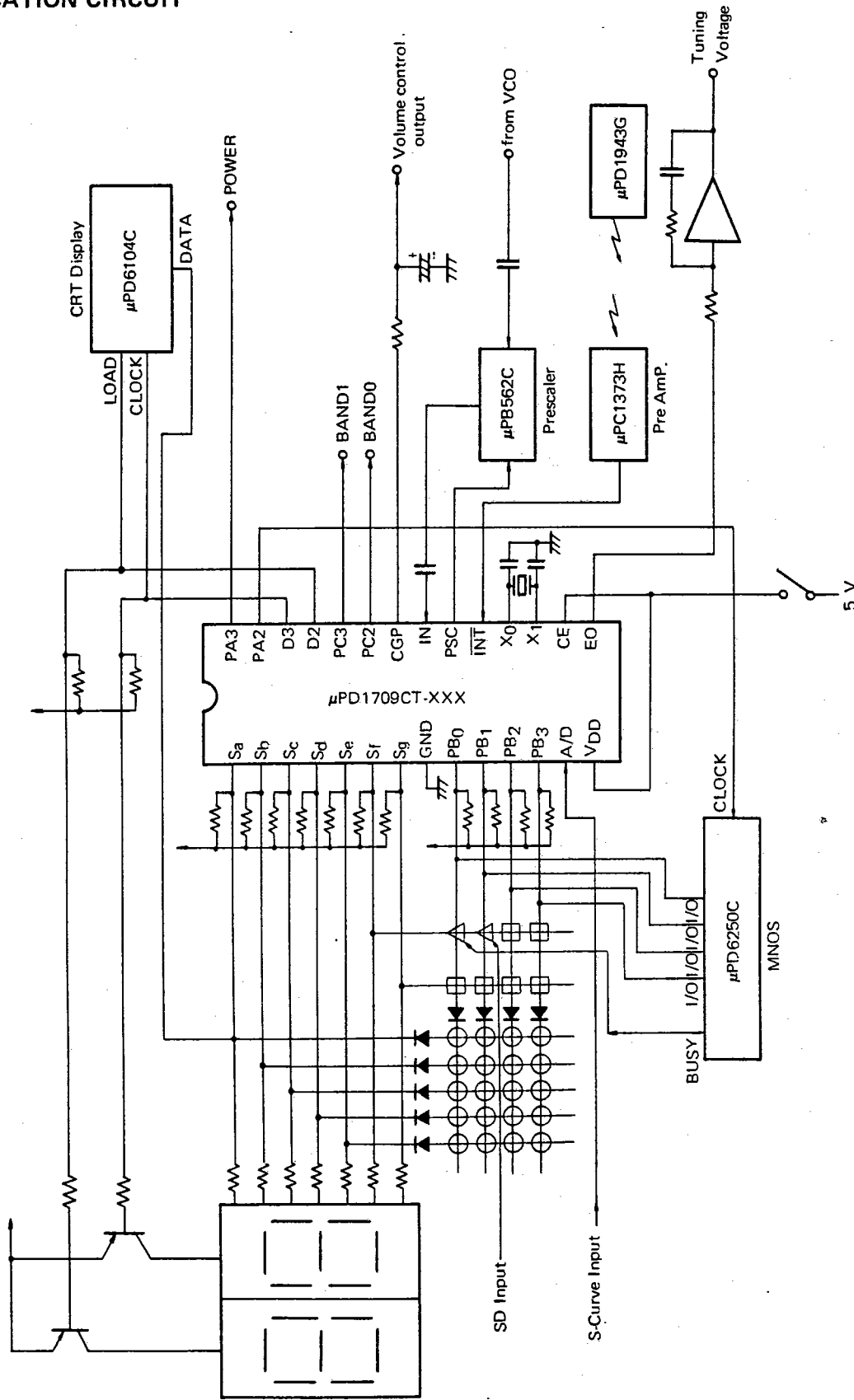
|             | Mnemonic | Operand |  | Function   | Operation  | Machine code   |                |                |                |
|-------------|----------|---------|--|--|--|----------------|----------------|----------------|----------------|
|             |          | 1ST     | 2ND  |  |  | Operation code |                |                |                |
| Addition    | AD       | r       | M  | Add memory to register   | $r \leftarrow (r) + (M)$                           | 110100         | D <sub>H</sub> | D <sub>L</sub> | R <sub>n</sub> |
|             | ADS      | r       | M  | Add memory to register, then skip if carry                         | $r \leftarrow (r) + (M)$<br>skip if carry          | 111100         | D <sub>H</sub> | D <sub>L</sub> | R <sub>n</sub> |
|             | ADN      | r       | M  | Add memory to register, then skip if not carry                     | $r \leftarrow (r) + (M)$<br>skip if not carry      | 111000         | D <sub>H</sub> | D <sub>L</sub> | R <sub>n</sub> |
|             | AC       | r       | M  | Add memory to register with carry                                  | $r \leftarrow (r) + (M) + c$                       | 110110         | D <sub>H</sub> | D <sub>L</sub> | R <sub>n</sub> |
|             | ACS      | r       | M  | Add memory to register with carry, then skip if carry              | $r \leftarrow (r) + (M) + c$<br>skip if carry      | 111110         | D <sub>H</sub> | D <sub>L</sub> | R <sub>n</sub> |
|             | ACN      | r       | M  | Add memory to register with carry, then skip if not carry          | $r \leftarrow (r) + (M) + c$<br>skip if not carry  | 111010         | D <sub>H</sub> | D <sub>L</sub> | R <sub>n</sub> |
|             | AI       | M       | I  | Add immediate data to memory                                       | $M \leftarrow (M) + I$                             | 010100         | D <sub>H</sub> | D <sub>L</sub> | I              |
|             | AIS      | M       | I  | Add immediate data to memory, then skip if carry                   | $M \leftarrow (M) + I$<br>skip if carry            | 011100         | D <sub>H</sub> | D <sub>L</sub> | I              |
|             | AIN      | M       | I  | Add immediate data to memory, then skip if not carry               | $M \leftarrow (M) + I$<br>skip if not carry        | 011000         | D <sub>H</sub> | D <sub>L</sub> | I              |
|             | AIC      | M       | I  | Add immediate data to memory with carry                            | $M \leftarrow (M) + I + c$                         | 010110         | D <sub>H</sub> | D <sub>L</sub> | I              |
| AICS        | M        | I       | Add immediate data to memory with carry, then skip if carry              | $M \leftarrow (M) + I + c$<br>skip if carry                        | 011110   | D <sub>H</sub> | D <sub>L</sub> | I              |                |
| AICN        | M        | I       | Add immediate data to memory with carry, then skip if not carry          | $M \leftarrow (M) + I + c$<br>skip if not carry                    | 011010   | D <sub>H</sub> | D <sub>L</sub> | I              |                |
| Subtraction | SU       | r       | M  | Subtract memory from register                                      | $r \leftarrow (r) - (M)$                           | 110101         | D <sub>H</sub> | D <sub>L</sub> | R <sub>n</sub> |
|             | SUS      | r       | M  | Subtract memory from register, then skip if borrow                 | $r \leftarrow (r) - (M)$<br>skip if borrow         | 111101         | D <sub>H</sub> | D <sub>L</sub> | R <sub>n</sub> |
|             | SUN      | r       | M  | Subtract memory from register, then skip if not borrow             | $r \leftarrow (r) - (M)$<br>skip if not borrow     | 111001         | D <sub>H</sub> | D <sub>L</sub> | R <sub>n</sub> |
|             | SB       | r       | M  | Subtract memory from register with borrow                          | $r \leftarrow (r) - (M) - b$                       | 110111         | D <sub>H</sub> | D <sub>L</sub> | R <sub>n</sub> |
|             | SBS      | r       | M  | Subtract memory from register with borrow, then skip if borrow     | $r \leftarrow (r) - (M) - b$<br>skip if borrow     | 111111         | D <sub>H</sub> | D <sub>L</sub> | R <sub>n</sub> |
|             | SBN      | r       | M  | Subtract memory from register with borrow, then skip if not borrow | $r \leftarrow (r) - (M) - b$<br>skip if not borrow | 111011         | D <sub>H</sub> | D <sub>L</sub> | R <sub>n</sub> |
|             | SI       | M       | I  | Subtract immediate data from memory                                | $M \leftarrow (M) - I$                             | 010101         | D <sub>H</sub> | D <sub>L</sub> | I              |
|             | SIS      | M       | I  | Subtract immediate data from memory, then skip if borrow           | $M \leftarrow (M) - I$<br>skip if borrow           | 011101         | D <sub>H</sub> | D <sub>L</sub> | I              |
|             | SIN      | M       | I  | Subtract immediate data from memory, then skip if not borrow       | $M \leftarrow (M) - I$<br>skip if not borrow       | 011001         | D <sub>H</sub> | D <sub>L</sub> | I              |
|             | SIB      | M       | I  | Subtract immediate data from memory with borrow                    | $M \leftarrow (M) - I - b$                         | 010111         | D <sub>H</sub> | D <sub>L</sub> | I              |
| SIBS        | M        | I       | Subtract immediate data from memory with borrow, then skip if borrow     | $M \leftarrow (M) - I - b$<br>skip if borrow                       | 011111   | D <sub>H</sub> | D <sub>L</sub> | I              |                |
| SIBN        | M        | I       | Subtract immediate data from memory with borrow, then skip if not borrow | $M \leftarrow (M) - I - b$<br>skip not borrow                      | 011011   | D <sub>H</sub> | D <sub>L</sub> | I              |                |



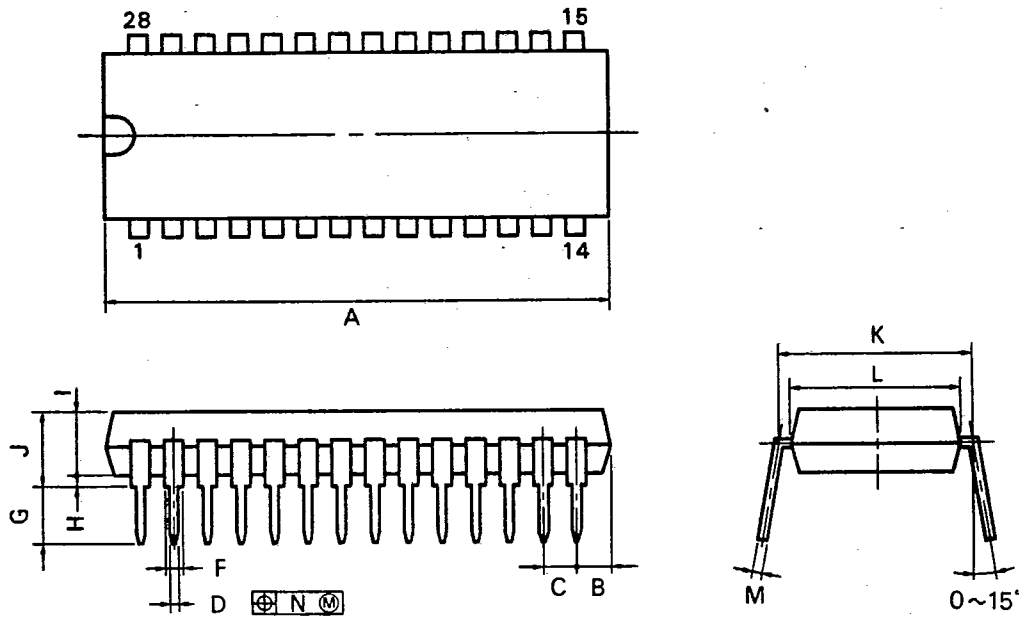
|                   | Mnemonic | Operand        |   | Function  | Operation   | Machine code   |                |                 |                 |
|-------------------|----------|----------------|---|---|---|----------------|----------------|-----------------|-----------------|
|                   |          | 1ST            | 2ND   |   |   | Operation code |                |                 |                 |
| Comparison        | SEQ      | r              | M   | Skip if register equals memory  | $r - M$<br>skip if zero   | 1 0 1 1 1 0    | D <sub>H</sub> | D <sub>L</sub>  | R <sub>n</sub>  |
|                   | SNE      | r              | M   | Skip if register not equals memory                                      | $r - M$<br>skip if not zero   | 1 0 1 1 1 1    | D <sub>H</sub> | D <sub>L</sub>  | R <sub>n</sub>  |
|                   | SGE      | r              | M   | Skip if register is greater than or equal to memory                     | $r - M$<br>skip if not borrow ( $r \geq M$ )                                      | 1 0 1 1 0 1    | D <sub>H</sub> | D <sub>L</sub>  | R <sub>n</sub>  |
|                   | SLT      | r              | M   | Skip if register is less than memory                                    | $r - M$<br>skip if borrow ( $r < M$ )   | 1 0 1 1 0 0    | D <sub>H</sub> | D <sub>L</sub>  | R <sub>n</sub>  |
|                   | SEI      | M              | I   | Skip if memory equals immediate data                                    | $M - I$<br>skip if zero   | 0 0 1 1 1 0    | D <sub>H</sub> | D <sub>L</sub>  | I               |
|                   | SNEI     | M              | I   | Skip if memory not equals immediate data                                | $M - I$<br>skip if not zero   | 0 0 1 1 1 1    | D <sub>H</sub> | D <sub>L</sub>  | I               |
|                   | SGEI     | M              | I   | Skip if memory is greater than or equal to immediate data               | $M - I$<br>skip if not borrow ( $M \geq I$ )                                      | 0 0 1 1 0 1    | D <sub>H</sub> | D <sub>L</sub>  | I               |
|                   | SLTI     | M              | I   | Skip if memory is less than immediate data                              | $M - I$<br>skip if borrow ( $M < I$ )   | 0 0 1 1 0 0    | D <sub>H</sub> | D <sub>L</sub>  | I               |
| Logical operation | ANI      | M              | I   | Logic AND of memory and immediate data                                  | $M \leftarrow (M) \wedge I$   | 0 1 0 0 1 1    | D <sub>H</sub> | D <sub>L</sub>  | $\bar{I}$       |
|                   | ORI      | M              | I   | Logic OR of memory and immediate data                                   | $M \leftarrow (M) \vee I$   | 0 1 0 0 0 1    | D <sub>H</sub> | D <sub>L</sub>  | I               |
|                   | EXL      | r              | M   | Exclusive OR Logic of memory and register                               | $r \leftarrow (r) \oplus (M)$   | 1 0 0 1 0 1    | D <sub>H</sub> | D <sub>L</sub>  | R <sub>n</sub>  |
| Transfer          | LD       | r              | M   | Load memory to register   | $r \leftarrow (M)$  | 1 0 0 1 1 0    | D <sub>H</sub> | D <sub>L</sub>  | R <sub>n</sub>  |
|                   | ST       | M              | r   | Store register to memory  | $M \leftarrow (r)$  | 1 1 0 0 0 0    | D <sub>H</sub> | D <sub>L</sub>  | R <sub>n</sub>  |
|                   | MVRD     | r              | M   | Move memory to destination memory referring to register in the same row | $(D_H, R_n) \leftarrow (M)$   | 1 1 0 0 1 1    | D <sub>H</sub> | D <sub>L</sub>  | R <sub>n</sub>  |
|                   | MVRS     | M              | r   | Move source memory referring to register to memory in the same row      | $M \leftarrow (D_H, R_n)$   | 1 1 0 0 0 1    | D <sub>H</sub> | D <sub>L</sub>  | R <sub>n</sub>  |
|                   | MVSR     | M <sub>1</sub> | M <sub>2</sub>  | Move memory to memory in the same row                                   | $(D_H, D_{L1}) \leftarrow (D_H, D_{L2})$  | 1 0 0 1 0 0    | D <sub>H</sub> | D <sub>L1</sub> | D <sub>L2</sub> |
|                   | MVI      | M              | I   | Move immediate data to memory   | $M \leftarrow I$  | 0 1 0 0 1 0    | D <sub>H</sub> | D <sub>L</sub>  | I               |
| PLL               | M        | r              | Load N0-N3, N <sub>r</sub> & memory to PLL registers        | $PLLr \leftarrow (N0-N3, N_r \& (M))$                                   | 1 0 1 0 1 1   | D <sub>H</sub> | D <sub>L</sub> | R <sub>n</sub>  |                 |
| Bit test          | TMT      | M              | N   | Test memory bits, then skip if all bits specified are true              | if $M(N) = \text{all "1"}$ , then skip  | 0 0 1 0 1 1    | D <sub>H</sub> | D <sub>L</sub>  | N               |
|                   | TMF      | M              | N   | Test memory bits, then skip if all bits specified are false             | if $M(N) = \text{all "0"}$ , then skip  | 0 0 1 0 1 0    | D <sub>H</sub> | D <sub>L</sub>  | N               |
| Jump              | JMP      | ADDR           | Jump to the address specified in page 0                     |   | $PC \leftarrow ADDR, PAGE \leftarrow 0$   | 0 0 0 1 1 0    | ADDR: 10 bits  |                 |                 |
|                   |          |                | Jump to the address specified in page 1                     |   | $PC \leftarrow ADDR, PAGE \leftarrow 1$   | 0 0 0 0 1 0    |                |                 |                 |
| Subroutine        | CAL      | ADDR           | Call subroutine in page 0                                   |   | $Stack \leftarrow (PC+1, PAGE), PC \leftarrow ADDR, PAGE \leftarrow 0$            | 0 0 0 1 1 1    | ADDR (10 bits) |                 |                 |
|                   | RT       |                | Return to main routine                                      |   | $PC \leftarrow (stack)$   | 0 0 0 1 0 0    | -              | -               | -               |
|                   | RTS      |                | Return to main routine, then skip unconditionary            |   | $PC \leftarrow (stack), \text{ and skip}$   | 0 0 0 1 0 1    | -              | -               | -               |
| Interrupt         | EI       |                | Enable interrupt  |   | $INTE FF \leftarrow 1$  | 0 0 0 0 0 1    | -              | 0 0 0 1         | -               |
|                   | DI       |                | Disable interrupt   |   | $INTE FF \leftarrow 0$  | 0 0 0 0 1 1    | -              | 0 0 0 1         | -               |
| F/F test          | TTM      |                | Test and reset timer F F, then skip if it has not been set  |   | if $Timer FF = 1$ , then $Timer FF \leftarrow 0$<br>if $Timer FF = 0$ , then skip | 1 0 1 0 0 1    | -              | -               | -               |
|                   | TUL      |                | Test and reset unlock F F, then skip if it has not been set |   | if $UL FF = 1$ , then $UL FF \leftarrow 0$<br>if $UL FF = 0$ , then skip          | 1 0 1 0 1 0    | -              | -               | -               |
| Test timer        | TIP      |                | Test interval pulse, then skip if low                       |   | if $IPG = 0$ , then skip  | 1 0 1 0 0 1    | -              | 0 0 0 0         | 0 0 0 0         |

|                               | Mnemonic | Operand        |     | Function  | Operation   | Machine code   |                |                  |                |
|-------------------------------|----------|----------------|-----|---|---|----------------|----------------|------------------|----------------|
|                               |          | 1ST            | 2ND |   |   | Operation code |                |                  |                |
| Status word and terminal test | SS       | N <sub>1</sub> |     | Set status word 1   | (STATUS WORD 1) <sub>N</sub> ← 1  | 0 0 0 0 0 1    | —              | 0 N <sub>1</sub> | —              |
|                               | RS       | N <sub>1</sub> |     | Reset status word 1   | (STATUS WORD 1) <sub>N</sub> ← 0  | 0 0 0 0 1 1    | —              | 0 N <sub>1</sub> | —              |
|                               | TST      | N <sub>2</sub> |     | Test status word 2 true   | if (STATUS WORD 2) <sub>N</sub> = all "1", then skip                    | 0 0 1 0 0 1    | —              | 0 N <sub>2</sub> | —              |
|                               | TSF      | N <sub>2</sub> |     | Test status word 2 false  | if (STATUS WORD 2) <sub>N</sub> = all "0", then skip                    | 0 0 1 0 0 0    | —              | 0 N <sub>2</sub> | —              |
|                               | STC      |                |     | Set carry F/F   | carry F/F ← 1   | 0 0 0 0 0 1    | —              | 0 0 1 0          | —              |
|                               | RSC      |                |     | Reset carry F/F   | carry F/F ← 0   | 0 0 0 0 1 1    | —              | 0 0 1 0          | —              |
|                               | BANK0    |                |     | Select BANK0  | BANK F/F ← 0  | 0 0 0 0 1 1    | —              | 1 1 0 0          | —              |
|                               | BANK1    |                |     | Select BANK1  | BANK F/F ← 1  | 0 0 0 0 0 1    | —              | 0 1 0 0          | —              |
|                               | TITT     |                |     | Test INT, skip if true  | if $\overline{\text{INT}} = 0$ , then skip                              | 0 0 1 0 0 1    | —              | 0 0 0 1          | —              |
|                               | TITF     |                |     | Test INT, skip if false   | if $\overline{\text{INT}} = 1$ , then skip                              | 0 0 1 0 0 0    | —              | 0 0 0 1          | —              |
|                               | TCET     |                |     | Test CE, skip if true   | if CE = 1, then skip  | 0 0 1 0 0 1    | —              | 0 0 1 0          | —              |
|                               | TCEF     |                |     | Test CE, skip if false  | if CE = 0, then skip  | 0 0 1 0 0 0    | —              | 0 0 1 0          | —              |
|                               | SBK0     |                |     | Skip if BANK0   | if BANK F F = 0, then skip  | 0 0 1 0 0 0    | —              | 1 1 0 0          | —              |
|                               | SBK1     |                |     | Skip if BANK1   | if BANK F F = 1, then skip  | 0 0 1 0 0 1    | —              | 0 1 0 0          | —              |
| Input / output                | SEG      | D <sub>H</sub> | r   | Output segment pattern based on the memory specified indirectly | $SG_{0-4} \leftarrow (D_H, (R_n))$ ,<br>$S_{4-x} \leftarrow  SG_{0-4} $ | 1 0 0 0 0 1    | D <sub>H</sub> | —                | R <sub>n</sub> |
|                               | DIG      | r              |     | Output contents of register to digit port                       | Digit (D <sub>3</sub> , D <sub>2</sub> ) ← (r) <sub>3,2</sub>           | 1 0 0 0 0 0    | —              | —                | R <sub>n</sub> |
|                               | IN       | r              | P   | Input data on port to register                                  | r ← (Port (P))  | 1 1 0 0 1 0    | P              | —                | R <sub>n</sub> |
|                               | OUT      | P              | r   | Output contents of register to port                             | (Port (P)) ← (r)  | 1 0 0 0 1 0    | P              | —                | R <sub>n</sub> |
|                               | SPB      | P              | N   | Set port bits   | (Port (P)) <sub>N</sub> ← 1   | 0 0 0 0 0 1    | P              | 0 0 0 0          | N              |
|                               | RPB      | P              | N   | Reset port bits   | (Port (P)) <sub>N</sub> ← 0   | 0 0 0 0 1 1    | P              | 0 0 0 0          | N              |
|                               | TPT      | P              | N   | Test port bits, then skip if all bits specified are true        | if (Port (P)) <sub>N</sub> = all 1s, then skip                          | 0 0 1 0 0 1    | P              | 0 0 0 0          | N              |
|                               | TPF      | P              | N   | Test port bits, then skip if all bits specified are false       | if (Port (P)) <sub>N</sub> = all 0s, then skip                          | 0 0 1 0 0 0    | P              | 0 0 0 0          | N              |
| Serial I/O                    | SIO      | N              |     | Serial input-output   | SMR (3.1.0) ← N (3.1.0)   | 0 0 0 0 0 0    | 0 0            | 0 0 0 1          | N              |
|                               | TSET     |                |     | Test shift end, then skip if true                               | if SCC = 8 × (2n + 1), then skip n ≥ 0                                  | 1 0 1 0 0 0    | 1 0            | 0 0 0 1          | —              |
|                               | TSEF     |                |     | Test shift end, then skip if false                              | if SCC ≠ 8 × (2n + 1), then skip n ≥ 0                                  | 1 0 1 0 0 0    | 0 0            | 0 0 0 1          | —              |
| Test A/D                      | TADT     |                |     | Test A-D comparator, then skip if true                          | if V <sub>in</sub> > V <sub>comp</sub> , then skip                      | 1 0 1 0 0 0    | 0 0            | 0 0 0 0          | —              |
|                               | TADF     |                |     | Test A-D comparator, then skip if false                         | if V <sub>in</sub> < V <sub>comp</sub> , then skip                      | 1 0 1 0 0 0    | 1 0            | 0 0 0 0          | —              |
| Others                        | CKSTP    |                |     | Clock stop by CE  | stop clock if CE = 0  | 1 0 0 0 1 1    | —              | 1 1 1 0          | 1 1 1 0        |
|                               | NOP      |                |     | No operation  |   | 0 0 0 0 0 0    | —              | —                | —              |

APPLICATION CIRCUIT



28PIN PLASTIC SHRINK DIP (400 mil)



S28C-70-400B

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    | 28.46 MAX.            | 1.121 MAX.              |
| B    | 2.67 MAX.             | 0.106 MAX.              |
| C    | 1.778 (T.P.)          | 0.070 (T.P.)            |
| D    | 0.50 <sup>+0.10</sup> | 0.020 <sup>+0.004</sup> |
| F    | 0.85 MIN.             | 0.033 MIN.              |
| G    | 3.2 <sup>+0.3</sup>   | 0.126 <sup>+0.012</sup> |
| H    | 0.51 MIN.             | 0.020 MIN.              |
| I    | 4.31 MAX.             | 0.170 MAX.              |
| J    | 5.08 MAX.             | 0.200 MAX.              |
| K    | 10.16 (T.P.)          | 0.400 (T.P.)            |
| L    | 8.6                   | 0.339                   |
| M    | 0.25 <sup>+0.08</sup> | 0.010 <sup>+0.003</sup> |
| N    | 0.17                  | 0.007                   |

NEC Corporation

INTERNATIONAL ELECTRON DEVICES DIV.  
 SUMITOMO MITA Building, 37-8,  
 Shiba Gochome, Minato-ku, Tokyo 108, Japan  
 Tel: Tokyo 456-3111  
 Telex Address: NECTOK J22686  
 Cable Address: NEC TOKYO