CMOS 4-BIT MICROCONTROLLER

TMP47C452BN
TMP47C452BF

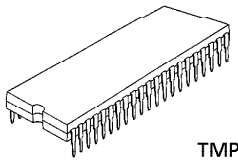
The 47C452B is a high performance 4-bit single chip microcomputer based on the TLCS-47 CMOS series. And the 47C452B has a built-in large-capacity RAM for repertory dial and DTMF generator, which is suitable for application in telephones. The 47C452B is also capable of operation with low voltage such as those supplied by telephone line.

PART No.	ROM	RAM	PACKAGE	OTP version
TMP47C452BN	4096 × 8-bit	768 × 4-bit	SDIP42-P-600-1.78	TMP47P452VN
TMP47C452BF			QFP44-P-1414-0.80D	TMP47P452VF

FEATURES

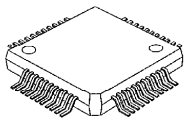
- ◆4-bit single chip microcomputer
- ◆Instruction execution time : 16.7μs (at 480kHz)
- ◆Low voltage operation : 2.2V min.
- ◆90 basic instructions
- ◆Table look-up instructions
- ◆Subroutine nesting : 15 levels max.
- ◆6 interrupt sources (External : 2, Internal : 4)
  All sources have independent latches each, and multiple interrupt control is available
- ◆I/O port (35 pins)
  - Input 2ports 5pins
  - I/O 7ports 27pins
  - Output 1port 3pins
- ◆Interval Timer (22 stages)
- ◆Two 12-bit Timer/Counters
  Timer, event counter, and pulse width measurement mode
- ◆Serial Interface with 4-bit buffer
  External/internal clock, leading/trailing edge shift mode
- ◆DTMF (Dual Tone Multi Frequency) output
  - DTMF output with one instruction
  - Single tone output function
- ◆RAM for repertory dial : 768 × 4-bit max.
- ◆BEEP output function
- ◆Hold function
  - Battery/Capacitor back-up
  - Hold function controlled by port K0
- ◆Real Time Emulator : BM47215B

SDIP42-P-600-1.78



TMP47C452BN
TMP47P452VN

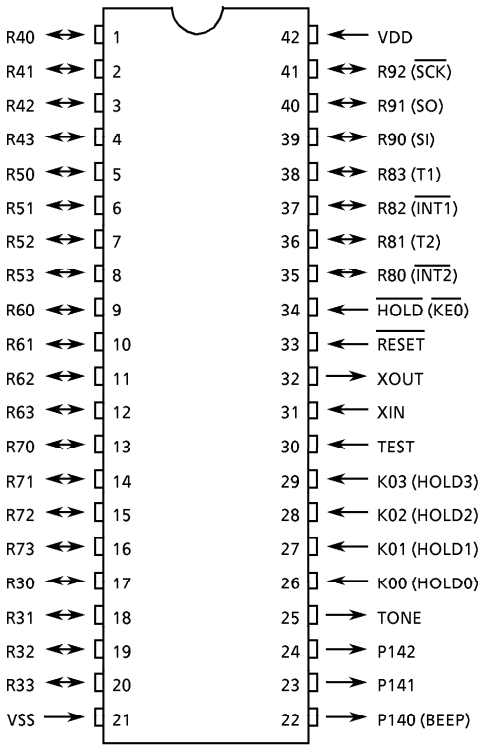
QFP44-1414-0.80D



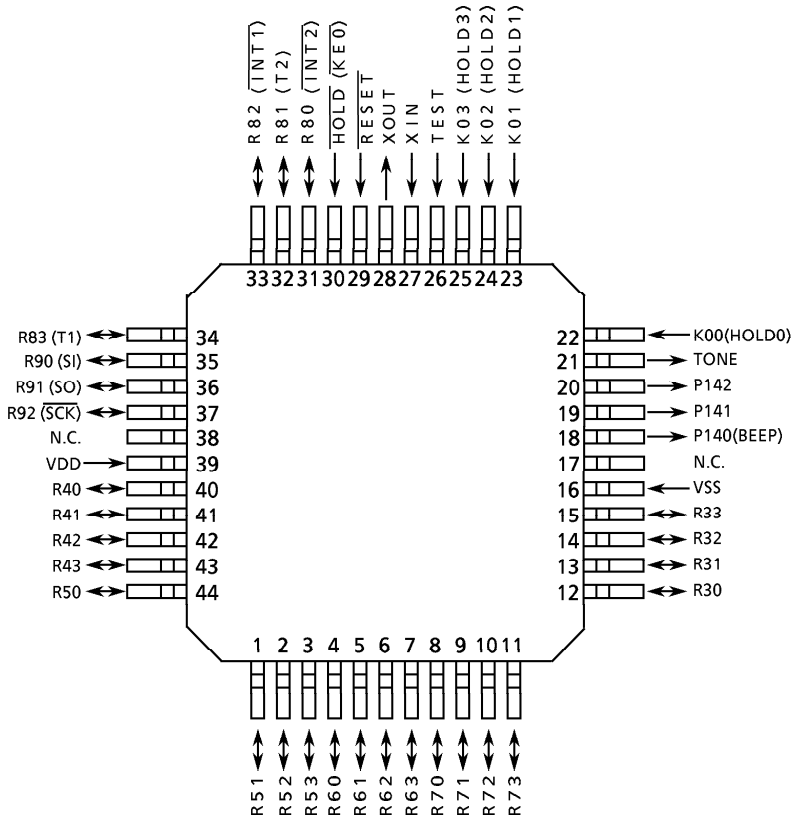
TMP47C452BF
TMP47P452VF

PIN ASSIGNMENTS (TOP VIEW)

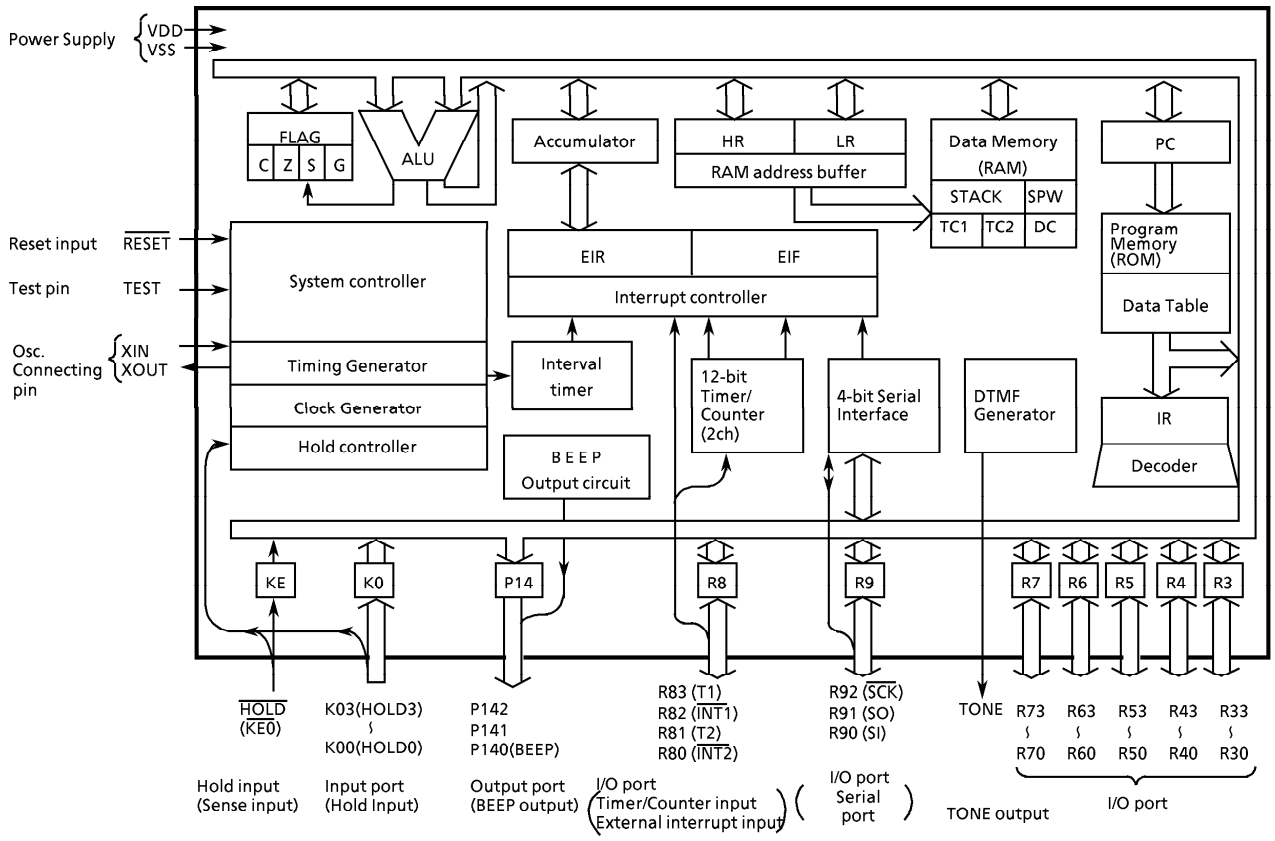
(1) SDIP42-P-600-1.78



(2) QFP44-P-1414-0.80D



BLOCK DIAGRAM



PIN FUNCTION

PIN NAME	Input/Output	FUNCTIONS	
K03 (HOLD3) - K00 (HOLD0)	Input (Input)	4-bit input port	Hold request/release signal input (Active "H")
R33 - R30	I/O	4-bit I/O port with latch. When used as input port, the latch must be set to "1".	
R43 - R40			
R53 - R50			
R63 - R60			
R73 - R70			
R83 (T1)	I/O (Input)	4-bit I/O port with latch. When used as input port, external interrupt input pin, or timer/counter external input pin, the latch must be set to "1".	Timer/Counter 1 external input
R82 ($\overline{INT1}$)			External interrupt 1 input
R81 (T2)			Timer/Counter 2 external input
R80 ($\overline{INT2}$)			External interrupt 2 input
R92 ($\overline{SCK}$)	I/O (I/O)	3-bit I/O port with latch. When used as input port or serial port, the latch must be set to "1".	Serial clock I/O
R91 (SO)	I/O (Output)		Serial data output
R90 (SI)	I/O (Input)		Serial data input
P142 - P141	Output	3-bit output port with latch	
P140(BEEP)	Output (Output)		BEEP Output
TONE	Output	Tone output	
XIN	Input	Resonator connecting pins.	
XOUT	Output		
$\overline{RESET}$	Input	Reset signal input	
$\overline{HOLD}$ ($\overline{KE0}$)	Input	Hold request/release signal input	Sense input
TEST	Input	Test pin for out-going test. Be opened or fixed to low level.	
VDD	Power supply	+ 2.2V to 6.0V	
VSS		0V(GND)	

OPERATIONAL DESCRIPTION

1. SYSTEM CONFIGURATION

◆INTERNAL CPU FUNCTION

- 2.1 Program Counter (PC)
- 2.2 Program Memory (ROM)
- 2.3 H Register, L Register
- 2.4 Data Memory (RAM)
  - a. Stack
  - b. Stack Pointer Word (SPW)
  - c. Data Counter (DC)
- 2.5 ALU, Accumulator
- 2.6 Flags
- 2.7 Clock Generator, Timing Generator
- 2.8 Interrupt Controller
- 2.9 Reset Circuit

◆PERIPHERAL HARDWARE FUNCTION

- 3.1 I/O Ports
- 3.2 Interval Timer
- 3.3 Timer/Counters (TC1, TC2)
- 3.4 DTMF Generator
- 3.5 BEEP Output Circuit
- 3.6 Serial Interface

Concerning the above component parts, the configuration and functions of hardwares are described.

2. INTERNAL CPU FUNCTION

2.1 Program Counter (PC)

The program counter is a 12-bit binary counter which indicates the address of the program memory storing the next instruction to be executed. Normally, the PC is incremented by the number of bytes of the instruction every time it is fetched. When a branch instruction or a subroutine instruction has been executed or an interrupt has been accepted, the specified values listed in Table 2-1 are set to the PC. The PC is initialized to "0" during reset.

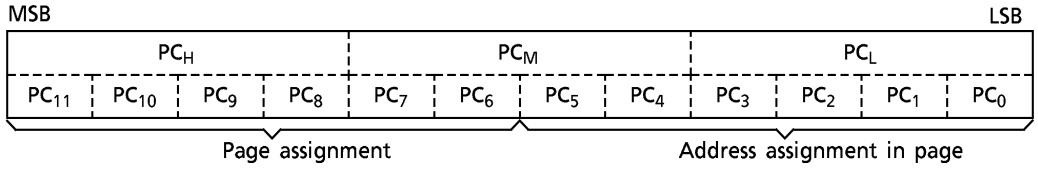


Figure 2-1. Configuration of Program Counter

The PC can directly address a 4096-byte address space. However, with the short branch and subroutine call instructions, the following points must be considered :

(1) Short branch instruction [BSS a]

In [BSS a] instruction execution, when the branch condition is satisfied, the value specified in the instruction is set to the lower 6 bits of the PC. That is, [BSS a] becomes the in-page branch instruction. When [BSS a] is stored at the last address of the page, the upper 6 bits of the PC point the next page, so that branch is made to the next page.

(2) Subroutine call instruction [CALL a]

In [CALL a] instruction execution, the contents of the PC are saved to the stack then the value specified in the instruction is set to the PC. The address which can be specified by the instruction consists of 11 bits and the most significant bit of the PC is always "0". Therefore, the entry address of the subroutine should be within an addrress range of 000_H through $7FF_H$.

Instruction or Operation		Condition		Program Counter (PC)											
				PC_{11}	PC_{10}	PC_9	PC_8	PC_7	PC_6	PC_5	PC_4	PC3	PC_2	PC_1	PC_0
Execution of Instruction	BS a	SF = 1 (Branch condition is satisfied)		Immediate data specified by the instruction											
		SF = 0 (Branch condition is not satisfied)		+ 2											
	BSS a	SF = 1	Lower 6-bit address ≠ 111111	Hold						Immediate data specified by the instruction					
			Lower 6-bit address = 111111 (last address in page)	+ 1						Immediate data specified by the instruction					
		SF = 0		+ 1											
	CALL a			0	Immediate data specified by the instruction										
	CALLS a			0	0	0	0	The data generated by the immediate data specified by the instruction					1	1	0
	RET			The return address restored from stack											
	RETI			The return address restored from stack											
	Others			Incremented by the number of bytes in the instruction											
Interrupt acceptance				0	0	0	0	0	0	0	0	Interrupt vector			0
Reset				0	0	0	0	0	0	0	0	0	0	0	0

Table 2-1. Status Change of Program Counter

2.2 Program Memory (ROM)

Programs and fixed data are stored in the program memory. The instruction to be executed next is read from the address indicated by the contents of the PC.
The fixed data can be read by using the table look-up instructions or 5-bit to 8-bit data conversion instruction.

(1) Table look-up instructions

[LDL A, @DC], [LDH A, @DC+]

The table look-up instructions read the lower and upper 4 bits of the fixed data stored at the address specified in the data counter (DC) to place them into the accumulator. [LDL A, @DC] instruction reads the lower 4 bits of fixed data, and [LDH A, @DC+] instruction reads the upper 4 bits.
The DC is a 12-bit register, allowing it to address the entire program memory space.

Example: When [LDL A, @DC] instruction is executed with the DC value being $7A0_H$ and the contents of program memory address $7A0_H$ being 58_H, "8" is stored in the accumulator; when [LDH A, @DC+] instruction is executed, "5" is stored in the accumulator and the DC value is incremented to $7A1_H$.

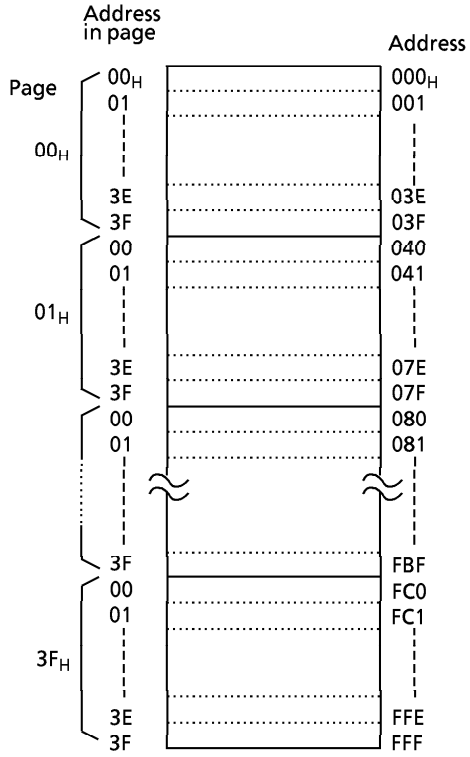


Figure 2-2. Configuration of Program Memory

(2) 5-bit to 8-bit data conversion instruction [OUTB @HL]

The 5-bit to 8-bit data conversion instruction reads the fixed data (8 bits) from the data conversion table in the program memory to output the upper 4 bits to COLUMN register and the lower 4 bits to ROW register. The table is located in the last 32-byte space (addresses, $FE0_H$ through FFF_H for the 47C452B) in the program memory with the lower address consisting of the 5 bits obtained by concatenating the contents of the data memory specified by the HL register pair and the content of the carry flag. This instruction is usable for such applications as setting generator data of DTMF generator.

2.2.1 Program Memory Capacity

The 47C452B has 4096 × 8 bits (addresses 000_H through FFF_H) of program memory (mask ROM).

2.2.2 Program Memory Map

Figure 2-3 shows the program memory map. Address 000_H-086_H and $FE0_H$-FFF_H of the program memory are also used for special purposes.

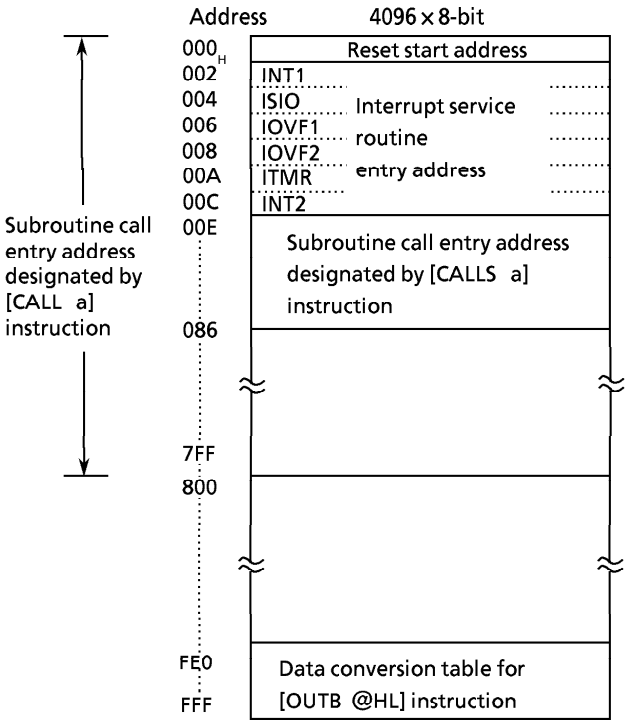


Figure 2-3. Program Memory Map

2.3 H Register and L Register

The H register and the L register are 4-bit general registers. They are also used as a register pair (HL) for the data memory (RAM) addressing pointer. The RAM consists of pages, each page being 16 words long (1word = 4bits). The H register specifies a page and the L register specifies an address in the page.

The L register has the auto-post-increment/decrement capability, implementing the execution of composite instructions. For example, [ST A, @HL+] instruction automatically increments the contents of the L register after data transfer.

During the execution of [SET @L], [CLR @L], or [TEST @L] instructions, the L register is also used to specify the bits corresponding to I/O port pins R73 through R40 (the indirect addressing of port bits by the L register).

Example: To write immediate values "5" and "F_H" to data memory addresses 10_H and 11_H.

```
LD      HL,#10H          ;  HL←10H
ST      #5,@HL+          ;  RAM [10H] ←5H, LR←LR + 1
ST      #0FH,@HL+        ;  RAM [11H] ←FH, LR←LR + 1
```

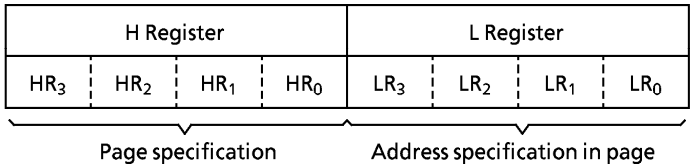


Figure 2-4. Configuration of H and L Registers

2.4 Data Memory (RAM : 768 × 4bit)

The 47C452B data memory consists of a 768 × 4-bit RAM. And 512 × 4bit RAM in that area is mainly used for storing repertory dialing data and is controlled by the RAM address register, RAM data buffer register and TONE/RAM command register.

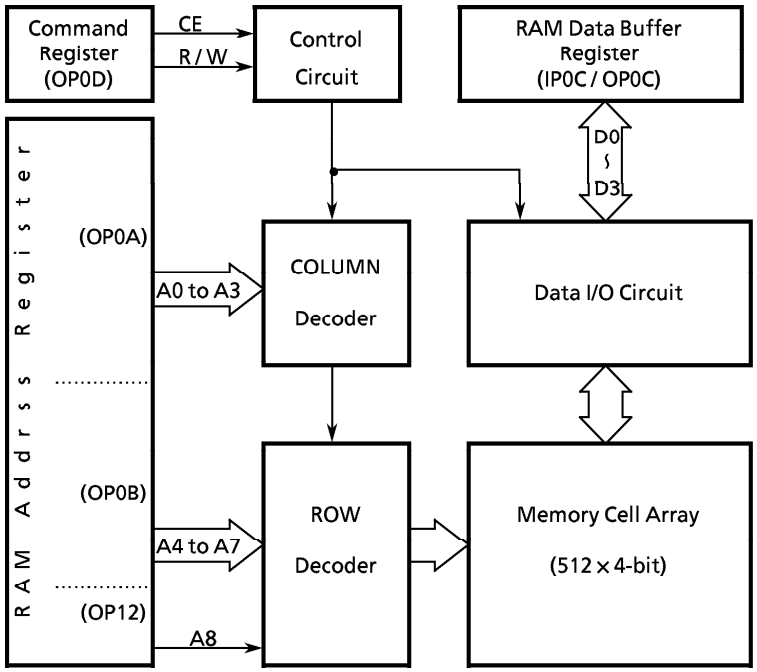


Figure 2-5. Block Diagram

2.4.1 Data Memory (RAM : 256 × 4bit)

The 47C452B has 256 × 4 bits (addresses 00_H through FF_H) of the data memory (RAM).

The RAM is addressed in one of the three ways (addressing modes):

(1) Register-indirect addressing mode

In this mode, a page is specified by the H register and an address in the page by the L register.

Example: LD A, @HL ; Acc←RAM [HL]

(2) Direct addressing mode

In this mode, an address is directly specified by the 8 bits of the second byte (operand) in the instruction field.

Example: LD A, 2CH ; Acc←RAM [$2C_H$]

(3) Zero-page addressing mode

In this mode, an address in zero-page (addresses 00_H through $0F_H$) is specified by the lower 4 bits of the second byte (operand) in the instruction field.

Example: ST #3, 05H ; RAM [05_H] ←3

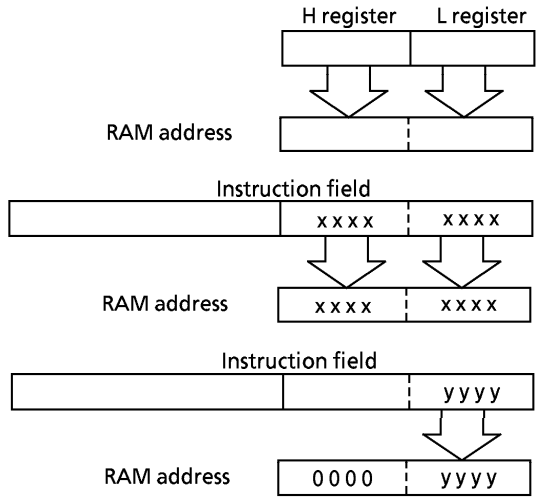


Figure 2-6. Addressing mode

2.4.2 RAM (512 × 4bit) Control Register

(1) RAM (512 × 4bit) Address Register

The RAM address register is a 9-bit register to specify addresses for the RAM data memory. The upper 1 bit is accessed with port address OP12, the next 4 bits are accessed with the port address OP0B/IP0B and the lower 4 bits are accessed with port address OP0A/IP0A.

These registers are initialized to "0" during reset.

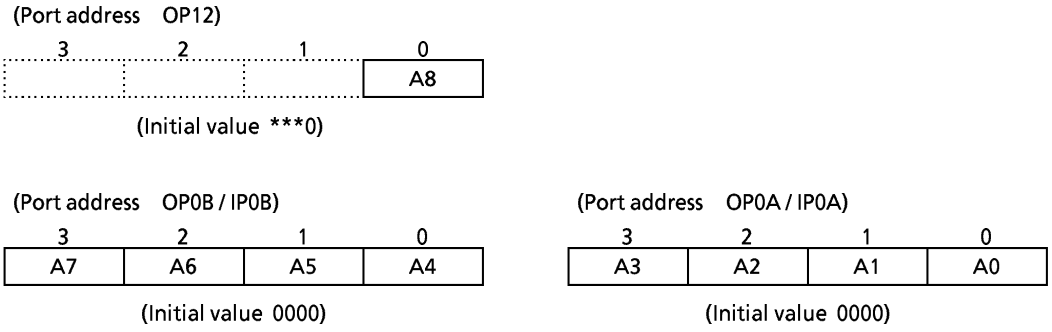


Figure 2-7. RAM Address Register

(2) RAM (512 × 4bit) Data Buffer Register

The RAM data buffer register is a 4-bit buffer register to read or write RAM data. When writing data to RAM, it is accessed as port address OP0C. Port address IP0C is used for access when reading data from RAM.

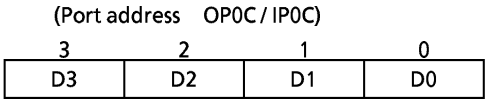
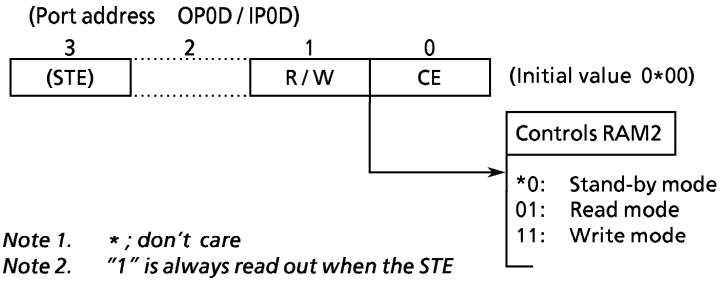


Figure 2-8. Data Buffer Register

(3) RAM (512 × 4bit) Command Register

The RAM command register (OP0D/IP0D) controls the reading or writing data, and whether RAM. is to be accessed or put in stand-by mode. This register is accessed as the port address OP0D/IP0D. The RAM command register is also used as the TONE command register.



Note 1. ** ; don't care*
Note 2. *"1" is always read out when the STE bit is read.*

Figure 2-9. Command Register

2.4.3 Access for RAM (512 × 4bit)

To write data to RAM, lead the address into the RAM address register and the data into the RAM data buffer register (OP0C) , then put the RAM command register in the write mode. The data will be written to the specified RAM address by this operation.
The data are latched in the RAM data buffer register, therefore, RAM data buffer register operation is not necessary when the same data are written continuously.
To read data from RAM, set the RAM command register to the read mode and load the address into the RAM address register, then read the data via RAM data buffer register (IP0C) . Data are not latched in the RAM data buffer register.
After writing/reading data to/from RAM, make sure to set the RAM command register back to the standby mode. It is necessary to set to the standby mode before executing in hold operation mode.

Example 1 : To write data "9" to address 182_H and data "7" to address $15A_H$ in RAM.

```
        LD      A,#1                ; Sets data "182H" to RAM address register.
        OUT     A,%OP12
        OUT     #8,%OP0B
        OUT     #2,%OP0A
        OUT     #9,%OP0C            ; Writes data "9" to RAM data buffer register.
        OUT     #0011B,%OP0D        ; Sets RAM to write mode.
        OUT     #0010B,%OP0D        ; Sets RAM to stand-by mode.
        OUT     #5,%OP0B            ; Sets data "15AH" to RAM address register.
        OUT     #0AH,%OP0A
        OUT     #7,%OP0C            ; Writes data "7" to RAM data buffer register.
        OUT     #0011B,%OP0D        ; Sets RAM to write mode.
        OUT     #0010B,%OP0D        ; Sets RAM to stand-by mode.
```

Example 2 : To write data "0" to address 120_H through 127_H in RAM.

```
        OUT     #0,%OP0C            ; Writes data "0" to RAM data buffer register.
        LD      A,#0                ; Sets data "120H" to RAM address register.
        OUT     #1,%OP12
        OUT     #2,%OP0B
        OUT     A,%OP0A
        OUT     #0011B,%OP0D        ; Sets RAM to write mode.
SLOOP : CMPR    A,#7                ; Increases address register.
        TESTP   ZF
        B       SWEND
        INC     A
        OUT     A,%OP0A
        BR      SLOOP
SWEND : OUT     #0010B,%OP0D        ; Sets RAM to stand-by mode.
```

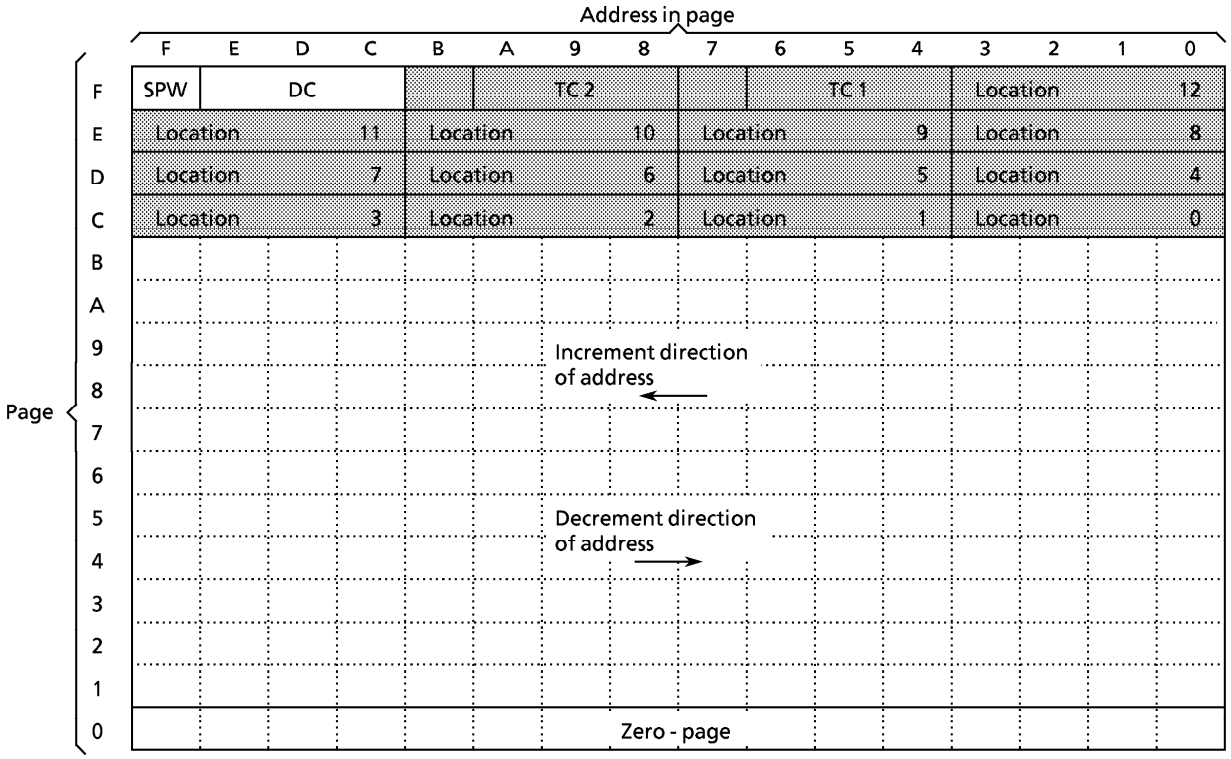
Example 3 : To read data from address $0B1_H$ in RAM and store to Accumulator.

```
        OUT     #0001B,%OP0D        ; Sets RAM to read mode.
        LD      A,#0                ; Sets data "0B1H"to RAM address register
        OUT     A,%OP12
        OUT     #0BH,%OP0B
        OUT     #1,%OP0A
        IN      %IP0C,A             ; Reads data from RAM and stores to
                                      Accumulator.
        OUT     #0000B,%OP0D        ; Sets RAM to stand-by mode
```

2.4.4 Data Memory Map

Figure 2-7 shows the data memory map. The data memory is also used for the following special purpose.

① Stack
② Stack Pointer Word (SPW)
③ Data Counter (DC)
④ Count registers of the timer/counters (TC1, TC2)
⑤ Zero-page



Note1. ▨ *denotes the stack area.*
Note2. The TC1 and TC2 areas are shared by the locations 13 and 14.

Figure 2-10. Data Memory Map

(1) Stack

The stack provides the area in which the return address is saved before a jump is performed to the processing routine at the execution of a subroutine call instruction or the acceptance of an interrupt. When a subroutine call instruction is executed, the contents (the return address) of the program counter are saved; when an interrupt is accepted, the contents of the program counter and flags are saved.

When returning from the processing routine, executing the subroutine return instruction [RET] restores the contents of the program counter from the stack; executing the interrupt return instruction [RETI] restores the contents of the program counter and flags.

The stack consists of up to 15 levels (locations 0 through 14) which are provided in the data memory (addresses $C0_H$ through FB_H). Each location consists of 4-word data memory. Locations 13 and 14 are shared with the count registers of the timer/counters (TC1, TC2) to be described later.

The save/restore locations in the stack are determined by the stack pointer word (SPW). The SPW is automatically decremented after save, and incremented before restore. That is, the value of the SPW indicates the stack location number for the next save.

(2) Stack Pointer Word (SPW)

Address FF_H in the data memory is called the stack pointer word, which identifies the location in the stack to be accessed (save or restore).

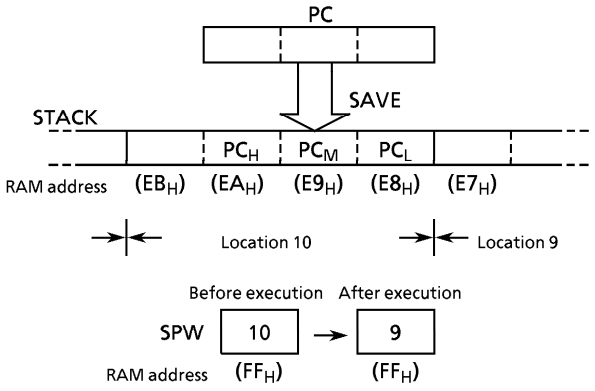
Generally, location number 0 to 12 can be set to the SPW, providing up to 13 levels of stack nesting. Locations 13 and 14 are shared with the timer/counters to be described later; therefore, when the timer/counters are not used, the stack area of up to 15 levels is available. Address FF_H is assigned to the SPW, so that the contents of the SPW cannot be set "15" in any case.

The SPW is automatically updated when a subroutine call is executed or an interrupt is accepted. However, if it is used in excess of the stack area permitted by the data memory allocating configuration, the user-processed data may be lost. (For example, when the user-processed data area is in an address range 00_H through CF_H, up to location 4 of the stacks are usable. If an interrupt is accepted with location 4 already used, the user-processed data stored in addresses CC_H through CF_H corresponding to the location 3 area is lost.)

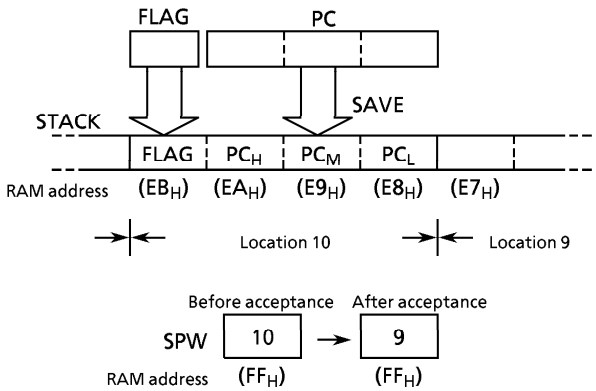
The SPW is not initialized by hardware, requiring to write the initial value (the location with which the use of the stack starts) by using the initialization routine. Normally, the initial value of "12" is used.

Example: To initialize the SPW (when the stack is used from location 12)

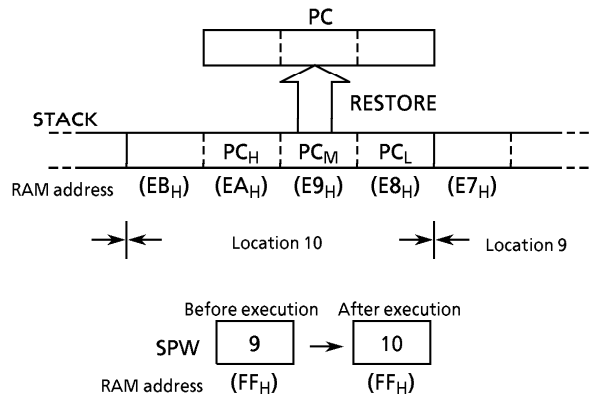
```
LD      A,#12     ;  SPW←12
ST      A,0FFH
```



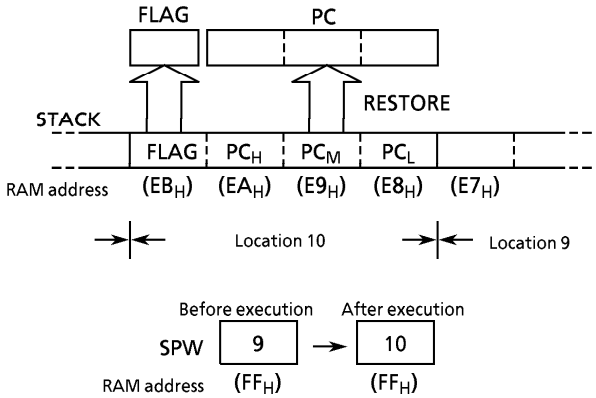
(a) At execution of the subroutine call instruction



(b) At acceptance of an interrupt



(c) At execution of the subroutine return instruction



(d) At execution of the interrupt return instruction

Figure 2-11. Accessing Stack (Save/Restore)

(3) Data Counter (DC)

The data counter is a 12-bit register to specify the address of the data table to be referenced in the program memory (ROM). Data table reference is performed by the table look-up instructions [LDL A, @DC] and [LDH A, @DC+]. The data table may be located anywhere within the program memory address space.

The DC is assigned with a RAM address in unit of 4 bits. Therefore, the RAM manipulation instruction is used to set the initial value or read the contents of the DC.

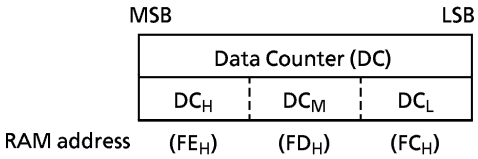


Figure 2-12. Data Counter

Example: To set the DC to 780_H.

```
LD      HL,#0FCH    ;  Sets RAM address of DCL to HL register pair.
ST      #0H,@HL+    ;  DC ← 780H
ST      #8H,@HL+
ST      #7H,@HL+
```

(4) Count registers of the timer/counters (TC1, TC2)

The 47C452B has two channels of 12-bit timer/counters. The count register of the timer/counter is assigned with a RAM addresses in unit of 4 bits, so that the initial value is set and the contents are read by using the RAM manipulation instruction.

The count registers are shared with the stack area (locations 13 and 14) described earlier, so that the stack is usable from location 13 when the timer/counter 1 is not used. When none of timer/counter 1 and timer/counter 2 are used, the stack is usable from location 14.

When both timer/counter 1 and timer/counter 2 are used, the data memory locations at addresses $F7_H$ and FB_H can be used to store the user-processed data.

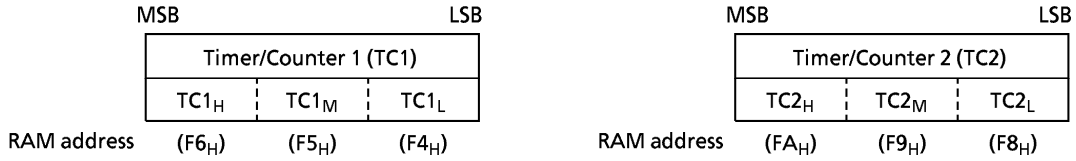


Figure 2-13. Count Registers of the Timer/Counters (TC1, TC2)

(5) Zero-page

The 16 words (at addresses 00_H through $0F_H$) of the zero page of the data memory can be used as the user flags or pointers by using zero-page addressing mode instructions (comparison, addition, transfer, and bit manipulation), providing enhanced efficiency in programming.

Example: To write immediate data "8" to address 09_H if bit 2 at address 04_H in the RAM is "1".

```
        TEST    04H,2       ;  Skips if bit 2 at address 04H in the RAM is "0".
        B       SKIP
        ST      #8,09H      ;  Writes "8" to address 09H in the RAM
SKIP:
```

2.4.5 Data Memory Capacity

The 47C452B have 512 × 4 bit of data memory (RAM) , 256 × 4 bits (addresses 00_H through FF_H) of data memory (The 47C452B have the total 768 × 4bit of data memory). When power-on is performed, the contents of the RAM become unpredictable, so that they must be initialized by the initialization routine.

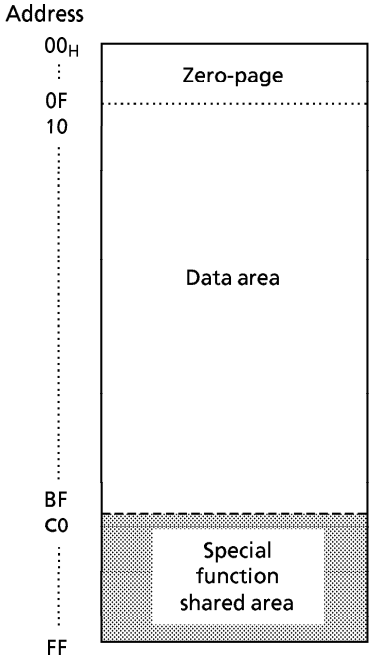


Figure 2-14. Data Memory Capacity and Address Assignment

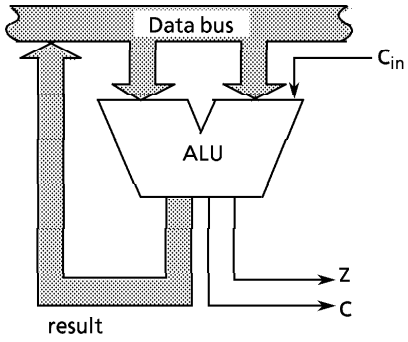
Example: To clear RAM (use common to the 256 × 4bit RAM)

```
        LD      HL, #00H        ; HL←00H
SCLRRAM: ST     #0, @HL+        ; RAM [HL] ←0, LR←LR + 1
        B       SCLRRAM
        ADD     H, #1           ; HR←HR + 1
        B       SCLRRAM
```

2.5 ALU and Accumulator

2.5.1 Arithmetic / Logic Unit (ALU)

The ALU performs the arithmetic and logic operations specified by instructions on 4-bit binary data and outputs the result of the operation, the carry information (C), and the zero detect information (Z).



Note. C_{in} *indicates the carry input specified by instruction*

Figure 2-15. ALU

(1) Carry information (C)

The carry information indicates a carry-out from the most significant bit in an addition. A subtraction is performed as addition of two's complement, so that, with a subtraction, the carry information indicates that there is no borrow to the most significant bit. With a rotate instruction, the information indicates the data to be shifted out from the accumulator.

(2) Zero detect information (Z)

This information is "1" when the operation result or the data to be transferred to the accumulator/data memory is "0000_B".

Example: The carry information (C) and zero detect information (Z) for 4-bit additions and subtractions.

Operation	Result	C	Z
4 + 2 =	6	0	0
7 + 9 =	0	1	1
8 − 1 =	7	1	0
2 − 2 =	0	1	1
5 − 8 =	−3 (1101_B)	0	0

2.5.2 Accumulator (Acc)

The accumulator is a 4-bit register used to hold source data or results of the operations and data manipulations.

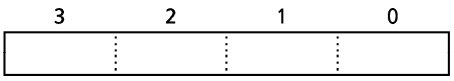


Figure 2-16. Accumulator

2.6 Flags

There are a carry flag (CF), a zero flag (ZF), a status flag (SF), and a general flag (GF), each consisting of 1 bit. These flags are set or cleared according to the condition specified by an instruction. When an interrupt is accepted, the flags are saved on the stack along with the program counter. When the [RETI] instruction is executed, the flags are restored from the stack to the states set before interrupt acceptance.

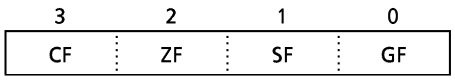


Figure 2-17. Flags

(1) Carry flag (CF)

The carry flag holds the carry information received from the ALU at the execution of an addition/subtraction with carry instruction, a compare instruction, or a rotate instruction. With a carry flag test instruction, the CF holds the value specified by it.

① Addition/subtraction with carry instructions [ADDC A, @HL], [SUBRC A, @HL]
The CF becomes the input (C_{in}) to the ALU to hold the carry information.

② Compare instructions [CMPR A, @HL], [CMPR A, #k]
The CF holds the carry information (non-borrow).

③ Rotate instructions [ROLC A], [RORC A]
The CF is shifted into the accumulator to hold the carry information (the data shifted out from the accumulator).

④ Carry flag test instructions [TESTP CF], [TEST CF]
With [TESTP CF] instruction, the content of the CF is transferred to the SF then the CF is set to "1".
With [TEST CF] instruction, the value obtained by inverting the content of the CF is transferred to the SF then the CF is cleared to "0".

(2) Zero flag (ZF)

The zero flag holds the zero detect information (Z) received from the ALU at the execution of an operational instruction, a rotate instruction, an input instrcution, or a transfer-to-accumulator instruction.

(3) Status flag (SF)

The status flag provides the branch condition for a branch instruction. Branch is performed when this flag is set to "1". Normally the SF is set to "1", so that any branch instruction can be regarded as an unconditional branch instruction. When a branch instruction is executed upon set or clear of the SF according to the condition specified by an instruction, this instruction becomes a conditional branch instruction. During reset, the SF is initialized to "1", other flags are not affected.

(4) General flag (GF)

This is a 1-bit general-purpose flag which can be set, cleared, or tested by program.

Example 1: When the following instructions are executed with the accumulator, H register, L register, data memory (address 07_H), and carry flag being set to "C_H", "0", "7", "5", and "1" respectively, the contents of the accumulator and flags become as follows:

Instruction	Acc after execution	Flag after execution		
		CF	ZF	SF
ADDC A, @HL	2_H	1	0	0
SUBRC A, @HL	9_H	0	0	0
CMPR A, @HL	C_H	0	0	1
AND A, @HL	4_H	1	0	1
LD A, @HL	5_H	1	0	1

Instruction	Acc after execution	Flag after execution		
		CF	ZF	SF
LD A, #0	0_H	1	1	1
ADD A, #4	0_H	1	1	0
DEC A	B_H	1	0	1
ROLC A	9_H	1	0	0
RORC A	E_H	0	0	1

Example 2: When the accumulator (Acc) is $0 \leqq Acc \leqq 9$, the general flag (GF) is set to "1".

```
        CLR     GF          ; GF←0
        CMPR    A, #9       ; Skip if Acc ≧ 9.
        TEST    CF
        B       SKIPC
        SET     GF          ; GF←1
SKIPC:
```

2.7 Clock Generator and Timing Generator

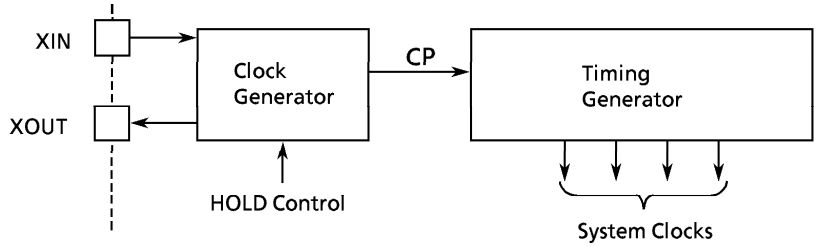


Figure 2-18. Clock Generator and Timing Generator

2.7.1 Clock Generator

The clock generator provides the basic clock pulse (CP) by which the system clock to be supplied to the CPU and the peripheral hardware is produced. The CP can be easily obtained by connecting the resonator to the XIN and XOUT pins. The clock from the external oscillator is also available. In the hold operating mode, the clock generator stops oscillating.

(a) Crystal or ceramic resonator (b) External oscillator

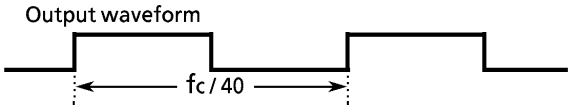
Figure 2-19. Examples of Oscillator Connection

Note: Accurate adjustment of the oscillation frequency

Although the hardware to externally and directly monitor the CP is not provided, the oscillation frequency can be adjusted by making the program to output the pulse with a fixed frequency to the port with the all interrupts disabled and timer/counters stopped and monitoring this pulse. With a system requiring the oscillation frequency adjustment, the adjusting program must be created beforehand.

Example: To output the oscillation frequency adjusting monitor pulse to port R70.

```
SFCCHK:   SET    %OP07,0
          CLR    %OP07,0
          BSS    SFCCHK
```



2.7.2 Timing Generator

The timing generator produces the system clocks from basic clock pulse which are supplied to the CPU and the peripheral hardware.

2.7.3 Instruction Cycle

The instruction execution and the on-chip peripheral hardware operations are performed in synchronization with the basic clock pulse (CP: fc [Hz]). The smallest unit of instruction execution is called an instruction cycle. The instruction set of the TLCS-47 series consists of 1-cycle instructions and 2-cycle instructions. The former requires 1 cycle for their execution; the latter, 2 cycles. Each instruction cycle consists of 4 states (S1 through S4). Each state consists of 2 basic clock pulses.

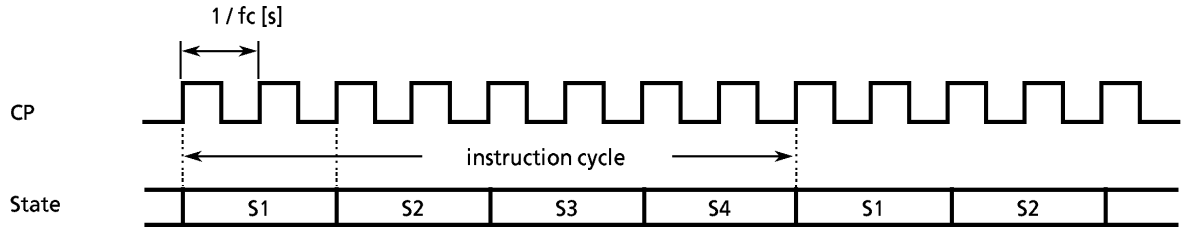


Figure 2-20. Instruction Cycle

2.7.4 Hold Operating Mode

The hold feature stops the system and holds the the system's internal states active before stop with a low power. The hold operation is controlled by the command register (OP10) and the $\overline{\text{HOLD}}$ pin input. The $\overline{\text{HOLD}}$ pin input state can be known by the status register (IP0E). The $\overline{\text{HOLD}}$ pin is shared with the $\overline{\text{KE0}}$ pin. Figure 2-21 shows the KE port, KE0 input data and hold control signal.

The KE port has been allocated to the least significant bit of the port address IP0E as shown in the figure, it is read as the logical operational result of the $\overline{\text{KE0}}$ pin input and K0 pin input.

The HOLD operational mode is released when the HOLD control signal increases from "0" to "1". Therefore, the HOLD operation can be released by either shutting off the hook switch or pressing the key by connecting the hook switch to the $\overline{\text{HOLD}}$ pin and the key switch to the K0 pin; this gives access to the on-hook dialing function.

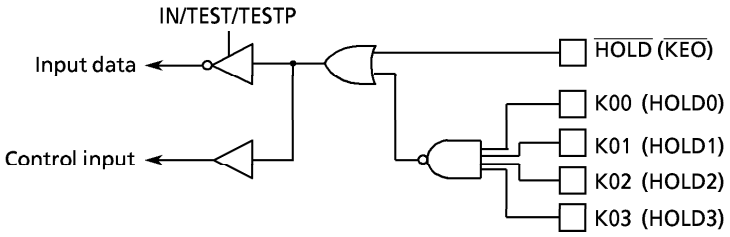


Figure 2-21. Hold control circuit

Note : P140 and P141 of TMP47C452B are used for an output port, and at the same time can be used for an input port to set a mode for tone output in test mode. When setting hold operation mode, be sure to set a pin interface as used for a normal input/output port.

(1) Starts Hold Operating Mode

The hold operation is started when the command is set to the command register and holds the following states during the hold operation:

① The oscillator stops and the system's internal operations are all held up.
② The interval timer is cleared to "0".
③ The states of the data memory, registers, and latches valid immediately before the system is put in the hold state are all held.
④ The program counter holds the address of the instruction to be executed after the instruction which starts the hold operating mode.

Hold operating mode command register (Port address OP10)

3	2	1	0	
HLDMS			HWUT	(Initial value *0**)

HLDMS	Sets mode/starts hold operation
01 :	Starts hold operation in edge-release mode
11 :	Starts hold operation in level-release mode
*0 :	Unused

HWUT	Sets the warm-up time at release of the hold operating mode	
	Example : At fc = 480kHz	
00 :	2^{18} / fc [s]	··· 546 [ms]
01 :	2^{14} / fc	··· 34.1
10 :	Unused	
11 :	2^{6} / fc	··· 0.13

Note 1. * ; *don't care*

Note 2. fc ; Basic clock frequency [Hz]

Hold operating mode status register (Port address IP0E)

3	2	1	0
(SIOF)	(SEF)		HOLD (KE0)

HOLD	$\overline{\text{HOLD}}$ pin input state
0 :	$\overline{\text{HOLD}}$ pin is high
1 :	$\overline{\text{HOLD}}$ pin is low (HOLD operation request)

Figure 2-22. Hold Operating Mode Command Register/Status Register

The hold operating mode consists of the level-sensitive release mode and the edge-sensitive release mode.

a. Level-sensitive release mode

In this mode, the hold operation is released by setting the $\overline{\text{HOLD}}$ pin to the high level. This mode is used for the capacitor backup with power off or for the battery backup for long hours.

If the instruction to start the hold operation is executed with the $\overline{\text{HOLD}}$ pin input being high, the hold operation does not start but the release sequence (warm-up) starts immediately. Therefore, to start the hold operation in the level-sensitive release mode, that the $\overline{\text{HOLD}}$ pin input being low (the hold operation request) must be recognized in program. This recognition is performed in one of the two ways below:

① Testing HOLD (bit 0 of the status register)

② Applying the $\overline{\text{HOLD}}$ pin input also to the $\overline{\text{INT1}}$ pin to generate the external interrupt 1 request.

Example: To test HOLD to start the hold operation in the level-sensitive release mode (the warm-up time = 2^{14}/fc).

```
SHOLDH: TEST    %IP0E, 0    ;  Waits until HOLD pin input goes low.
        B       SHOLDH
        LD      A, #1101B   ;  OP10←1101B
        OUT     A, %OP10
```

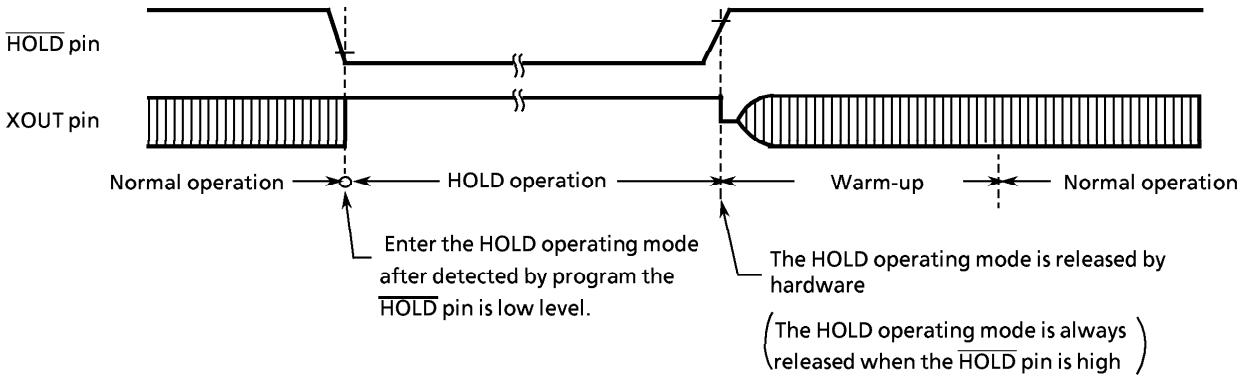


Figure 2-23. Level-sensitive release mode

b. Edge-sensitive release mode

In this mode, the hold operation is released at the rising edge of the $\overline{HOLD}$ pin input. This mode is used for applications in which a relatively short-time program processing is repeated at a certain cycle. This cyclic signal (for example, the clock supplied from the low power dissipation oscillator). In the edge-sensitive mode, even if the $\overline{HOLD}$ pin input is high, the hold operation is performed.

Example: To start the hold operation in the edge-sensitive release mode (the warm-up time = 2^{14}/fc).

```
LD      A, #0101B      ;  OP10←0101B
OUT     A, %OP10
```

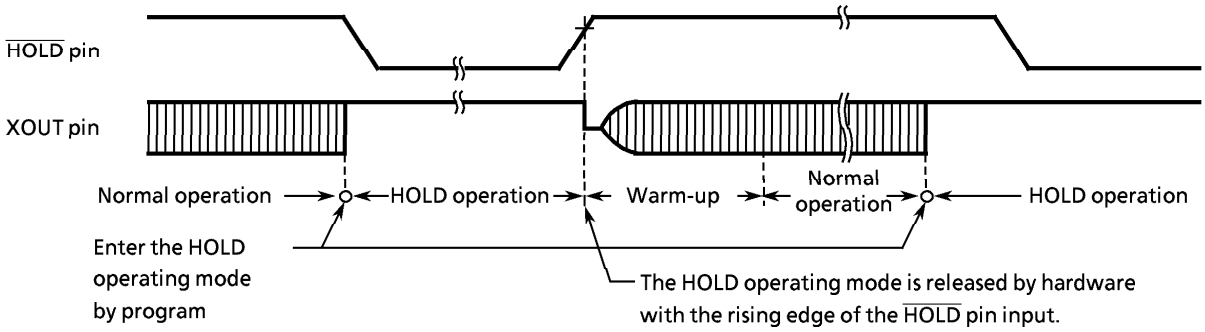


Figure 2-24. Edge-sensitive release mode

Note: In the hold operation, the dissipation of the power associated with the oscillator and the internal hardware is lowered; however, the power dissipation associated with the pin interface (depending on the external circuitry and program) is not directly determined by the hardware operation of the hold feature. This point should be considered in the system design and the interface circuit design.
In the CMOS circuitry, a current does not flow when the input level is stable at the power voltage level (V_{DD}/V_{SS}); however, when the input level gets higher than the power voltage level (by approximately 0.3 to 0.5 V), a current begins to flow. Therefore, if cutting off the output transistor at an I/O port (the open drain output pin with an input transistor connected) puts the pin signal into the high-impedance state, a current flows across the ports input transistor, requiring to fix the level by pull-up or other means.

(2) Releases Hold Operating Mode

The hold operating mode is released in the following sequence:

① The oscillator starts

② Warm-up is performed to acquire the time for stabilizing oscillation. During the warm-up , the internal operations are all stopped. One of three warm-up times can be selected by program depending on the characteristics of the oscillator used.

③ When the warm-up time has passed, an ordinary operation restarts from the instruction next to the instruction which starts the hold operation. At this time, the interval timer starts from the reset state "0".

※ *The warm-up time is obtained by dividing the basic clock by the interval timer, so that, if the frequency at clearing the HOLD operation is unstable, the warm-up time shown in Figure 2.22 includes an error. Therefore, the warm-up time must be handled as an approximate value.*

The warm-up time is obtained by dividing the basic clock by the interval timer, so that, if the frequency at releasing the hold operation is unstable, the warm-up time shown in Figure 2-21. includes an error. Therefore, the warm-up time must be handled as an approximate value. The hold operation is also released by setting the $\overline{\text{RESET}}$ pin to the low level. In this case, the normal reset operation follows immediately.

Note: To release the hold operation at a low hold voltage, the following points must be considered:
When the power voltage rises from the hold voltage to the operating voltage, the $\overline{\text{RESET}}$ *pin input is also at the high level and its voltage rises with the power voltage.*
In this case, if a time-constant circuit or the like is externally attached, the voltage rise of the $\overline{\text{RESET}}$ *pin input occurs after the power voltage rise. If the voltage level of the* $\overline{\text{RESET}}$ *pin input gets under the non-inverted high input voltage of the* $\overline{\text{RESET}}$ *pin input (the hysteresis input), a reset operation may happen.*

2.8 INTERRUPT FUNCTION

(1) Interrupt Controller

There are 6 interrupt sources (2 external and 4 internal). The prioritized multiple interrupt capability is supported. The interrupt latches (IL_5 through IL_0) to hold interrupt requests are provided for the interrupt sources. Each interrupt latch is set to "1" when an interrupt request is made, asking the CPU to accept the interrupt. The acceptance of interrupt can be permitted or prohibited by program through the interrupt enable master flip-flop (EIF) and interrupt enable register (EIR). When two or more interrupts occur simultaneously, the one with the highest priority determined by hardware is serviced first.

Interrupt Source			Priority	Interrupt Latch	Enable conditions	Entry address
External	Extenal Interrupt 1	(INT1)	(highest) 1	IL_5	EIF = 1	002_H
Internal	Serial Interface Interrupt	(ISIO)	2	IL_4	EIF = 1, EIR_3 = 1	004_H
	TC1 overflow Interrupt	(IOVF1)	3	IL_3	EIF = 1, EIR_2 = 1	006_H
	TC2 overflow Interrupt	(IOVF2)	4	IL_2	EIF = 1, EIR_1 = 1	008_H
	Interval Timer Interrupt	(ITMR)	5	IL_1		$00A_H$
External	External Interrupt 2	(INT2)	(lowest) 6	IL_0	EIF = 1, EIR_0 = 1	$00C_H$

Table 2-2. Interrupt Sources

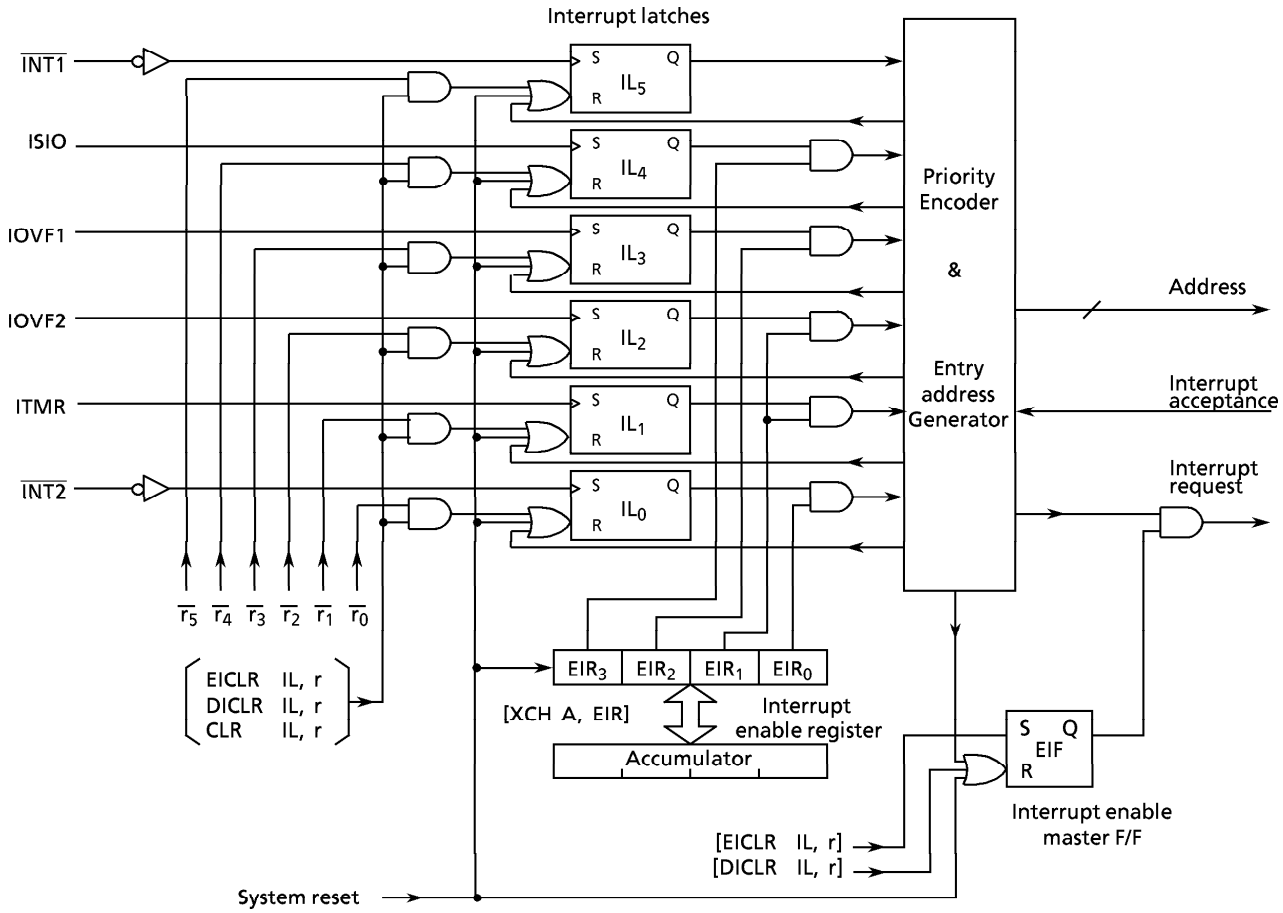


Figure 2-25. Interrupt Controller Block Diagram

a. Interrupt enable master flip-flop (EIF)

The EIF controls the enable/disable of all interrupts. When this flip-flop is cleared to "0", all interrupts are disabled; when it is set to "1", the interrupts are enabled.

When an interrupt is accepted, the EIF is cleared to "0", temporarily disabling the acceptance of subsequent interrupts. When the interrupt service program has been executed, the EIF is set to "1" by the execution of the interrupt return instruction [RETI], being put in the enabled state again.

Set or clear of the EIF in program is performed by instructions [EICLR IL, r] and [DICLR IL, r], respectively. The EIF is initialized to "0" during reset.

b. Interrupt enable register (EIR)

The EIR is a 4-bit register specifies the enable or disable of each interrupt except INT1. An interrupt is enabled when the corresponding bit of the EIR is "1", and an interrupt is disabled when the corresponding bit of the EIR is "0". Bit 1 of the EIR (EIR_1) is shared by both IOVF2 and ITMR interrupts.

Read/write on the EIR is performed by executing [XCH A, EIR] instruction. The EIR is initialized to "0" during reset.

c. Interrupt latch (IL_5 through IL_0)

An interrupt latch is provided for each interrupt source. The IL is set to "1" when an interrupt request is made to ask the CPU for accepting the interrupt. Each IL is cleared to "0" upon acceptance of the interrupt. It is initialized to "0" during reset.

The ILs can be cleared independently by interrupt latch operation instructions ([EICLR IL, r], [DICLR IL, r], and [CLR IL, r]) to make them cancel interrupt requests or initialize by program. When the value of instruction field (r) is "0", the interrupt latch is cleared; when the value is "1", the IL is held. Note that the ILs cannot be set by instruction.

Example 1: To enable IOVF1, INT1, and INT2 interrupts.

```
LD      A,#0101B    ;  EIR←0101B
XCH     A,EIR
EICLR   IL,111111B  ;  EIF←1
```

Example 2: To set the EIF to "1", and to clear the interrupt latches except ISIO to "0".

```
EICLR   IL,010000B  ;  EIF←1, IL5←0, IL3 – IL0←0
```

(2) Interrupt Processing

An interrupt request is held until the interrupt is accepted or the IL is cleared by the reset or the interrupt latch operation instruction. The interrupt acknowledge processing is performed in 2 instruction cycles after the end of the current instruction execution (or after the timer/counter processing if any). The interrupt service program terminates upon execution of the interrupt return instruction [RETI].

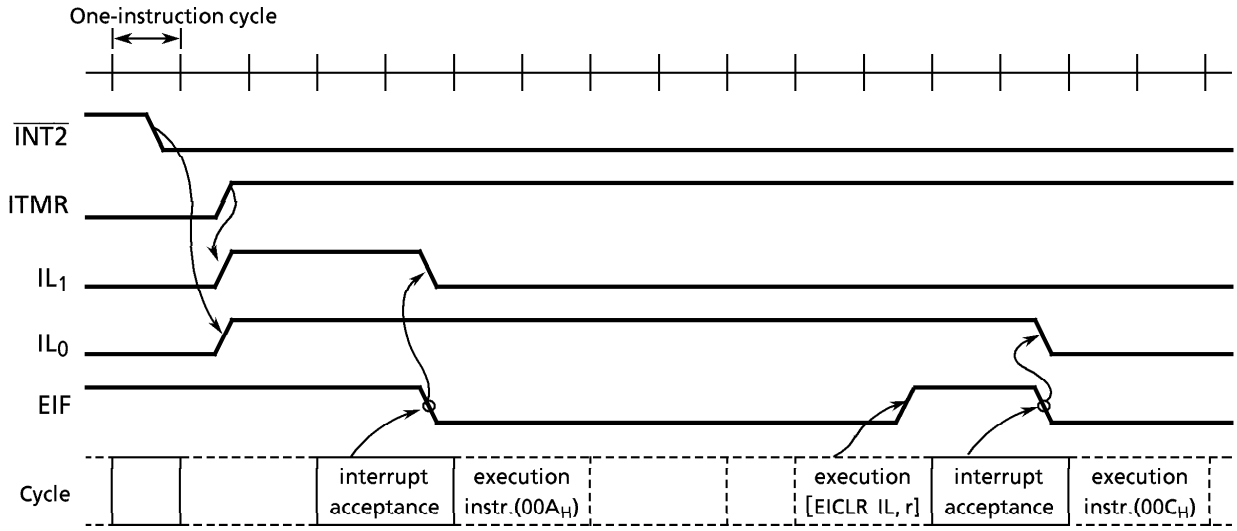
The interrupt acknowledge processing consists of the following sequence:

① The contents of the program counter and the flags are saved on the stack.
② The interrupt entry address corresponding to the interrupt source is set to the program counter.
③ The status flag is set to "1".
④ The EIF is cleared to "0", temporarily disabling the acceptance of subsequent interrupts.
⑤ The interrupt latch for the accepted interrupt source is cleared to "0".
⑥ The instruction stored at the interrupt entry address is executed. (Generally, in the program memory space at the interrupt entry address, the branch instruction to each interrupt processing program is stored.)

To perform the multi-interrupt, the EIF is set to "1" in the interrupt service program, and the acceptable interrupt source is selected by the EIR. However, for the INT1 interrupt, the interrupt service is disabled under software control because it is not disabled by the EIR.

Example: The INT1 interrupt service is disabled under software control (Bit 0 of RAM [05_H] are assigned to the disabling switch of interrupt service).

```
PINT1:  TEST    05H,0    ;  Skips if RAM [05H] 0 is "1"
        B       SINT1
        RETI
SINT1:   ⋮
```



Notes.

1. *It is assumed that there is no other interrupt request and EIR = 0011_B.*
2. *The value r in the [EICLR IL, r] instruction is assumed as 111111_B.*
3. *[ ] denotes the execution of an instruction.*

Figure 2-26. Interrupt Timing chart (Example)

The interrupt return instruction [RETI] performs the following operations :

① Restores the contents of the program counter and the flags from the stack.

② Sets the EIF to "1" to provide the interrupt enable state again.

In the interrupt processing, the program counter and flags are automatically saved or restored but the accumulator and other registers are not. If it is necessary to save or restore them, it must be performed by program as shown in the following example. To perform the multi-interrupt, the saving RAM area never be overlapped.

Example: To save and restore the accumulator and HL register pair.

```
XCH     HL, GSAV1      ;  RAM [GSAV1] ↔ HL
XCH     A, GSAV1 + 2   ;  RAM [GSAV1 + 2] ↔ Acc
```

Note. The lower 2 bits of GSAV1 should be "0".

(3) External Interrupt

When an external interrupt (INT1 or INT2) occurs, the interrupt latch is set at the falling edge of the corresponding pin input ($\overline{INT1}$ or $\overline{INT2}$).

The INT1 interrupt cannot be disabled by the EIR, so that it is always accepted in the interrupt enable state (EIF = "1"). Therefore, INT1 is used for an interrupt with high priority such as an emergency interrupt. When R82 ($\overline{INT1}$) pin is used for the I/O port, the INT1 interrupt occurs at the falling edge of the pin input, so that the interrupt return [RETI] instruction must be stored at the interrupt entry address to perform dummy interrupt processing.

Because the external interrupt input is the hysteresis type, each of high and low level time requires 2 or more instruction cycles for a correct interrupt operation.

2.9 RESET FUNCTION

When the $\overline{\text{RESET}}$ pin is held to the low level for three or more instruction cycles when the power voltage is within the operating voltage range and the oscillation is stable, reset is performed to initialize the internal states.

When the $\overline{\text{RESET}}$ pin input goes high, the reset is cleared and program execution starts from address 000_H. The $\overline{\text{RESET}}$ pin is a hysteresis input with a pull-up resistor (220kΩ typ.). Externally attaching a capacitor and a diode implement a simplified power-on-reset operation.

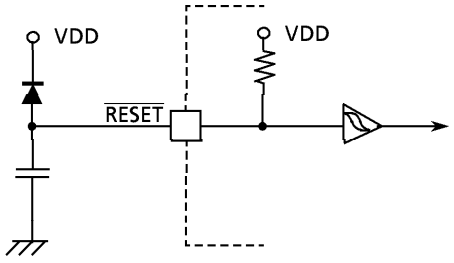


Figure 2-27. Simplified Power-On-Reset Circuit

On-chip hardware	Initial value	On-chip hardware	Initial value
Program counter (PC)	000_H	Output latch (I/O ports or Output ports)	Refer to "INPUT/OUTPUT Circuitry".
Status flag (SF)	1		
Interrupt enable master flip-flop (EIF)	0		
Interrupt enable register (EIR)	0_H	Command register	Refer to the description of each relative command register.
Interrupt latch (IL)	"0"		
Interval timer	"0"		

Table 2-3. Initialization of Internal States by Reset Operation

2.9.1 Warm-Start

The warm-start capability to hold the data memory contents in the reset operation is not supported by hardware. However, it can be implemented by the following:

① Back up the voltage to be supplied to VDD pin.
② Apply to the $\overline{\text{HOLD}}$ pin the waveform synchronized with the power voltage variation.
③ Set the hold operating mode during the power is down.
④ Reset by program using the output port of sink open drain (initial "Hi-Z") after releasing the hold operation.
⑤ Apply to an input port the power-on detection signal, and skip the initialize routines such as clearing RAM.

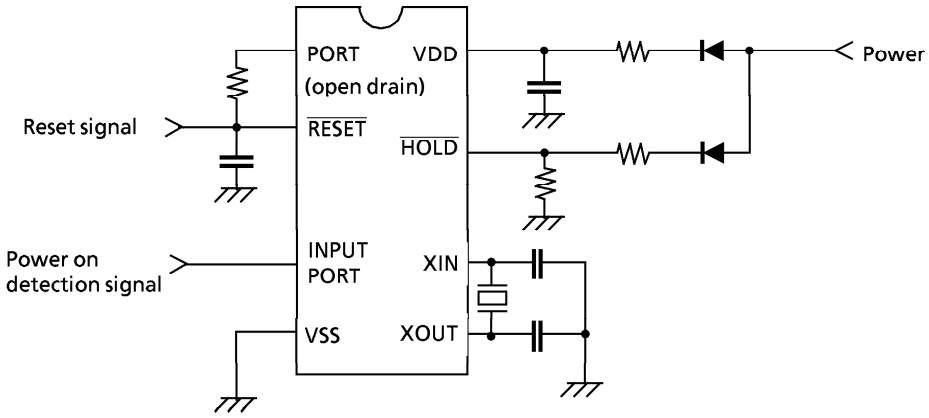


Figure 2-28. Warm-Start Circuit Example

3. PERIPHERAL HARDWARE FUNCTION

3.1 Ports

The 47C452B have 10 I/O ports (35 pins) each as follows:

① K0	;	4-bit input
② R3	;	4-bit input/output
③ R4, R5, R6, R7	;	4-bit input/output
④ R8	;	4-bit input/output (shared with external interrupt input and timer/counter input)
⑤ R9	;	3-bit input/output (shared with serial port)
⑥ P14	;	3-bit output (P140 is shared by BEEP Output)
⑦ KE	;	1-bit sense input (shared with hold request/release signal input)

Each output port contains a latch, which holds the output data. The input ports have no latch; therefore, it is desired to hold data externally until it is read or read twice or more before processing it.

3.1.1 I/O Timing

(1) Input timing

External data is read from an input port or an I/O port in the S3 state of the second instruction cycle during the input instruction (2-cycle instruction) execution. This timing cannot be recognized from the outside, so that the transient input such as chattering must be processed by program.

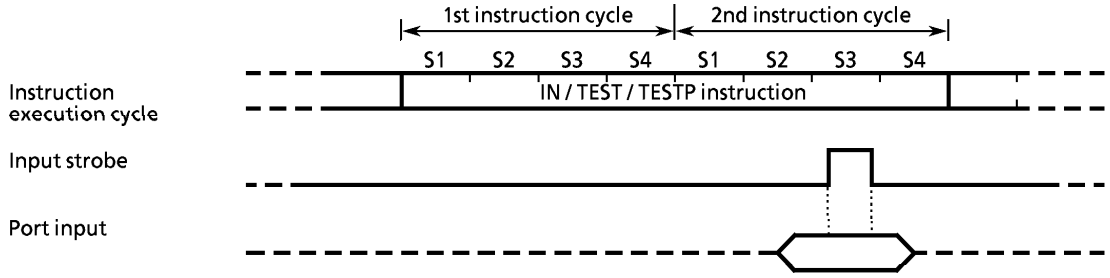


Figure 3-1. Input Timing

(2) Output timing

Data is output to an output port or an I/O port in the S4 state of the second instruction cycle during the output instruction (2-cycle instruction) execution.

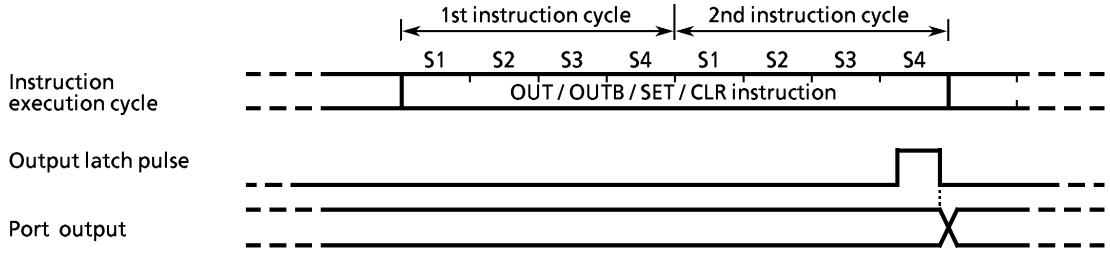


Figure 3-2. Output Timing

Port address (**)	Port		Input/Output instruction						
	Input (IP**)	Output (OP**)	IN %p, A IN %p, @HL	OUT A, %p OUT@HL,%p	OUT #k, %p	OUTB @HL	SET %p, b CLR %p, b	TEST %p, b TESTP %p, b	SET @L CLR @L TEST @L
00_H	K0 Input port	——	○	–	–	–	–	○	–
01	ROW register	ROW register	○	○	○	○ (Note2)	○	○	–
02	COLUMN register	COLUMN register	○	○	○	○	○	○	–
03	R3 Input port	R3 Output port	○	○	○	–	○	○	–
04	R4 Input port	R4 Output port	○	○	○	–	○	○	○
05	R5 Input port	R5 Output port	○	○	○	–	○	○	○
06	R6 Input port	R6 Output port	○	○	○	–	○	○	○
07	R7 Input port	R7 Output port	○	○	○	–	○	○	○
08	R8 Input port	R8 Output port	○	○	○	–	○	○	–
09	R9 Input port	R9 Output port	○	○	○	–	○	○	–
0A	RAM2 address register	RAM2 address register	○	○	○	–	○	○	–
0B	RAM2 address register	RAM2 address register	○	○	○	–	○	○	–
0C	RAM2 data buffer register	RAM2 data buffer register	○	○	○	–	○	○	–
0D	RAM2 command register	RAM2 command register	○	○	○	–	–	○	–
0E	SIO, Hold status	——	○	–	–	–	–	○	–
0F	Serial transfer buffer	Serial transfer buffer	○	○	○	–	–	–	–
10_H	Undefined	Hold operating mode control	–	○	–	–	–	–	–
11	Undefined	——	–	–	–	–	–	–	–
12	Undefined	RAM2 address register	–	○	–	–	–	–	–
13	Undefined	BEEP output control	–	○	–	–	–	–	–
14	Undefined	P14 Output port	–	○	–	–	–	–	–
15	Undefined	——	–	–	–	–	–	–	–
16	Undefined	——	–	–	–	–	–	–	–
17	Undefined	——	–	–	–	–	–	–	–
18	Undefined	——	–	–	–	–	–	–	–
19	Undefined	Interval Timer interrupt control	–	○	–	–	–	–	–
1A	Undefined	——	–	–	–	–	–	–	–
1B	Undefined	——	–	–	–	–	–	–	–
1C	Undefined	Timer/Counter 1 control	–	○	–	–	–	–	–
1D	Undefined	Timer/Counter 2 control	–	○	–	–	–	–	–
1E	Undefined	——	–	–	–	–	–	–	–
1F	Undefined	Serial interface control	–	○	–	–	–	–	–

Note 1. "——" means the reserved state. Unavailable for the user program.

Note 2. The 5-bit to 8-bit data conversion instruction [OUT @ HL] , automatic access to ROW register and COLUMN register.

Table 3-1. Port Address Assignments and Available I/O Instructions

3.1.2 I/O Ports

(1) Port K0 (K03-K00)

The 4-bit input port with pull-up resistors, shared by hold request/release signal input.

Port K0 (Port address IP00)

3	2	1	0
K03 (HOLD3)	K02 (HOLD2)	K01 (HOLD1)	K00 (HOLD0)

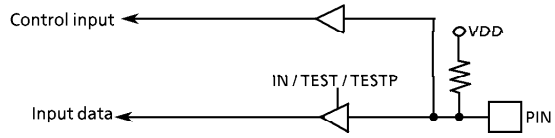


Figure 3-3. Port K0

(2) Port R3 (R33-R30)

The 4-bit I/O port with latch. When used as an input port, the latch must be set to "1". The latch is initialized to "1" during reset.

Port R3 (Port address OP03 / IP03)

3	2	1	0
R33	R32	R31	R30

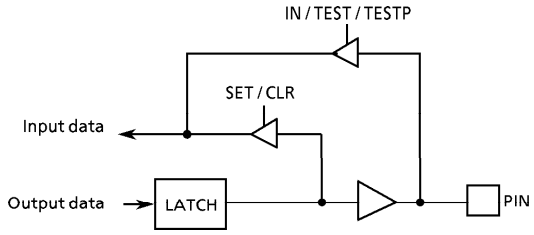


Figure 3-4. Port R3

(3) Port P14 (P142-P140)

The 3-bit output port with latch. The latch is initialized to "1" during reset. The pin P140 is shared by the BEEP output. When used as the BEEP output, the latch must be set to "1".

Port P14 (Port address OP14)

3	2	1	0
	P142	P141	P140 (BEEP)

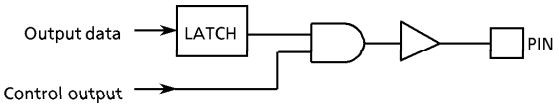


Figure 3-5. Port P14

(4) Ports R4 (R43 – R40), R5 (R53 – 50), R6 (R63 – R60), R7 (R73 – R70)

These ports are 4-bit I/O ports with a latch. When used as an input port, the latch must be set to "1". The latch is initialized to "1" during reset.

These 4 ports (16 pins) can be set, cleared, and tested for each bit as specified by L register indirect addressing bit manipulation instructions ([SET @L], [CLR @L], and [TEST @L]). Table 3-2 lists the pins (I/O ports) that correspond to the contents of L register.

Example: To clear R43 output as specified by the L register indirect addrressing bit manipulation instruction.

```
LD      L, #0011B       ;  Sets R43 pin address to L register
CLR     @L              ;  R43←0
```

L register 3 2 1 0	PIN
0 0 0 0	R40
0 0 0 1	R41
0 0 1 0	R42
0 0 1 1	R43

L register 3 2 1 0	PIN
0 1 0 0	R50
0 1 0 1	R51
0 1 1 0	R52
0 1 1 1	R53

L register 3 2 1 0	PIN
1 0 0 0	R60
1 0 0 1	R61
1 0 1 0	R62
1 0 1 1	R63

L register 3 2 1 0	PIN
1 1 0 0	R70
1 1 0 1	R71
1 1 1 0	R72
1 1 1 1	R73

Table 3-2. Relationship between L register contents and I/O port bits

Port R* (Port address OP0* / IP0*)

3	2	1	0
R*3	R*2	R*1	R*0

*; 4 to 7

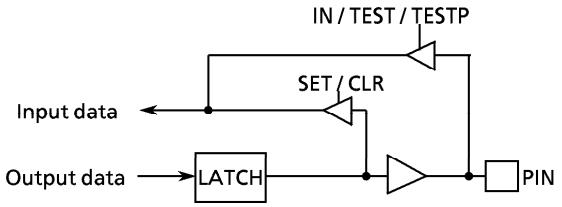


Figure 3-6. Ports R4 – R7

(5) Port R8 (R83 – R80)

Port R8 is a 4-bit I/O port with a latch. When used as an input port, the latch must be set to "1". The latch is initialized to "1" during reset.

Port R8 is shared with the external interrupt input pin and the timer/counter input pin. To use this port for one of these functional pins, the latch should be set to "1". To use it for an ordinary I/O port, the acceptance of external interrupt should be disabled or the event counter/pulse width measurement modes of the timer/counter should be disabled.

Note: When R82 ($\overline{INT1}$) pin is used for an I/O port, external interrupt 1 occurs upon detection of the falling edge of pin input, and if the interrupt enable master flip-flop is enabled, the interrupt request is always accepted. So that a dummy interrupt processing must be performed (only the interrupt return instruction [RETI] is executed).

With R80 ($\overline{INT2}$) pin, external interrupt 2 occurs like R82 in but bit 0 of the interrupt enable register (EIR_0) is only kept at "0", not accepting the interrupt request.

Port R8 (Port address OP08 / IP08)

3	2	1	0
R83 (T1)	R82 ($\overline{INT1}$)	R81 (T2)	R80 ($\overline{INT2}$)

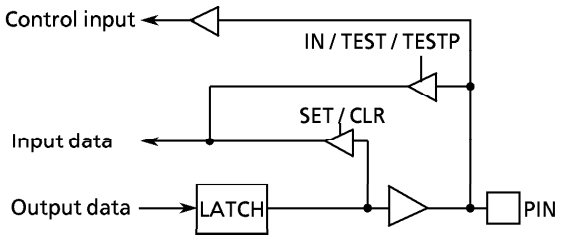


Figure 3-7. Port R8

(6) Port R9 (R92 – R90)

Port R9 is a 3-bit I/O port with a latch. When used as an input, the latch must be set to "1". The latch is initialized to "1" during reset.

Port R9 is shared with the serial port. To use port R9 for the serial port, the latch should be set to "1". To use port R9 for an ordinary I/O port, the serial interface must be disabled.

Note that R93 pin does not exist actually but "1" is read when an input instruction is executed.

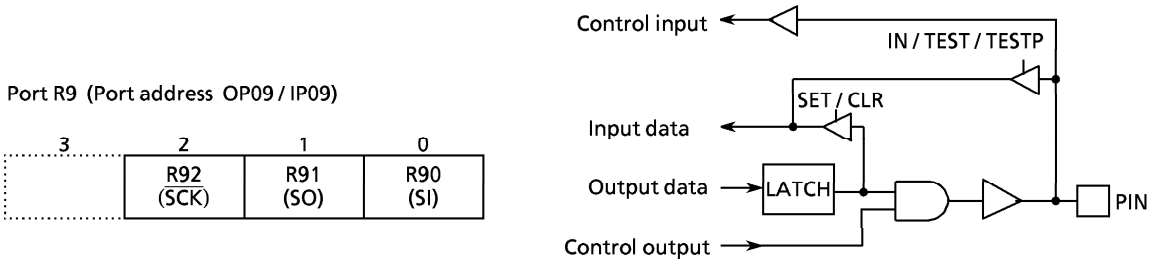


Figure 3-8. Port R9

(7) Port KE ($\overline{KE0}$)

Port KE is a 1-bit sense input port shared with the hold request/release signal input in ($\overline{HOLD}$). This input port is assigned to the least significant bit of port address IP0E and is processed as the data with inverted polarity. For example, if an input instruction is executed with the pin on the high level, "0" is read.

Example: To wait until $\overline{KE0}$ pin goes low.

```
SWAIT   :   TEST    %IP0E,0        ; Waits if KE0 pin = "H".
            B       SWAIT
```

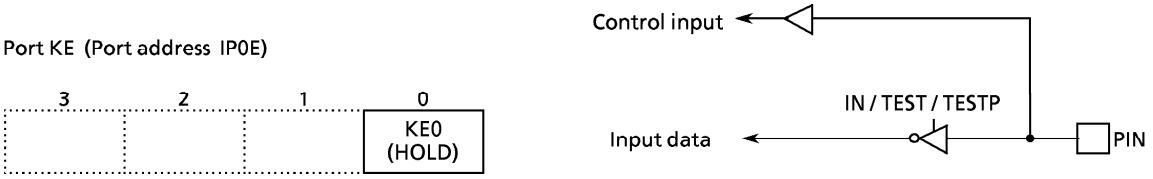


Figure 3-9. Port KE

3.2 Interval Timer

3.2.1 Configuration of Interval Timer

The interval timer consists of a 18-stage binary counter with a divided-by-16 prescaler. The basic clock (frequency: fc) provides the interval timer. Therefore, the output frequency at the last stage is $fc/2^{22}$[Hz]. During reset, the binary counter is cleared to "0", however, the prescaler is not cleared.

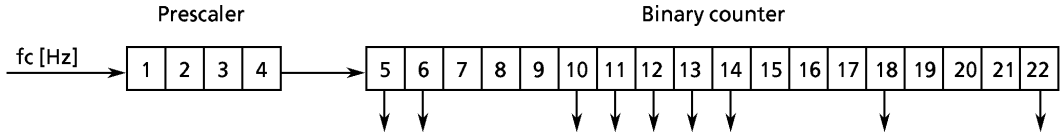


Figure 3-10. Configuration of Interval Timer

3.2.2 Function of Interval Timer

The interval timer provides the following functions:

① Generation of an interrupt with a fixed frequency (the interval timer interrupt)
② Generation of an internal source clock for timer/counters
③ Generation of an internal serial clock for a serial interface
④ Generation of a warm-up time for releasing of the hold operating mode

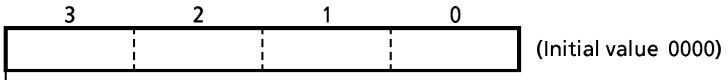
3.2.3 Interval Timer Interrupt (ITMR)

The interval timer can be used to generate an interrupt with a fixed frequency. For an interval timer interrupt, one of 4 frequencies can be selected by command. The command register (OP19) is initialized to "0" during reset. An interval timer interrupt is generated at the first rising edge of the binary counters output after the command has been set. The interval timer is not cleared by command, so that the first interrupt may occur earlier than the preset interrupt period.

Example: To set the interval timer interrupt frequency to $fc/2^{12}$ [Hz].

```
LD      A,  #0110B    ;  OP19←0110B
OUT     A,  %OP19
```

Interval Timer interrupt command register (Port address OP19)



Example: At fc = 480kHz

00**	:	Interrupt disabled		
0100	:	Interrupt frequency	$fc / 2^{10}$ [Hz] ···	469 [Hz]
0101	:	Interrupt frequency	$fc / 2^{11}$ ···	234
0110	:	Interrupt frequency	$fc / 2^{12}$ ···	117
0111	:	Interrupt frequency	$fc / 2^{13}$ ···	58.6
1***	:	Reserved		

*Note 1. * ; don't care*

Note 2. fc ; Basic clock frequency [Hz]

Figure 3-11. Interval Timer Interrupt Command Register

3.3 Timer/Counters (TC1, TC2)

The 47C452B contain two 12-bit timer/counters (TC1, TC2). RAM addresses are assigned to the count register in unit of 4 bits, permitting the initial value setting and counter reading through the RAM manipulation instruction. When the timer/counter is not used, the mode selection may be set to "stopped" to use the RAM at the address corresponding to the timer/counter for storing the ordinary user-processed data.

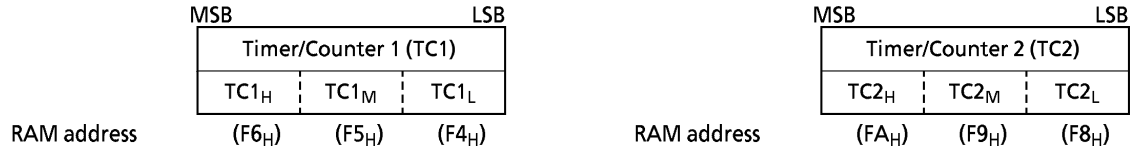


Figure 3-12. The Count registers of the Timer/Counters (TC1, TC2)

3.3.1 Functions of Timer/Counters

The timer/counters provide the following functions:

① Event counter
② Programmable timer
③ Pulse width measurement

3.3.2 Control of Timer/Counters

The timer/counters are controlled by the command registers. The command register is accessed as port address OP1C for TC1 and port address OP1D for TC2. These registers are initialized to "0" during reset.

Timer/Counter 1 control command register (Port address OP1C)

3	2	1	0	
TC1MS		IPR1		(Initial value 0000)

Timer/Counter 2 control command register (Port address OP1D)

3	2	1	0	
TC2MS		IPR2		(Initial value 0000)

TC2MS	Mode selection
00	Stop
01	Event counter mode
10	Timer mode
11	Pulse width measurement mode

IPR2	Internal pulse rate (interval timer output) selection	Example: At fc = 480kHz
00	$fc / 2^{10}$[Hz]	469 [Hz]
01	$fc / 2^{14}$	29.3
10	$fc / 2^{18}$	1.8
11	$fc / 2^{22}$	0.1

(Note) fc; Basic clock frequency [Hz]

Figure 3-13. Timer/Counter Control Command Registers

The timer/counter increments at the rising edge of each count pulse. Counting starts with the first rising edge of the count pulse generated after the command has been set. Count operation is performed in one instruction cycle after the current instruction execution, during which the execution of a next instruction and the acceptance of an interrupt are delayed. If counting is requested by both TC1 and TC2 simultaneously, the request by TC1 is preferred. The request by TC2 is accepted in the next instruction cycle. Therefore, during count operation, the apparent instruction execution speed drops as counting occurs more frequently.

The timer/counter causes an interrupt upon occurrence of an overflow (a transition of the count value from FFF_H to 000_H). If the timer/counter is in the interrupt enabled state and the overflow interrupt is accepted immediately after its occurrence, the interrupt is processed in the sequence shown in Figure 3-14. Note that counting continues if there is a count request after overflow occurrence.

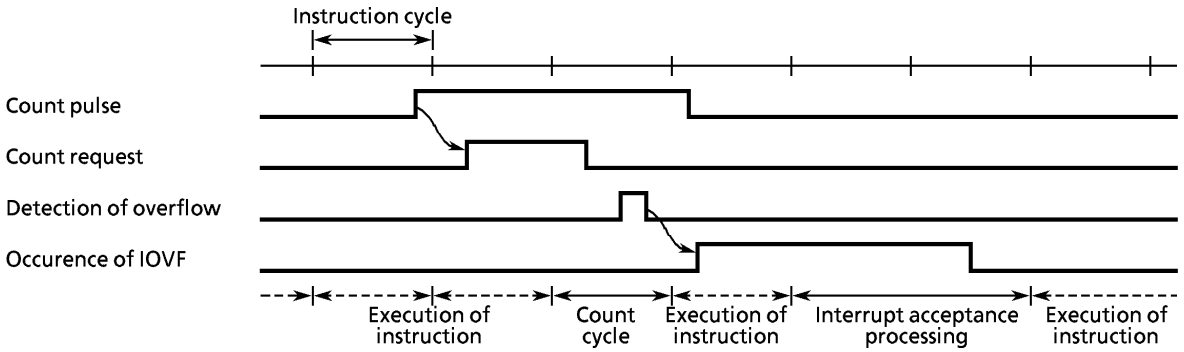


Figure 3-14. Timer Mode Timing Chart

(1) Event counter mode

In the event counter mode, the timer/counter increments at each rising edge of the external pin (T1, T2) input. The maximum applied frequency of the external pin input is fc/32 for the 1-channel operation; for the 2-channel operation, the frequency is fc/32 for TC1 and fc/40 for TC2. The apparent instruction execution speed drops most to $(9/11) \times 100 = 82\%$ when TC1 and TC2 are operated at the maximum applied frequency because the count operation is inserted once every 4 instruction cycles for TC1 and every 5 cycles for TC2. For example, the instruction execution speed of 16.7 μs drops to 30.4 μs.

Example: To operate TC2 in the event counter mode

```
LD      A,  #0100B    ;  OP1D←01**B
OUT     A,  %OP1D
```

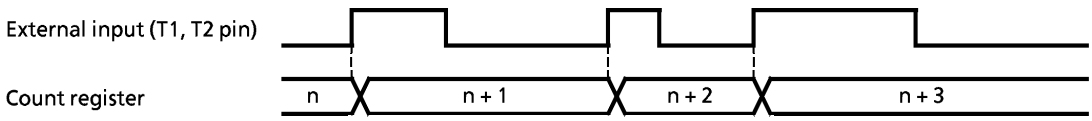


Figure 3-15. Event Counter Mode Timing chart

(2) Timer mode

In the timer mode, the timer/counter increments at the rising edge of the internal pulse generated from the interval timer. One of 4 internal pulse rates can be selected by the command register. The selected rate can be initially set to the timer/counter to generate an overflow interrupt in order to create a desired time interval.

$$1 \div \{ \frac{fc/8}{\text{(Internal pulse rate)}} - 1 \} \times 100 \quad [\%]$$

When an internal pulse rate of fc/2^{10} is used, a count operation is inserted once every 128 instruction cycles, so that the apparent instruction execution speed drops by (1/127) × 100 = 0.8%. For example, the instruction execution speed of 16.7μs drops to 16.8μs.
In the timer mode, R83 (T1) and R81 (T2) pins provide the ordinary I/O ports.

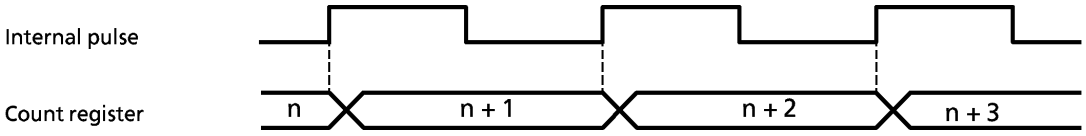


Figure 3-16. Timer Mode Timing chart

Example: To generate an overflow interrupt (at fc = 480kHz) by the TC1 after 100 ms.

```
LD      HL,  #0F4H    ;  TC1←FD1H (setting of the count register)
ST      #1,  @HL+
ST      #D,  @HL+
ST      #0FH,  @HL+
LD      A,  #1000B    ;  OP1C←1000B
OUT     A,  %OP1C
LD      A,  #0100B    ;  EIR←0100B (enables interrupt)
XCH     A,  EIR
EICLR   IL,  110111B  ;  EIF←1, IL3←0
```

Calculating the preset value of the counter register

The preset value of the count register is obtained from the following relation:

2^{12} – (interrupt setting time) × (internal pulse rate)

For example, to generate an overflow interrupt after 100ms at fc = 480kHz with the internal pulse rate of fc/2^{10}, set the following value to the count register as the preset value:

$2^{12} - (100 \times 10^{-3}) \times (4.8 \times 10^{5}/2^{10}) = 4049 = FD1_H$

Internal pulse rate	Max. setting time	Example : At fc = 480 kHz	
		Internal pulse rate	Max. setting time
fc / 2^{10} [Hz]	2^{22} / fc [s]	469 [Hz]	8.7 [s]
fc / 2^{14}	2^{26} / fc	2.9	140
fc / 2^{18}	2^{30} / fc	1.8	2237
fc / 2^{22}	2^{34} / fc	0.1	35791

Table 3-3. Internal Pulse Rate Selection

(3) Pulse width measurement mode

In the pulse width measurement mode, the timer/counter increments with the pulse obtained by sampling the external pins (T1, T2) by the internal pulse. As shown in Figure 3-17, the timer/counter increments only while the external pin input is high. The maximum applied frequency to the external pin input must be one that is enough for analyzing the count value. Normally, a frequency sufficient slower than the internal pulse rate setting is applied to the external pin.

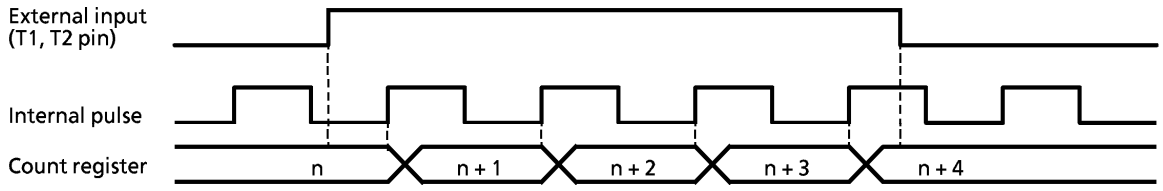


Figure 3-17. Pulse Width Measurement Mode Timing chart

3.4 DTMF Generator

The 47C452B has a built-in DTMF generator which generates dialing signals for tone dialing type telephones. There are two groups of tone dial signals, one group of 4 sine wave low frequencies and another group of 4 sine wave high frequencies. All of these frequencies can be selected individually and combined with a frequency from the other group for a total of 16 different DTMF composite waves.

(DTMF ; Dual Tone Multi Frequency)

3.4.1 Configuration of DTMF Generator

Figure 3-4 shows configuration of the DTMF generator. The 47C452B generates two stepped, quasi sine waves for tone dial signals which can be combined and output. The high or low group of frequencies is selected by setting frequency selection codes into the ROW and COLUMN registers.

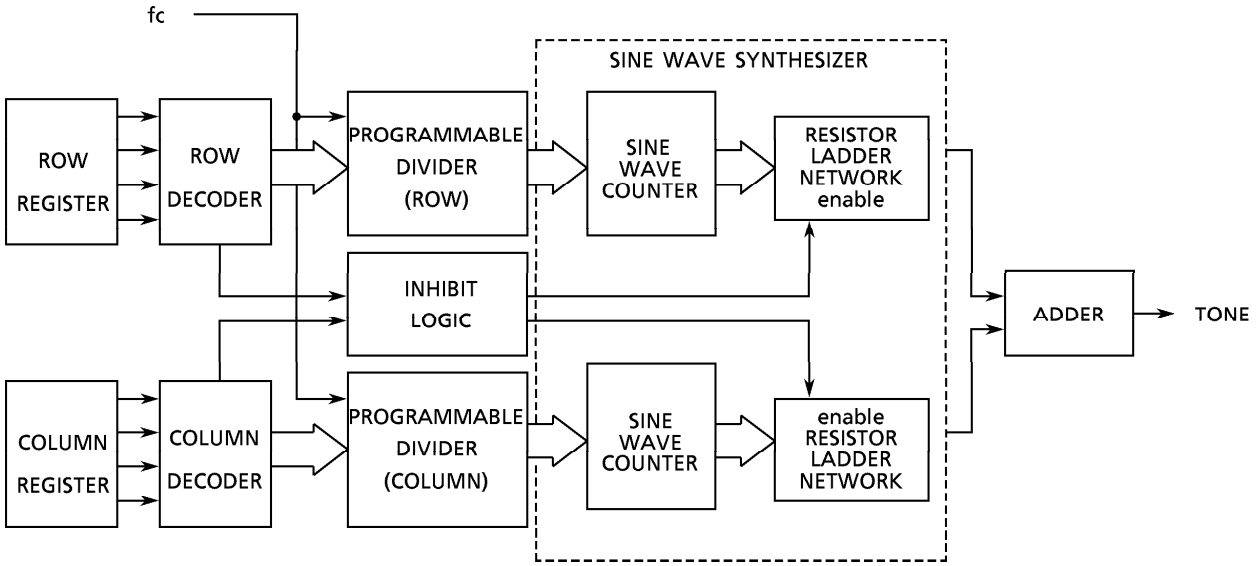


Figure 3-18. Configuration of DTMF Generator

3.4.2 Control of DTMF Generator

Tone output is controlled by ROW register (OP01/IP01) and COLUMN register (OP02/IP02) . And single tone is controlled by TONE command register (OP0D/IP0D) . ROW register, COLUMN register and TONE command register are initialized to "0" during the reset.

TONE command register (Port address OP0D/IP0D)

3	2	1	0
STE		(R / W)	(CE)

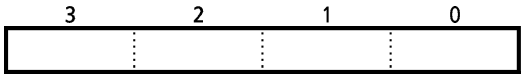
(Initial value 0*00)

STE	Controls single tone output
0 :	Disable mode of single tone output
1 :	Enable mode of single tone output

Note 1. **; don't care*
Note 2. *When read STE bit, "1" is always read.*

Figure 3-19. TONE command register

ROW register (Port address OP01 / IP01)

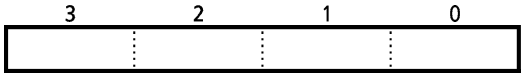


(Initial value 0000)

Selects ROW tone frequency

0001 : Outputs 697.7Hz single tone
0010 : Outputs 769.2Hz single tone
0100 : Outputs 857.1Hz single tone
1000 : Outputs 937.5Hz single tone

COLUMN register (Port address OP02 / IP02)



(Initial value 0000)

Selects COLUMN tone frequency

0001 : Outputs 1212.1Hz single tone
0010 : Outputs 1333.3Hz single tone
0100 : Outputs 1481.5Hz single tone
1000 : Outputs 1621.6Hz single tone

Figure 3-20. ROW, COLUMN Register

Tones are outputted by loading the frequency selection codes shown in Figure 3-6 into the ROW and COLUMN registers. In the enable mode of single tone output and either ROW or COLUMN register is disabled, another register remains to be enabled, and so single tone can be outputted, by loading an ineffective code into the register. When both the registers are enabled, dual tone can be outputted. In the disable mode of single tone output, effective codes are loaded into both ROW and COLUMN registers and then dual tone can be outputted. At this time, an ineffective code is loaded into ROW or COLUMN register and then the 47C452B has no tone output signal.
The [OUTB @HL] instruction can set 8-bit data into both registers (the upper 4 bits of the ROM data go to the COLUMN register and the lower 4 bits go to the ROW register) at the same time, and DTMF signal is outputted without single tone output.

Example 1 : To output 1481.5Hz single tone

```
OUT     #8,%OP0D ;  Sets the enable mode of single tone output.
OUT     #0,%OP01 ;  Sets an ineffective code into ROW register.
OUT     #4,%OP02 ;  Sets data "4" into COLUMN register
```

Example 2 : 8 bits data corresponding to the 5 bits of data linking the content of carry flag and the contents of data memory RAM address 90_H are read from the ROM, frequency selection codes are loaded into ROW and COLUMN registers, and dual tone is outputted.

```
LD      HL,#90H  ;  HL←90H (Sets the address of the data memory)
OUTB    @HL      ;  Sets the ROM data into the ROW and COLUMN register.
```

Table 3-4 shows the corresponding frequency selection codes of the ROW and COLUMN registers for the telephone dial keys. Table 3-5 shows the deviation between the 47C452B tone output frequency and standard frequency.

		COLUMN register (OP02 / IP02)		
	Frequency selection code	0001 (1209)	0010 (1336)	0100 (1477)
ROW register (OP01/IP01)	0001 (697)	1	2	3
	0010 (770)	4	5	6
	0100 (852)	7	8	9
	1000 (941)	*	0	#
		Standard telephone dial key		

Contents of ( ) are standard frequencies, unit : Hz

Table 3-4. Corresponding frequency selection codes of the ROW and COLUMN registers for the telephone dial keys

ROW Tone						
Frequency selection code				Tone output frequency [Hz]	Standard frequency [Hz]	Deviation [%]
3	2	1	0			
0	0	0	1	697.7	697	+ 0.10
0	0	1	0	769.2	770	– 0.10
0	1	0	0	857.1	852	+ 0.60
1	0	0	0	937.5	941	– 0.37

COLUMN Tone						
Frequency selection code				Tone output frequency [Hz]	Standard frequency [Hz]	Deviation [%]
3	2	1	0			
0	0	0	1	1212.1	1209	+ 0.26
0	0	1	0	1333.3	1336	– 0.20
0	1	0	0	1481.5	1477	+ 0.30
1	0	0	0	1621.6	1633	– 0.70

Table 3-5. Tone output frequencies and Deviation from standard

3.4.3 Test mode for tone output

The 47C452B includes a test mode for checking tone output waveforms. Tones can be outputted by the curcuit shown in Figure 3-21. ROW data are inputted from the R6 port and COLUMN data are inputted from the R3 port, and any desired single or dual tones can be outputted by setting the frequency selection codes shown in Figure 3-20. Figure 3-22 shows a single tone waveform and Figure 3-23 shows a dual tone waveform.

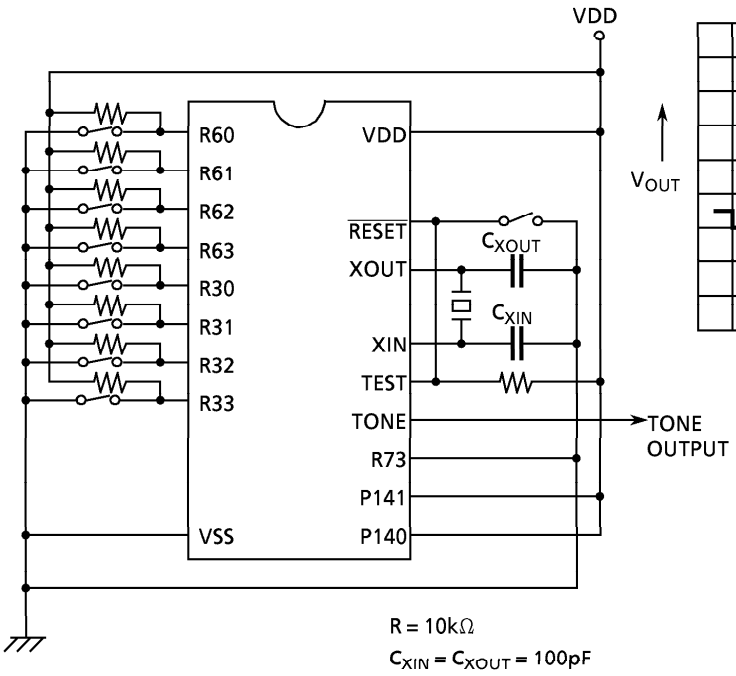


Figure 3-21. Tone test circuit

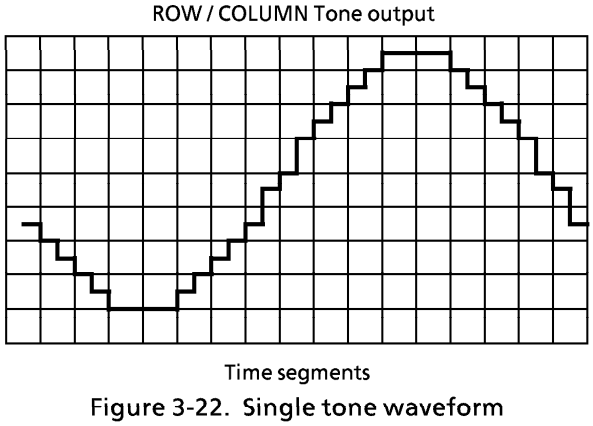


Figure 3-22. Single tone waveform

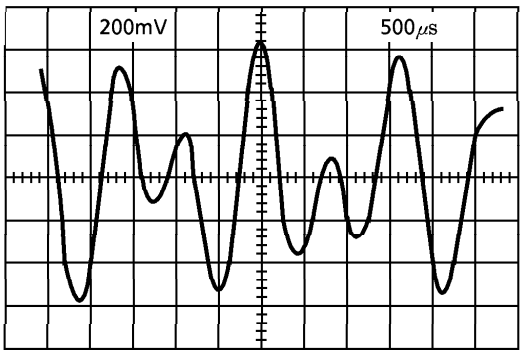


Figure 3-23. Dual tone waveform

3.5 BEEP Output Circuit

BEEP output circuit generates square wave in the audible frequency range. This circuit can drive the key input confirmation tone generator circuit for telephone applications.

BEEP output is from the P140 (BEEP) pin. This pin is for both P140 output and BEEP output. Set the P140 output latch to "1" for BEEP output.

3.5.1 Configuration of BEEP Output Circuit

Figure 3-10 shows configuration of the BEEP output circuit. The clock pulse of BEEP output circuit is supplied by an interval timer. BEEP output is controlled by frequency selection and output enable/disable setting.

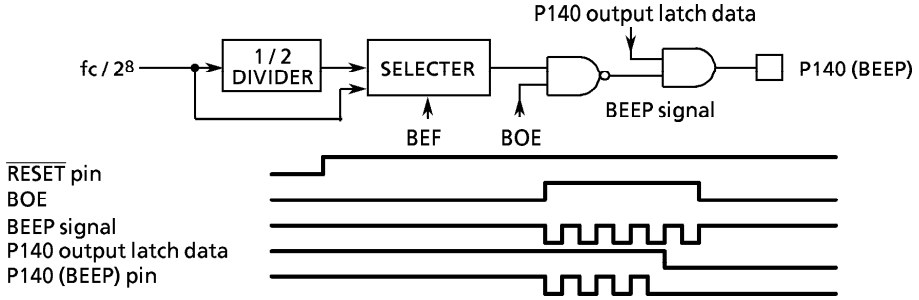


Figure 3-24. BEEP Output Circuit Configuration and Timing Chart

3.5.2 Control of BEEP Output

BEEP output is controlled with the BEEP output control command register (OP13).

BEEP Output Control command register (Port address OP13)

3	2	1	0
BOE			BEF

(Initial value 0**0)

BOE	Controls BEEP output
0:	Disable output
1:	Enable output

BEF	Selects BEEP output frequency	Example: At fc = 480kHz
0:	fc / 256 [Hz] ·····	1875 [Hz]
1:	fc / 512 ·····	937.5

Figure 3-25. BEEP Output Control Command Register

3.6 Serial Interface (SIO)

The 47C452B have a serial interface with a 4-bit buffer. The serial interface is connected to the external device via 3 pins (the serial port): R92 ($\overline{SCK}$), R91 (SO), and R90 (SI). The serial port is shared by port R9. For the serial port, the output latch of port R9 must be set to "1". In the transmit mode, R90 pin provides the I/O port; in the receive mode, R91 pin provides the I/O port.

3.6.1 Configuration of Serial Interface

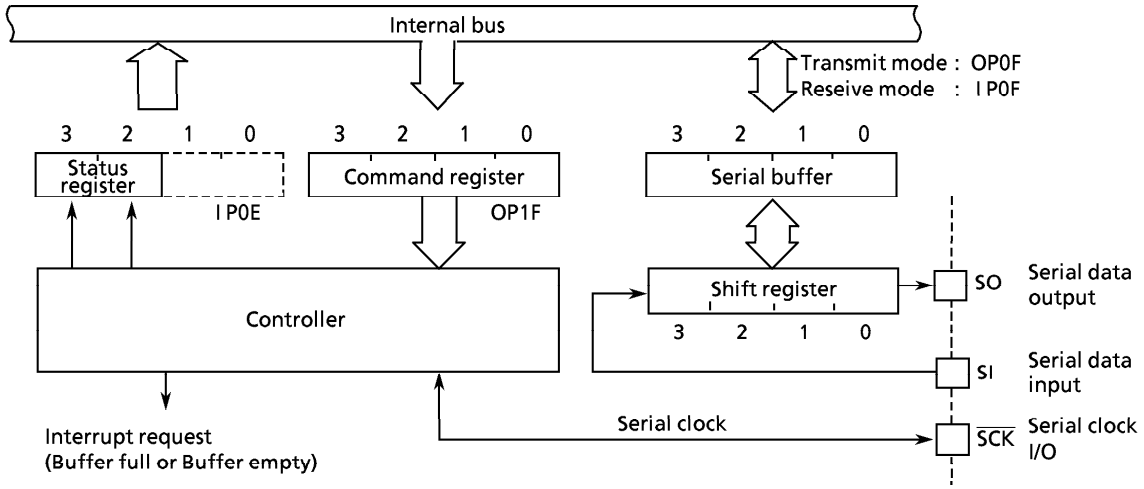


Figure 3-26. Configuration of Serial Interface

3.6.2 Control of Serial Interface

The serial interface is controlled by the command register (OP1F). The command register is initialized to "0" during reset. The operating states of the serial interface can be monitored by the status register (IP0E). The status of the $\overline{HOLD}$ ($\overline{KE0}$) pin is assigned to bit 0 of this status register.

Serial Interface control command register (Port address OP1F)

3	2	1	0
ESIO	RM	LM	ECKM

(Initial value 0000)

ESIO	Instructs serial transfer start / terminate

0 : Serial transfer terminate
1 : Serial transfer start

RM	Selects transfer mode

0 : Transmit mode
1 : Receive mode

LM	Selects shift edge

0 : Shift at the trailing edge of serial clock
1 : Shift at the leading edge of serial clock

ECKM	Selects serial clock

0 : Internal clock (output to $\overline{SCK}$ pin)
1 : External clock (input from $\overline{SCK}$ pin)

Note. When setting the transfer mode, ESIO must be "0".
When setting the transmit mode, LM must be "1".

Figure 3-27. Serial Interface Control Command Register

Serial Interface status register (Port address IP0E)

3	2	1	0
SIOF	SEF		(HOLD) (KE0)

SIOF	Monitors serial transfer operation state

0 : Transfer is terminated
1 : Transfer is in progress

SEF	Monitors shift operation state

0 : Shift operation is terminated
1 : Shift operation is in progress

Figure 3-28. Serial Interface Status Register

(1) Serial clock
For the serial clock, one of the following can be selected according to the contents of the command register:

a. Clock source selection

① Internal clock
fc/2^7 [Hz] is used for the serial clock (at fc = 480kHz, the serial clock frequency is 3.75kHz). The serial clock is output on the $\overline{SCK}$ pin. Note that at the start of transfer, the $\overline{SCK}$ pin output goes high. This serial interface provides the wait function in which the shift is not occurred until these processings are completed.
The highest transfer rate based on the internal clock is 3580 bits/second (at fc = 480kHz).

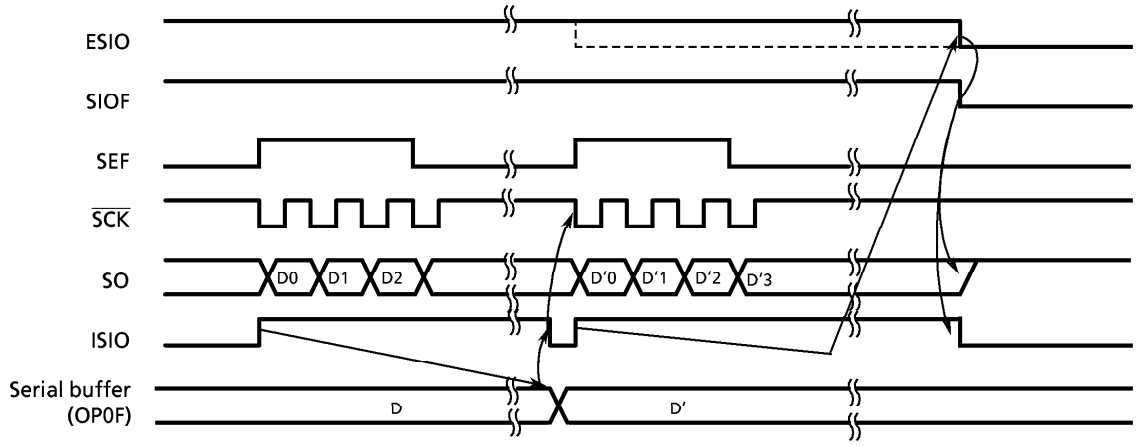
② External clock
The signal obtained by the clock supplied to the $\overline{SCK}$ pin from the outside is used for the serial clock. In this case, the output latch of R92 ($\overline{SCK}$) must be set to "1" beforehand. For the shift operation to be performed correctly, each of the clock's high and low levels needs two instructions or more to be completed.

b. Shift edge selection
① Leading edge
Data is shifted at the leading edge (the falling edge of $\overline{SCK}$ pin input) of the serial clock.
② Trailing edge
Data is shifted at the trailing edge (the rising edge of $\overline{SCK}$ pin input) of the serial clock. However, in the transmit mode, the trailing edge shift is not supported.

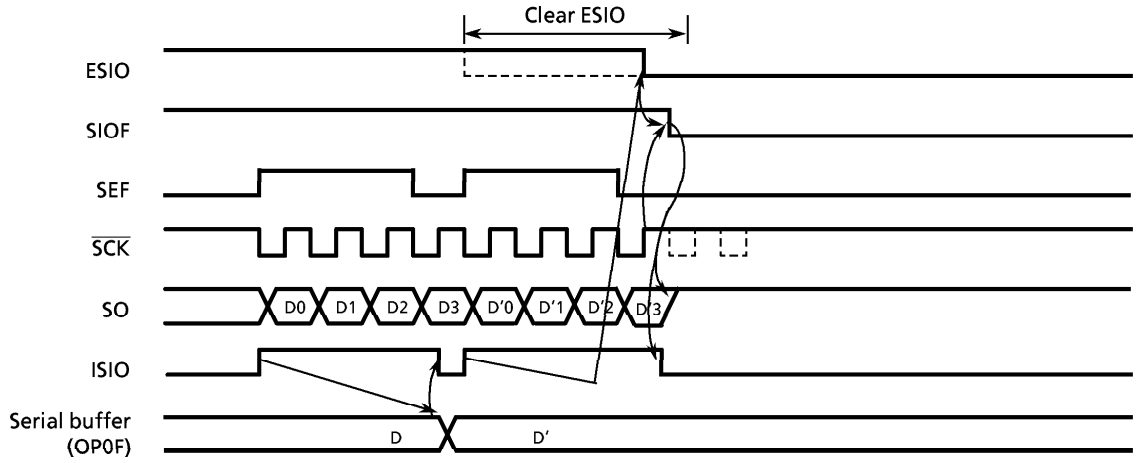
(2) Transfer modes
Selection between the transmit mode and the receive mode is performed by RM (bit 2 of the command register).

a. Transmit mode
The transmit mode is set to the command register then the first transmit data (4 bits) is written to the buffer register OP0F (if the transmit mode is not set, the data is not written to the buffer register). Then, setting ESIO to "1" starts transmission. The transmit data is output to the SO pin synchronization with the serial clock from the LSB side sequentially. When the LSB is output, the transmit data is moved from the buffer register to the shift register. When the buffer register becomes empty, the buffer empty interrupt (ISIO) to request for the next transmit data is generated. When the interrupt service program writes the transmit data to the buffer register, the interrupt request is reset.
In the operation based on the internal clock, if no more data is set after the transmission of the 4-bit data, the serial clock is stopped and the wait state sets in.
In the operation based on the external clock, the data must be set in the buffer register by the time the next data shift operation starts. Therefore, the transfer rate is determined by the maximum delay time between the occurrence of the interrupt request and the writing of data to the buffer register by the interrupt service program.

To end transmission, ESIO is cleared to "0" instead of writing the next transmit data by the buffer empty interrupt service program. When ESIO is cleared, transmission stops upon termination of the currently shifted-out data. The transmission end can be known by the SIOF state (SIOF goes "0" upon transmission end). In the operation based on the external clock, ESIO must be cleared to "0" before the next data is shifted out. If ESIO is not cleared before, the transmission stops upon transmitting the next data (dummy).



(a) Internal-clock-based operation with wait



(b) External-clock-based operation

Figure 3-29. Transmit Mode

Example 1: To transmit data stored in the RAM (its address is specified by the HL register pair) in synchronization with the internal clock.

```
        LD      A, #0010B       ; OP1F←0010B (Sets the transmit mode of
                                  internal-clock-based operation)
        OUT     A, %OP1F
        OUT     @HL, %OP0F      ; OP0F←RAM [HL] (Writes the first transmit
                                  data)
        LD      A, #1010B       ; ESIO←1 (Instructs transmission start)
        OUT     A, %OP1F
```

Example 2: To end transmission (internal-clock-based operation)

```
        LD      A, #0010B       ; ESIO←0 (Instructs transmission end)
        OUT     A, %OP1F
SENDC:  TESTP   %IP0E, 3        ; Waits until SIOF = "0"
        B       SENDC
```

b. Receive mode

Data can be received when ESIO is set to "1" after setting the receive mode to the command register. The data is put from the SI pin to the shift register in synchronization with the serial clock. Then the 4-bit data is transferred from the shift register to the buffer register (IP0F), upon which the buffer full interrupt (ISIO) to request for reading received data is generated. The received data is read from the buffer register by the interrupt service program. When the data has been read, the interrupt request is reset and the next data is put in the shift register to be transferred to the buffer register.

In the operation based on the internal clock, if the previous received data has not been read from the buffer register at the end of capturing the next data, the serial clock is stopped and the wait operation is performed until the data has been read.

In the operation based on the external clock, the shift operation is performed in synchronization with the externally-supplied clock, so that the data must be read from the buffer register before the next receive data is transferred to it. The maximum transfer rate in the external-clock-based operation is determined by the maximum delay time between the generation of interrupt request and the reading of received data.

In the receive mode, the shift operation may be performed at either the leading edge or the trailing edge. In the leading edge shift operation, data is captured at the leading edge of the serial clock, so that the first shift data must be put in the SI pin before the first serial clock is applied at the start of transfer.

Example: To instruct the receive start operation with the internal clock and leading edge shift (with the interrupt enable register already set).

```
LD      A,#0110B    ;  OP1F←0110B (Sets the receive mode)
OUT     A,%OP1F
EI                  ;  EIF←1 (Enables interrupt)
LD      A,#1110B    ;  ESIO←1 (Instructs receive start)
OUT     A,%OP1F
 ⋮
```

To end the receive operation, ESIO should be cleared to "0". When ESIO is cleared, the completion of the transfer of the current 4-bit data to the buffer register terminates the receive operation. To confirm the end of the receive operation by program, SIOF should be sensed. SIOF goes "0" at the end of receive operation.

Note: If the receive and transmit modes are switched, the contents of the buffer register are lost. Therefore, the modes should not be switched until the last receive data is read even after the end of reception is instructed (by clearing ESIO to "0").

The receive operation can be terminated in one of the following approaches determined by the transfer rate:

① When the transfer rate is sufficiently low (the external-clock-based operation):
If ESIO can be cleared to "0" before the next serial clock is applied upon occurrence of buffer full interrupt in the external-clock-based operation, ESIO is cleared to "0" by the interrupt service program, then the last receive data is read.

Example: To instruct the receive end with sufficient low transfer rate (the reading edge shift).

```
LD      A,#0111B    ;  ESIO←0 (Instructs receive end)
OUT     A,%OP1F
IN      %IP0F,A     ;  Acc←IP0F (Reads received data)
```

② When the transfer rate is sufficiently high (the internal/external clock-based operation):
If the transfer rate is high and, therefore, it is possible that the capture of the next data starts before ESIO is cleared to "0" upon acceptance of an interrupt, ESIO must be cleared to "0" by confirming that SEF (bit 2 of the status register) is set at reading the data proceeding the last data. Then, the data is read.
In the interrupt servicing following the reception of the last data, no operation is needed for termination; only the reading of the received data is performed. This method is generally employed for the internal-clock-based operations. For an external-clock-based operation, ESIO must be cleared and the receive data must be read before the last data is transferred to the buffer register.

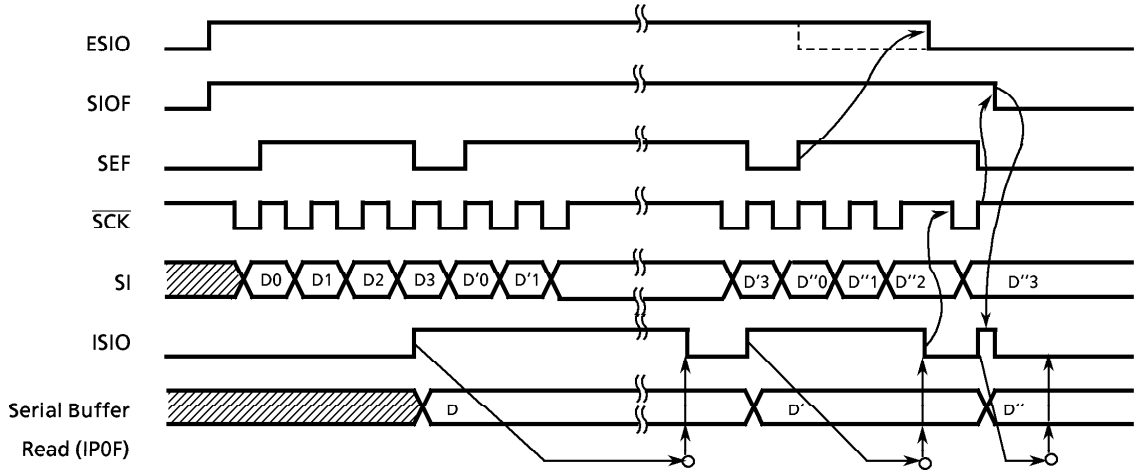
Example: To instruct receive end when the transfer rate is high (the internal clock, reading edge shift).

```
SSEF0:  TEST    %IP0E, 2        ;  Waits until SEF = "1"
        B       SSEF0
        LD      A, #0110B       ;  ESIO←0 (Instructs receive end)
        OUT     A, %OP1F
        IN      %IP0F, A        ;  Acc←IP0F (Reads received data)
```

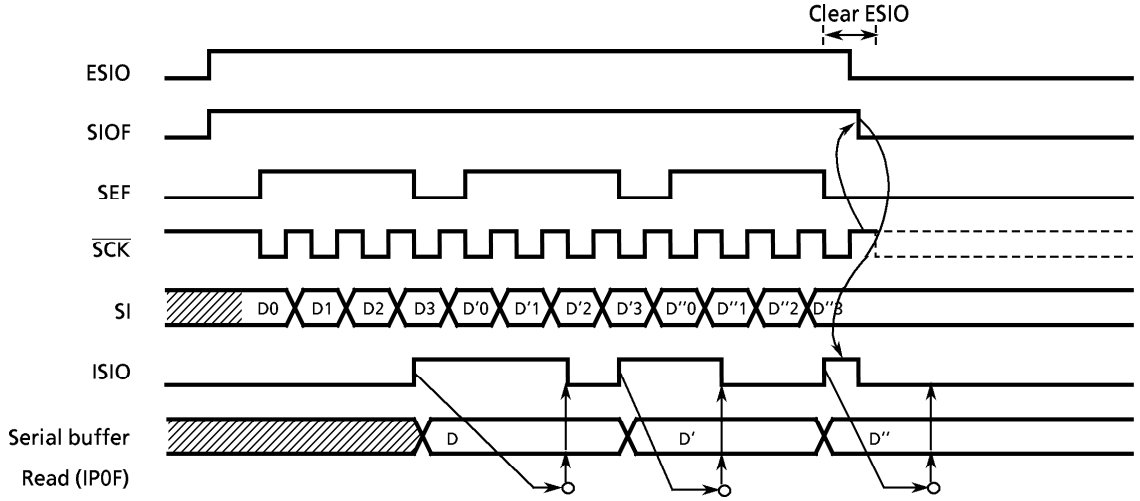
③ One-word reception
When receiving only one word, ESIO is set to "1" then it is returned to "0" after confirming that SEF (bit 2 of the status register) has gone "1". In this case, buffer full interrupt is caused only once, so that the received data is read by the interrupt service program.

Example: To instruction the start/end of one-word reception (the internal clock, the leading edge shift).

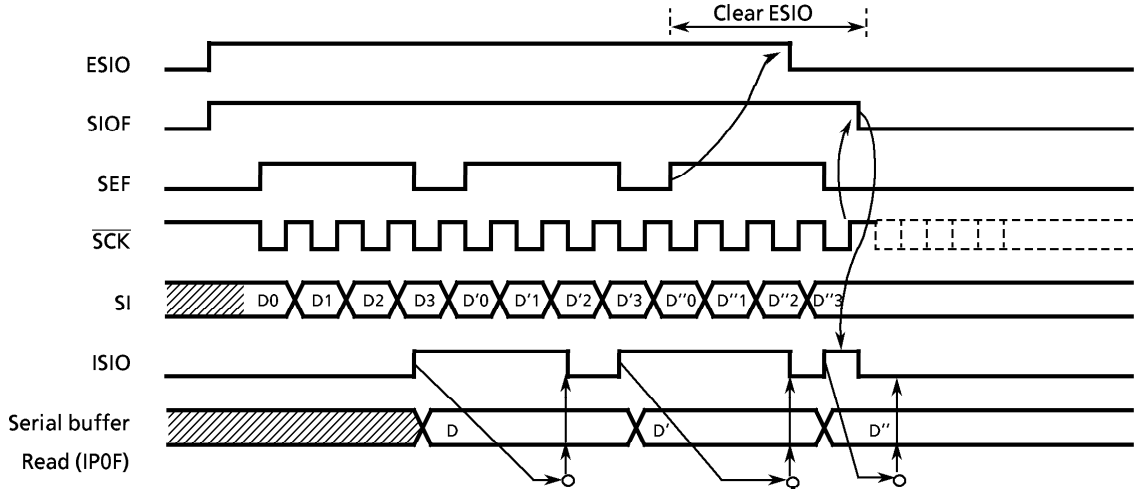
```
        LD      A, #0110B       ;  OP1F←0110B (Sets in the receive mode)
        OUT     A, %OP1F
        EI                      ;  EIF←1 (Enables interrupt)
        LD      A, #1110B       ;  ESIO←1 (Instructs receive start)
        OUT     A, %OP1F
SSEF0:  TEST    %IP0E, 2        ;  Confirms that SEF = "1"
        B       SSEF0
        LD      A, #0110B       ;  ESIO←0 (Instructs receive end)
        OUT     A, %OP1F
```



(a) Internal-clock-based operation, trailing-edge shift (with wait)



(b) External-clock-based operation, reading-edge shift (when transfer rate is low)



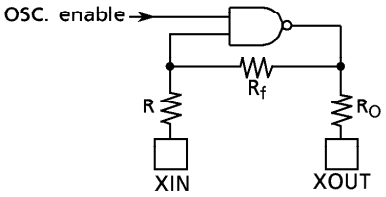
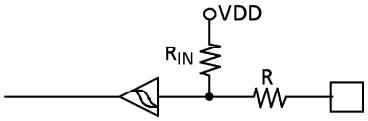
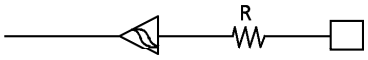
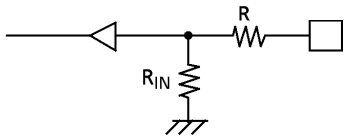
(C) Internal-clock-based operation, reading-edge shift (when transfer rate is high)

Figure 3-30. Receive Mode

INPUT/OUTPUT CIRCUITRY

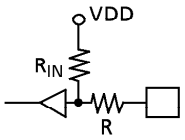
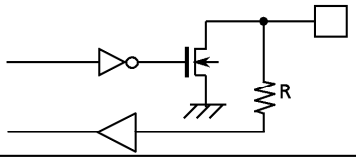
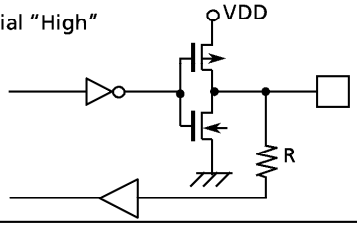
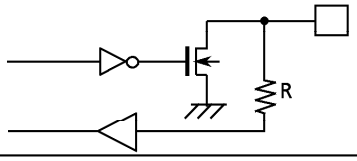
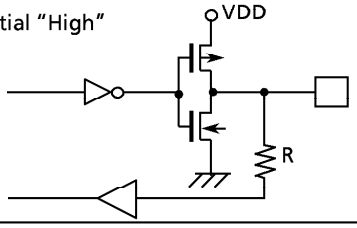
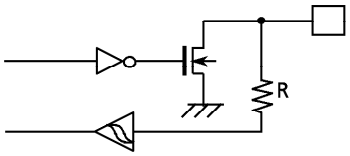
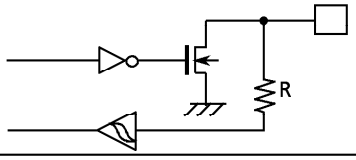
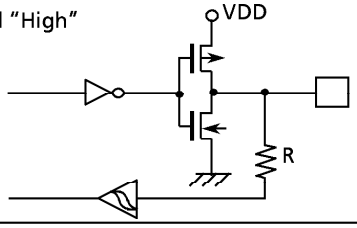
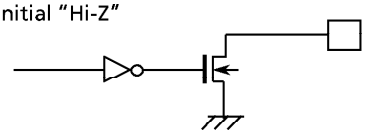
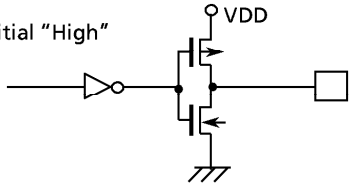
(1) Control pins

The input/output circuitries of the 47C452B control pins are shown below.

CONTROL PIN	I/O	CIRCUITRY	REMARKS
XIN XOUT	Input Output		Resonator connecting pins R = 1kΩ (typ.) R_f = 1.5MΩ (typ.) R_O = 2kΩ (typ.)
$\overline{\text{RESET}}$	Input		Hysteresis input Pull-up resistor R_{IN} = 220kΩ (typ.) R = 1kΩ (typ.)
$\overline{\text{HOLD}}$ ($\overline{\text{KE0}}$)	Input (Input)		Hysteresis input (Sense input) R = 1kΩ (typ.)
TEST	Input		Pull-down resistor R_{IN} = 70kΩ (typ.) R = 1kΩ (typ.)

(2) I/O Ports

The input/output circuitries of the 47C452B I/O ports are shown as below, any one of the circuitries can be chosen by a code (WB, WE, WH) as a mask option.

PORT	I/O	INPUT/OUTPUT CIRCUITRY and CODE	REMARKS
K0	Input		Pull-up resistor R_{IN} = 70kΩ (typ.) R = 1kΩ (typ.)
R3 R4 R5 R6	I/O	WB: Initial "Hi-Z"  WE, WH: Initial "High" 	Sink open drain or push-pull output R = 1kΩ (typ.)
R7	I/O	WB, WE: Initial "Hi-Z"  WH: Initial "High" 	Sink open drain or push-pull output R = 1kΩ (typ.)
R8	I/O	Initial "Hi-Z" 	Sink open drain Hysteresis input R = 1kΩ (typ.)
R9	I/O	WB, WE: Initial "Hi-Z"  WH: Initial "High" 	Sink open drain or push-pull output Hysteresis input R = 1kΩ (typ.)
P14	Output	WB: Initial "Hi-Z"  WE, WH: Initial "High" 	Sink open drain or push-pull output

ELECTRICAL CHARACTERISTICS

ABSOLUTE MAXIMUM RATINGS (V_{SS} = 0V)

PARAMETER	SYMBOL	PINS	RATINGS	UNIIT
Supply Voltage	V_{DD}		– 0.3 to 7	V
Input Voltage	V_{IN}		– 0.3 to V_{DD} + 0.3	V
Output Voltage	V_{OUT1}	Except sink open drain pin	– 0.3 to V_{DD} + 0.3	V
	VOUT2	Sink open drain pin	– 0.3 to 10	
Output Current (per 1 pin)	I_{OUT}		3.2	mA
Power Dissipation [T_{opr} = 60°C]	PD		600	mW
Soldering Temperature (time)	T_{sld}		260 (10 s)	°C
Storage Temperature	Ts_{tg}		– 55 to 125	°C
Operating Temperature	T_{opr}		– 30 to 60	°C

RECOMMENDED OPERATING CONDITIONS (V_{SS} = 0V, T_{opr} = – 30 to 60°C)

PARAMETER	SYMBOL	PINS	CONDITIONS	Min.	Max.	UNIT
Supply Voltage	V_{DD}		In the Normal mode	2.2	6.0	V
			In the HOLD mode	2.0		
Input High Voltage	V_{IH1}	Except hysteresis input	$V_{DD} \geqq 4.5V$	$V_{DD} \times 0.7$	V_{DD}	V
	V_{IH2}	Hysteresis input		$V_{DD} \times 0.75$		
	V_{IH3}		$V_{DD} < 4.5V$	$V_{DD} \times 0.9$		
Input Low Voltage	V_{IL1}	Except hysteresis input	$V_{DD} \geqq 4.5V$	0	$V_{DD} \times 0.3$	V
	V_{IL2}	Hysteresis input			$V_{DD} \times 0.25$	
	V_{IL3}		$V_{DD} < 4.5V$		$V_{DD} \times 0.1$	
Clock Frequency	fc			480		kHz

D.C. CHARACTERISTICS

(V_{SS} = 0V, V_{DD} = 2.2 to 6.0V, T_{opr} = – 30 to 60°C)

PARAMETER	SYMBOL	PINS	CONDITIONS	Min.	Typ.	Typ.	UNIT
Hysteresis Voltage	V_{HS}	Hysteresis Input		—	0.7	—	V
Input Current	I_{IN1}	Port K0, TEST, $\overline{RESET}$, $\overline{HOLD}$	V_{DD} = 5.5V, V_{IN} = 5.5V / 0V	—	—	± 2	μA
	I_{IN2}	Port R (open drain)					
Input Low Current	I_{IL}	Port R (push-pull)	V_{DD} = 5.5V, V_{IN} = 0.4V	—	—	– 2	mA
Input Resistance	R_{IN1}	Port K0		30	70	150	kΩ
	R_{IN2}	$\overline{RESET}$		100	220	450	
Output Leakage Current	I_{LO}	Ports P, R (open drain)	V_{DD} = 5.5V, V_{OUT} = 5.5V	—	—	2	μA
Output High Voltage	V_{OH}	Port R (push-pull)	V_{DD} = 4.5V, I_{OH} = – 200μA	2.4	—	—	V
Output Low Voltage	V_{OL2}	Except XOUT	V_{DD} = 4.5V, I_{OL} = 1.6mA	—	—	0.4	V
Supply Current (in the Normal mode)	I_{DD}		Except TONE generating V_{DD} = 2.2V fc = 480kHz	—	0.2	0.4	mA
	I_{DDT}		TONE generating V_{DD} = 2.2 V fc = 480kHz	—	0.5	1.0	
Supply Current (in the HOLD mode)	I_{DDH}		V_{DD} = 5.5V	—	0.5	10	μA
			V_{DD} = 2.2V, Topr = 25°C	—	—	0.5	

Note 1. Typ.values show those at T_{opr} = 25°C, V_{DD} = 5V.

Note 2. Input Current I_{IN1} : The current through resistor is not included, when the pull-up/pull-down resistor is contained.

Note 3. Supply Current : V_{IN} = 2.0V / 0.2V

The K0 port is opened when the pull-up/pull-down resistor is contained.

The Voltage applied to the R port is within the valid range V_{IL} or V_{IH}.

TONE OUTPUT CHARACTERISTICS

(V_{SS} = 0V, V_{DD} = 2.2 to 6.0V, T_{opr} = – 30 to 60°C)

PARAMETER	SYMBOL	CONDITIONS	Min.	Typ.	Max.	UNIT
Tone Output Voltage (ROW)	V_{TONE}	RL ≧ 10kΩ, V_{DD} = 2.2V	125	185	250	mVrms
Pre-Emphasis High Band (COL / ROW)	PEHB	PEHB = 20log (COL /ROW)	1	2	3	dB
Output Distortion	DIS		—	—	10	%
Frequency Stability	△f	Except error of osc. frequency	—	—	0.7	%

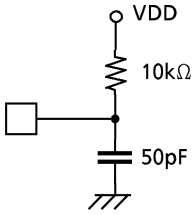
A.C. CHARACTERISTICS

(V_{SS} = 0V, V_{DD} = 2.2 to 6.0V, T_{opr} = – 30 to 60°C)

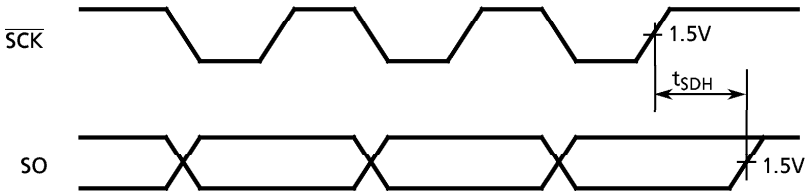
PARAMETER	SYMBOL	CONDITIONS	Min.	Typ.	Max.	UNIT
Instruction Cycle Time	t_{cy}		16.7			μs
Shift Data Hold Time	t_{SDH}		$0.5t_{cy}$-300	—	—	ns

Note. Shift Data Hold Time :

External circuit for $\overline{SCK}$ pin and SO pin



Serial port (completion of transmisson)



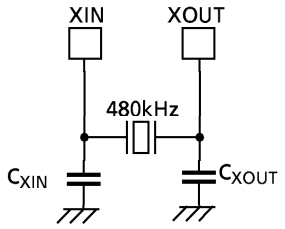
RECOMMENDED OSCILLATING CONDITION

(V_{SS} = 0V, V_{DD} = 2.2 to 6.0V, T_{opr} = – 30 to 60°C)

480kHz

Ceramic Resonator

CSB480E16	(MURATA)	$C_{XIN} = C_{XOUT}$ = 100pF
KBR-480B13	(KYOCERA)	$C_{XIN} = C_{XOUT}$ = 100pF



TYPICAL CHARACTERISTICS