

**Features**

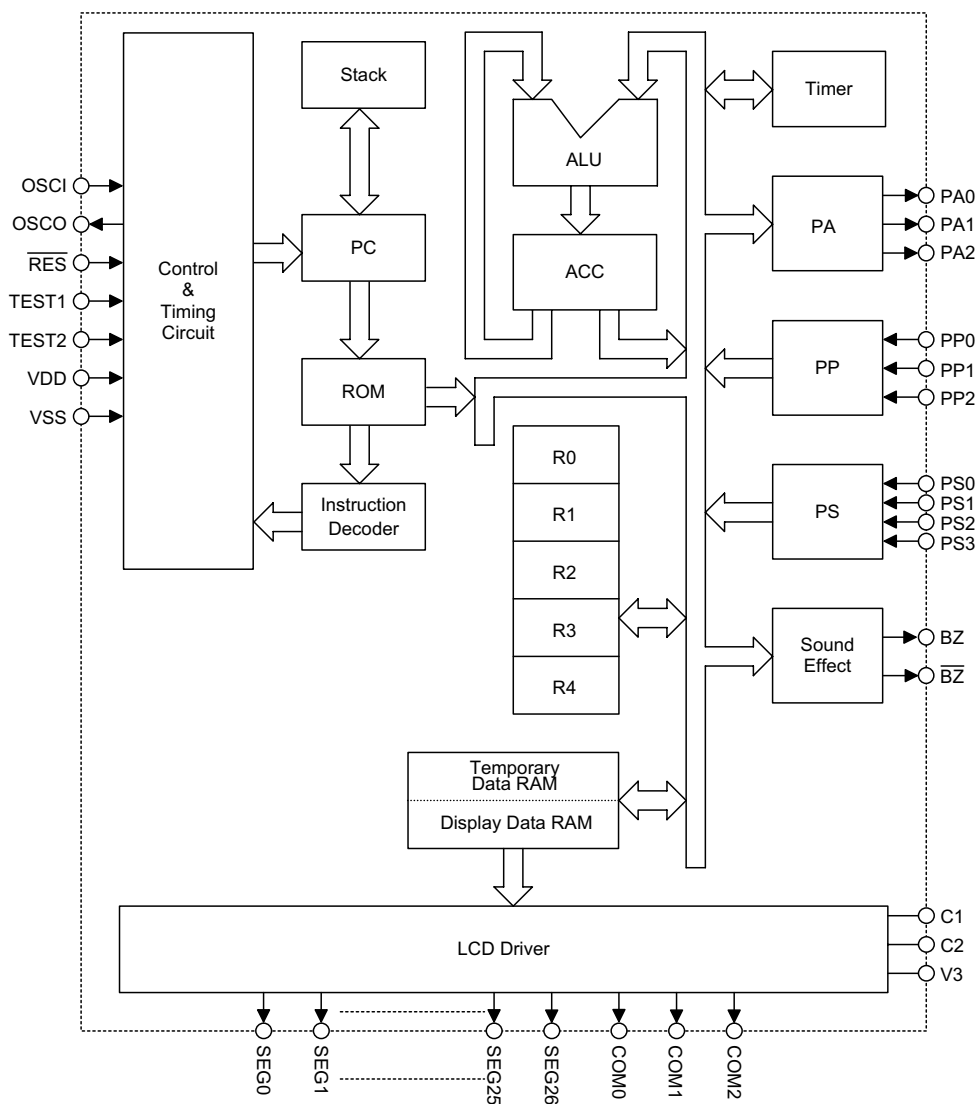
- Operating voltage: 1.2V~1.8V
- 7 input lines
- 3 output lines
- HALT feature reduces power consumption
- Up to 16 $\mu$ s instruction cycle with 256kHz system clock at  $V_{DD}=1.5V$
- All instructions in 1 or 2 machine cycles
- 4K $\times$ 8 program ROM
- Data memory RAM size 128 $\times$ 4 bits
- 27 $\times$ 3 segment LCD driver
- 8-bit table read instruction
- 5 working registers
- Internal timer overflow interrupt
- One level subroutine nesting
- RC oscillator for system clock
- 8-bit timer with internal clock source
- Sound effect circuit

**General Description**

The HTG1390 is the processor from Holtek's 4-bit stand alone single chip microcontroller range specifically designed for LCD product applications. The device is ideally suited for mul-

ti-ple LCD low power applications among which are calculators, scales, and hand held LCD products.

## Block Diagram



Note: ACC: Accumulator

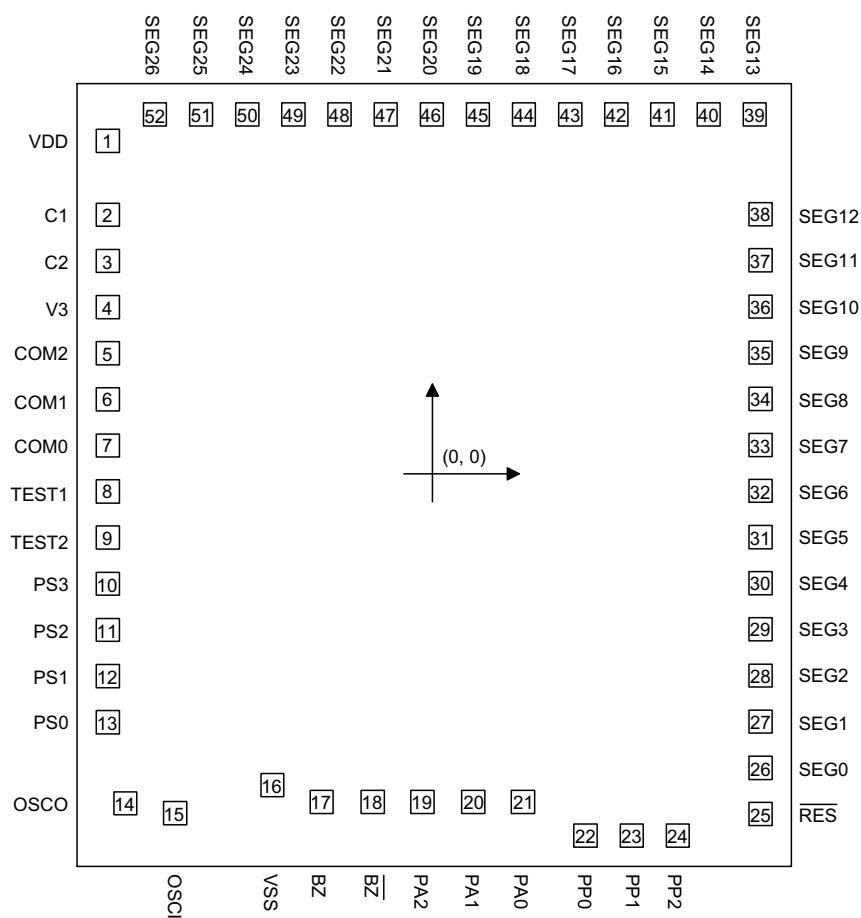
PC: Program counter

R0~R4: Working registers

PA: Output port

PS,PP: Input ports

## Pad Assignment



Chip size:  $1960 \times 2300 (\mu\text{m})^2$

\* The IC substrate should be connected to VSS in the PCB layout artwork.

**Pad Coordinates**

Unit:  $\mu\text{m}$ 

Pad No.	X	Y	Pad No.	X	Y
1	-843.74	866.29	27*	853.56	-644.11
2	-843.74	674.39	28*	853.56	-524.11
3*	-843.74	554.39	29*	853.56	-404.11
4*	-843.74	434.39	30*	853.56	-284.11
5	-843.74	314.39	31*	853.56	-164.11
6	-843.74	194.39	32*	853.56	-44.11
7	-843.74	74.39	33	853.56	75.89
8	-843.74	-45.61	34*	853.56	195.89
9*	-843.74	-165.61	35*	853.56	315.89
10*	-843.74	-285.61	36*	853.56	435.89
11	-843.74	-405.61	37*	853.56	555.89
12	-843.74	-525.61	38*	853.56	675.89
13	-843.74	-645.61	39*	838.76	935.39
14	-798.04	-856.71	40*	718.76	935.39
15	-668.04	-882.11	41*	598.76	935.39
16	-415.94	-809.01	42*	478.76	935.39
17	-287.94	-853.41	43*	358.76	935.39
18	-154.74	-853.41	44*	238.76	935.39
19*	-25.94	-853.41	45*	118.76	935.39
20*	107.26	-853.41	46*	-1.24	935.39
21*	236.26	-853.41	47*	-121.24	935.39
22*	398.66	-940.91	48*	-241.24	935.39
23	518.66	-940.91	49*	-361.24	935.39
24	638.66	-940.91	50*	-481.24	935.39
25	853.56	-884.11	51*	-601.24	935.39
26	853.56	-764.11	52*	-721.24	935.39

\* These pins must be bonded out for functional testing.

## Pad Description

Pad No.	Pad Name	I/O	Mask Option	Description
1	VDD	—	—	Positive power supply
2, 3	C1, C2	I	—	LCD system voltage booster condensor connecting terminal
4	V3	I	—	LCD system power 1/2 bias generated
5~7	COM2~COM0	O	Note 2	Output for LCD panel common plate
8 9	TEST1 TEST2	I I	—	For test mode only TEST1 and TEST2 are left open when the HTG1390 is in normal operation (with an internal pull-high resistor).
10~13	PS3~PS0	I	Pull-high or None. Note 3	4-bit port for input only
15 14	OSCI OSCO	I O	—	OSCI, OSCO are connected to an external resistor for an internal system clock
16	VSS	—	—	Negative power supply, ground
17, 18	BZ, $\overline{BZ}$	O	Note 1	Sound effect outputs
19~21	PA2~PA0	O	CMOS or NMOS Open Drain	3-bit latch port for output only
22~24	PP0~PP2	I	Pull-high or None. Note 2	3-bit port for input only
25	$\overline{RES}$	I	—	Input to reset an internal LSI Reset is active on logical low level
26~52	SEG0~SEG26	O	—	LCD driver outputs for LCD panel segment

Note: The system clock provides 6 different sources selectable by mask option to drive the sound effect clock. If the Holtek sound library is used only 128kHz and 64kHz are acceptable.

Each bit of ports PS and PP can be a trigger source of the HALT interrupt, selectable by mask option.

## Absolute Maximum Ratings

Supply Voltage .....	-0.3V~5.5V	Input Voltage .....	$V_{SS}-0.3V \sim V_{DD}+0.3V$
Storage Temperature .....	-50°C~125°C	Operating Temperature .....	0°C~70°C

Note: These are stress ratings only. Stresses exceeding the range specified under "Absolute Maximum Ratings" may cause substantial damage to the device. Functional operation of this device at other conditions beyond those listed in the specification is not implied and prolonged exposure to extreme conditions may affect device reliability.

**D.C. Characteristics**
 $T_a=25^{\circ}\text{C}$ 

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		$V_{DD}$	Conditions				
$V_{DD}$	Operating Voltage	—	—	1.2	1.5	1.8	V
$I_{DD}$	Operating Current	1.5V	No load, $f_{SYS}=256\text{kHz}$	—	20	—	$\mu\text{A}$
$I_{STB}$	Standby Current	1.5V	No load, HALT mode	—	—	1	$\mu\text{A}$
$V_{IL}$	Input Low Voltage	1.5V	—	0	—	0.4	V
$V_{IH}$	Input High Voltage	1.5V	—	1.0	—	1.5	V
$I_{OL1}$	Port A, BZ & $\overline{\text{BZ}}$ Output Sink Current	1.5V	$V_{DD}=1.5\text{V}$ , $V_{OL}=0.15\text{V}$	95	100	—	$\mu\text{A}$
$I_{OH1}$	Port A, BZ & $\overline{\text{BZ}}$ Output Source Current	1.5V	$V_{DD}=1.5\text{V}$ , $V_{OH}=1.35\text{V}$	600	700	—	$\mu\text{A}$
$I_{OL2}$	Segment Output Sink Current	1.5V	$V_{LCD}=3\text{V}$ , $V_{OL}=0.3\text{V}$	100	150	—	$\mu\text{A}$
$I_{OH2}$	Segment Output Source Current	1.5V	$V_{LCD}=3\text{V}$ , $V_{OH}=2.7\text{V}$	-20	-40	—	$\mu\text{A}$
$R_{PH}$	Pull-high Resistance	1.5V	PS, PP, $\overline{\text{RES}}$	30	—	300	$\text{k}\Omega$

**A.C. Characteristics**
 $T_a=25^{\circ}\text{C}$ 

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		$V_{DD}$	Conditions				
$f_{SYS}$	System Clock	1.5V	R:36k $\Omega$ ~2M $\Omega$	38	—	400	kHz
$f_{LCD}$	LCD Clock	1.5V	—	—	128*	—	Hz
$t_{COM}$	LCD Common Period	—	1/3 duty	—	$(1/f_{LCD}) \times 3$	—	s
$t_{CY}$	Cycle Time	—	$f_{SYS}=256\text{kHz}$	—	16	—	$\mu\text{s}$
$t_{RES}$	Reset Pulse Width	—	—	5	—	—	ms
$f_{SOUND}$	Sound Effect Clock	—	—	—	64 or 128 **	—	kHz

Note: \* In general,  $f_{LCD}$  is selected and optimized by Holtek depending upon  $f_{SYS}$  and the operating voltage.

\*\* Only these two clocking signal frequencies are supported by the Holtek sound library.

## Functional Description

### Program counter – PC

This counter addresses the program ROM and is arranged as an 12-bit binary counter from PC0 to PC11 whose contents specify a maximum of 4096 addresses. The program counter counts with an increment of 1 or 2 with each execution of an instruction.

When executing the jump instruction (JMP, JNZ, JC, JTMR,...), a subroutine call, initial reset, internal interrupt, external interrupt or returning from a subroutine, the program counter is loaded with the corresponding instruction data as shown in the table.

Note: P0~P11: Instruction code

@: PC11 keeps current value

S0~S11: Stack register bits

### Program memory – ROM

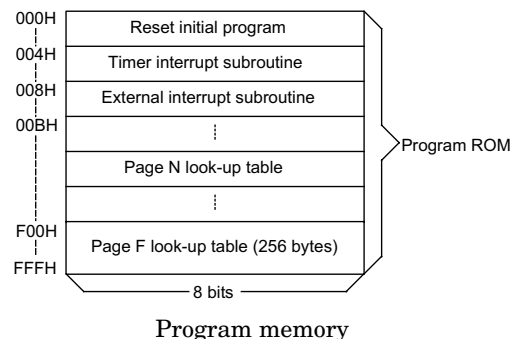
The program memory is the executable memory and is arranged in a 4096×8 bit format. The address is specified by the program counter (PC). Four special locations are reserved as described as follows.

- Location 0

Activating the processor  $\overline{RES}$  pin causes the first instruction to be fetched from location 0.

- Location 4

Contains the timer interrupt resulting from a TIMER overflow. If the interrupts are enabled it causes the program to jump to this subroutine.



- Location 8

Activating the PS or PP input pins of the processor with the interrupts enabled during HALT mode causes the program to jump to this location.

- Locations n00H to nFFH

These are the 256 bytes of each page in program memory. This area from n00H to nFFH and F00H to FFFH can be used as a look-up table. Instructions such as READ R4A, READ MR0A, READF R4A, READF MR0A can read the table and transfer the contents of the table to ACC and R4 or to ACC and a data memory address specified by the register pair R1,R0. However as R1,R0 can only store 8 bits, these instructions cannot fully specify the full 12 bit program memory address. For this reason a jump instruction should be first used to place the program counter in the right page. The above instructions can then be used to read the look up table data.

Mode	Program Counter											
	PC11	PC10	PC9	PC8	PC7	PC6	PC5	PC4	PC3	PC2	PC1	PC0
Initial reset	0	0	0	0	0	0	0	0	0	0	0	0
Internal interrupt	0	0	0	0	0	0	0	0	0	1	0	0
External interrupt	0	0	0	0	0	0	0	0	1	0	0	0
Jump, call instruction	P11	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
Conditional branch	@	P10	P9	P8	P7	P6	P5	P4	P3	P2	P1	P0
Return from subroutine	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

Program memory

Note that the page number  $n$  must be greater than zero as some locations in page 0 are reserved for specific usage as mentioned. This area may function as normal program memory as required.

The program memory mapping is shown in the diagram.

In the execution of an instruction the program counter is added before the execution phase, so careful manipulation of READ MR0A and READ R4A is needed in the page margin.

### Stack register

The stack register is a group of registers used to save the contents of the program counter (PC) and is arranged in 13 bits $\times$ 1 level. One bit is used to store the carry flag. An interrupt will force the contents of the PC and the carry flag onto the stack register. A subroutine call will also cause the PC contents to be pushed onto the stack; however the carry flag will not be stored. At the end of a subroutine or an interrupt (indicated by a return instruction RET or RETI), the contents of the stack register are returned to the PC.

Executing "RETI" instruction will restore the carry flag from stack register, but "RET" doesn't.

### Working registers – R0,R1,R2,R3,R4

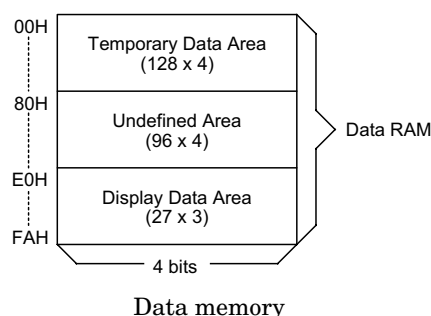
There are 5 working registers (R0,R1,R2,R3,R4) usually used to store the frequently accessed intermediate results. Using the instructions INC Rn and DEC Rn the working registers can increment (+1) or decrement (–1). The JNZ Rn ( $n=0,1,4$ ) instruction makes efficient use of the working registers as a program loop counter. Also the register pairs R0,R1 and R2,R3 are used as a data memory pointer when the memory transfer instruction is executed.

### Data memory – RAM

The static data memory (RAM) is arranged in 256 $\times$ 4 bit format and is used to store data. All of the data memory locations are indirectly addressable through the register pair R1,R0 or R3,R2; for example MOV A,[R3R2] or MOV [R3R2],A.

There are two areas in the data memory, the temporary data area and the display data area. Access to the temporary data area is from 00H to 7FH. Locations E0H to FAH represent the display data area. The locations between the temporary and display data areas are undefined and cannot be used.

When data is written into the display data area it is automatically read by the LCD driver which then generates the corresponding LCD driving signals.



### Accumulator – ACC

The accumulator is the most important data register in the processor. It is one of the sources of input to the ALU and the destination of the results of the operations performed in the ALU. Data to and from the I/O ports and memory also passes through the accumulator.

### Arithmetic and logic unit – ALU

This circuit performs the following arithmetic and logical operations ...

- Add with or without carry
- Subtract with or without carry
- AND, OR, Exclusive-OR
- Rotate right, left through carry
- BCD decimal adjust for addition
- Increment, decrement
- Data transfers
- Branch decisions

The ALU not only outputs the results of data operations, but also sets the status of the carry flag (CF) in some instructions.



## Timer

The HTG1390 contains a programmable 8-bit count-up counter which can be used as a clock to generate an accurate time base.

The Timer may be set and read with software instructions and stopped by a hardware reset or a **TIMER OFF** instruction. To restart the timer load the counter with the value **XXH** and then issue a **TIMER ON** instruction. Note that **XX** is the desired start count immediate value of the 8 bits. Once the Timer/Counter is started it increments to a maximum count of **FFH** and then overflows to zero (**00H**). It then continues to count until stopped by a **TIMER OFF** instruction or a reset.

The increment from the maximum count of **FFH** to a zero (**00H**) triggers a timer flag **TF** and an internal interrupt request. The interrupt may be enabled or disabled by executing the **EI** and **DI** instruction. If the interrupt is enabled the timer overflow will cause a subroutine call to location 4. The state of the timer flag is also testable with the conditional jump instruction **JTMR**. The timer flag is cleared after the interrupt or the **JTMR** instruction is executed.

If an internal source is used the frequency is determined by the system clock and the parameter **n** as defined in the equation. The frequency of the internal frequency source can be selected by mask option.

$$\text{Frequency of TIMER clock} = \frac{\text{system clock}}{2^n}$$

where **n**=0,1,2 ...13 selectable by mask option.

Note that **n** cannot have the value of 6, which is reserved for internal use.

## Interrupt

The HTG1390 provides both internal and external interrupt modes. The **DI** and **EI** instructions are used to disable and enable the interrupts. During **HALT** mode, if the **PP** or **PS** input pin is triggered on a high to low transition in the enable interrupt mode and the program is not within a **CALL** subroutine, the external interrupt is activated. This causes a subroutine call to location 8 and resets the interrupt latch.

Likewise when the timer flag is set in the enable interrupt mode and the program is not within a **CALL** subroutine the internal interrupt is activated. This causes a subroutine call to location 4 and resets the timer flag.

When running under a **CALL** subroutine or **DI** the interrupt acknowledge is on hold until the **RET** or **EI** instruction is invoked. The **CALL** instruction should not be used within an interrupt routine as unpredictable behaviour may occur. If within a **CALL** subroutine internal interrupt occur, the internal interrupt will be serviced after leaving the **CALL** subroutine.

The interrupts are disabled by a hardware reset or a **DI** instruction. They remain disabled until the **EI** instruction is executed.

Each input port pin can be programmed by mask option to have an external interrupt function in the **HALT** mode.

## Initial reset

The HTG1390 provides an **RES** pin for system initialization. This pin is equipped with an internal pull-high resistor and in combination with an external 0.1μF~1μF capacitor, provides an internal reset pulse of sufficient length to guarantee a reset to all internal circuits. If the reset pulse is generated externally, the **RES** pin must be held low for at least 5ms. Normal circuit operation will not commence until the **RES** pin returns high.

The reset performs the following functions:

<b>PC</b>	<b>000H</b>
<b>TIMER</b>	Stop
<b>Time flag</b>	Reset (Low)
<b>SOUND</b>	Sound off and one sing mode
<b>Output Port A</b>	High (or floating state)
<b>Interrupt</b>	Disabled
<b>BZ and <math>\overline{\text{BZ}}</math> output</b>	Low level

## HALT

This is a special feature of the HTG1390. It will stop the chip's normal operation and reduce power consumption. When the instruction "HALT" is executed, then

- The system clock will be stopped
- The contents of the on-chip RAM and registers remain unchanged
- LCD segments and commons keep 2VDD voltage (i.e. LCD becomes blank)

The system can escape HALT mode by ways of initial reset or external interrupt and wake-up from the following entry of program counter value.

- Initial reset: 000H.
- Interrupt (enabled): 008H
- Interrupt (disabled): next address of HALT instruction.

In HALT mode, each bit of port PS, PP0~PP2, can be used as external interrupt by mask option to wake-up system. This signal is active in low-going transition.

## Sound effects

The HTG1390 includes sound effect circuitry which offers up to 16 sounds with 3 tone, boom and noise effects. Holtek supports a sound library which has melodies, alarms, machine guns etc..

Whenever the instruction "SOUND n" or "SOUND A" is executed, the specified sound will begin. Whenever "SOUND OFF" is executed, it terminates the singing sound immediately.

There are two singing modes, SONE mode and SLOOP mode activated by SOUND ONE and SOUND LOOP. In SONE mode the specified sound plays just once. In SLOOP mode the specified sound keeps re-playing.

Since sounds 0~11 contain 32 notes and sounds 12~15 contain 64 notes the latter possesses better sound than the former.

The frequency of the sound effect circuit can be selected by mask option.

$$\text{Frequency of sound effect circuit} = \frac{\text{system clock}}{2^m}$$

...where m=0,1,2,3,4,5.

Holtek's sound library supports only sound clock frequencies of 128kHz or 64kHz. To use Holtek's sound library the proper system clock and mask option should be selected.

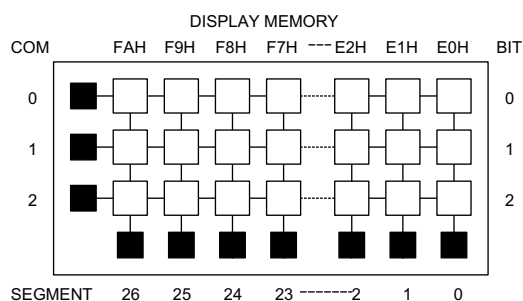
## LCD display memory

As mentioned in the data memory section the LCD display memory is embedded in the data memory. It can be read and written to in the same way as normal data memory.

The figures show the mapping between the display memory and LCD pattern for the HTG1390.

To turn the display on or off a 1/0 is written to the corresponding bit of the display memory.

The LCD display module may have any form as long as the number of commons does not exceed 3 and the number of segments does not exceed 27.



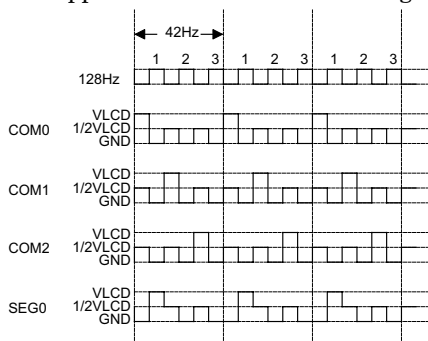
LCD display memory

### LCD driver output

All LCD segments are random after an initial clear. The bias voltage circuits of the LCD display is built-in and no external resistor is needed.

The output number of the HTG1390 LCD driver is 27×3 which can directly drive an LCD with 1/3 duty cycle and 1/2 bias.

The frequency of the LCD driving clock is fixed at about 128Hz. This is set by Holtek according to the application and cannot be changed.



LCD driver output

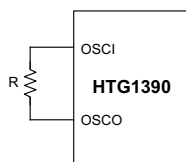
Note: VLCD is produced by double voltage circuit, therefore its value is double by VDD.

### Oscillator

Only one external resistor is needed for the HTG1390 oscillator circuit.

The system clock is also used as the reference signal of the LCD driving clock, sound effect clock and internal frequency source of TIMER.

One HTG1390 machine cycle consists of a sequence of 4 states numbered T1 to T4. Each state lasts for one oscillator period. The machine cycle is 16μs if the system frequency is up to 256kHz.



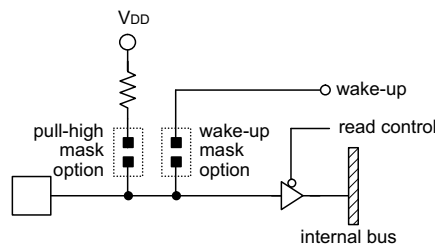
RC oscillator

### Interfacing

The HTG1390 microcontroller communicate with the outside world through 4-bit input port PS, 3-bit input port PP and one 4-bit output port PA.

### Input ports – PP, PS

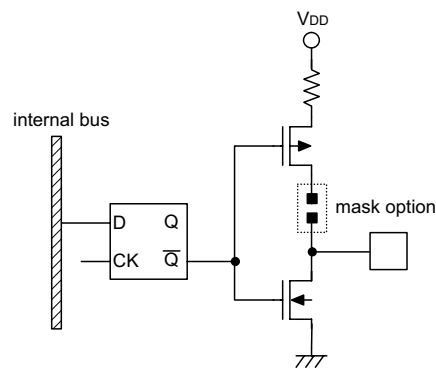
All ports can have internal pull-high resistors determined by mask option. Every bit of the input ports PP and PS can be specified to be a trigger source to wake up the HALT interrupt by mask option. A high to low transition on one of these pins will wake up the device from a HALT status.



Input ports PP and PS

### Output port – PA

A mask option is available to select whether the output is a CMOS or open drain NMOS type. After an initial clear the output port PA defaults to be high for CMOS or floating for NMOS.



Output port PA

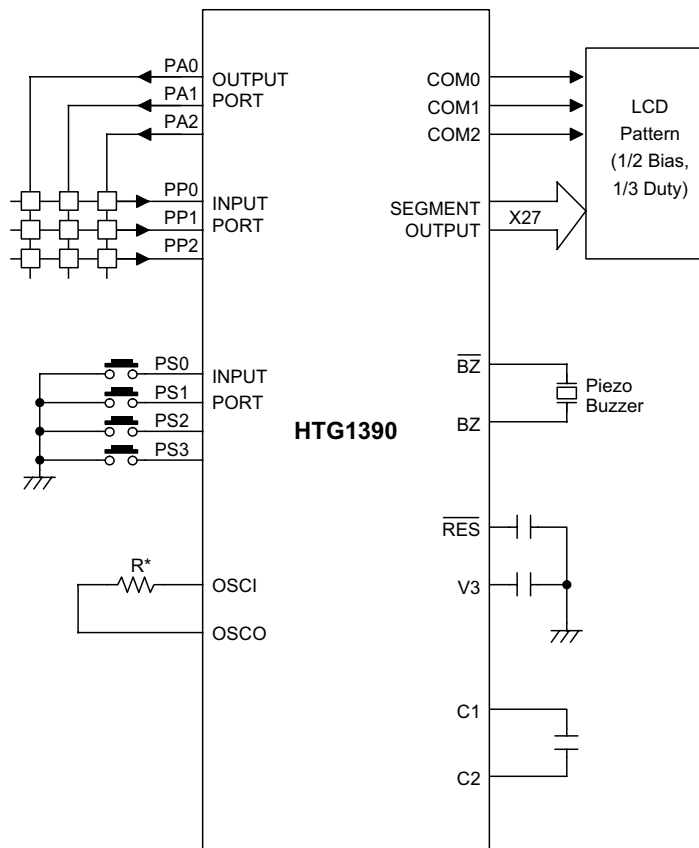
### Mask options

The following either/or options are available by mask option which the user must select prior to manufacture.

- 4-bit input ports PP and PS with or without pull-high resistors
- Each bit of PP and PS can wake up the processor from a HALT state

- Output Port PA to be CMOS or open drain NMOS
- 8-bit programmable timer with external clock or internal frequency source. Thirteen internal frequency sources are available to provide an internal clock. Note that a value of  $n=6$  cannot be used for the devices.
- Six kinds of sound clock frequency:  
 $f_{SYS}/2^m$ ,  $m=0, 1, 2, 3, 4, 5$

### Application Circuits



R\*: Depends on the required system clock frequency. ( $R=36k\Omega \sim 2M\Omega$ , at  $V_{DD}=1.5V$ )

**Instruction Set Summary**

Mnemonic	Description	Byte	Cycle	CF
<b>Arithmetic</b>				
ADD A,[R1R0]	Add data memory to ACC	1	1	✓
ADC A,[R1R0]	Add data memory with carry to ACC	1	1	✓
SUB A,[R1R0]	Subtract data memory from ACC	1	1	✓
SBC A,[R1R0]	Subtract data memory from ACC with borrow	1	1	✓
ADD A,XH	Add immediate data to ACC	2	2	✓
SUB A,XH	Subtract immediate data from ACC	2	2	✓
DAA	Decimal adjust ACC for addition	1	1	✓
<b>Logic Operation</b>				
AND A,[R1R0]	AND data memory to ACC	1	1	—
OR A,[R1R0]	OR data memory to ACC	1	1	—
XOR A,[R1R0]	Exclusive-OR data memory to ACC	1	1	—
AND [R1R0],A	AND ACC to data memory	1	1	—
OR [R1R0],A	OR ACC to data memory	1	1	—
XOR [R1R0],A	Exclusive-OR ACC to data memory	1	1	—
AND A,XH	AND immediate data to ACC	2	2	—
OR A,XH	OR immediate data to ACC	2	2	—
XOR A,XH	Exclusive-OR immediate data to ACC	2	2	—
<b>Increment &amp; Decrement</b>				
INC A	Increment ACC	1	1	—
INC Rn	Increment register, n=0~4	1	1	—
INC [R1R0]	Increment data memory	1	1	—
INC [R3R2]	Increment data memory	1	1	—
DEC A	Decrement ACC	1	1	—
DEC Rn	Decrement register, n=0~4	1	1	—
DEC [R1R0]	Decrement data memory	1	1	—
DEC [R3R2]	Decrement data memory	1	1	—
<b>Data Move</b>				
MOV A,Rn	Move register to ACC, n=0~4	1	1	—
MOV Rn,A	Move ACC to register, n=0~4	1	1	—
MOV A,[R1R0]	Move data memory to ACC	1	1	—
MOV A,[R3R2]	Move data memory to ACC	1	1	—
MOV [R1R0],A	Move ACC to data memory	1	1	—
MOV [R3R2],A	Move ACC to data memory	1	1	—
MOV A,XH	Move immediate data to ACC	1	1	—
MOV R1R0,XXH	Move immediate data to R1 and R0	2	2	—
MOV R3R2,XXH	Move immediate data to R3 and R2	2	2	—
MOV R4,XH	Move immediate data to R4	2	2	—

Mnemonic	Description	Byte	Cycle	CF
<b>Rotate</b>				
RL A	Rotate ACC left	1	1	√
RLC A	Rotate ACC left through the carry	1	1	√
RR A	Rotate ACC right	1	1	√
RRC A	Rotate ACC right through the carry	1	1	√
<b>Input &amp; Output</b>				
IN A,Pi	Input port-i to ACC ,port i=PS,PP	1	1	—
OUT PA,A	Output ACC to port-A	1	1	—
<b>Branch</b>				
JMP addr	Jump unconditionally	2	2	—
JC addr	Jump on carry=1	2	2	—
JNC addr	Jump on carry=0	2	2	—
JTMR addr	Jump on timer overflow	2	2	—
JAn addr	Jump on ACC bit n=1	2	2	—
JZ A,addr	Jump on ACC is zero	2	2	—
JNZ A,addr	Jump on ACC is not zero	2	2	—
JNZ Rn,addr	Jump on register Rn not zero, n=0,1,4	2	2	—
<b>Subroutine</b>				
CALL addr	Subroutine call	2	2	—
RET	Return from subroutine or interrupt	1	1	—
RETI	Return from interrupt service routine	1	1	√
<b>Flag</b>				
CLC	Clear carry flag	1	1	0
STC	Set carry flag	1	1	1
EI	Enable interrupt	1	1	—
DI	Disable interrupt	1	1	—
NOP	No operation	1	1	—
<b>Timer</b>				
TIMER XXH	Set 8 bits immediate data to TIMER	2	2	—
TIMER ON	Set TIMER start counting	1	1	—
TIMER OFF	Set TIMER stop counting	1	1	—
MOV A,TMRL	Move low nibble of TIMER to ACC	1	1	—
MOV A,TMRH	Move high nibble of TIMER to ACC	1	1	—
MOV TMRL,A	Move ACC to low nibble of TIMER	1	1	—
MOV TMRH,A	Move ACC to high nibble of TIMER	1	1	—
<b>Table Read</b>				
READ R4A	Read ROM code of current page to R4 & ACC	1	2	—
READ MR0A	Read ROM code of current page to M(R1,R0), ACC	1	2	—
READF R4A	Read ROM code of page F to R4 & ACC	1	2	—
READF MR0A	Read ROM code of page F to M(R1,R0),ACC	1	2	—

Mnemonic	Description	Byte	Cycle	CF
<b>Sound Control</b>				
SOUND n	Activate SOUND channel n	2	2	—
SOUND A	Activate SOUND channel with ACC	1	1	—
SOUND ONE	Turn on SOUND one cycle	1	1	—
SOUND LOOP	Turn on SOUND repeat cycle	1	1	—
SOUND OFF	Turn off SOUND	1	1	—
<b>Miscellaneous</b>				
HALT	Enter power down mode	2	2	—

## Instruction Definitions

<b>ADC A,[R1R0]</b>	Add data memory contents and carry to accumulator
Machine code	0 0 0 0 1 0 0 0
Description	The contents of the data memory addressed by the register pair "R1,R0" and the carry are added to the accumulator. Carry is affected.
Operation	$ACC \leftarrow ACC + M(R1,R0) + CF$
<b>ADD A,XH</b>	Add immediate data to accumulator
Machine code	0 1 0 0 0 0 0 0   0 0 0 0 d d d d
Description	The specified data is added to the accumulator. Carry is affected.
Operation	$ACC \leftarrow ACC + XH$
<b>ADD A,[R1R0]</b>	Add data memory contents to accumulator
Machine code	0 0 0 0 1 0 0 1
Description	The contents of the data memory addressed by the register pair "R1,R0" is added to the accumulator. Carry is affected.
Operation	$ACC \leftarrow ACC + M(R1,R0)$
<b>AND A,XH</b>	Logical AND immediate data to accumulator
Machine code	0 1 0 0 0 0 1 0   0 0 0 0 d d d d
Description	Data in the accumulator is logically ANDed with the immediate data specified by the code.
Operation	$ACC \leftarrow ACC \text{ "AND" } XH$
<b>AND A,[R1R0]</b>	Logical AND accumulator with data memory
Machine code	0 0 0 1 1 0 1 0
Description	Data in the accumulator is logically ANDed with the data memory addressed by the register pair "R1,R0".
Operation	$ACC \leftarrow ACC \text{ "AND" } M(R1,R0)$
<b>AND [R1R0],A</b>	Logical AND data memory with accumulator
Machine code	0 0 0 1 1 1 0 1
Description	Data in the data memory addressed by the register pair "R1,R0" is logically ANDed with the accumulator
Operation	$M(R1,R0) \leftarrow M(R1,R0) \text{ "AND" } ACC$



<b>CALL address</b>	Subroutine call
Machine code	1 1 1 1 a a a a a a a a a a
Description	The program counter bits 0~11 are saved in the stack and the specified address loaded into the program counter.
Operation	Stack $\leftarrow$ PC+2 PC $\leftarrow$ address
<b>CLC</b>	Clear carry flag
Machine code	0 0 1 0 1 0 1 0
Description	The carry flag is reset to zero.
Operation	CF $\leftarrow$ 0
<b>DAA</b>	Decimal-Adjust accumulator
Machine code	0 0 1 1 0 1 1 0
Description	The accumulator value is adjusted to BCD (Binary Code Decimal), if the contents of the accumulator is greater than 9 or CF (Carry flag) is one.
Operation	If ACC>9 or CF=1 then ACC $\leftarrow$ ACC+6, CF $\leftarrow$ 1 else ACC $\leftarrow$ ACC, CF $\leftarrow$ CF
<b>DEC A</b>	Decrement accumulator
Machine code	0 0 1 1 1 1 1 1
Description	Data in the accumulator is decremented by one. Carry flag is not affected.
Operation	ACC $\leftarrow$ ACC-1
<b>DEC Rn</b>	Decrement register
Machine code	0 0 0 1 n n n 1
Description	Data in the working register "Rn" is decremented by one. Carry flag is not affected.
Operation	Rn $\leftarrow$ Rn-1; Rn=R0,R1,R2,R3,R4, for nnn=0,1,2,3,4
<b>DEC [R1R0]</b>	Decrement data memory
Machine code	0 0 0 0 1 1 0 1
Description	Data in the data memory specified by the register pair "R1,R0" is decremented by one. Carry flag is not affected.
Operation	M(R1,R0) $\leftarrow$ M(R1,R0)-1

<b>DEC [R3R2]</b>	Decrement data memory
Machine code	0 0 0 0 1 1 1 1
Description	Data in the data memory specified by the register pair "R3,R2" is decremented by one. Carry flag is not affected.
Operation	$M(R3,R2) \leftarrow M(R3,R2)-1$
<b>DI</b>	Disable interrupt
Machine code	0 0 1 0 1 1 0 1
Description	Internal time-out interrupt and external interrupt are disabled.
<b>EI</b>	Enable interrupt
Machine code	0 0 1 0 1 1 0 0
Description	Internal time-out interrupt and external interrupt are enabled.
<b>HALT</b>	HALT system clock
Machine code	0 0 1 1 0 1 1 1    0 0 1 1 1 1 1 0
Description	Turn off system clock, and enter power down mode.
Operation	$PC \leftarrow PC+2$
<b>IN A,Pi</b>	Input port to accumulator
Machine code	PS    0 0 1 1 0 0 1 1 PP    0 0 1 1 0 1 0 0
Description	The data on port "Pi" is transferred to the accumulator.
Operation	$ACC \leftarrow Pi; Pi=PS \text{ or } PP$
<b>INC A</b>	Increment accumulator
Machine code	0 0 1 1 0 0 0 1
Description	Data in the accumulator is incremented by one. Carry flag is not affected.
Operation	$ACC \leftarrow ACC+1$
<b>INC Rn</b>	Increment register
Machine code	0 0 0 1 n n n 0
Description	Data in the working register "Rn" is incremented by one. Carry flag is not affected.
Operation	$Rn \leftarrow Rn+1; Rn=R0 \sim R4 \text{ for } nnn=0 \sim 4$
<b>INC [R1R0]</b>	Increment data memory
Machine code	0 0 0 0 1 1 0 0
Description	Data in the data memory specified by the register pair "R1,R0" is incremented by one. Carry flag is not affected.
Operation	$M(R1,R0) \leftarrow M(R1,R0)+1$

<b>INC [R3R2]</b>	Increment data memory
Machine code	0 0 0 0 1 1 1 0
Description	Data memory specified by the register pair "R3,R2" is incremented by one. Carry flag is not affected.
Operation	$M(R3,R2) \leftarrow M(R3,R2)+1$
<b>JAn address</b>	Jump if accumulator bit n is set
Machine code	1 0 0 n n a a a a a a a a a a
Description	Bits 0~10 of the program counter are replaced with the directly-specified address but bit 11 of the program counter is unaffected, if accumulator bit n is set to one.
Operation	PC (bit 0~10) $\leftarrow$ address, if ACC bit n=1(n=0~3) PC $\leftarrow$ PC+2, if ACC bit n=0
<b>JC address</b>	Jump if carry is set
Machine code	1 1 0 0 0 a a a a a a a a a a
Description	Bits 0~10 of the program counter are replaced with the directly-specified address but bit 11 of the program counter is unaffected, if the CF (Carry flag) is set to one.
Operation	PC (bit 0~10) $\leftarrow$ address, if CF=1 PC $\leftarrow$ PC+2, if CF=0
<b>JMP address</b>	Direct jump
Machine code	1 1 1 0 a a a a a a a a a a
Description	Bits 0~11 of the program counter are replaced with the directly-specified address.
Operation	PC $\leftarrow$ address
<b>JNC address</b>	Jump if carry is not set
Machine code	1 1 0 0 1 a a a a a a a a a a
Description	Bits 0~10 of the program counter are replaced with the directly-specified address and bit 11 of the program counter is unaffected, if the CF (Carry flag) is set to zero.
Operation	PC (bit 0~10) $\leftarrow$ address, if CF=0 PC $\leftarrow$ PC+2, if CF=1
<b>JNZ A,address</b>	Jump if accumulator is not zero
Machine code	1 0 1 1 1 a a a a a a a a a a
Description	Bits 0~10 of the program counter are replaced with the directly-specified address but bit 11 of the program counter is unaffected, if the accumulator is not zero.
Operation	PC (bit 0~10) $\leftarrow$ address, if ACC $\neq$ 0 PC $\leftarrow$ PC+2, if ACC=0

<b>JNZ Rn,address</b>	Jump if register is not zero
Machine code	R0 1 0 1 0 0 a a a a a a a a a a R1 1 0 1 0 1 a a a a a a a a a a R4 1 1 0 1 1 a a a a a a a a a a
Description	Bits 0~10 of the program counter are replaced with the directly-specified address but bit 11 of the program counter is unaffected, if the register is not zero.
Operation	PC (bit 0~10) ← address, if Rn≠0; Rn=R0,R1,R4 PC ← PC+2, if Rn=0
<b>JTMR address</b>	Jump if time-out
Machine code	1 1 0 1 0 a a a a a a a a a a
Description	Bits 0~10 of the program counter are replaced with the directly-specified address but bit 11 of the program counter is unaffected, if the TF (Timer flag) is set to one.
Operation	PC (bit 0~10) ← address, if TF=1 PC ← PC+2, if TF=0
<b>JZ A,address</b>	Jump if accumulator is zero
Machine code	1 0 1 1 0 a a a a a a a a a a
Description	Bits 0~10 of the program counter are replaced with the directly-specified address but bit 11 of the program counter is unaffected, if the accumulator is zero.
Operation	PC (bit 0~10) ← address, if ACC=0 PC ← PC+2, if ACC≠0
<b>MOV A,Rn</b>	Move register to accumulator
Machine code	0 0 1 0 n n n 1
Description	Data in the working register "Rn" is moved to the accumulator.
Operation	ACC ← Rn; Rn=R0~R4, for nnn=0~4
<b>MOV A,TMRH</b>	Move timer high nibble to accumulator
Machine code	0 0 1 1 1 0 1 1
Description	The high nibble data of the timer counter is loaded to the accumulator.
Operation	ACC ← TIMER (high nibble)
<b>MOV A,TMRL</b>	Move timer low nibble to accumulator
Machine code	0 0 1 1 1 0 1 0
Description	The low nibble data of the timer counter is loaded to the accumulator.
Operation	ACC ← TIMER (low nibble)

<b>MOV A,XH</b>	Move immediate data to accumulator
Machine code	0 1 1 1 d d d d
Description	The 4-bit data specified by the code is loaded to the accumulator.
Operation	$ACC \leftarrow XH$
<b>MOV A,[R1R0]</b>	Move data memory to accumulator
Machine code	0 0 0 0 0 1 0 0
Description	Data in the data memory specified by the register pair "R1,R0" is moved to the accumulator.
Operation	$ACC \leftarrow M(R1,R0)$
<b>MOV A,[R3R2]</b>	Move data memory to accumulator
Machine code	0 0 0 0 0 1 1 0
Description	Data in the data memory specified by the register pair "R3,R2" is moved to the accumulator.
Operation	$ACC \leftarrow M(R3,R2)$
<b>MOV R1R0,XXH</b>	Move immediate data to R1 and R0
Machine code	0 1 0 1 d d d d    0 0 0 0 d d d d
Description	The 8-bit data specified by the code is loaded to the working registers R1 and R0, the high nibble of the data is loaded to R1, and the low nibble to R0.
Operation	$R1 \leftarrow XH$ (high nibble) $R0 \leftarrow XH$ (low nibble)
<b>MOV R3R2,XXH</b>	Move immediate data to R3 and R2
Machine code	0 1 1 0 d d d d    0 0 0 0 d d d d
Description	The 8-bit data specified by the code is loaded to the working registers R3 and R2, the high nibble of the data is loaded to R3, and the low nibble to R2.
Operation	$R3 \leftarrow XH$ (high nibble) $R2 \leftarrow XH$ (low nibble)
<b>MOV R4,XH</b>	Move immediate data to R4
Machine code	0 1 0 0 0 1 1 0    0 0 0 0 d d d d
Description	The 4-bit data specified by the code is loaded to the working register R4.
Operation	$R4 \leftarrow XH$
<b>MOV Rn,A</b>	Move accumulator to register
Machine code	0 0 1 0 n n n 0
Description	Data in the accumulator is moved to the working register "Rn".
Operation	$Rn \leftarrow ACC$ ; $Rn=R0\sim R4$ , for $nnn=0\sim 4$

<b>MOV TMRH,A</b>	Move accumulator to timer high nibble
Machine code	0 0 1 1 1 1 0 1
Description	The contents of the accumulator is loaded to the high nibble of the timer counter.
Operation	TIMER(high nibble) ← ACC
<b>MOV TMRL,A</b>	Move accumulator to timer low nibble
Machine code	0 0 1 1 1 1 0 0
Description	The contents of the accumulator is loaded to the low nibble of the timer counter.
Operation	TIMER(low nibble) ← ACC
<b>MOV [R1R0],A</b>	Move accumulator to data memory
Machine code	0 0 0 0 0 1 0 1
Description	Data in the accumulator is moved to the data memory specified by the register pair "R1,R0".
Operation	M(R1,R0) ← ACC
<b>MOV [R3R2],A</b>	Move accumulator to data memory
Machine code	0 0 0 0 0 1 1 1
Description	Data in the accumulator is moved to the data memory specified by the register pair "R3,R2".
Operation	M(R3,R2) ← ACC
<b>NOP</b>	No operation
Machine code	0 0 1 1 1 1 1 0
Description	Do nothing, but one instruction cycle is delayed.
<b>OR A,XH</b>	Logical OR immediate data to accumulator
Machine code	0 1 0 0 1 0 0    0 0 0 0 d d d d
Description	Data in the accumulator is logically ORed with the immediate data specified by the code.
Operation	ACC ← ACC "OR" XH
<b>OR A,[R1R0]</b>	Logical OR accumulator with data memory
Machine code	0 0 0 1 1 1 0 0
Description	Data in the accumulator is logically ORed with the data memory addressed by the register pair "R1,R0".
Operation	ACC ← ACC "OR" M(R1,R0)

<b>OR [R1R0],A</b>	Logically OR data memory with accumulator
Machine code	0 0 0 1 1 1 1 1
Description	Data in the data memory addressed by the register pair "R1,R0" is logically ORed with the accumulator.
Operation	$M(R1,R0) \leftarrow M(R1,R0) \text{ "OR" } ACC$
<b>OUT PA,A</b>	Output accumulator data to port A
Machine code	0 0 1 1 0 0 0 0
Description	The data in the accumulator is transferred to port PA and latched.
Operation	$PA \leftarrow ACC$
<b>READ MR0A</b>	Read ROM code of current page to M(R1,R0) and ACC
Machine code	0 1 0 0 1 1 1 0
Description	<p>The 8-bit ROM code (current page) addressed by ACC and R4 is moved to the data memory M(R1,R0) and the accumulator. The high nibble of the ROM code is loaded to M(R1,R0) and the low nibble of the ROM code is loaded to the accumulator. The address of the ROM code is specified as below:</p> <p>Current page → ROM code address bit 11~8  ACC → ROM code address bit 7~4  R4 → ROM code address bit 3~0</p>
Operation	$M(R1,R0) \leftarrow \text{ROM code (high nibble)}$ $ACC \leftarrow \text{ROM code (low nibble)}$
<b>READ R4A</b>	Read ROM code of current page to R4 and accumulator
Machine code	0 1 0 0 1 1 0 0
Description	<p>The 8-bit ROM code (current page) addressed by ACC and M(R1,R0) is moved to the working register R4 and the accumulator. The high nibble of the ROM code is loaded to R4 and the low nibble of the ROM code is loaded to the accumulator. The address of the ROM code is specified as below:</p> <p>Current page → ROM code address bit 11~8  ACC → ROM code address bit 7~4  M(R1,R0) → ROM code address bit 3~0</p>
Operation	$R4 \leftarrow \text{ROM code (high nibble)}$ $ACC \leftarrow \text{ROM code (low nibble)}$

<b>READF MR0A</b>	Read ROM Code of page F to M(R1,R0) and ACC
Machine code	0 1 0 0 1 1 1 1
Description	The 8-bit ROM code (page F) addressed by ACC and R4 is moved to the data memory M(R1,R0) and the accumulator. The high nibble of the ROM code is loaded to M(R1,R0) and the low nibble of the ROM code is loaded to the accumulator. Page F → ROM code address bit 11~8 are "1111" ACC → ROM code address bit 7~4 R4 → ROM code address bit 3~0
Operation	M(R1,R0) ← high nibble of ROM code (page F) ACC ← low nibble of ROM code (page F)
<b>READF R4A</b>	Read ROM code of page F to R4 and accumulator
Machine code	0 1 0 0 1 1 0 1
Description	The 8-bit ROM code (page F) addressed by ACC and M(R1,R0) is moved to the working register R4 and the accumulator. The high nibble of the ROM code is loaded to R4 and the low nibble of the ROM code is loaded to the accumulator. Page F → ROM code address bit 11~8 are "1111" ACC → ROM code address bit 7~4 M(R1,R0) → ROM code address bit 3~0
Operation	R4 ← high nibble of ROM code (page F) ACC ← low nibble of ROM code (page F)
<b>RET</b>	Return from subroutine or interrupt
Machine code	0 0 1 0 1 1 1 0
Description	The program counter bits 0~11 are restored from the stack.
Operation	PC ← Stack
<b>RETI</b>	Return from interrupt subroutine
Machine code	0 0 1 0 1 1 1 1
Description	The program counter bits 0~11 are restored from the stack. The carry flag before entering the interrupt service routine is restored.
Operation	PC ← Stack CF ← CF (before interrupt service routine)
<b>RL A</b>	Rotate accumulator left
Machine code	0 0 0 0 0 0 0 1
Description	The contents of the accumulator are rotated left one bit. Bit 3 is rotated to both bit 0 and the carry flag.
Operation	A <sub>n+1</sub> ← A <sub>n</sub> , A <sub>n</sub> : accumulator bit n (n=0,1,2) A0 ← A3 CF ← A3



<b>RLC A</b>	Rotate accumulator left through carry
Machine code	0 0 0 0 0 1 1
Description	The contents of the accumulator are rotated left one bit. Bit 3 replaces the carry bit, which is rotated into the bit 0 position.
Operation	$A_{n+1} \leftarrow A_n$ , $A_n$ : Accumulator bit $n$ ( $n=0,1,2$ ) $A_0 \leftarrow CF$ $CF \leftarrow A_3$
<b>RR A</b>	Rotate accumulator right
Machine code	0 0 0 0 0 0 0
Description	The contents of the accumulator are rotated right one bit. Bit 0 is rotated to both bit 3 and the carry flag.
Operation	$A_n \leftarrow A_{n+1}$ , $A_n$ : Accumulator bit $n$ ( $n=0,1,2$ ) $A_3 \leftarrow A_0$ $CF \leftarrow A_0$
<b>RRC A</b>	Rotate accumulator right through carry
Machine code	0 0 0 0 0 1 0
Description	The contents of the accumulator are rotated right one bit. Bit 0 replaces the carry bit, which bit is rotated into the bit 3 position.
Operation	$A_n \leftarrow A_{n+1}$ , $A_n$ : Accumulator bit $n$ ( $n=0,1,2$ ) $A_3 \leftarrow CF$ $CF \leftarrow A_0$
<b>SBC A,[R1R0]</b>	Subtract data memory contents and carry from ACC
Machine code	0 0 0 0 1 0 1 0
Description	The contents of the data memory addressed by the register pair "R1,R0" and the complement of the carry are subtracted from the accumulator. Carry is set if a borrow does not take place in subtraction; otherwise carry is cleared.
Operation	$ACC \leftarrow ACC + \overline{M(R1,R0)} + CF$
<b>SOUND A</b>	Activate SOUND channel with accumulator
Machine code	0 1 0 0 1 0 1 1
Description	The activated sound begins playing in accordance with the contents of accumulator when the specified sound channel is matched.
<b>SOUND LOOP</b>	Turn on sound repeat cycle
Machine code	0 1 0 0 1 0 0 1
Description	The activated sound plays repeatedly.

<b>SOUND OFF</b>	Turn off sound
Machine code	0 1 0 0 1 0 1 0
Description	The activated sound will terminate immediately.
<b>SOUND ONE</b>	Turn on sound one cycle
Machine code	0 1 0 0 1 0 0 0
Description	The activated sound plays once.
<b>SOUND n</b>	Activate SOUND channel n
Machine code	0 1 0 0 0 1 0 1    0 0 0 0 n n n n
Description	The specified sound begins playing and overwrites the previous activated sound. (nnnn=0~15)
<b>STC</b>	Set carry flag
Machine code	0 0 1 0 1 0 1 1
Description	The carry flag is set to one.
Operation	$CF \leftarrow 1$
<b>SUB A,XH</b>	Subtract immediate data from accumulator
Machine code	0 1 0 0 0 0 0 1    0 0 0 0 d d d d
Description	The specified data is subtracted from the accumulator. Carry is set if a borrow does not take place in subtraction; otherwise carry is cleared.
Operation	$ACC \leftarrow ACC + \overline{XH} + 1$
<b>SUB A,[R1R0]</b>	Subtract data memory contents from accumulator
Machine code	0 0 0 0 1 0 1 1
Description	The contents of the data memory addressed by the register pair "R1,R0" is subtracted from the accumulator. Carry is set if a borrow does not take place in subtraction; otherwise carry is cleared.
Operation	$ACC \leftarrow ACC + \overline{M(R1,R0)} + 1$
<b>TIMER OFF</b>	Set timer stop counting
Machine code	0 0 1 1 1 0 0 1
Description	The timer stops counting, when the "TIMER OFF" instruction is executed.
<b>TIMER ON</b>	Set timer start counting
Machine code	0 0 1 1 1 0 0 0
Description	The timer starts counting, when the "TIMER ON" instruction is executed.

<b>TIMER XXH</b>	Set immediate data to timer counter
Machine code	0 1 0 0 0 1 1 1    d d d d d d d d
Description	The 8-bit data specified by the code is loaded to the timer counter.
Operation	TIMER $\leftarrow$ XXH
<b>XOR A,XH</b>	Logical XOR immediate data to accumulator
Machine code	0 1 0 0 0 0 1 1    0 0 0 0 d d d d
Description	Data in the accumulator is Exclusive-ORed with the immediate data specified by the code.
Operation	ACC $\leftarrow$ ACC "XOR" XH
<b>XOR A,[R1R0]</b>	Logical XOR accumulator with data memory
Machine code	0 0 0 1 1 0 1 1
Description	Data in the accumulator is Exclusive-ORed with the data memory addressed by the register pair "R1,R0".
Operation	ACC $\leftarrow$ ACC "XOR" M(R1,R0)
<b>XOR [R1R0],A</b>	Logical XOR data memory with accumulator
Machine code	0 0 0 1 1 1 1 0
Description	Data in the data memory addressed by the register pair "R1,R0" is logically Exclusive-ORed with the accumulator.
Operation	M(R1,R0) $\leftarrow$ M(R1,R0) XOR ACC

**Holtek Semiconductor Inc. (Headquarters)**

No.3 Creation Rd. II, Science-based Industrial Park, Hsinchu, Taiwan, R.O.C.

Tel: 886-3-563-1999

Fax: 886-3-563-1189

**Holtek Semiconductor Inc. (Taipei Office)**

11F, No.576, Sec.7 Chung Hsiao E. Rd., Taipei, Taiwan, R.O.C.

Tel: 886-2-2782-9635

Fax: 886-2-2782-9636

Fax: 886-2-2782-7128 (International sales hotline)

**Holtek Semiconductor (Hong Kong) Ltd.**

RM.711, Tower 2, Cheung Sha Wan Plaza, 833 Cheung Sha Wan Rd., Kowloon, Hong Kong

Tel: 852-2-745-8288

Fax: 852-2-742-8657

**Holtek Semiconductor (Shanghai) Ltd.**

7th Floor, Building 2, No.889, Yi Shan Road, Shanghai, China

Tel: 021-6485-5560

Fax: 021-6485-0313

**Holmate Technology Corp.**

48531 Warm Springs Boulevard, Suite 413, Fremont, CA 94539

Tel: 510-252-9880

Fax: 510-252-9885

Copyright © 2000 by HOLTEK SEMICONDUCTOR INC.

The information appearing in this Data Sheet is believed to be accurate at the time of publication. However, Holtek assumes no responsibility arising from the use of the specifications described. The applications mentioned herein are used solely for the purpose of illustration and Holtek makes no warranty or representation that such applications will be suitable without further modification, nor recommends the use of its products for application that may present a risk to human life due to malfunction or otherwise. Holtek reserves the right to alter its products without prior notification. For the most up-to-date information, please visit our web site at <http://www.holtek.com.tw>.