查询HT49C10供应商

捷多邦，专业PCB打样工厂，24小时加急出货

HOLTEK

HT49C10
8-Bit Microcontroller

Features

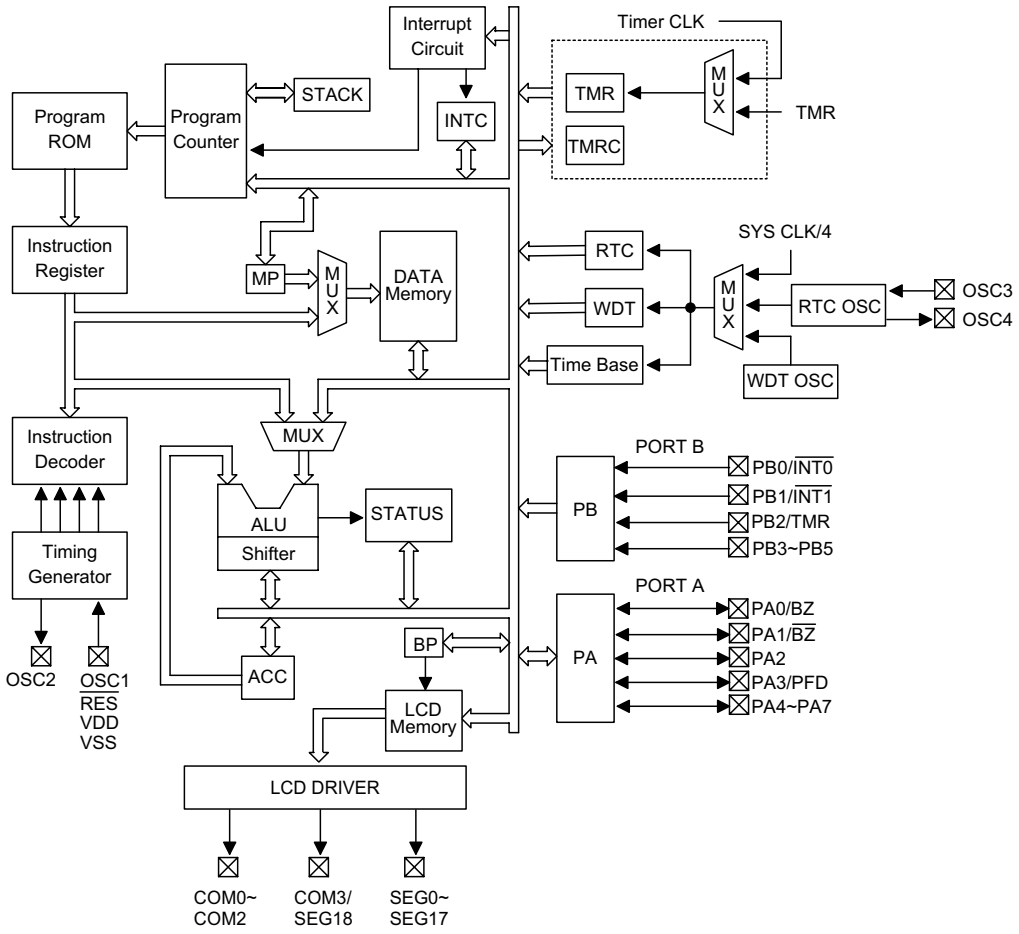
- Operating voltage: 2.2V~5.2V
- Eight bidirectional I/O lines
- Six input lines
- Two external interrupt input
- 8-bit programmable timer/event counter with PFD (programmable frequency divider)
- Watchdog timer
- On-chip crystal and RC oscillator
- 1K×14 program memory ROM
- 64×8 data memory RAM
- Real Time Clock (RTC)
- 8-bit prescaler for RTC
- Buzzer output
- Halt function and wake-up feature reduce power consumption
- LCD driver with 19×3 or 18×4 segments
- 4-level subroutine nesting
- Bit manipulation instruction
- 14-bit table read instruction
- Up to 1μs instruction cycle with 4MHz system clock
- 63 powerful instructions
- All instructions in 1 or 2 machine cycles
- 48-pin SSOP package

General Description

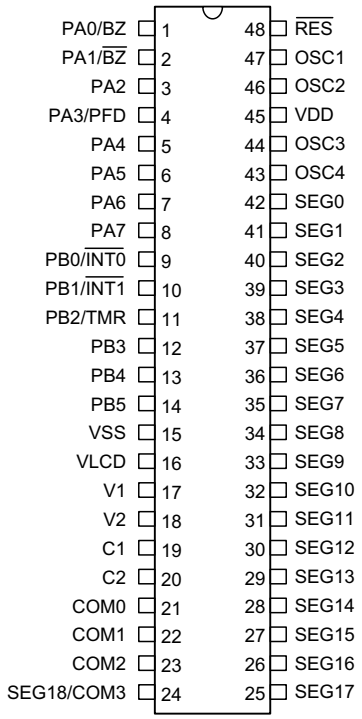
The HT49C10 is an 8-bit high performance single chip microcontroller. Its single-cycle instruction and two-stage pipeline architecture make it suitable for high speed applications. The device is also suited for multiple LCD low power applications among which are calculators, clock timers, games, scales, leisure products, other hand held LCD products, and battery systems in particular.

找PDF 维库一下 pdf.dzsc.com

Block Diagram

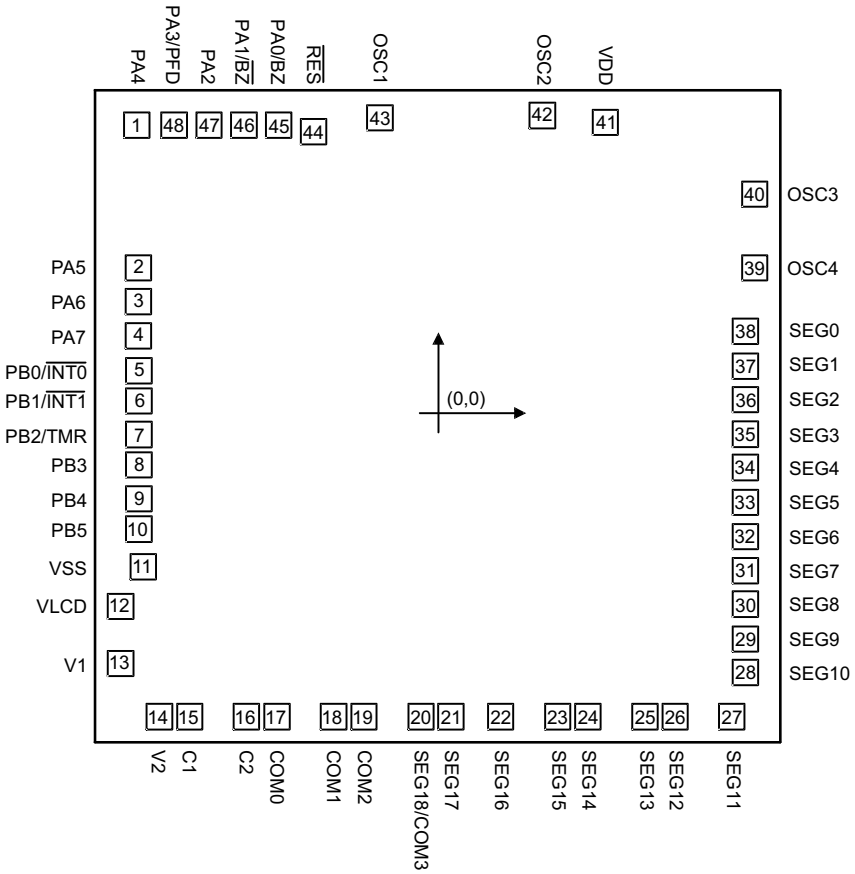


Pin Assignment



HT49C10
– 48 SSOP

Pad Assignment



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

Pad Description

Pad No.	Pad Name	I/O	Mask Option	Description
45 46 47 48 1~4	PA0/BZ PA1/$\overline{BZ}$ PA2 PA3/PFD PA4~PA7	I/O	Wake-up Pull-high or None CMOS or NMOS	PA0~PA7 constitute an 8-bit bidirectional input/ output port with Schmitt trigger input capability. Each bit on port can be configured as wake-up input by mask option. PA0~PA3 can be configured as CMOS (output) or NMOS (input/output) and with or without pull-high resistor by mask option, PA4~PA7 always pull-high and NMOS (input/output). Of the eight bits, PA0~PA1 can be set as I/O pins or buzzer outputs by mask option. PA3 can be set as an I/O pin or a PFD output also by mask option.
5 6 7 8~10	PB0/$\overline{INT0}$ PB1/$\overline{INT1}$ PB2/TMR PB3~PB5	I	—	PB0~PB5 constitute a 6-bit Schmitt trigger input port. Each bit on port are pull-high resistor. Of the six bits, PB0 can be set as input pin or external interrupt control pin ($\overline{INT0}$) by software application. PB1 can be set as input pin or an external interrupt control pin ($\overline{INT1}$) by software application. While PB2 can be set as input pin or timer/event counter input pin also by software application.
11	VSS	I	—	Negative power supply, GND
12	VLCD	I	—	LCD power supply
13~16	V1,V2,C1,C2	I	—	Voltage pump
20 19~17	SEG18/COM3 COM2~COM0	O	1/3 or 1/4 Duty	SEG18 can be set as segment or common output driver for LCD panel by mask option. COM2~COM0 are outputs for LCD panel plate.
21~38	SEG17~SEG0	O	—	LCD driver outputs for LCD panel segments
39 40	OSC4 OSC3	O I	—	Real time clock oscillators
41	VDD	—	—	Positive power supply
42 43	OSC2 OSC1	O I	Crystal or RC	OSC1 and OSC2 are connected to an RC network or a crystal (by mask option) for the internal system clock. In the case of RC operation, OSC2 is the output terminal for 1/4 system clock.
44	$\overline{RES}$	I	—	Schmitt trigger reset input, active low

Absolute Maximum Ratings

Supply Voltage.....–0.3V to 5.5V

Input Voltage.....V_{SS}–0.3V to V_{DD}+0.3V

Storage Temperature.....–50°C to 125°C

Operating Temperature .....–25°C to 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

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	—	2.2	—	5.2	V
I_{DD1}	Operating Current (Crystal OSC)	3V	No load, f_{SYS}=4MHz	—	0.7	1.5	mA
		5V		—	2	3	mA
I_{DD2}	Operating Current (RC OSC)	3V	No load, f_{SYS}=2MHz	—	0.5	1	mA
		5V		—	1	2	mA
I_{STB1}	Standby Current (RTC Enable, LCD On)	3V	No load, system Halt	—	—	5	μA
		5V		—	—	10	μA
I_{STB2}	Standby Current (RTC Disable, LCD Off)	3V	No load, system Halt	—	—	1	μA
		5V		—	—	2	μA
V_{IL}	Input Low Voltage for I/O Ports	3V	—	0	—	0.9	V
		5V	—	0	—	1.5	V
V_{IH}	Input High Voltage for I/O Ports	3V	—	2.1	—	3	V
		5V	—	3.5	—	5	V
V_{IL1}	Input Low Voltage ($\overline{RES}$, $\overline{INT0}$, $\overline{INT1}$, TMR)	3V	$\overline{RES}$=0.5V_{DD} $\overline{INT0/1}$=0.3V_{DD} TMR=0.3V_{DD}	0	—	1.5/0.9	V
		5V		0	—	2.5/1.5	V
V_{IH1}	Input High Voltage ($\overline{RES}$, $\overline{INT0}$, $\overline{INT1}$, TMR)	3V	0.8V_{DD}	2.4	—	3	V
		5V		4.0	—	5	V
I_{OL}	I/O Ports Sink Current	3V	V_{OL}=0.3V	1.5	2.5	—	mA
		5V	V_{OL}=0.5V	4	6	—	mA
I_{OH}	I/O Ports Source Current	3V	V_{OH}=2.7V	–1	–1.5	—	mA
		5V	V_{OH}=4.5V	–2	–3	—	mA
R_{PH}	Pull-high Resistance of I/O Ports and $\overline{INT0}$, $\overline{INT1}$	3V	—	40	60	80	kΩ
		5V	—	10	30	50	kΩ

A.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{SYS1}	System Clock (Crystal OSC)	3V	—	455	—	4000	kHz
		5V	—	455	—	4000	kHz
f_{SYS2}	System Clock (RC OSC)	3V	—	400	—	2000	kHz
		5V	—	400	—	3000	kHz
f_{TIMER}	Timer I/P Frequency (TMR)	3V	—	0	—	4000	kHz
		5V	—	0	—	4000	kHz
t_{WDTOSC}	Watchdog Oscillator	3V	—	45	90	180	μs
		5V	—	35	65	130	μs
t_{RES}	External Reset Low Pulse Width	—	—	1	—	—	μs
t_{SST}	System Start-up Timer Period	—	Power-up or wake-up from halt	—	1024	—	t_{SYS}
t_{INT}	Interrupt Pulse Width	—	—	1	—	—	μs

Note: $t_{SYS}=1/f_{SYS}$

Functional Description

Execution flow

The system clock is derived from either a crystal or an RC oscillator. It is internally divided into four non-overlapping clocks. One instruction cycle consists of four system clock cycles.

Instruction fetching and execution are pipelined in such a way that a fetch takes one instruction cycle while decoding and execution takes the next instruction cycle. The pipelining scheme causes each instruction to effectively execute in a cycle. If an instruction changes the value of the program counter, two cycles are required to complete the instruction.

Program counter – PC

The 10-bit program counter (PC) controls the sequence in which the instructions stored in the program ROM are executed. The contents of the PC can specify a maximum of 1024 addresses.

After accessing a program memory word to fetch an instruction code, the value of the PC is incremented by one. The PC then points to the memory word containing the next instruction code.

When executing a jump instruction, conditional skip execution, loading a PCL register, a subroutine call, an initial reset, an internal interrupt, an external interrupt, or returning from a subroutine, the PC manipulates program transfer by loading the address corresponding to each instruction.

The conditional skip is activated by instructions. Once the condition is met, the next instruction, fetched during the current instruction execution, is discarded and a dummy cycle replaces it to get a proper instruction; otherwise proceed with the next instruction.

The lower byte of the PC (PCL) is a readable and writeable register (06H). Moving data into the PCL performs a short jump. The destination is within 256 locations.

When a control transfer takes place, an additional dummy cycle is required.

Program memory – ROM

The program memory (ROM) is used to store the program instructions which are to be executed. It also contains data, table, and interrupt entries, and is organized into 1024×14 bits which are addressed by the PC and table pointer.

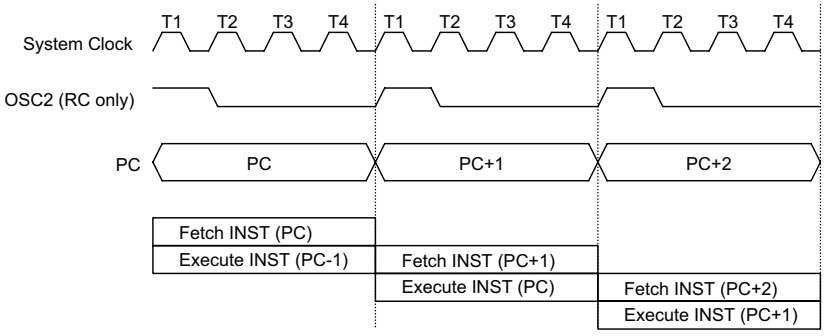
Certain locations in the ROM are reserved for special usage:

- Location 000H

  Location 000H is reserved for program initialization. After chip reset, the program always begins execution at this location.

- Location 004H

  Location 004H is reserved for the external interrupt service program. If the $\overline{\text{INT0}}$ input pin is activated, and the interrupt is enabled, and the stack is not full, the program begins execution at location 004H.



Execution flow

- Location 008H

  Location 008H is reserved for the external interrupt service program. If the $\overline{INT1}$ input pin is activated, and the interrupt is enabled, and the stack is not full, the program begins execution at location 008H.

- Location 00CH

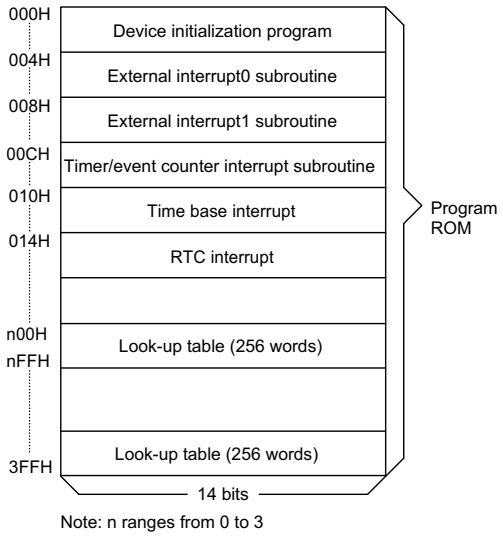
  Location 00CH is reserved for the timer/event counter interrupt service program. If a timer interrupt resulting from a timer/event counter overflow, and if the interrupt is enabled and the stack is not full, the program begins execution at location 00CH.

- Location 010H

  Location 010H is reserved for the time base interrupt service program. If a time base interrupt occurs, and the interrupt is enabled, and the stack is not full, the program begins execution at location 010H.

- Location 014H

  Location 014H is reserved for the real time clock interrupt service program. If a real time clock interrupt occurs, and the interrupt is enabled, and the stack is not full, the program begins execution at location 014H.



Program memory

- Table location

  Any location in the ROM can be used as a look-up table. The instructions "TABRDC [m]" (the current page, 1 page=256 words) and "TABRDL [m]" (the last page) transfer the con-

Mode	Program Counter									
	***9**	***8**	***7**	***6**	***5**	***4**	***3**	***2**	***1**	***0**
Initial reset	0	0	0	0	0	0	0	0	0	0
External interrupt 0	0	0	0	0	0	0	0	1	0	0
External interrupt 1	0	0	0	0	0	0	1	0	0	0
Timer/event Counter overflow	0	0	0	0	0	0	1	1	0	0
Time Base Interrupt	0	0	0	0	0	1	0	0	0	0
RTC Interrupt	0	0	0	0	0	1	0	1	0	0
Skip	PC+2									
Loading PCL	*9	*8	@7	@6	@5	@4	@3	@2	@1	@0
Jump, Call Branch	#9	#8	#7	#6	#5	#4	#3	#2	#1	#0
Return From Subroutine	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

Program counter

Note: *9~*0: Program counter bits
#9~#0: Instruction code bits
S9~S0: Stack register bits
@7~@0: PCL bits

tents of the lower-order byte to the specified data memory, and the contents of the higher-order byte to TBLH (table higher-order byte register) (08H). Only the destination of the lower-order byte in the table is well-defined; the other bits of the table word are all transferred to the lower portion of TBLH, and the remaining two bits are both read as "0". The TBLH is read only, and the table pointer (TBLP) is a read/write register (07H), indicating the table location. Before accessing the table, the location should be placed in TBLP. All the table related instructions require two cycles to complete the operation. These areas may function as a normal ROM depending upon the requirements.

Stack register – STACK

The stack register is a special part of the memory used to save the contents of the PC. The stack is organized into four levels and is neither part of the data nor part of the program, and is neither readable nor writeable. Its activated level is indexed by a stack pointer (SP) and is neither readable nor writeable. At a commencement of a subroutine call or an interrupt acknowledgment, the contents of the PC is pushed onto the stack. At the end of the subroutine or interrupt routine, signaled by a return instruction (RET or RETI), the contents of the PC is restored to its previous value from the stack. After chip reset, the SP will point to the top of the stack.

If the stack is full and a non-masked interrupt takes place, the interrupt request flag is recorded but the acknowledgment is still inhibited. Once the SP is decremented (by RET or RETI), the interrupt is serviced. This feature prevents stack overflow, allowing the programmer to use the structure easily. Likewise, if the stack is full, and a "CALL" is subsequently executed, a stack overflow occurs and the first entry is lost (only the most recent four return addresses are stored).

Data memory – RAM

The data memory (RAM) is designed with 81×8 bits, and is divided into two functional groups, namely special function registers and general purpose data memory, most of which are readable/writeable, although some are read only.

Of the two types of functional groups, the special function registers consist of an indirect addressing register 0 (00H), a memory pointer register 0 (MP0; 01H), an indirect addressing register 1 (02H), a memory pointer register 1 (MP1;03H), a bank pointer (BP;04H), an accumulator (ACC;05H), a program counter lower-order byte register (PCL;06H), a table pointer (TBLP;07H), a table higher-order byte register (TBLH;08H), a real time clock control register (RTCC;09H), a status register (STATUS;0AH), an interrupt control register 0 (INTC0;0BH), a timer/event counter (TMR;0DH), a timer/event counter control register (TMRC; 0EH), I/O registers (PA;12H, PB;14H), and interrupt control register 1 (INTC1;1EH). On the other hand, the general purpose data memory, addressed from 20H to 5FH, is used for data and control information under instruction commands.

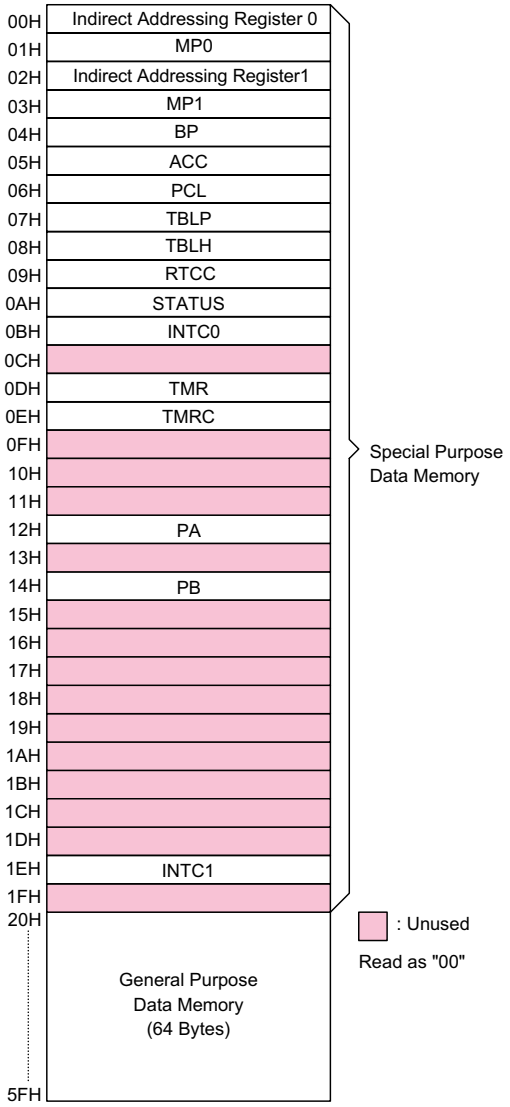
The areas in the RAM can directly handle arithmetic, logic, increment, decrement, and rotate operations. Except for some dedicated bits, each bit in the RAM can be set and reset by "SET [m].i" and "CLR [m].i". They are also indirectly accessible through the memory pointer

Instruction(s)	Table Location									
	*9	*8	*7	*6	*5	*4	*3	*2	*1	*0
TABRDC [m]	P9	P8	@7	@6	@5	@4	@3	@2	@1	@0
TABRDL [m]	1	1	@7	@6	@5	@4	@3	@2	@1	@0

Table location

Note: *9~*0: Table location bits
@7~@0: Table pointer bits
P9~P8: Current program counter bits

register 0 (MP0;01H) or the memory pointer register 1 (MP1;03H).



RAM mapping

Indirect addressing register

Location 00H and 02H are indirect addressing registers that are not physically implemented. Any read/write operation of [00H] and [02H] accesses the RAM pointed to by MP0 (01H) and MP1 (03H) respectively. Reading location 00H or 02H indirectly returns the result 00H, while, writing it leads to no operation.

The function of data movement between two indirect addressing registers is not supported. The memory pointer registers, MP0 and MP1, are both 7-bit registers used to access the RAM by combining corresponding indirect addressing registers. The bit 7 of MP0 and MP1 are undefined and reading will return the result 1 . Any writing operation to MP0 and MP1 will only transfer the lower 7-bit data.

Only MP0 can be applied to data memory, while MP1 can be applied to data memory and LCD display memory.

Accumulator – ACC

The accumulator (ACC) is related to ALU operations. It is also mapped to location 05H of the RAM and is capable of carrying out immediate data operations. The data movement between two data memory locations has to pass through the ACC.

Arithmetic and logic unit – ALU

This circuit performs 8-bit arithmetic and logic operations and provides the following functions:

- Arithmetic operations (ADD, ADC, SUB, SBC, DAA)
- Logic operations (AND, OR, XOR, CPL)
- Rotation (RL, RR, RLC, RRC)
- Increment and Decrement (INC, DEC)
- Branch decision (SZ, SNZ, SIZ, SDZ etc.)

The ALU not only saves the results of a data operation but also changes the status register.

Status register – STATUS

The status register (0AH) is 8-bit wide and contains a carry flag (C), an auxiliary carry flag (AC), a zero flag (Z), an overflow flag (OV), a power down flag (PD), and a watchdog time-out flag (TO). It also records the status information and controls the operation sequence.

Except for the TO and PD flags, bits in the status register can be altered by instructions, similar to other registers. Data written into the status reg-

Labels	Bits	Function
C	0	C is set if the operation results in a carry during an addition operation or if a borrow does not take place during a subtraction operation; otherwise C is cleared. C is also affected by a rotate through carry instruction.
AC	1	AC is set if the operation results in a carry out of the low nibbles in addition or no borrow from the high nibble into the low nibble in subtraction; otherwise AC is cleared.
Z	2	Z is set if the result of an arithmetic or logic operation is zero; otherwise Z is cleared.
OV	3	OV is set if the operation results in a carry into the highest-order bit but not a carry out of the highest-order bit, or vice versa; otherwise OV is cleared.
PD	4	PD is cleared by either a system power-up or executing the "CLR WDT" instruction. PD is set by executing the "HALT" instruction.
TO	5	TO is cleared by a system power-up or executing the "CLR WDT" or "HALT" instruction. TO is set by a WDT time-out.
—	6, 7	Undefined, read as "0"

STATUS register

ister does not alter the TO or PD flags. Operations related to the status register, however, may yield different results from those intended. The TO and PD flags can only be changed by a watchdog timer overflow, chip power-up, or clearing the watchdog timer and executing the "HALT" instruction. The Z, OV, AC, and C flags reflect the status of the latest operations.

On entering the interrupt sequence or executing the subroutine call, the status register will not be pushed onto the stack automatically. If the contents of the status is important, and if the subroutine is likely to corrupt the status register, precautions should be taken to save it properly.

Interrupts

The HT49C10 provides two external interrupts, an internal timer/event counter interrupt, an internal time base interrupt, and an internal real time clock interrupt. The interrupt control register 0 (INTC0;0BH) and interrupt control register 1 (INTC1;1EH) both contain the interrupt control bits that are used to set the enable/disable status and interrupt request flags.

Once an interrupt subroutine is serviced, other interrupts are all blocked (by clearing the EMI bit). This scheme may prevent any further interrupt nesting. Other interrupt requests may take place during this interval, but only the interrupt request flag will be recorded. If a certain interrupt requires servicing within the service routine, the programmer may set the EMI bit and the corresponding bit of INTC0 or of INTC1 in order to allow interrupt nesting. Once the stack is full, the interrupt request will not be acknowledged, even if the related interrupt is enabled, until the SP is decremented. If immediate service is desired, the stack should be prevented from becoming full.

All these interrupts have the wake-up capability. When an interrupt is serviced, a control transfer occurs by pushing the PC onto the stack, followed by a branch to subroutines at the specified locations in the ROM. Only the PC is pushed onto the stack. If the contents of the register or of the status register (STATUS) is altered by the interrupt service program which corrupts the desired control sequence, the contents should be saved first.

External interrupts are triggered by a high to low transition of $\overline{\text{INT0}}$ or $\overline{\text{INT1}}$, and the related

Register	Bit No.	Label	Function
INTC0 (0BH)	0	EMI	Control the master (global) interrupt (1= enabled; 0= disabled)
	1	EEI0	Control the external interrupt 0 (1= enabled; 0= disabled)
	2	EEI1	Control the external interrupt 1 (1= enabled; 0= disabled)
	3	ETI	Control the timer/event counter interrupt (1= enabled; 0= disabled)
	4	EIF0	External interrupt 0 request flag (1= active; 0= inactive)
	5	EIF1	External interrupt 1 request flag (1= active; 0= inactive)
	6	TF	Internal timer/event counter request flag (1= active; 0= inactive)
	7	—	Unused bit, read as ″0″
INTC1 (1EH)	0	ETBI	Control the time base interrupt (1= enabled; 0= disabled)
	1	ERTI	Control the real time clock interrupt (1= enabled; 0= disabled)
	2, 3	—	Unused bit, read as ″0″
	4	TBF	Time base request flag (1= active; 0= inactive)
	5	RTF	Real time clock request flag (1= active; 0= inactive)
	6, 7	—	Unused bit, read as ″0″

INTC register

interrupt request flag (EIF0; bit 4 of INTC0, EIF1; bit 5 of INTC0) is set as well. After the interrupt is enabled, and the stack is not full, and the external interrupt is active, a subroutine call to location 04H or 08H occurs. The interrupt request flag (EIF0 or EIF1) and EMI bits are all cleared to disable other interrupts.

The internal timer/event counter interrupt is initialized by setting the timer/event counter interrupt request flag (TF; bit 6 of INTC0), that is caused by a timer overflow. After the interrupt is enabled, and the stack is not full, and the TF bit is set, a subroutine call to location 0CH occurs. The related interrupt request flag (TF) is reset, and the EMI bit is cleared to disable further interrupts.

The time base interrupt is initialized by setting the time base interrupt request flag (TBF; bit 4 of INTC1), that is caused by a regular time base signal. After the interrupt is enabled, and the stack is not full, and the TBF bit is set, a subroutine call to location 10H occurs. The related interrupt request flag (TBF) is reset and the EMI bit is cleared to disable further interrupts.

The real time clock interrupt is initialized by setting the real time clock interrupt request flag (RTF; bit 5 of INTC1), that is caused by a

regular real time clock signal. After the interrupt is enabled, and the stack is not full, and the RTF bit is set, a subroutine call to location 14H occurs. The related interrupt request flag (RTF) is reset and the EMI bit is cleared to disable further interrupts.

During the execution of an interrupt subroutine, other interrupt acknowledgments are all held until the "RETI" instruction is executed or the EMI bit and the related interrupt control bit are set both to 1 (if the stack is not full). To return from the interrupt subroutine, "RET" or "RETI" may be invoked. RETI sets the EMI bit and enables an interrupt service, but RET does not.

Interrupts occurring in the interval between the rising edges of two consecutive T2 pulses are serviced on the latter of the two T2 pulses if the corresponding interrupts are enabled. In the case of simultaneous requests, the following table shows the priority that is applied. These can be masked by resetting the EMI bit.

No.	Interrupt Source	Priority	Vector
a	External interrupt 0	1	04H
b	External interrupt 1	2	08H
c	Timer/event counter overflow	3	0CH
d	Time base interrupt	4	10H
e	Real time clock interrupt	5	14H

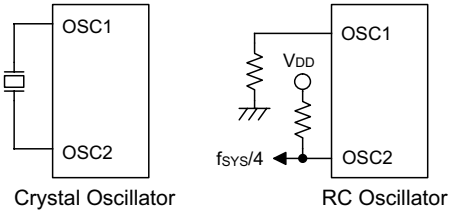
The timer/event counter interrupt request flag (TF), external interrupt 1 request flag (EIF1), external interrupt 0 request flag (EIF0), enable timer/event counter interrupt bit (ETI), enable external interrupt 1 bit (EEI1), enable external interrupt 0 bit (EEI0), and enable master interrupt bit (EMI) make up of the interrupt control register (INTC0) which is located at 0BH in the RAM. The real time clock interrupt request flag (RTF), time base interrupt request flag (TBF), enable real time clock interrupt bit (ERTI), and enable time base interrupt bit (ETBI), on the other hand, constitute the other interrupt control register (INTC1) which is located at 1EH in the RAM. EMI, EEI0, EEI1, ETI, ETBI, and ERTI are all used to control the enable/disable status of the interrupts. These bits prevent the requested interrupt from being serviced. Once the interrupt request flags (RTF, TBF, TF, EIF1, EIF0) are all set, they remain in the INTC1 or INTC0 respectively until the interrupts are serviced or cleared by a software instruction.

It is recommended that a program not use the "CALL subroutine" within the interrupt subroutine. It's because interrupts often occur in an unpredictable manner or require to be serviced immediately in some applications. At this time, if only one stack is left, and enabling the interrupt is not well controlled, operation of the "call subroutine" in the interrupt subroutine may damage the original control sequence.

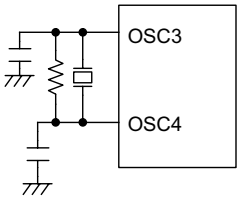
Oscillator configuration

The HT49C10 provides two oscillator circuits for system clocks, i.e., RC oscillator and crystal oscillator, determined by mask option. No matter what type of oscillator is selected, the signal is used for the system clock. The HALT mode stops the system oscillator and ignores an external signal to conserve power.

Of the two oscillators, if the RC oscillator is used, an external resistor between OSC1 and VSS is required, and the range of the resistance should be from 51kΩ to 1MΩ. The system clock,



System oscillator



RTC oscillator

divided by 4, is available on OSC2 with pull-high resistor, which can be used to synchronize external logic. The RC oscillator provides the most cost effective solution. However, the frequency of the oscillation may vary with VDD, temperature, and the chip itself due to process variations. It is, therefore, not suitable for timing sensitive operations where accurate oscillator frequency is desired.

On the other hand, if the crystal oscillator is selected, a crystal across OSC1 and OSC2 is needed to provide the feedback and phase shift required for the oscillator, and no other external components are required. A resonator may be connected between OSC1 and OSC2 to replace the crystal and to get a frequency reference, but two external capacitors in OSC1 and OSC2 are needed.

There is another oscillator circuit designed for the real time clock. In this case, only the 32.768kHz crystal oscillator can be applied. The crystal should be connected between OSC3 and OSC4, and two external capacitors along with one external resistor are required for the oscillator circuit in order to get a stable frequency.

The RTC oscillator circuit can be controlled to oscillate quickly by setting "QOSC" bit (bit 4 of RTCC). It is recommended to turn on the quick oscillating function upon power on, and turn it off after two seconds.

The WDT oscillator is a free running on-chip RC oscillator, and no external components are required. Although the system enters the power down mode, the system clock stops, and the WDT oscillator still works with a period of approximately 78 μs. The WDT oscillator can be disabled by mask option to conserve power.

Watchdog timer – WDT

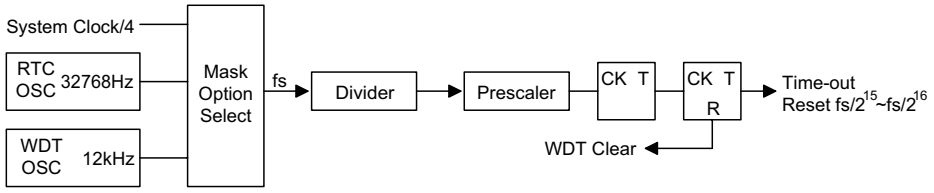
The WDT clock source is implemented by a dedicated RC oscillator (WDT oscillator) or an instruction clock (system clock/4) or a real time clock oscillator (RTC oscillator). The timer is designed to prevent a software malfunction or sequence from jumping to an unknown location with unpredictable results. The WDT can be disabled by mask option. But if the WDT is disabled, all executions related to the WDT lead to no operation.

After the WDT clock source is selected, it time-out period is $fs/2^{15}$~$fs/2^{16}$.

If the WDT clock source chooses the internal WDT oscillator, the time-out period may vary with temperature, VDD, and process variations. On the other hand, if the clock source selects the instruction clock and the "halt" instruction is executed, WDT may stop counting and lose its protecting purpose, and the logic can only be restarted by external logic.

When the device operates in a noisy environment, using the on-chip RC oscillator (WDT OSC) is strongly recommended, since the HALT can stop the system clock.

The WDT overflow under normal operation initializes a "chip reset" and sets the status bit "TO". In the HALT mode, the overflow initializes a "warm reset", and only the PC and SP are reset to zero. To clear the contents of the WDT, there are three methods to be adopted, i.e., external reset (a low level to $\overline{RES}$), software instruction, and "HALT" instruction. There are two sets of software instructions, "CLR WDT" and the other set – "CLR WDT1" and "CLR WDT2". Of these two types of instructions, only one type of instruction can be active at a time depending on the mask option – "CLR WDT



Watchdog timer

times selection option". If the "CLR WDT" is selected (i.e., CLR WDT times equal one), any execution of the "CLR WDT" instruction clears the WDT. In the case that "CLR WDT1" and "CLR WDT2" are chosen (i.e., CLR WDT times equal two), these two instructions have to be executed to clear the WDT; otherwise, the WDT may reset the chip because of time-out.

Multi-function timer

The HT49C10 provides a multi-function timer for WDT, time base and RTC but with different time-out periods. The multi-function timer consists of a 7-stage divider and an 8-bit prescaler, with the clock source coming from the WDT OSC or RTC OSC or the instruction clock (i.e., system clock divided by 4). The multi-function timer also provides a selectable frequency signal (ranges from $fs/2^2$ to $fs/2^8$) for LCD driver circuits, and a selectable frequency signal (ranges from $fs/2^2$ to $fs/2^9$) for buzzer output by mask option. It is recommended to select a 4kHz signal for LCD driver circuits for proper display.

Time base

The time base offers a periodic time-out period to generate a regular internal interrupt. Its time-out period ranges from $fs/2^{12}$ to $fs/2^{15}$ selected by mask option. If time base time-out occurs, the related interrupt request flag (TBF; bit 4 of INTC1) is set. But if the interrupt is enabled, and the stack is not full, a subroutine call to location 10H occurs.

Real time clock – RTC

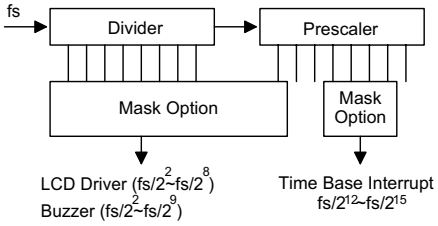
The real time clock (RTC) is operated in the same manner as the time base that is used to supply a regular internal interrupt. Its time-out period ranges from $fs/2^8$ to $fs/2^{15}$ by software programming (recommand use 2^{12}~2^{15}). Writing data to RT2, RT1 and RT0 (bits 2, 1, 0 of RTCC;09H) yields various time-out periods. If the RTC time-out occurs, the related interrupt request flag (RTF; bit 5 of INTC1) is set. But if the interrupt is enabled, and the stack is not full, a subroutine call to location 14H occurs. The real time clock time-out signal can also be applied as a clock source of the timer/event counter to get a longer time-out period.

RT2	RT1	RT0	RTC Clock Divided Factor
0	0	0	2^8
0	0	1	2^9
0	1	0	2^{10}
0	1	1	2^{11}
1	0	0	2^{12}
1	0	1	2^{13}
1	1	0	2^{14}
1	1	1	2^{15}

Power down operation – HALT

The HALT mode is initialized by the "HALT" instruction and results in the following.

- The system oscillator turns off but the WDT oscillator keeps running (if the WDT oscillator or the real time clock is selected).
- The contents of the on-chip RAM and of the registers remain unchanged.
- The WDT is cleared and start recounting (if the WDT clock source is from the WDT oscillator or the real time clock oscillator).



Time base

- All I/O ports maintain their original status.
- The PD flag is set but the TO flag is cleared.
- LCD driver is still running (if the WDT OSC or RTC OSC is selected).

The system quits the HALT mode by an external reset, an interrupt, an external falling edge signal on port A, or a WDT overflow. An external reset causes device initialization, and the WDT overflow performs a "warm reset". After examining the TO and PD flags, the reason for chip reset can be determined. The PD flag is cleared by system power-up or by executing the "CLR WDT" instruction, and is set by executing the "HALT" instruction. On the other hand, the TO flag is set if WDT time-out occurs, and causes a wake-up that only resets the PC (Program Counter) and SP, and leaves the others at their original state.

The port A wake-up and interrupt methods can be considered as a continuation of normal execution. Each bit in port A can be independently selected to wake up the device by mask option. Awakening from an I/O port stimulus, the program resumes execution of the next instruction. On the other hand, awakening from an interrupt, two sequences may occur. If the related interrupt is disabled or the interrupt is enabled, but the stack is full, the program resumes execution at the next instruction. But if the interrupt is enabled, and the stack is not full, the regular interrupt response takes place.

When an interrupt request flag is set before entering the "halt" status, the system cannot be awaken using that interrupt.

If a wake-up event occurs, it takes 1024 t_{SYS} (system clock period) to resume normal operation. In other words, a dummy period is inserted. If the wake-up results from an interrupt acknowledgment, the actual interrupt subroutine execution is delayed by more than one cycle. However, if the wake-up results in the next instruction execution, the execution will be performed immediately after the dummy period is finished.

To minimize power consumption, all the I/O pins should be carefully managed before entering the HALT status.

Reset

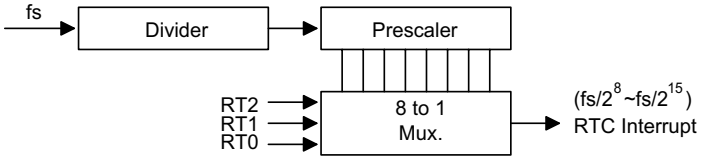
There are three ways in which a reset may occur.

- $\overline{RES}$ is reset during normal operation
- $\overline{RES}$ is reset during HALT
- WDT time-out is reset during normal operation

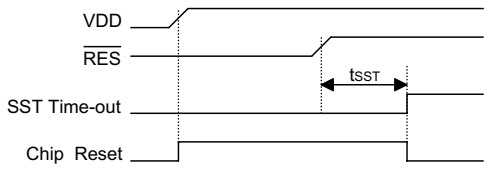
The WDT time-out during HALT differs from other chip reset conditions, for it can perform a "warm reset" that resets only the PC and SP and leaves the other circuits at their original state. Some registers remain unaffected during any other reset conditions. Most registers are reset to the "initial condition" once the reset conditions are met. Examining the PD and TO flags, the program can distinguish between different "chip resets".

TO	PD	RESET Conditions
0	0	$\overline{RES}$ reset during power-up
u	u	$\overline{RES}$ reset during normal operation
0	1	$\overline{RES}$ wake-up HALT
1	u	WDT time-out during normal operation
1	1	WDT wake-up HALT

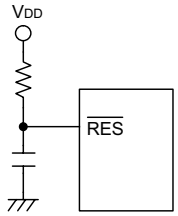
Note: "u" means "unchanged"



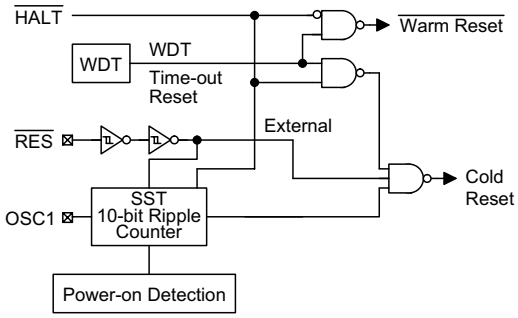
Real time clock



Reset timing chart



Reset circuit



Reset configuration

To guarantee that the crystal oscillator is started and stabilized, the SST (System Start-up Timer) provides an extra-delay of 1024 system clock pulses when the system powers up or awakes from the HALT state. Awaking from the HALT state or system power-up, the SST delay is added.

An extra SST delay is added during the power-up period, and any wake-up from HALT may enable the SST delay only.

The functional unit chip reset status are shown below.

PC	000H
Interrupt	Disabled
Prescaler, divider	Cleared
WDT, RTC, time base	Clear. After master reset, begins counting
Timer/event counter	Off
Input/output ports	Input mode
SP	Points to the top of the stack

The states of the registers are summarized below.

Register	Reset (Power On)	WDT Time-out (Normal Operation)	$\overline{RES}$ Reset (Normal Operation)	$\overline{RES}$ Reset (HALT)	WDT Time-out (HALT)
TMR	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMRC	0000 1---	0000 1---	0000 1---	0000 1---	uuuu u---
Program Counter	000H	000H	000H	000H	000H*
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	--xx xxxx	--uu uuuu	--uu uuuu	--uu uuuu	--uu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--01 uuuu	--11 uuuu
INTC0	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
INTC1	--00 --00	--00 --00	--00 --00	--00 --00	--uu --uu
RTCC	--00 0111	--00 0111	--00 0111	--00 0111	--uu uuuu
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PB	--11 1111	--11 1111	--11 1111	--11 1111	--uu uuuu

Note: "*" means "warm reset"
"u" means "unchanged"
"x" means "unknown"

Timer/event counter

A timer/event counter (TMR) is implemented in the HT49C10. The timer/event counter contains an 8-bit programmable count-up counter, and the clock source may come from the system clock or instruction clock (system clock/4) or RTC time-out signal or external source. System clock source or instruction clock is selected by mask option.

The external clock input allows the user to count external events, measure time intervals or pulse widths, or generate an accurate time base.

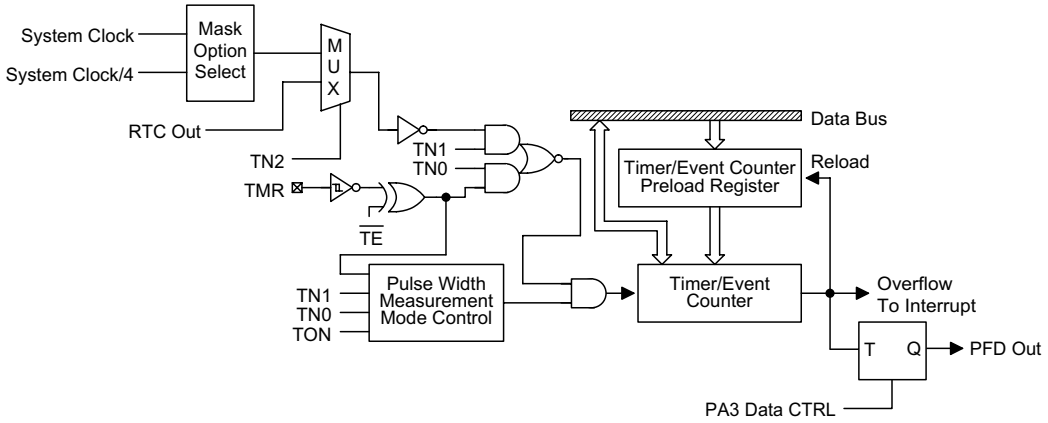
There are two registers related to the timer/event counter, i.e., TMR ([0DH]) and TMRC ([0EH]). And two physical registers are mapped to TMR location; writing TMR locates the starting value in the timer/event counter preload register, while reading it yields the contents of the timer/event counter. The TMRC is a timer/event counter control register which defines some options.

The TN0 and TN1 bits define the operation mode. The event count mode is used to count external events, which means that the clock source is from an external (TMR) pin. The timer mode functions as a normal timer with the clock source coming from the internal selected clock source. Finally, the pulse width measurement mode can be used to count the high or low level duration of the external signal (TMR), and the counting is based on the internal selected clock source.

In the event count or timer mode, the timer/event counter starts counting at the current contents in the timer/event counter and ends at FFH. Once an overflow occurs, the counter is reloaded from the timer/event counter preload register, and generates an interrupt request flag (TF; bit 6 of INTC0).

In the pulse width measurement mode with the values of the TON and TE bits equal to one, after the TMR has received a transient from low to high (or high to low if the TE bit is "0"), it will start counting until the TMR returns to the original level and resets the TON. The measured result remains in the timer/event counter even if the activated transient occurs again. In other words, only one cycle measurement can be done. Until setting the TON, the cycle measurement will re-function as long as it receives further transient pulse. In this operation mode, the timer/event counter begins counting according not to the logic level but to the transient edges. In the case of counter overflows, the counter is reloaded from the timer/event counter preload register and issues an interrupt request, as in the other two modes, i.e., event and timer modes.

To enable the counting operation, the Timer ON bit (TON; bit 4 of TMRC) should be set to 1. In the pulse width measurement mode, the TON is automatically cleared after the measurement cycle is completed. But in the other two modes,



Timer/event counter

Label (TMRC)	Bits	Function
—	0~2	Unused bits, read as ″0″
TE	3	Defines the TMR active edge of the timer/event counter (0=active on low to high; 1=active on high to low)
TON	4	Enable/disable timer counting (0=disabled; 1=enabled)
TN2	5	Two to one multiplexer control inputs to select the timer/event counter clock source (0=RTC output; 1=system clock or system clock/4)
TN0, TN1	7, 6	Defines the operating mode 01=Event count mode (external clock) 10=Timer mode (internal clock) 11=Pulse Width measurement mode (external clock) 00=Unused

TMRC register

the TON can only be reset by instructions. The overflow of the timer/event counter is one of the wake-up sources and can also be applied to a PFD (Programmable Frequency Divider) output at PA3 by mask option. No matter what the operation mode is, writing a 0 to ETI disables the interrupt service. When the PFD function is selected, executing ″CLR [PA].3″ instruction to enable PFD output and executing ″SET [PA].3″ instruction to disable the PFD output.

In the case of timer/event counter OFF condition, writing data to the timer/event counter preload register also reloads that data to the timer/event counter. But if the timer/event counter is turned on, data written to the timer/event counter is kept only in the timer/event counter preload register. The timer/event counter still goes on operating until an overflow occurs.

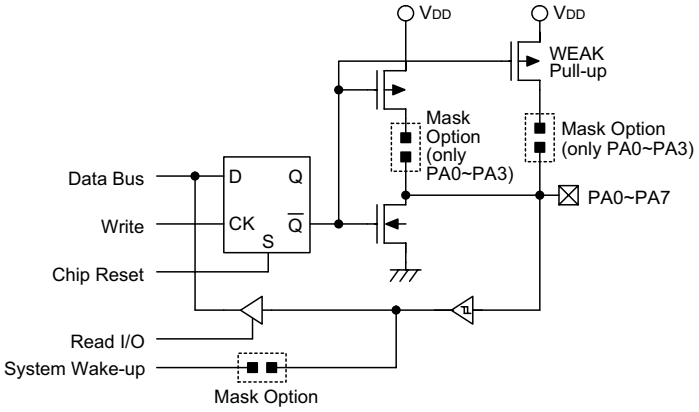
When the timer/event counter (reading TMR) is read, the clock is blocked to avoid errors. As this may results in a counting error, this should be taken into account by the programmer.

It is strongly recommended to load first a specific value into the TMR register, then turn on the timer/event counter for proper operation. Because the initial value of TMR is unknown.

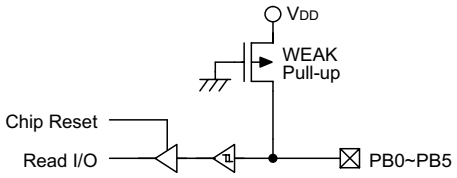
Input/output ports

There is an 8-bit bidirectional input/output port and a 6-bit input port in the HT49C10, labeled PA and PB, which are mapped to [12H] and [14H] of the RAM, respectively. PA0~PA3 can be configured as CMOS (output) or NMOS (input/output) and with or without pull-high resistor by mask option, PA4~PA7 always pull-high and NMOS (input/output). PB can only be used for input operation, and each bit on the port with pull-high resistor. Both are for the input operation, these ports are non-latched, that is, the inputs should be ready at the T2 rising edge of the instruction ″MOV A, [m]″ (m=12H or 14H). For PA output operation, all data are latched and remain unchanged until the output latch is rewritten.

When the structures of PA are open drain NMOS type, it should be noted that, before reading data from the pads, a ″1″ should be written to the related bits to disable the NMOS device. That is executing first the instruction ″SET [m].i″ (i=0~7 for PA) to disable any related NMOS device, and then ″MOV A, [m]″ to get stable data.



PA input/output port



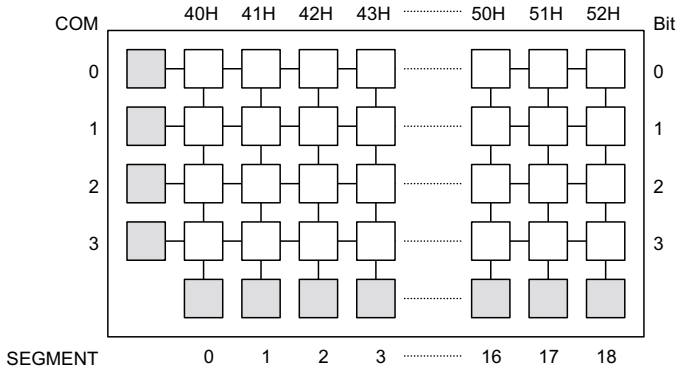
PB input lines

After chip reset, these input lines remain at a high level or are left floating (by mask option). Each bit of these output latches can be set or cleared by the "SET [m].i" and "CLR [m].i" (m=12H) instructions.

Some instructions first input data and then follow the output operations. For example, "SET [m].i", "CLR [m].i", "CPL [m]", "CPLA [m]" read the entire port states into the CPU, execute the defined operations (bit-operation), and then write the results back to the latches or to the accumulator.

LCD display memory

The HT49C10 provides an area of embedded data memory for LCD display. This area is located from 40H to 52H of the RAM at Bank 1. Bank pointer (BP; located at 04H of the RAM) is the switch between the RAM and the LCD display memory. When the BP is set "1", any data



Display memory

Register	Bit No.	Label	Read/Write	Reset	Function
RTCC (09H)	0~2	RT0 RT1 RT2	R/W	0	8 to 1 multiplexer control inputs to select the real time clock prescaler output
	3	—	—	—	Unused bits, read as ″0″
	4	QOSC	R/W	0	Control the RTC OSC to oscillate quickly. ″0″ enable ″1″ disable
	5~7	—	—	—	Unused bits, read as ″0″

RTCC register

written into 40H~52H will affect the LCD display. When the BP is cleared ″0″, any data written into 40H~52H means to access the general purpose data memory. The LCD display memory can be read and written to, only by indirect addressing mode using MP1. When data is writxten into the display data area it is automatically read by the LCD driver which then generates the corresponding LCD driving signals. To turn the display On or Off, a ″1″ or a ″0″ is written to the corresponding bit of the display memory, respectively. The figure illustrates the mapping between the display memory and LCD pattern for the HT49C10.

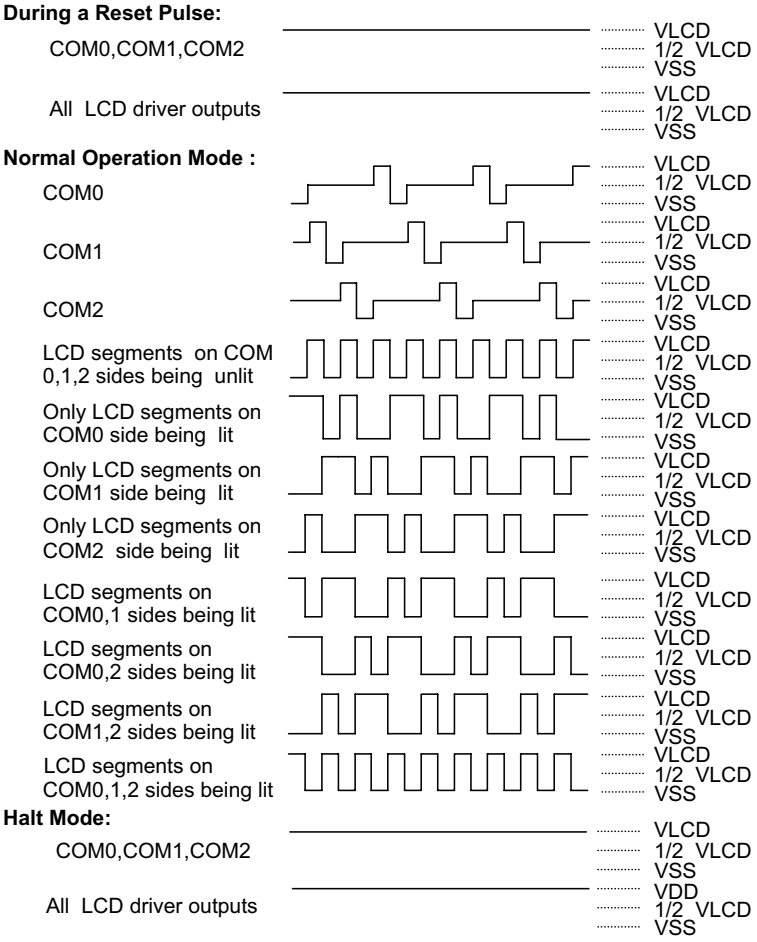
LCD driver output

The output number of the HT49C10 LCD driver can be 19×2 or 19×3 or 18×4 by mask option (i.e., 1/2 duty or 1/3 duty or 1/4 duty). The bias type of LCD driver can be ″R″ type or ″C″ type. If the ″R″ bias type is selected, no external capacitor is required. If the ″C″ bias type is selected, a capacitor mounted between C1 and C2 pins is needed. The bias voltage of LCD driver can be 1/2 bias or 1/3 bias by mask option. If 1/2 bias is selected, a capacitor mounted between V2 pin and the ground is required. If 1/3 bias is selected, two capacitors are needed for V1 and V2 pins. Please refer to the application diagram.

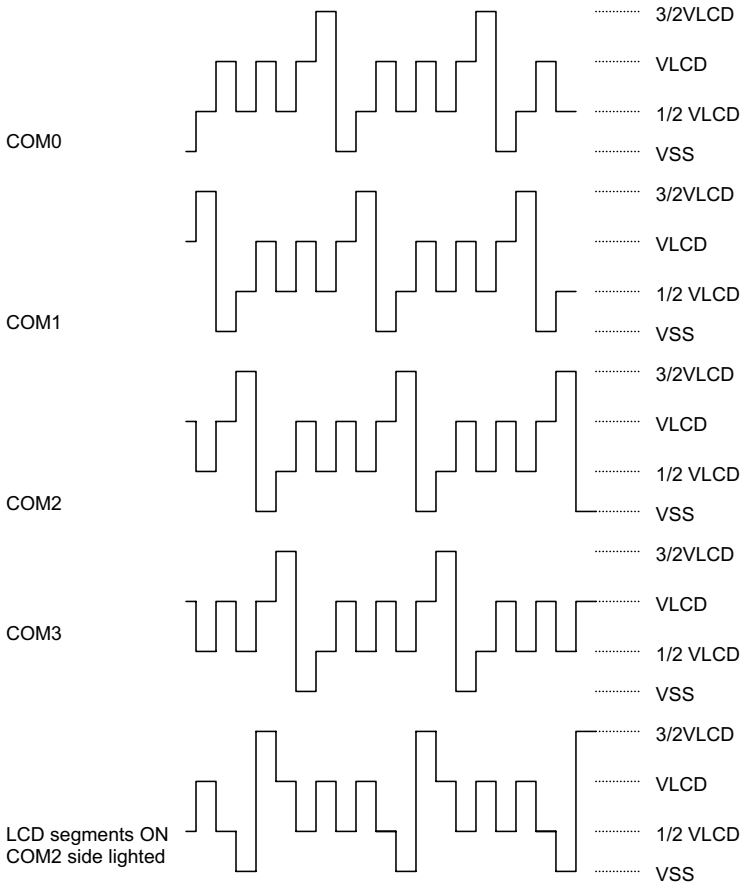
Buzzer

HT49C10 provides a pair of buzzer output BZ and $\overline{BZ}$, which share pins with PA0 and PA1 respectively, as determined by mask option. Its output frequency can be selected by mask option.

When the buzzer function is selected, setting PA.0 and PA.1 ″0″ simultaneously will enable the buzzer output and setting PA.0 ″1″ will disable the buzzer output.



LCD driver output (1/3 duty, 1/2 bias, R/C type)



LCD driver output (1/4 duty, 1/3 bias, C type)

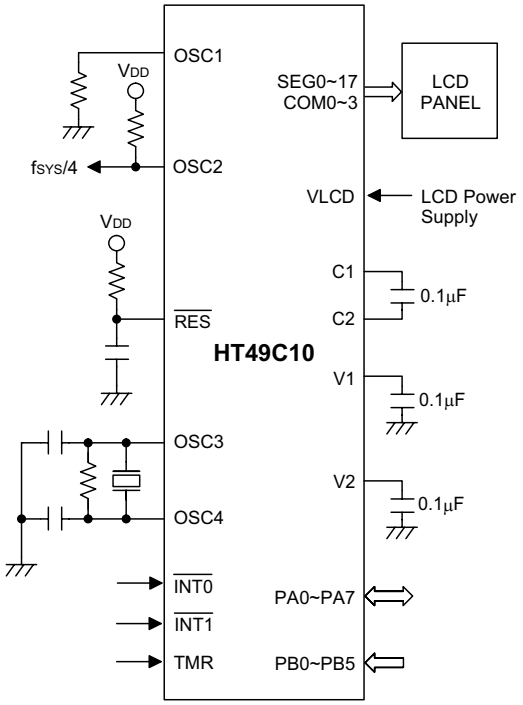
Mask option

The following shows 16 kinds of mask options in the HT49C10. All these options should be defined in order to ensure proper system functioning.

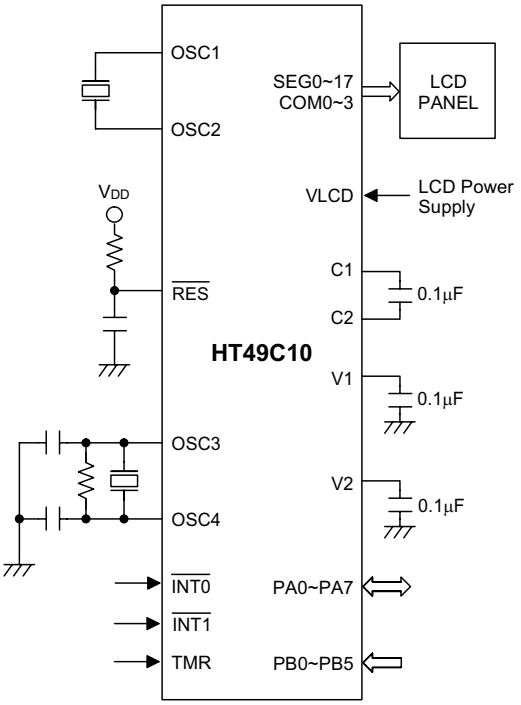
No.	Mask Option
1	OSC type selection. This option is to decide if an RC or crystal oscillator is selected as system clock.
2	Clock source selection of WDT, RTC and Time Base. There are three types of selection: system clock/4 or RTC OSC or WDT OSC.
3	WDT enable/disable selection. WDT can be enabled or disabled by mask option.
4	CLR WDT times selection. This option defines how to clear the WDT by instruction. One time means that the "CLR WDT" can clear the WDT. "Two times" means only if both of the "CLR WDT1" and "CLR WDT2" have been executed, the WDT can be cleared.
5	Time Base time-out period selection. The Time Base time-out period ranges from $fs/2^{12}$ to $fs/2^{15}$. "fs" means the clock source selected by mask option.
6	Buzzer output frequency selection. There are eight types frequency signals for buzzer output: $fs/2^2$~$fs/2^9$. "fs" means the clock source selected by mask option.
7	Wake-up selection. This option defines the wake-up function activity. External I/O pins (PA only) all have the capability to wake-up the chip from a HALT by a falling edge.
8	Pull-high selection. This option is to decide whether the pull-high resistance is viable or not on the PA0~PA3.
9	PA CMOS or NMOS selection. PA0~PA3 can be selected as CMOS or NMOS structure, but PA4~PA7 are always NMOS. When the CMOS is selected, the related pins can only be used for output operations. When the NMOS is selected, the related pins can be used for input or output operations.
10	Clock source selection of timer/event counter. There are two types of selection: system clock or system clock/4.
11	I/O pins share with other functions selection. PA0/$\overline{BZ}$, PA1/BZ: PA0 and PA1 can be set as I/O pins or buzzer outputs. PA3/PFD: PA3 can be set as I/O pins or PFD output.
12	LCD common selection. There are three types of selection: 2 commons (1/2 duty) or 3 commons (1/3 duty) or 4 commons (1/4 duty). If the 4 commons are selected, the segment output pin "SEG18" will be set as a common output.
13	LCD bias power supply selection. There are two types of selection: 1/2 bias or 1/3 bias.
14	LCD bias type selection. This option is to decide what kind of bias is selected, R type or C type.
15	LCD driver clock selection. There are seven types frequency signals for LCD driver circuits: $fs/2^2$~$fs/2^8$. "fs" means the clock source selected by mask option.

Application Circuits

RC oscillator application



Crystal oscillator application



Instruction Set Summary

Mnemonic	Description	Flag Affected
Arithmetic		
ADD A,[m]	Add data memory to ACC	Z,C,AC,OV
ADDM A,[m]	Add ACC to data memory	Z,C,AC,OV
ADD A,x	Add immediate data to ACC	Z,C,AC,OV
ADC A,[m]	Add data memory to ACC with carry	Z,C,AC,OV
ADCM A,[m]	Add ACC to register with carry	Z,C,AC,OV
SUB A,x	Subtract immediate data from ACC	Z,C,AC,OV
SUB A,[m]	Subtract data memory from ACC	Z,C,AC,OV
SUBM A,[m]	Subtract data memory from ACC with result in data memory	Z,C,AC,OV
SBC A,[m]	Subtract data memory from ACC with carry	Z,C,AC,OV
SBCM A,[m]	Subtract data memory from ACC with carry and result in data memory	Z,C,AC,OV
DAA [m]	Decimal adjust ACC for addition with result in data memory	C
Logic Operation		
AND A,[m]	AND data memory to ACC	Z
OR A,[m]	OR data memory to ACC	Z
XOR A,[m]	Exclusive-OR data memory to ACC	Z
ANDM A,[m]	AND ACC to data memory	Z
ORM A,[m]	OR ACC to data memory	Z
XORM A,[m]	Exclusive-OR ACC to data memory	Z
AND A,x	AND immediate data to ACC	Z
OR A,x	OR immediate data to ACC	Z
XOR A,x	Exclusive-OR immediate data to ACC	Z
CPL [m]	Complement data memory	Z
CPLA [m]	Complement data memory with result in ACC	Z
Increment & Decrement		
INCA [m]	Increment data memory with result in ACC	Z
INC [m]	Increment data memory	Z
DECA [m]	Decrement data memory with result in ACC	Z
DEC [m]	Decrement data memory	Z
Rotate		
RRA [m]	Rotate data memory right with result in ACC	None
RR [m]	Rotate data memory right	None
RRCA [m]	Rotate data memory right through carry with result in ACC	C
RRC [m]	Rotate data memory right through carry	C
RLA [m]	Rotate data memory left with result in ACC	None
RL [m]	Rotate data memory left	None
RLCA [m]	Rotate data memory left through carry with result in ACC	C
RLC [m]	Rotate data memory left through carry	C

Mnemonic	Description	Flag Affected
Data Move		
MOV A,[m]	Move data memory to ACC	None**
MOV [m],A	Move ACC to data memory	None
MOV A,x	Move immediate data to ACC	None
Bit Operation		
CLR [m].i	Clear bit of data memory	None
SET [m].i	Set bit of data memory	None
Branch		
JMP addr	Jump unconditionally	None
SZ [m]	Skip if data memory is zero	None
SZA [m]	Skip if data memory is zero with data movement to ACC	None
SZ [m].i	Skip if bit i of data memory is zero	None
SNZ [m].i	Skip if bit i of data memory is not zero	None
SIZ [m]	Skip if increment data memory is zero	None
SDZ [m]	Skip if decrement data memory is zero	None
SIZA [m]	Skip if increment data memory is zero with result in ACC	None
SDZA [m]	Skip if decrement data memory is zero with result in ACC	None
CALL addr	Subroutine call	None
RET	Return from subroutine	None
RET A,x	Return from subroutine and load immediate data to ACC	None
RETI	Return from interrupt	None
Table Read		
TABRDC [m]	Read ROM code (current page) to data memory and TBLH	None
TABRDL [m]	Read ROM code (last page) to data memory and TBLH	None
Miscellaneous		
NOP	No operation	None
CLR [m]	Clear data memory	None
SET [m]	Set data memory	None
CLR WDT	Clear Watchdog timer	TO,PD
CLR WDT1	Pre-clear watchdog timer	TO*,PD*
CLR WDT2	Pre-clear watchdog timer	TO*,PD*
SWAP [m]	Swap nibbles of data memory	None
SWAPA [m]	Swap nibbles of data memory with result in ACC	None
HALT	Enter power down mode	TO,PD

Note: x: 8-bit immediate data
m: 7 bits data memory address
A: accumulator
i: 0~7 number of bits
addr: 10-bit program memory address
√: Flag is affected
–: Flag is not affected

*: Flag may be affected by the execution status

**: For the old version of the E.V. chip, the zero flag (Z) can be affected by executing the MOV A,[M] instruction.

For the new version of the E.V. chip, the zero flag cannot be changed by executing the MOV A,[M] instruction.

Instruction Definition

ADC A,[m] Add data memory and carry to the accumulator

Description The contents of the specified data memory, accumulator and the carry flag are added simultaneously, leaving the result in the accumulator.

Operation ACC ← ACC+[m]+C

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	√	√	√	√

ADCM A,[m] Add the accumulator and carry to data memory

Description The contents of the specified data memory, accumulator and the carry flag are added simultaneously, leaving the result in the specified data memory.

Operation [m] ← ACC+[m]+C

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	√	√	√	√

ADD A,[m] Add data memory to the accumulator

Description The contents of the specified data memory and the accumulator are added. The result is stored in the accumulator.

Operation ACC ← ACC+[m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	√	√	√	√

ADD A,x Add immediate data to the accumulator

Description The contents of the accumulator and the specified data are added, leaving the result in the accumulator.

Operation ACC ← ACC+x

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	√	√	√	√

ADDM A,[m] Add the accumulator to the data memory

Description The contents of the specified data memory and the accumulator are added. The result is stored in the data memory.

Operation [m] ← ACC+[m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	√	√	√	√

AND A,[m] Logical AND accumulator with data memory

Description Data in the accumulator and the specified data memory perform a bitwise logical_AND operation. The result is stored in the accumulator.

Operation ACC ← ACC "AND" [m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	√	—	—

AND A,x Logical AND immediate data to the accumulator

Description Data in the accumulator and the specified data perform a bitwise logical_AND operation. The result is stored in the accumulator.

Operation ACC ← ACC "AND" x

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	√	—	—

ANDM A,[m] Logical AND data memory with the accumulator

Description Data in the specified data memory and the accumulator perform a bitwise logical_AND operation. The result is stored in the data memory.

Operation [m] ← ACC "AND" [m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	√	—	—

CALL addr Subroutine call

Description The instruction unconditionally calls a subroutine located at the indicated address. The program counter increments once to obtain the address of the next instruction, and pushes this onto the stack. The indicated address is then loaded. Program execution continues with the instruction at this address.

Operation Stack ← PC+1
PC ← addr

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

CLR [m] Clear data memory

Description The contents of the specified data memory are cleared to zero.

Operation [m] ← 00H

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

CLR [m].i Clear bit of data memory

Description The bit i of the specified data memory is cleared to zero.

Operation [m].i ← 0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

CLR WDT Clear watchdog timer

Description The WDT is cleared (re-counting from zero). The power down bit (PD) and time-out bit (TO) are cleared.

Operation WDT ← 00H
PD and TO ← 0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	0	0	—	—	—	—

CLR WDT1 Preclear watchdog timer

Description The TD, PD flags and WDT are all cleared (re-counting from zero), if the other preclear WDT instruction has been executed. Only execution of this instruction without the other preclear instruction sets the indicated flag which implies that this instruction has been executed and the TO and PD flags remain unchanged.

Operation WDT ← 00H*
PD and TO ← 0*

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	0*	0*	—	—	—	—

CLR WDT2 Preclear watchdog timer

Description The TO, PD flags and WDT are cleared (re-counting from zero), if the other preclear WDT instruction has been executed. Only execution of this instruction without the other preclear instruction sets the indicated flag which implies that this instruction has been executed and the TO and PD flags remain unchanged.

Operation WDT ← 00H*
PD and TO ← 0*

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	0*	0*	—	—	—	—

CPL [m] Complement data memory

Description Each bit of the specified data memory is logically complemented (1′s complement). Bits which previously contained a one are changed to zero and vice-versa.

Operation $[m] \leftarrow [\overline{m}]$

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	√	—	—

CPLA [m] Complement data memory and place result in the accumulator

Description Each bit of the specified data memory is logically complemented (1′s complement). Bits which previously contained a one are changed to zero and vice-versa. The complemented result is stored in the accumulator and the contents of the data memory remain unchanged.

Operation ACC ← [$\overline{m}$]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	√	—	—

DAA [m] Decimal-Adjust accumulator for addition

Description The accumulator value is adjusted to the BCD (Binary Code Decimal) code. The accumulator is divided into two nibbles. Each nibble is adjusted to the BCD code and an internal carry (AC1) will be done if the low nibble of the accumulator is greater than 9. The BCD adjustment is done by adding 6 to the original value if the original value is greater than 9 or a carry (AC or C) is set; otherwise the original value remains unchanged. The result is stored in the data memory and only the carry flag (C) may be affected.

Operation If ACC.3~ACC.0 >9 or AC=1
then [m].3~[m].0 ← (ACC.3~ACC.0)+6, AC1=$\overline{AC}$
else [m].3~[m].0) ← (ACC.3~ACC.0), AC1=0
and
If ACC.7~ACC.4+AC1 >9 or C=1
then [m].7~[m].4 ← ACC.7~ACC.4+6+AC1,C=1
else [m].7~[m].4 ← ACC.7~ACC.4+AC1,C=C

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	√

DEC [m] Decrement data memory

Description Data in the specified data memory is decremented by one.

Operation [m] ← [m]–1

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	√	—	—

DECA [m] Decrement data memory and place result in the accumulator

Description Data in the specified data memory is decremented by one, leaving the result in the accumulator. The contents of the data memory remain unchanged.

Operation ACC ← [m]–1

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	√	—	—

HALT Enter power down mode

Description This instruction stops program execution and turns off the system clock. The contents of the RAM and registers are retained. The WDT and prescaler are cleared. The power down bit (PD) is set and the WDT time-out bit (TO) is cleared.

Operation PC ← PC+1
PD ← 1
TO ← 0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	0	1	—	—	—	—

INC [m] Increment data memory

Description Data in the specified data memory is incremented by one.

Operation [m] ← [m]+1

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	√	—	—

INCA [m] Increment data memory and place result in the accumulator

Description Data in the specified data memory is incremented by one, leaving the result in the accumulator. The contents of the data memory remain unchanged.

Operation ACC ← [m]+1

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	√	—	—

JMP addr Directly jump

Description The contents of the program counter are replaced with the directly-specified address unconditionally, and control is passed to this destination.

Operation PC ← addr

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

MOV A,[m] Move data memory to the accumulator

Description The contents of the specified data memory are copied to the accumulator.

Operation ACC ← [m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

MOV A,x Move immediate data to the accumulator

Description The 8-bit data specified by the code is loaded into the accumulator.

Operation ACC ← x

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

MOV [m],A Move the accumulator to data memory

Description The contents of the accumulator are copied to the specified data memory (one of the data memories).

Operation [m] ← ACC

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

NOP No operation

Description No operation is performed. Execution continues with the next instruction.

Operation PC ← PC+1

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

OR A,[m] Logical OR accumulator with data memory

Description Data in the accumulator and the specified data memory (one of the data memories) perform a bitwise logical_OR operation. The result is stored in the accumulator.

Operation ACC ← ACC "OR" [m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	√	—	—

OR A,x Logical OR immediate data to the accumulator

Description Data in the accumulator and the specified data perform a bitwise logical_OR operation. The result is stored in the accumulator.

Operation ACC ← ACC "OR" x

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	√	—	—

ORM A,[m] Logical OR data memory with the accumulator

Description Data in the data memory (one of the data memories) and the accumulator perform a bitwise logical_OR operation. The result is stored in the data memory.

Operation [m] ← ACC "OR" [m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	√	—	—

RET Return from subroutine

Description The program counter is restored from the stack. This is a two-cycle instruction.

Operation PC ← Stack

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

RET A,x Return and place immediate data in the accumulator

Description The program counter is restored from the stack and the accumulator loaded with the specified 8-bit immediate data.

Operation PC ← Stack
ACC ← x

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

RETI Return from interrupt

Description The program counter is restored from the stack, and interrupts are enabled by setting the EMI bit. EMI is the enable master (global) interrupt bit (bit 0; register INTC).

Operation PC ← Stack
EMI ← 1

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

RL [m] Rotate data memory left

Description The contents of the specified data memory are rotated one bit left with bit 7 rotated into bit 0.

Operation [m].(i+1) ← [m].i; [m].i:bit i of the data memory (i=0~6)
[m].0 ← [m].7

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

RLA [m] Rotate data memory left and place result in the accumulator

Description Data in the specified data memory is rotated one bit left with bit 7 rotated into bit 0, leaving the rotated result in the accumulator. The contents of the data memory remain unchanged.

Operation ACC.(i+1) ← [m].i; [m].i:bit i of the data memory (i=0~6)
ACC.0 ← [m].7

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

RLC [m] Rotate data memory left through carry

Description The contents of the specified data memory and the carry flag are rotated one bit left. Bit 7 replaces the carry bit; the original carry flag is rotated into the bit 0 position.

Operation [m].(i+1) ← [m].i; [m].i:bit i of the data memory (i=0~6)
[m].0 ← C
C ← [m].7

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	√

RLCA [m] Rotate left through carry and place result in the accumulator

Description Data in the specified data memory and the carry flag are rotated one bit left. Bit 7 replaces the carry bit and the original carry flag is rotated into bit 0 position. The rotated result is stored in the accumulator but the contents of the data memory remain unchanged.

Operation ACC.(i+1) ← [m].i; [m].i:bit i of the data memory (i=0~6)
ACC.0 ← C
C ← [m].7

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	√

RR [m] Rotate data memory right

Description The contents of the specified data memory are rotated one bit right with bit 0 rotated to bit 7.

Operation [m].i ← [m].(i+1); [m].i:bit i of the data memory (i=0~6)
[m].7 ← [m].0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

RRA [m] Rotate right-place result in the accumulator

Description Data in the specified data memory is rotated one bit right with bit 0 rotated into bit 7, leaving the rotated result in the accumulator. The contents of the data memory remain unchanged.

Operation ACC.(i) ← [m].(i+1); [m].i:bit i of the data memory (i=0~6)
ACC.7 ← [m].0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

RRC [m] Rotate data memory right through carry

Description The contents of the specified data memory and the carry flag are together rotated one bit right. Bit 0 replaces the carry bit; the original carry flag is rotated into the bit 7 position.

Operation [m].i ← [m].(i+1); [m].i:bit i of the data memory (i=0~6)
[m].7 ← C
C ← [m].0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	√

RRCA [m] Rotate right through carry-place result in the accumulator

Description Data of the specified data memory and the carry flag are rotated one bit right. Bit 0 replaces the carry bit and the original carry flag is rotated into the bit 7 position. The rotated result is stored in the accumulator. The contents of the data memory remain unchanged.

Operation ACC.i ← [m].(i+1); [m].i:bit i of the data memory (i=0~6)
ACC.7 ← C
C ← [m].0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	√

SBC A,[m] Subtract data memory and carry from the accumulator

Description The contents of the specified data memory and the complement of the carry flag are subtracted from the accumulator, leaving the result in the accumulator.

Operation ACC ← ACC+[$\overline{m}$]+C

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	√	√	√	√

SBCM A,[m] Subtract data memory and carry from the accumulator

Description The contents of the specified data memory and the complement of the carry flag are subtracted from the accumulator, leaving the result in the data memory.

Operation [m] ← ACC+[$\overline{m}$]+C

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	√	√	√	√

SDZ [m] Skip if decrement data memory is zero

Description The contents of the specified data memory are decremented by one. If the result is zero, the next instruction is skipped. If the result is zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (two cycles). Otherwise proceed with the next instruction (one cycle).

Operation Skip if ([m]–1)=0, [m] ← ([m]–1)

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

SDZA [m] Decrement data memory and place result in ACC, skip if zero

Description The contents of the specified data memory are decremented by one. If the result is zero, the next instruction is skipped. The result is stored in the accumulator but the data memory remains unchanged. If the result is zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (two cycles). Otherwise proceed with the next instruction (one cycle).

Operation Skip if ([m]–1)=0, ACC ← ([m]–1)

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

SET [m] Set data memory

Description Each bit of the specified data memory is set to one.

Operation [m] ← FFH

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

SET [m].i Set bit of data memory

Description Biti of the specified data memory is set to one.

Operation [m].i ← 1

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

SIZ [m] Skip if increment data memory is zero

Description The contents of the specified data memory are incremented by one. If the result is zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (two cycles). Otherwise proceed with the next instruction (one cycle).

Operation Skip if ([m]+1)=0, [m] ← ([m]+1)

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

SIZA [m] Increment data memory and place result in ACC, skip if zero

Description The contents of the specified data memory are incremented by one. If the result is zero, the next instruction is skipped and the result is stored in the accumulator. The data memory remains unchanged. If the result is zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (two cycles). Otherwise proceed with the next instruction (one cycle).

Operation Skip if ([m]+1)=0, ACC ← ([m]+1)

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

SNZ [m].i Skip if bit "i" of the data memory is not zero

Description If bit "i" of the specified data memory is not zero, the next instruction is skipped. If bit "i" of the data memory is not zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (two cycles). Otherwise proceed with the next instruction (one cycle).

Operation Skip if [m].i≠0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

SUB A,[m] Subtract data memory from the accumulator

Description The specified data memory is subtracted from the contents of the accumulator, leaving the result in the accumulator.

Operation ACC ← ACC+[$\overline{m}$]+1

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	√	√	√	√

SUBM A,[m] Subtract data memory from the accumulator

Description The specified data memory is subtracted from the contents of the accumulator, leaving the result in the data memory.

Operation [m] ← ACC+[$\overline{m}$]+1

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	√	√	√	√

SUB A,x Subtract immediate data from the accumulator

Description The immediate data specified by the code is subtracted from the contents of the accumulator, leaving the result in the accumulator.

Operation ACC ← ACC+$\overline{x}$+1

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	√	√	√	√

SWAP [m] Swap nibbles within the data memory

Description The low-order and high-order nibbles of the specified data memory (one of the data memories) are interchanged.

Operation [m].3~[m].0 ↔ [m].7~[m].4

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

SWAPA [m] Swap data memory-place result in the accumulator

Description The low-order and high-order nibbles of the specified data memory are interchanged, writing the result to the accumulator. The contents of the data memory remain unchanged.

Operation ACC.3~ACC.0 ← [m].7~[m].4
ACC.7~ACC.4 ← [m].3~[m].0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

SZ [m] Skip if data memory is zero

Description If the contents of the specified data memory are zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (two cycles). Otherwise proceed with the next instruction (one cycle).

Operation Skip if [m]=0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

SZA [m] Move data memory to ACC, skip if zero

Description The contents of the specified data memory are copied to the accumulator. If the contents is zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (two cycles). Otherwise proceed with the next instruction (one cycle).

Operation Skip if [m]=0, ACC ← [m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

SZ [m].i Skip if bit "i" of the data memory is zero

Description If bit "i" of the specified data memory is zero, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (two cycles). Otherwise proceed with the next instruction (one cycle).

Operation Skip if [m].i=0

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

TABRDC [m] Move the ROM code (current page) to TBLH and data memory

Description The low byte of ROM code (current page) addressed by the table pointer (TBLP) is moved to the specified data memory and the high byte transferred to TBLH directly.

Operation [m] ← ROM code (low byte)
TBLH ← ROM code (high byte)

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

TABRDL [m] Move the ROM code (last page) to TBLH and data memory

Description The low byte of ROM code (last page) addressed by the table pointer (TBLP) is moved to the data memory and the high byte transferred to TBLH directly.

Operation [m] ← ROM code (low byte)
TBLH ← ROM code (high byte)

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	—	—	—

XOR A,[m] Logical XOR accumulator with data memory

Description Data in the accumulator and the indicated data memory perform a bitwise logical Exclusive_OR operation and the result is stored in the accumulator.

Operation ACC ← ACC XOR [m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	√	—	—

XORM A,[m] Logical XOR data memory with the accumulator

Description Data in the indicated data memory and the accumulator perform a bitwise logical Exclusive_OR operation. The result is stored in the data memory. The zero flag is affected.

Operation [m] ← ACC "XOR" [m]

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	√	—	—

XOR A,x Logical XOR immediate data to the accumulator

Description Data in the the accumulator and the specified data perform a bitwise logical Exclusive_OR operation. The result is stored in the accumulator. The zero flag is affected.

Operation ACC ← ACC "XOR" x

Affected flag(s)

TC2	TC1	TO	PD	OV	Z	AC	C
—	—	—	—	—	√	—	—

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)
5F, 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 Microelectronics Enterprises 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

Copyright © 1999 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.