HT48R01/HT48R02/HT48R03

10-Pin MSOP I/O Type 8-Bit OTP MCU

Technical Document

- Tools Information
- FAQs
- Application Note
  - HA0003E Communicating between the HT48 & HT46 Series MCUs and the HT93LC46 EEPROM
  - HA0016E Writing and Reading to the HT24 EEPROM with the HT48 MCU Series
  - HA0018E Controlling the HT1621 LCD Controller with the HT48 MCU Series
  - HA0049E Read and Write Control of the HT1380

Features

- Operating voltage:
  f_{SYS}=4MHz: 2.2V~5.5V
  f_{SYS}=8MHz: 3.3V~5.5V
  f_{SYS}=12MHz: 4.5V~5.5V
- 7 bidirectional I/O lines and 1 input
- Interrupt input shared with I/O line
- 4 oscillator configuration options
  - External crystal OSC
  - External RC OSC
  - Internal RC+I/O (PA5, PA6)
  - Internal RC+RTC OSC (32768Hz)
- Internal RC oscillator
  - 3 frequency selections: 4MHz/8MHz/12MHz
  - 4MHz with ±10% variation (2.2V~5.5V, 25°C)
  - 8MHz with ±10% variation (3.3V~5.5V, 25°C)
  - 12MHz with ±10% variation (4.5V~5.5V, 25°C)
- Watchdog Timer
- Program memory ROM: Up to 4096×15
- Data memory RAM: Up to 160×8
- Buzzer driving pair and PFD supported
- Power-down and wake-up functions reduce power consumption
- Up to 0.5μs instruction cycle with 8MHz system clock at V_{DD}=5V
- All instructions executed within one or two machine cycles
- 14-bit or 15-bit table read instruction
- Up to 8-levels of subroutine nesting
- Bit manipulation instruction
- Low voltage reset function
- 10-pin MSOP package

General Description

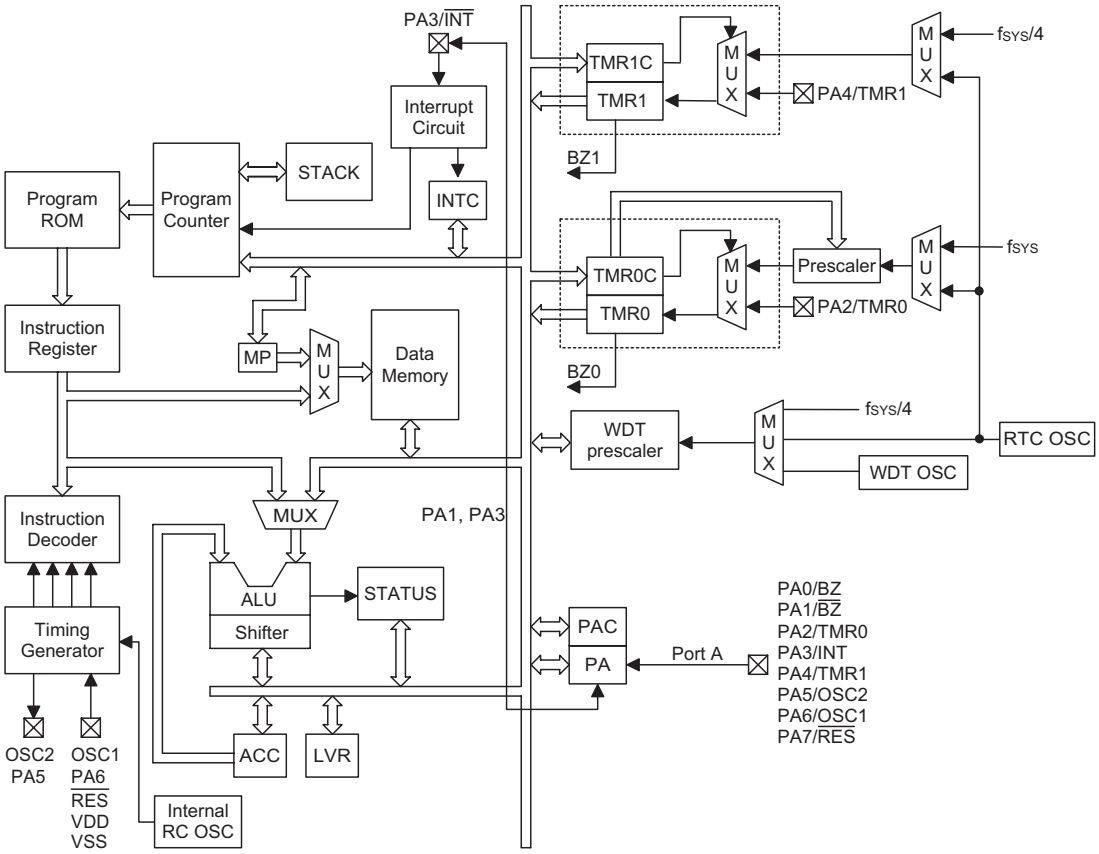
The HT48R01/HT48R02/HT48R03 are 8-bit high performance, RISC architecture microcontroller devices specifically designed for I/O control.

The advantages of low power consumption, I/O flexibility, timer functions, oscillator options, Power-down and wake-up functions, Watchdog Timer, buzzer driver, as well as low cost, enhance the versatility of these devices to suit a wide range of application possibilities such as industrial control, consumer products, subsystem controllers, etc.

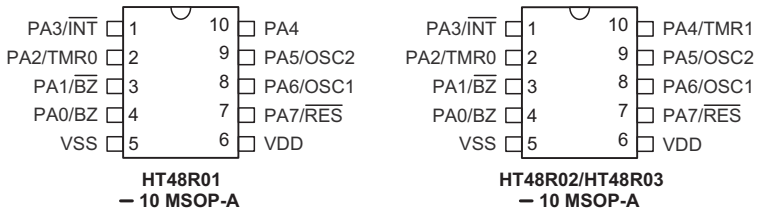
Selection Table

Part No.	VDD	Program Memory	Data Memory	I/O	Timer	External Interrupt	Buzzer	Stack	Package Types
HT48R01	2.2V~5.5V	1K×14	64×8	7 I/O, 1 Input	8-bit×1	1	√	4	10MSOP
HT48R02	2.2V~5.5V	2K×14	96×8	7 I/O, 1 Input	8-bit×2	1	√	6	10MSOP
HT48R03	2.2V~5.5V	4K×15	160×8	7 I/O, 1 Input	8-bit×2	1	√	8	10MSOP

Block Diagram



Pin Assignment



Pin Description

Pin Name	I/O	Configuration Options	Description
PA0/$\overline{BZ}$ PA1/$\overline{BZ}$	I/O	—	Bidirectional 2-line I/O. Each pin can be setup as a wake-up input using a software register. Software instructions determine if each pin is a CMOS output or a Schmitt trigger input. Pull-high resistors can be connected using a pull-high software register. PA0/PA1 are pin-shared with the BZ and $\overline{BZ}$ buzzer function pins.
PA2/TMR0	I/O	—	Bidirectional single line I/O. PA2 can be setup as a wake-up input using a software register. Software instructions determine if the pin is a CMOS output or Schmitt trigger input. A pull-high resistor can be connected using a pull-high software register. This line is pin-shared with the Timer/event 0 counter input.
PA3/$\overline{INT}$	I/O	—	Bidirectional single line I/O. PA3 can be setup as a wake-up input using a software register. Software instructions determine if the pin is a CMOS output or Schmitt trigger input. A pull-high resistor can be connected using a pull-high software register. This line is pin-shared with $\overline{INT}$.
PA4/TMR1	I/O	—	Bidirectional single line I/O. PA4 can be setup as a wake-up input using a software register. Software instructions determine if the pin is a CMOS output or Schmitt trigger input. A pull-high resistor can be connected using a pull-high software register. This line is pin-shared with the Timer/event counter 1 input.
OSC1/PA6 OSC2/PA5	I/O	RC, Crystal, RTC or I/O	Bidirectional 2-line I/O and oscillator pins. If configured as I/Os, software instructions determine if each pin is a CMOS output or a Schmitt trigger input. Pull-high resistors can be connected using a pull-high software register. A configuration option determines the choice of oscillator mode and I/O function. The four oscillator modes are: 1. Internal RC OSC: both pins configured as I/Os 2. External crystal OSC: both pins configured as OSC1/OSC2 3. Internal RC + RTC OSC: both pins configured as OSC2, OSC1. 4. External RC OSC+PA5: PA6 configured as OSC1 pin, PA5 configured as I/O Note: When the system clock is sourced from the internal RC OSC, there are 3 frequency options → 12MHz, 8MHz and 4MHz.
PA7/$\overline{RES}$	I	PA7 or $\overline{RES}$	Active low schmitt trigger reset input or PA7 input.
VDD	—	—	Positive power supply
VSS	—	—	Negative power supply, ground

* All pull-high resistors are controlled by an register option bit.

Absolute Maximum Ratings

Supply Voltage .....V_{SS}–0.3V to V_{SS}+6.0V

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

I_{OL} Total 150mA

Total Power Dissipation 500mW

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

Operating Temperature.....–40°C to 85°C

I_{OH} Total.....–100mA

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	—	f_{SYS}=4MHz	2.2	—	5.5	V
		—	f_{SYS}=8MHz	3.3	—	5.5	V
		—	f_{SYS}=12MHz	4.5	—	5.5	V
I_{DD1}	Operating Current (Crystal OSC, RC OSC)	3V	No load, f_{SYS}=4MHz	—	1	2	mA
		5V		—	2.5	5	mA
I_{DD2}	Operating Current (Crystal OSC, RC OSC)	3V	No load, f_{SYS}=8MHz	—	2	4	mA
		5V		—	4	8	mA
I_{DD3}	Operating Current (Crystal OSC, RC OSC)	5V	No load, f_{SYS}=12MHz	—	6	12	mA
I_{DD4}	Operating Current (Internal RC+RTC OSC, Normal Mode)	3V	No load, f_{SYS}=4MHz	—	1	2	mA
		5V		—	2.5	5	mA
I_{DD5}	Operating Current (Internal RC+RTC OSC, Normal Mode)	3V	No load, f_{SYS}=8MHz	—	2	4	mA
		5V		—	4	8	mA
I_{DD6}	Operating Current (Internal RC+RTC OSC, Normal Mode)	5V	No load, f_{SYS}=12MHz	—	6	12	mA
I_{DD7}	Operating Current (Internal RC+RTC OSC, Slow Mode)	3V	No load, f_{SYS}=32768Hz	—	10	20	μA
		5V		—	20	40	μA
I_{STB1}	Standby Current (WDT Enabled, RTC Off)	3V	No load, system HALT	—	—	5	μA
		5V		—	—	10	μA
I_{STB2}	Standby Current (WDT Disabled, RTC Off)	3V	No load, system HALT	—	—	1	μA
		5V		—	—	2	μA
I_{STB3}	Standby Current (WDT Disabled, RTC On)	3V	No load, system HALT	—	—	5	μA
		5V		—	—	10	μA
V_{IL1}	Input Low Voltage for PA0~PA6, TMR0, TMR1 and $\overline{INT}$	—	—	0	—	$0.3V_{DD}$	V
V_{IH1}	Input High Voltage forPA0~PA6, TMR0, TMR1 and $\overline{INT}$	—	—	$0.7V_{DD}$	—	V_{DD}	V
V_{IL2}	Input Low Voltage (PA7/$\overline{RES}$)	—	—	0	—	$0.4V_{DD}$	V
V_{IH2}	Input High Voltage (PA7/$\overline{RES}$)	—	—	$0.9V_{DD}$	—	V_{DD}	V
V_{LVR1}	Low Voltage Reset 1	—	Configuration option: 4.2V	3.98	4.2	4.42	V
V_{LVR2}	Low Voltage Reset 2	—	Configuration option: 3.15V	2.98	3.15	3.32	V
V_{LVR3}	Low Voltage Reset 3	—	Configuration option: 2.1V	1.98	2.1	2.22	V
I_{OL}	I/O Port Sink Current	3V	$V_{OL}=0.1V_{DD}$	4	8	—	mA
		5V		10	20	—	mA
I_{OH}	I/O Port Source Current	3V	$V_{OH}=0.9V_{DD}$	–2	–4	—	mA
		5V		–5	–10	—	mA
R_{PH}	Pull-high Resistance	3V	—	20	60	100	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, RC OSC)	—	2.2V~5.5V	400	—	4000	kHz
		—	3.3V~5.5V	400	—	8000	kHz
		—	4.5V~5.5V	400	—	12000	kHz
f_{SYS2}	System Clock (Internal RC OSC) (±10%)	4.5V~5.5V	12MHz, Ta=25°C	10800	12000	13200	kHz
		3.3V~5.5V	8MHz, Ta=25°C	7200	8000	8800	kHz
		2.2V~5.5V	4MHz, Ta=25°C	3600	4000	4400	kHz
f_{SYS3}	System Clock (32768 Crystal)	—	—	—	32768	—	Hz
f_{TIMER}	Timer I/P Frequency (TMR)	—	2.2V~5.5V	0	—	4000	kHz
		—	3.3V~5.5V	0	—	8000	kHz
		—	4.5V~5.5V	0	—	12000	kHz
t_{WDTOSC}	Watchdog Oscillator Period	3V	—	45	90	180	μs
		5V	—	32	65	130	μs
t_{RES}	External Reset Low Pulse Width	—	—	1	—	—	μs
t_{SST}	System Start-up Timer Period	—	Wake-up from HALT	—	1024	—	t_{SYS}
t_{INT}	Interrupt Pulse Width	—	—	1	—	—	μs
t_{LVR}	Low Voltage Width to Reset	—	—	0.25	1	2	ms
V_{POR}	VDD Start Voltage to Ensure Power-on Reset	—	—	—	—	100	mV
R_{POR}	VDD Rise Rate to Ensure Power-on Reset	—	—	0.035	—	—	V/ms

Note: $t_{SYS}=1/f_{SYS1}$, $1/f_{SYS2}$ or $1/f_{SYS3}$

Functional Description

Execution Flow

The system clock for the microcontroller is derived from either a crystal or an RC oscillator. The system clock 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 an instruction cycle while decoding and execution takes the next instruction cycle. However, the pipelining scheme causes each instruction to effectively execute in a cycle. If an instruction changes the program counter, two cycles are required to complete the instruction.

Program Counter – PC

The program counter controls the sequence in which the instructions stored in program memory are executed and its contents specify the full range of program memory.

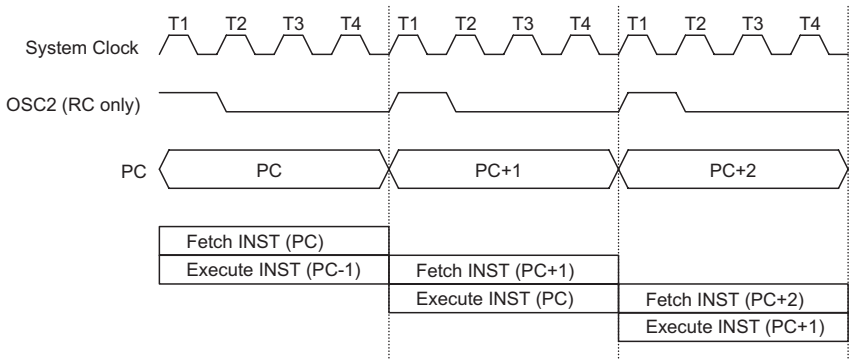
After accessing a program memory word to fetch an instruction code, the contents of the program counter are incremented by one. The program counter then points to the memory word containing the next instruction code.

When executing a jump instruction, conditional skip execution, loading PCL register, subroutine call, initial reset, internal interrupt, external interrupt or return from subroutine, the PC manipulates the 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 the proper instruction. Otherwise proceed with the next instruction.

The lower byte of the program counter (PCL) is a readable and writable register (06H). Moving data into the PCL performs a short jump. The destination will be within 256 locations.

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



Execution Flow

Mode	Program Counter											
	*11	*10	*9	*8	*7	*6	*5	*4	*3	*2	*1	*0
Initial Reset	0	0	0	0	0	0	0	0	0	0	0	0
External Interrupt	0	0	0	0	0	0	0	0	0	1	0	0
Timer/Event Counter Overflow	0	0	0	0	0	0	0	0	1	0	0	0
Skip	Program Counter+2											
Loading PCL	*11	*10	*9	*8	@7	@6	@5	@4	@3	@2	@1	@0
Jump, Call Branch	#11	#10	#9	#8	#7	#6	#5	#4	#3	#2	#1	#0
Return from Subroutine	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

Program Counter

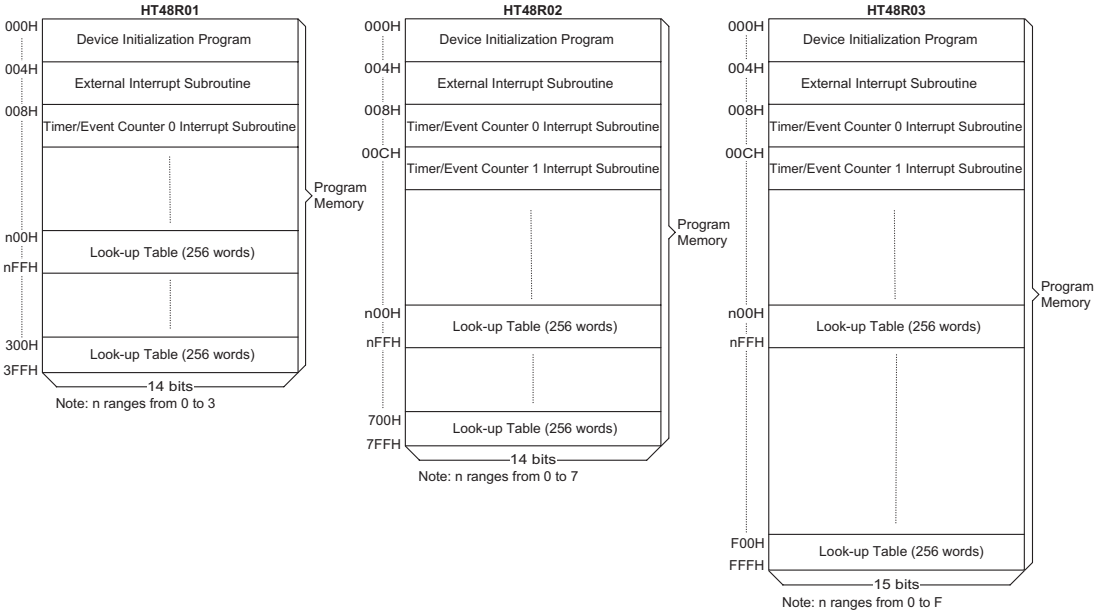
Note: *11~*0: Program Counter bits
S11~S0: Stack register bits
#11~#0: Instruction code bits
@7~@0: PCL bits
For HT48R01, the Program Counter is 10 bits wide, i.e. from *9~*0
For HT48R02, the Program Counter is 11 bits wide, i.e. from *10~*0
For HT48R03, the Program Counter is 12 bits wide, i.e. from *11~*0

Program Memory – ROM

The program memory 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 for the HT48R01, 2048×14 bits for the HT48R02 or 4096×15 bits for the HT48R03, addressed by the program counter and table pointer.

Certain locations in the program memory are reserved for special usage:

- Location 000H
  This area is reserved for program initialization. After chip reset, the program always begins execution at location 000H.
- Location 004H
  This area is reserved for the external interrupt service program. If the $\overline{INT}$ input pin is activated, the interrupt is enabled and the stack is not full, the program begins execution at location 004H.
- Location 008H
  This location is reserved for the Timer/Event Counter 0 interrupt service program. If a timer interrupt results from a Timer/Event Counter 0 overflow, and the interrupt is enabled and the stack is not full, the program begins execution at location 008H.
- Location 00CH (HT48R02/HT48R03 only)
  This location is reserved for the Timer/Event Counter 1 interrupt service program. If a timer interrupt results from a Timer/Event Counter 1 overflow, and the interrupt is enabled and the stack is not full, the program begins execution at location 00CH.
- Table location
  Any location in the program memory can be used as look-up tables. The instructions "TABRDC [m]" (the current page) and "TABRDL [m]" (the last page) transfer the contents of the lower-order byte to the specified data memory, and the higher-order byte to TBLH (08H). Only the destination of the lower-order byte in the table is well-defined, the other bits of the table



Program Memory

Instruction	Table Location											
	*11	*10	*9	*8	*7	*6	*5	*4	*3	*2	*1	*0
TABRDC [m]	P11	P10	P9	P8	@7	@6	@5	@4	@3	@2	@1	@0
TABRDL [m]	1	1	1	1	@7	@6	@5	@4	@3	@2	@1	@0

Table Location

Note: *11~*0: Table location bits P11~P8: Current program counter bits

@7~@0: Table pointer bits

For the HT48R01, the table address location is 10 bits, i.e. from *9~*0

For the HT48R02, the table address location is 11 bits, i.e. from *10~*0

For the HT48R03, the table address location is 12 bits, i.e. from *11~*0

word are transferred to the lower portion of TBLH, and the remaining 2 bits are read as ″0″. The Table Higher-order byte register (TBLH) is read only. The table pointer (TBLP) is a read/write register (07H), which indicates the table location. Before accessing the table, the location must be placed in TBLP. The TBLH is read only and cannot be restored. If the main routine and the ISR (Interrupt Service Routine) both employ the table read instruction, the contents of the TBLH in the main routine are likely to be changed by the table read instruction used in the ISR, and errors may occur. Therefore, using the table read instruction in the main routine and the ISR simultaneously should be avoided. However, if the table read instruction has to be applied in both the main routine and the ISR, the interrupt is supposed to be disabled prior to the table read instruction. It will not be enabled until the TBLH has been backed up. All table related instructions require two cycles to complete the operation. These areas may function as normal program memory depending upon the requirements.

Stack Register – STACK

This is a special part of the memory which is used to save the contents of the Program Counter only. The stack is organised up to 8 levels and is neither part of the data nor part of the program space, and is neither readable nor writable. The activated level is indexed by the stack pointer (SP) and is neither readable nor writeable. At a subroutine call or interrupt acknowledgment, the contents of the program counter are pushed onto the stack. At the end of a subroutine or an interrupt routine, signaled by a return RET or RETI instruction, the program counter is restored to its previous value from the stack. After a 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 will be recorded but the acknowledgment will be inhibited. When the stack pointer is decremented by RET or RETI, the interrupt will be serviced. This feature prevents stack overflow allowing the programmer to use the structure more easily. In a similar case, if the stack is full and a ″CALL″ is subsequently executed, a stack overflow occurs and the first entry will be los. Only the most recent 4 return addresses are stored.

Data Memory – RAM

The data memory is divided into two functional groups: special function registers and general purpose data memory 64×8 for the HT48R01, 96×8 for the HT48R02 or 160×8 for the HT48R03. Most are read/write, but some are read only.

The unused space before 20H is reserved for future expanded usage and reading these locations will get ″00H″. The general purpose data memory, addressed from 20H to 5FH (HT48R01), 20H to 7FH (HT48R02) or 20H to BFH (HT48R03), is used for data and control information under instruction commands.

All of the data memory areas can handle arithmetic, logic, increment, decrement and rotate operations directly. Except for some dedicated bits, each bit in the data memory can be set and reset by ″SET [m].i″ and ″CLR [m].i″. They are also indirectly accessible through memory pointer register (MP;01H).

Indirect Addressing Register

Location 00H/02H are indirect addressing registers that are not physically implemented. Any read/write operation of [00H]/[02H] accesses data memory pointed to by MP0 (01H)/MP1 (03H). Reading location 00H itself indirectly will return the result 00H. Writing indirectly results in no operation.

The memory pointer registers (MP0/MP1) are 7-bit registers (HT48R01/HT48R02) or 8 bit registers (HT48R03). The bit 7 of MP0/MP1 (HT48R01/HT48R02) are undefined and reading will return the result ″1″. Any writing operation to MP0/MP1 will only transfer the lower 7-bit data to MP.

Accumulator

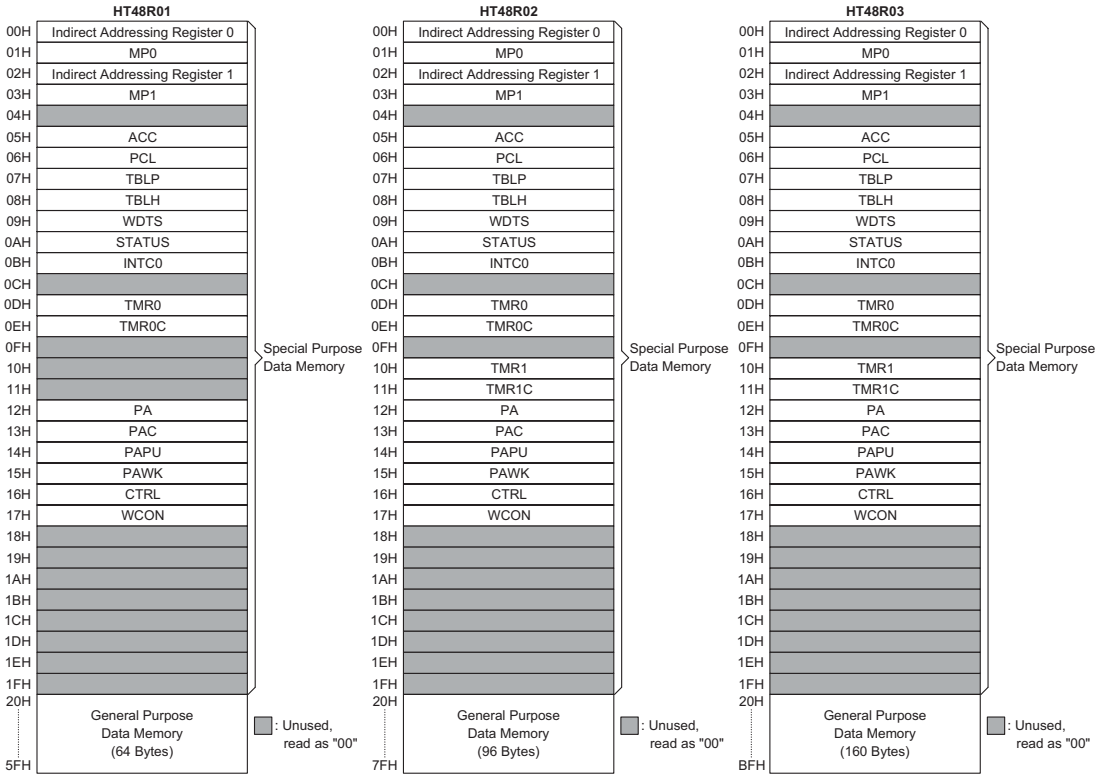
The accumulator is closely related to ALU operations. It is also mapped to location 05H of the data memory and can carry out immediate data operations. The data movement between two data memory locations must pass through the accumulator.

Arithmetic and Logic Unit – ALU

This circuit performs 8-bit arithmetic and logic operations. The ALU 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 ....)

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



RAM Mapping

Status Register – STATUS

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

With the exception of the TO and PDF flags, bits in the status register can be altered by instructions like most other registers. Any data written into the status register will not change the TO or PDF flag. In addition operations related to the status register may give different results from those intended. The TO flag can be affected only by system power-up, a WDT time-out or executing the ″CLR WDT″ or ″HALT″ instruction. The PDF flag can be affected only by executing the ″HALT″ or ″CLR WDT″ instruction or a system power-up.

The Z, OV, AC and C flags generally reflect the status of the latest operations.

Bit No.	Label	Function
0	C	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.
1	AC	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.
2	Z	Z is set if the result of an arithmetic or logic operation is zero; otherwise Z is cleared.
3	OV	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.
4	PDF	PDF is cleared by system power-up or executing the ″CLR WDT″ instruction. PDF is set by executing the ″HALT″ instruction.
5	TO	TO is cleared by system power-up or executing the ″CLR WDT″ or ″HALT″ instruction. TO is set by a WDT time-out.
6~7	—	Unused bit, read as ″0″

Status (0AH) Register

In addition, 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 are important and if the subroutine can corrupt the status register, precautions must be taken to save it properly.

System Control Register

Bit No.	Label	Function
0	CLKMOD	Clock mode selection - select the system clock source 0: High speed clock as system clock - internal RC 1: Low speed clock as system clock - 32.768kHz, and RC oscillator stop Note: This selection is used only in internal RC + RTC mode.
1	QOSC	32768Hz OSC quick start-up oscillating setting 0: quickly startup 1: slow startup
2 3	BZEN0 BZEN1	BZ/$\overline{BZ}$ enable/disable 00: both disabled 01: Reserved 10: BZ only enabled 11: BZ and $\overline{BZ}$ enabled When BZ or $\overline{BZ}$ are disabled, the I/O port will have general I/O functions. If enabled, the BZ or $\overline{BZ}$ outputs will still be controlled by the related I/O port control and data settings. Refer to the I/O chapter for details.
4~5	—	Unused bit, read as ″0″
6	BZCS	BZCS, buzzer clock source, 0/1: Timer0/Timer1
7		Unused bit, read as ″0″

CTRL (16H) Register

Note: For the HT48R01, BZCS is always 0 no matter what value is written into it; i.e., clock source for Buzzer is only from timer0.

Interrupt

The device provides an external interrupt and internal timer/event counter interrupts. The Interrupt Control Register (INTC;0BH) contains the interrupt control bits to set the enable or disable and the interrupt request flags.

Once an interrupt subroutine is serviced, all the other interrupts will be blocked, by clearing the EMI bit. This scheme may prevent any further interrupt nesting. Other interrupt requests may happen during this interval but only the interrupt request flag is recorded. If a certain interrupt requires servicing within the service routine, the EMI bit and the corresponding bit in the INTC register may be set to allow interrupt nesting. If 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 must be prevented from becoming full.

All these kinds of interrupts have a wake-up capability. As an interrupt is serviced, a control transfer occurs by pushing the program counter onto the stack, followed by a branch to a subroutine at specified location in the program memory. Only the program counter is pushed onto the stack. If the contents of the register or status register are altered by the interrupt service program which corrupts the desired control sequence, the contents should be saved in advance.

External interrupts are triggered by a high to low transition of $\overline{INT}$ and the related interrupt request flag (EIF; bit 4 of INTC) will be set. When the interrupt is enabled, the stack is not full and the external interrupt is active, a subroutine call to location 04H will occur. The interrupt request flag (EIF) and EMI bits will be cleared to disable other interrupts.

The internal timer/event counter interrupt is initialised by setting the timer/event counter interrupt request flag (TF; bit 5 of INTC), caused by a timer overflow. When the interrupt is enabled, the stack is not full and the TF bit is set, a subroutine call to location 08H will occur. The related interrupt request flag,TF, will be reset and the EMI bit cleared to disable further interrupts.

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

Interrupts, occurring in the interval between the rising edges of two consecutive T2 pulses, will be 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.

Interrupt Source	Priority	Vector
External Interrupt	1	04H
Timer/Event Counter 0 Overflow	2	08H

Interrupt Subroutine Vector for HT48R01

Interrupt Source	Priority	Vector
External Interrupt	1	04H
Timer/Event Counter 0 Overflow	2	08H
Timer/Event Counter 1 Overflow	3	0CH

Interrupt Subroutine Vector for HT48R02/HT48R03

Once the interrupt request flags (T0F/ T1F, EIF) are set, they will remain in the INTC register until the interrupts are serviced or cleared by a software instruction. It is recommended that a program does not use the ″CALL subroutine″ within the interrupt subroutine. Interrupts often occur in an unpredictable manner or need to be serviced immediately in some applications. If only one stack is left and enabling the interrupt is not well controlled, the original control sequence will be damaged once the ″CALL″ operates in the interrupt subroutine.

Oscillator Configuration

There are 4 different oscillator modes implemented in the microcontroller, which are selected by configuration options. All of them are designed for system clocks, namely the external RC oscillator (ERC), external crystal oscillator (ECRY), internal RC oscillator with I/O(IRC) and internal RC oscillator with RTC OSC (IRC+RTC). No matter what oscillator type is selected, the signal provides the system clock. The Power-down mode stops the system oscillator, except for the RTC oscillator, and resists external signals to conserve power.

Bit No.	Label	Function
0	EMI	Controls the master (global) interrupt (1= enabled; 0= disabled)
1	EEI	Controls the external interrupt (1= enabled; 0= disabled)
2	ET0I	Controls the timer/event counter 0 interrupt (1= enabled; 0= disabled)
3, 6~7	—	Unused bit, read as ″0″
4	EIF	External interrupt request flag (1= active; 0= inactive)
5	T0F	Internal timer/event counter 0 request flag (1= active; 0= inactive)

INTC 0 (0BH) Register for HT48R01

Bit No.	Label	Function
0	EMI	Controls the master (global) interrupt (1= enabled; 0= disabled)
1	EEI	Controls the external interrupt (1= enabled; 0= disabled)
2	ET0I	Controls the timer/event counter 0 interrupt (1= enabled; 0= disabled)
3	ET1I	Controls the timer/event counter 1 interrupt (1= enabled; 0= disabled)
4	EIF	External interrupt request flag (1= active; 0= inactive)
5	T0F	Internal timer/event counter 0 request flag (1= active; 0= inactive)
5	T1F	Internal timer/event counter 1 request flag (1= active; 0= inactive)
7	—	Unused bit, read as ″0″

INTC 0 (0BH) Register for HT48R02/HT48R03

If the configuration options select the IRC+RTC, the device supports two kinds of system clock. When combined with the Power-down function, it forms three operation modes. The two kinds of system clock are internal RC oscillator or RTC OSC (32768Hz) which is selected by the CTRL register CLKMOD bit. The three operation modes are named as Normal, Slow, or Idle mode. The following tables shows their relationship.

If an RC oscillator is used, an external resistor between OSC1 and VDD is required whose resistance must range from 24kΩ to 1.5MΩ. The RC oscillator provides the most cost effective solution. The frequency of oscillation may vary with VDD, temperatures and the chip itself due to process variations. It is, therefore, not suitable for timing sensitive operations where an accurate oscillator frequency is desired.

If the crystal oscillator is used, 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 demanded. Instead of a crystal, a resonator can also be connected between OSC1 and OSC2 to obtain a frequency reference, but two external capacitors connected to OSC1 and OSC2 are required. If an internal RC oscillator is used, OSC1 and OSC2 can be selected as general I/O lines or as a 32768Hz crystal (RTC) oscillator. The frequencies of internal oscillator can be 12MHz, 8MHz and 4MHz which is selected by configuration options.

The WDT oscillator is a free running on-chip RC oscillator, and no external components are required. Even if the system enters the power down mode, the system clock is stopped, but the WDT oscillator still works with a period of approximately 65μs at 5V. The WDT oscillator can be disabled by configuration options to conserve power.

Watchdog Timer – WDT

The WDT clock source may come from a dedicated RC oscillator (WDT oscillator), RTC clock or instruction clock (system clock divided by 4) which is determined by option. This timer is designed to prevent a software malfunction or sequence from jumping to an unknown location with unpredictable results. The Watchdog Timer can be disabled by options. If the Watchdog Timer is disabled, all the executions related to the WDT result in no operation. The RTC clock is enabled only in the internal RC+RTC mode.

The WDT clock (f_S) is further divided by an internal counter to give longer watchdog time-outs.

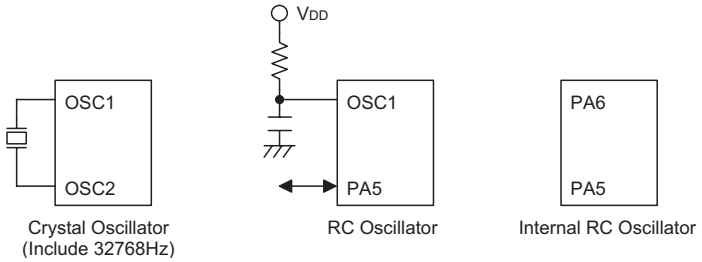
Once the internal WDT oscillator (RC oscillator with a period of 65μs at 5V normally) is selected, it is first divided by 256 (8-stage) to get the nominal time-out period of approximately 17ms at 5V. This time-out period may vary with temperatures, VDD and process variations. By invoking the WDT prescaler, longer time-out periods can be realized. Writing data to WS2, WS1, WS0 (bit 2,1,0 of the WDTS) can give different time-out periods. If WS2, WS1 and WS0 are all equal to ″1″, the division ratio is up to 1:128, and the maximum time-out period is 2.1s at 5V seconds. If the WDT oscillator is disabled, the WDT clock may still come from the instruction clock and operate in the same manner except that in the HALT state the WDT may stop counting and lose its protecting purpose. In this situation the logic can only be restarted by external logic.

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

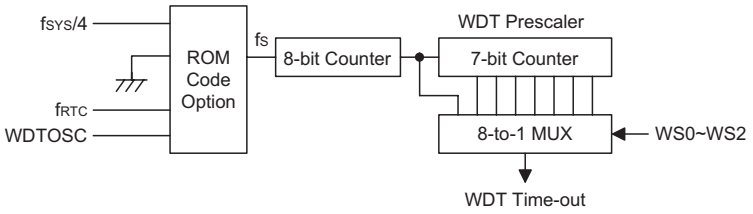
Bit No.	Label	Function
0~2	WS0~WS2	WDT prescaler rate select
3~7	—	Unused bit, read as ″0″

WDTS (09H) Register

HALT Instruction	CLKMOD	RC Oscillator	32768Hz	System Clock	Mode
During run state (HALT not execute)	0	On	On	RC oscillator	Normal
	1	Off	On	32768Hz	Slow
During run state (HALT execute)	x	Off	On	HALT	Idle



System Oscillator



Watchdog Timer

WS2	WS1	WS0	Division Ratio
0	0	0	1:1
0	0	1	1:2
0	1	0	1:4
0	1	1	1:8
1	0	0	1:16
1	0	1	1:32
1	1	0	1:64
1	1	1	1:128

The WDT overflow under normal operation will initialise a ″chip reset″ and set the status bit ″TO″. But in the HALT mode, the overflow will initialise a ″warm reset″, and only the Program Counter and SP are reset to zero. To clear the contents of WDT (including the WDT prescaler), three methods are adopted; external reset (a low level to RES), software instruction and a ″HALT″ instruction. The software instruction include ″CLR WDT″ and the other set – ″CLR WDT1″ and ″CLR WDT2″. Of these two types of instruction, only one can be active depending on the option – ″CLR WDT times selection option″. If the ″CLR WDT″ is selected (i.e. CLRWDT times equal one), any execution of the ″CLR WDT″ instruction will clear the WDT. In the case that ″CLR WDT1″ and ″CLR WDT2″ are chosen (i.e. CLRWDT times equal two), these two instructions must be executed to clear the WDT; otherwise, the WDT may reset the chip as a result of time-out.

The WDT control register contains 4 bits of WDT enable bits. WDT can be enable by either WDT mask option or WDT control register (WDTEN[3:0]=0101B) and be disable by both being disable.

Bit No.	Label	Function
0~3	WDTEN0~ WDTEN3	Bit3~0, WDTEN3~WDTEN0= 1010B: WDT disable others: enable (using 0101B to enable WDT is strongly recommended for the highest noise immunity)
4~5	—	Unused bit, read as ″0″

Bit No.	Label	Function
6~7	INTES0~ INTES1	External interrupt edge selection (default=10) 00: disable 01: rising edge trigger 10: falling edge trigger 11: dual edge trigger

WCON (17H) Register

Power Down Operation – HALT

The HALT mode is initialised by the ″HALT″ instruction and results in the following...

- The system oscillator will be turned off but the WDT oscillator keeps running (if the WDT oscillator is selected).
- The contents of the on chip RAM and registers remain unchanged.
- WDT and WDT prescaler will be cleared and recounted again (if the WDT clock is from the WDT oscillator).
- All of the I/O ports maintain their original status.
- The PDF flag is set and the TO flag is cleared.

The system can leave the HALT mode by means of an external reset, an interrupt, an external falling edge signal on port A or a WDT overflow. An external reset causes a device initialisation and the WDT overflow performs a ″warm reset″. After the TO and PDF flags are examined, the reason for chip reset can be determined. The PDF flag is cleared by system power-up or executing the ″CLR WDT″ instruction and is set when executing the ″HALT″ instruction. The TO flag is set if the WDT time-out occurs, and causes a wake-up that only resets the Program Counter and Stack Pointer; the others keep their original status.

Both 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 options. Awakening from an I/O port stimulus, the program will resume execution of the next instruction. If it is awakened by an interrupt, two sequences may happen. If the related interrupt is disabled or the interrupt is enabled but the stack is full, the program will resume execution at the next instruction. If the interrupt is enabled and the stack is not full, the regular interrupt response takes place. If an interrupt request flag is set to

″1″ before entering the HALT mode, the wake-up function of the related interrupt will be disabled. Once a wake-up event occurs, it takes 1024 t_{SYS} (system clock period) to resume normal operation. In other words, a dummy period will be inserted after wake-up. If the wake-up results from an interrupt acknowledgment, the actual interrupt subroutine execution will be delayed by one or more cycles. If the wake-up results in the next instruction execution, this will be executed immediately after the dummy period is finished.

To minimise 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 can occur:

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

The WDT time-out during HALT is different from other chip reset conditions, since it can perform a ″warm reset″ that resets only the Program Counter and SP, leaving the other circuits in their original state. Some registers remain unchanged during other reset conditions. Most registers are reset to the ″initial condition″ when the reset conditions are met. By examining the PDF and TO flags, the program can distinguish between different ″chip resets″.

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

Note: ″u″ means ″unchanged″

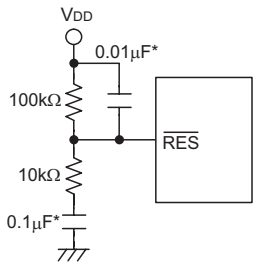
To guarantee that the system oscillator is started and stabilised, the SST (System Start-up Timer) provides an extra-delay of 1024 system clock pulses when the system reset (power-up, WDT time-out or $\overline{\text{RES}}$ reset) or the system awakes from the HALT state.

When a system reset occurs, the SST delay is added during the reset period. Any wake-up from HALT will enable the SST delay.

An extra option load time delay is added during a system reset (power-up, WDT time-out during normal mode or $\overline{\text{RES}}$ reset).

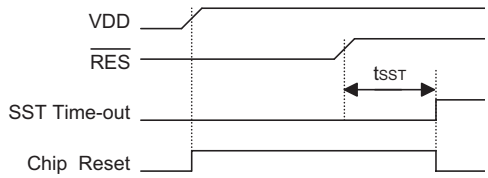
The functional unit chip reset status are shown below.

Program Counter	000H
Interrupt	Disable
Prescaler	Clear
WDT	Clear. After master reset, WDT begins counting
Timer/Event Counter	Off
Input/Output Ports	Input mode
Stack Pointer	Points to the top of the stack

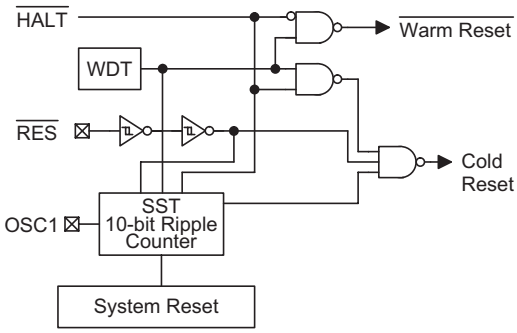


Reset Circuit

Note: ″*″ Make the length of the wiring, which is connected to the $\overline{\text{RES}}$ pin as short as possible, to avoid noise interference.



Reset Timing Chart



Reset Configuration

The register states are summarised in the following table.

Register	Reset (Power-on)	WDT time-out (Normal Operation)	$\overline{\text{RES}}$ Reset (Normal Operation)	$\overline{\text{RES}}$ Reset (HALT)	WDT Time-out (HALT)*
Program Counter	000H	000H	000H	000H	000H
MP0 (HT48R01/02)	1xxx xxxx	1uuu uuuu	-uuu uuuu	-uuu uuuu	1uuu uuuu
MP1 (HT48R01/02)	1xxx xxxx	1uuu uuuu	-uuu uuuu	-uuu uuuu	1uuu uuuu
MP0 (HT48R03)	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
MP1 (HT48R03)	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	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
WDTS	---- -111	---- -111	---- -111	---- -111	---- -uuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--01 uuuu	--11 uuuu
INTC0 (HT48R01)	--00 -000	--00 -000	--00 -000	--00 -000	--uu -uuu
INTC0 (HT48R02/03)	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
TMR0	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR0C	0000 1000	0000 1000	0000 1000	0000 1000	uuuu uuuu
TMR1	xxxx xxxx	xxxx xxxx	xxxx xxxx	xxxx xxxx	uuuu uuuu
TMR1C	0000 1---	0000 1---	0000 1---	0000 1---	uuuu u---
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAPU	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
PAWK	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
CTRL	-0-- 0000	-0-- 0000	-0-- 0000	-0-- 0000	-u-- uuuu
WCON	10-- 1010	10-- 1010	10-- 1010	10-- 1010	uu-- uuuu

Note: "*" means "warm reset"
"-" not implement
"u" means "unchanged"
"x" means "unknown"

Timer/Event Counter

One or two timer/event counters are implemented in the microcontroller. The timer/event counter contains an 8-bit programmable count-up counter and the clock may come from an external source, the system clock or RTC clock.

Using an external clock input allows the user to count external events, measure time internals or pulse widths, or generate an accurate time base, while using the internal clock allows the user to generate an accurate time base.

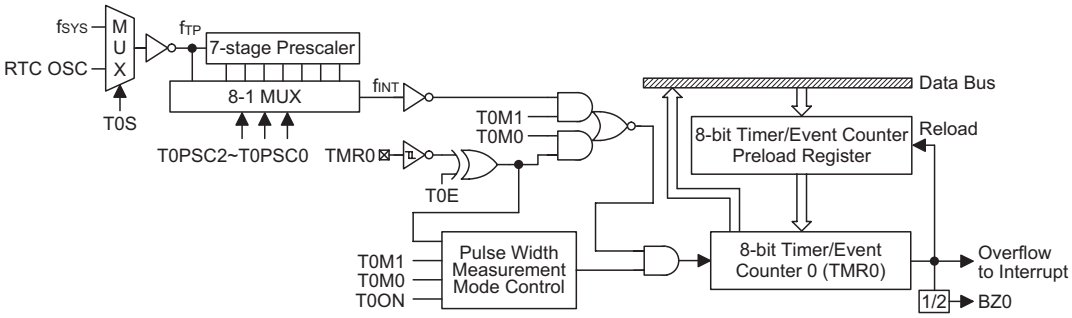
The timer/event counter can generate a buzzer signal by using an external or internal clock.

There are 2 registers related to the timer/event counter; TMR0 [0DH], TMR0C [0EH] (TMR1 [10H]), TMR1C [11H]). Two physical registers are mapped to the TMR location; writing TMR0 (TMR1) places the start value into the timer/event counter preload register while reading TMR0 (TMR1) retrieves the contents of the timer/event counter. The TMR0C (TMR1C) is a timer/ event counter control register, which defines some options.

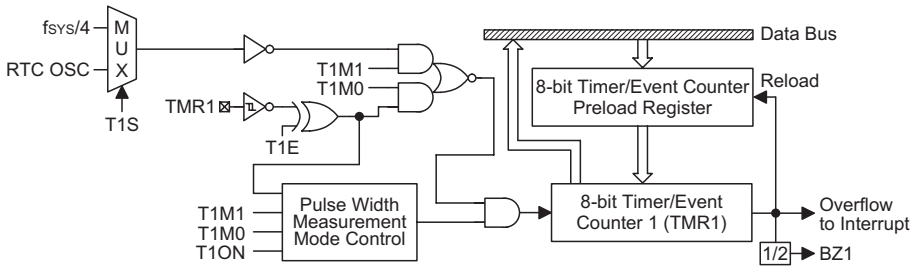
The T0M0, T0M1 (T1M0, T1M1) bits define the operating mode. The event count mode is used to count external events, which means the clock source comes from an external TMR0 (TMR1) pin. The timer mode functions as a normal timer with the clock source coming from the f_{INT} clock. The pulse width measurement mode can be used to count the high or low level duration of the external signal TMR0 (TMR1). The counting is based on the f_{INT} clock.

In the event count or timer mode, once the timer/event counter starts counting, it will count from the current contents in the timer/event counter to FFH. Once an overflow occurs, the counter is reloaded from the timer/event counter preload register and generates an interrupt request flag (T0F; bit 5 of INTC0 or T1F bit 6 of INTC0) at the same time.

In the pulse width measurement mode with the values of T0ON and T0E (T1ON and T1E) equal to 1, after the TMR0 (TMR1) has received a low to high transient (or high to low if T0E (T1E) is ″0″), it will start counting until TMR0 (TMR1) returns to its original level and resets T0ON (T1ON). The measured result remains in the timer/event counter even if the activated transient occurs again. In other words, only a single cycle measurement can be implemented. Not until the T0ON (T1ON) bit has been set again, will the cycle measurement function again as long as it receives further transient pulses. Note that, in this operating mode, the timer/event counter starts counting not according to the logic level but according to the transient edges. In the case of counter overflows, the counter is reloaded from the timer/event counter preload register and issues the interrupt request just like the other two modes. To enable the counting operation, the timer ON bit, T0ON (T1ON) should be set to 1. In the pulse width measurement mode, the T0ON (T1ON) will be cleared automatically after the measurement cycle is completed. But in the other two modes the T0ON (T1ON) can only be reset by instructions. The overflow of the timer/event counter is one of the wake-up sources. No matter what the operation mode is, writing a 0 to ETI can disable the interrupt service.



Timer/Event Counter 0



Timer/Event Counter 1 - HT48R02/HT48R03 only

In the case of timer/event counter OFF condition, writing data to the timer/event counter preload register will also reload that data to the timer/event counter. But if the timer/event counter is turned on, data written to it will only be kept in the timer/event counter preload register. The timer/event counter will still operate until overflow occurs. When the timer/event counter is read, the clock will be blocked to avoid errors. As clock blocking may results in a counting error, this must be taken into consideration by the programmer.

The bit 0~2 of the TMR0C can be used to define the pre-scaling stages of the internal clock sources of the timer/event counter. The definitions are as shown. The timer/event counter overflow signals can be used to generate signals for the buzzer.

Bit No.	Label	Function
0~2	T0PSC0~ T0PSC2	To define the prescaler stages, T0PSC2, T0PSC1, T0PSC0= 000: $f_{INT}=f_{TP}$ 001: $f_{INT}=f_{TP}/2$ 010: $f_{INT}=f_{TP}/4$ 011: $f_{INT}=f_{TP}/8$ 100: $f_{INT}=f_{TP}/16$ 101: $f_{INT}=f_{TP}/32$ 110: $f_{INT}=f_{TP}/64$ 111: $f_{INT}=f_{TP}/128$
3	T0E	To define the TMR active edge of the timer/event counter In event counter mode (T0M1,T0M0)=(0,1): 1: count on falling edge 0: count on rising edge In pulse width measurement mode (T0M1,T0M0)=(1,1): 1: start counting on rising edge, stop on falling edge 0: start counting on falling edge, stop on rising edge
4	T0ON	To enable or disable timer counting (0=disabled; 1=enabled)
5	T0S	Timer clock source selection 0: f_{SYS} 1: RTC
6 7	T0M0 T0M1	To define the operating mode (T0M1, T0M0) 01=Event count mode (external clock) 10=Timer mode (internal clock) 11=Pulse width measurement mode 00=Unused

TMR0C (0EH) Register

Bit No.	Label	Function
0~2	—	Unused bit, read as ″0″
3	T1E	To define the TMR active edge of the timer/event counter In event counter mode (T1M1,T1M0)=(0,1): 1: count on falling edge 0: count on rising edge In pulse width measurement mode (T1M1,T1M0)=(1,1): 1: start counting on rising edge, stop on falling edge 0: start counting on falling edge, stop on rising edge
4	T1ON	To enable or disable timer counting (0=disabled; 1=enabled)
5	T1S	Timer clock source selection 0: $f_{SYS}/4$ 1: RTC
6 7	T1M0 T1M1	To define the operating mode (T1M1, T1M0) 01=Event count mode (external clock) 10=Timer mode (internal clock) 11=Pulse width measurement mode 00=Unused

TMR1C (11H) Register

Input/Output Ports

There are 7 bi-directional input/output lines and 1 input line in the microcontroller, labeled as PA, which are mapped to the data memory of [12H]. All of the I/O ports can be used for input or output operations. For input operation, these ports are non-latching, that is, the inputs must be ready at the T2 rising edge of instruction ″MOV A,[m]″ (m=12H). For output operation, all the data is latched and remains unchanged until the output latch is rewritten. Each I/O line has its own control register (PAC) to control the input/output configuration (PA7 for input only). With this control register, a CMOS output or Schmitt trigger input (with or without pull-high resistor structures) can be reconfigured dynamically (i.e. on-the-fly) under software control (the PA7 only provide input mode). To function as an input, the corresponding latch of the control register must write ″1″. The input source also depends on the control register. If the control register bit is ″1″, the input will read the pad state. If the control register bit is ″0″, the contents of the latches will move to the internal bus. The latter is possible in the ″read-modify-write″ instruction. For output function, CMOS is the only configuration. These control register is mapped to locations 13H. After a chip reset, these input/output lines remain at high levels or floating state (dependent on pull-high options). Each bit of these input/output latches can be set or cleared by ″SET [m].i″ αvδ ″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 the accumulator.

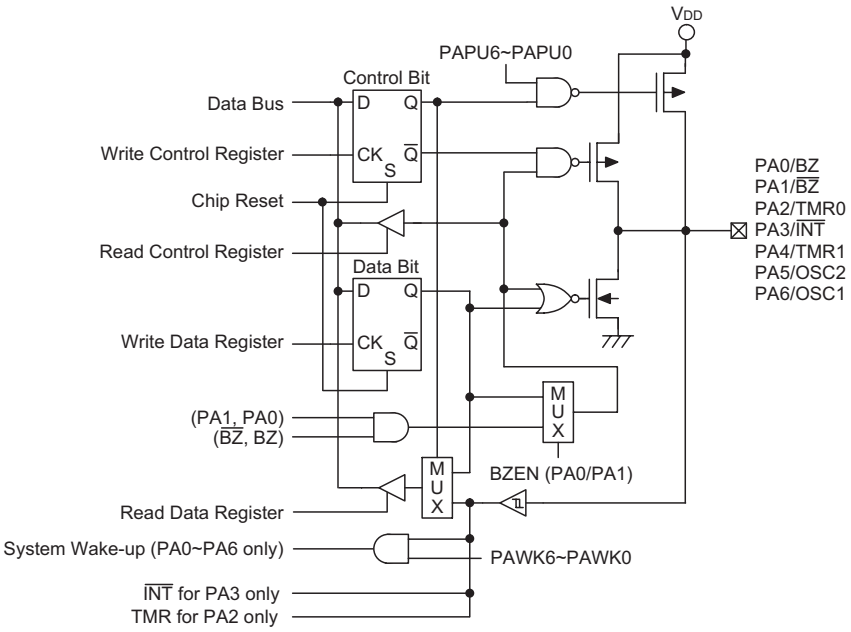
- Wake up and pull-high function
  Each line (except PA7) of PA port supports waking-up MCU and pull-high function which are controlled by PAWK, PAPU registers respectively. PA7 hasn't wake-up and pull-high function.

Bit No.	Label	Function
0~6	PAWK0~ PAWK6	PAWKn= 0, PAn wake-up is disable PAWKn=1, PAn wake-up is enable
7	—	Unused bit, read as ″0″

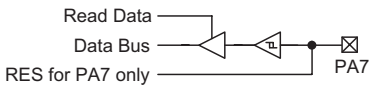
PAWK (15H) Register

Bit No.	Label	Function
0~6	PAPU0~ PAPU6	PAPUn= 0, PAn pull-up is disable PAPUn=1, PAn pull-up is enable
7	—	Unused bit, read as ″0″

PAPU (14H) Register



Input/Output Ports (PA0~PA6)



Input/Output Ports (PA7)

- Buzzer Function

  PA0 and PA1 are pin-shared with the BZ and $\overline{BZ}$ buzzer signals, respectively. If the Buzzer option is selected, then if these pins are setup as outputs, the signals on PA0 (or PA1) will be the Buzzer signal. If setup as inputs, they will always retain their original input functions.

  The buzzer output signals (in output mode) are controlled by the PA0 data register only. The truth table for PA0/BZ and PA1/$\overline{BZ}$ are listed below. Port A also has a CMOS or Schmitt trigger input configuration option (All port A I/O lines are controlled by a option bit).

  The truth table for PA0/BZ and PA1/$\overline{BZ}$ is as shown.

PA0 I/O	I	I	I	I	O	O	O	O	O	O	O	O
PA1 I/O	I	O	O	O	I	I	I	O	O	O	O	O
PA0 Mode	x	x	x	x	C	B	B	C	B	B	B	B
PA1 Mode	x	C	B	B	x	x	x	C	C	C	B	B
PA0 Data	x	x	0	1	D	0	1	D_0	0	1	0	1
PA1 Data	x	D	x	x	x	x	x	D_1	D	D	x	x
PA0 Pad Status	I	I	I	I	D	0	B	D_0	0	B	0	B
PA1 Pad Status	I	D	0	B	I	I	I	D_1	D	D	0	B

Note: I: input; O: output; D, D_0, D_1: data;
B: buzzer option, BZ or $\overline{BZ}$; x: don't care
C: CMOS output

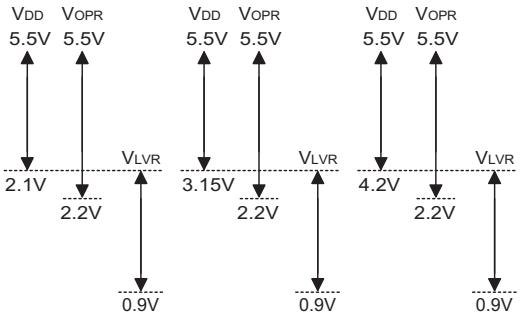
Low Voltage Reset – LVR

The microcontroller provides a low voltage reset circuit in order to monitor the supply voltage of the device. If the supply voltage of the device is within the range 0.9V~V_{LVR}, such as when changing the battery, the LVR will automatically reset the device internally.

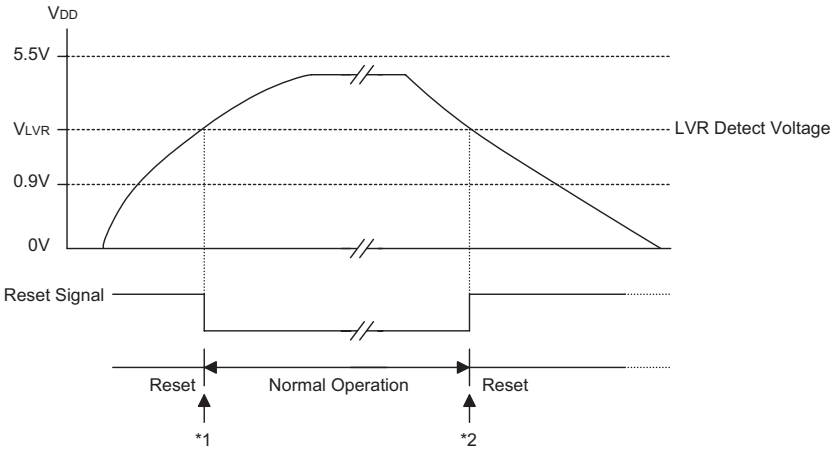
The LVR includes the following specifications:

- The low voltage (0.9V~V_{LVR}) has to remain in this condition for a time greater than 1ms. If the low voltage state does not exceed 1ms, the LVR will ignore it and will not perform a reset function.
- The LVR uses an "OR" function with the external $\overline{RES}$ signal to perform a chip reset.

The relationship between V_{DD} and V_{LVR} is shown below.



Note: V_{OPR} is the voltage range for proper chip operation with a 4MHz system clock.



Low Voltage Reset

Note: *1: To make sure that the system oscillator has stabilised, the SST provides an extra delay of 1024 system clock pulses before entering normal operation.

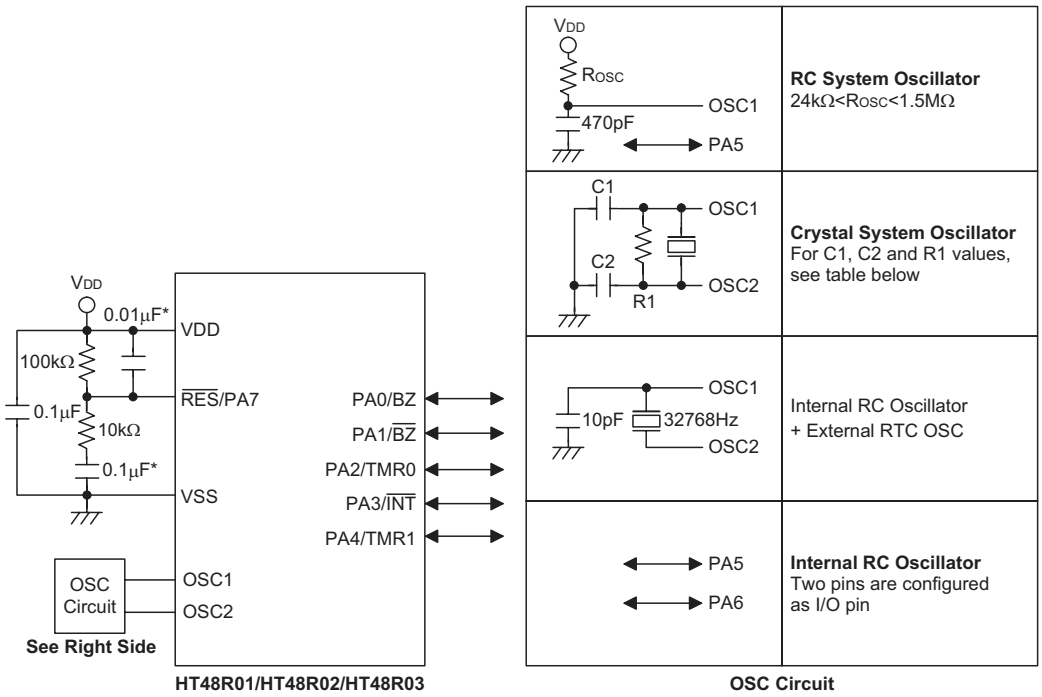
*2: Since the low voltage has to be maintained in its original state and exceed t_{LVR}, therefore a t_{LVR} delay enters the reset mode.

Configuration Options

The following table shows the various configuration options for the microcontroller. All options must be defined for proper system functioning.

Items	Options
1	System oscillator selection • Internal RC + PA5/PA6 • Internal RC + RTC • External Xtal • External RC + PA5
2	Internal RC frequency selection: 4MHz, 8MHz or 12MHz
3	WDT function: enable or disable
4	WDT clock source: WDTOSC, f_{SYS}/4 or RTC OSC
5	CLRWDT instruction(s): one or two clear WDT instruction(s)
6	LVR function: enable or disable
7	LVR selection: 2.1V/3.15V/4.2V
8	$\overline{RES}$ or PA7 input selection

Application Circuits



Note: The resistance and capacitance for the reset circuit should be designed to ensure that VDD is stable and remains in a valid range of the operating voltage before bringing $\overline{RES}$ high.

"*" Make the length of the wiring, which is connected to the $\overline{RES}$ pin as short as possible, to avoid noise interference.

The following table shows the C1, C2 and R1 values corresponding to the different crystal values. (For reference only)

Crystal or Resonator	C1, C2	R1
8MHz Crystal & Resonator	35pF	3.9kΩ
4MHz Crystal	10pF	10kΩ
4MHz Resonator	10pF	12kΩ
3.58MHz Crystal	10pF	12kΩ
3.58MHz Resonator	10pF	12kΩ
2MHz Crystal & Resonator	35pF	12kΩ
1MHz Crystal	68pF	18kΩ
480kHz Resonator	300pF	10kΩ
455kHz Resonator	300pF	10kΩ
429kHz Resonator	300pF	10kΩ
400kHz Resonator	300pF	10kΩ
The function of the resistor R1 is to ensure that the oscillator will switch off should low voltage conditions occur. Such a low voltage, as mentioned here, is one which is less than the lowest value of the MCU operating voltage. Note however that if the LVR is enabled then R1 can be removed.		

Instruction Set Summary

Mnemonic	Description	Instruction Cycle	Flag Affected
Arithmetic			
ADD A,[m]	Add data memory to ACC	1	Z,C,AC,OV
ADDM A,[m]	Add ACC to data memory	1(1)	Z,C,AC,OV
ADD A,x	Add immediate data to ACC	1	Z,C,AC,OV
ADC A,[m]	Add data memory to ACC with carry	1	Z,C,AC,OV
ADCM A,[m]	Add ACC to data memory with carry	1(1)	Z,C,AC,OV
SUB A,x	Subtract immediate data from ACC	1	Z,C,AC,OV
SUB A,[m]	Subtract data memory from ACC	1	Z,C,AC,OV
SUBM A,[m]	Subtract data memory from ACC with result in data memory	1(1)	Z,C,AC,OV
SBC A,[m]	Subtract data memory from ACC with carry	1	Z,C,AC,OV
SBCM A,[m]	Subtract data memory from ACC with carry and result in data memory	1(1)	Z,C,AC,OV
DAA [m]	Decimal adjust ACC for addition with result in data memory	1(1)	C
Logic Operation			
AND A,[m]	AND data memory to ACC	1	Z
OR A,[m]	OR data memory to ACC	1	Z
XOR A,[m]	Exclusive-OR data memory to ACC	1	Z
ANDM A,[m]	AND ACC to data memory	1(1)	Z
ORM A,[m]	OR ACC to data memory	1(1)	Z
XORM A,[m]	Exclusive-OR ACC to data memory	1(1)	Z
AND A,x	AND immediate data to ACC	1	Z
OR A,x	OR immediate data to ACC	1	Z
XOR A,x	Exclusive-OR immediate data to ACC	1	Z
CPL [m]	Complement data memory	1(1)	Z
CPLA [m]	Complement data memory with result in ACC	1	Z
Increment & Decrement			
INCA [m]	Increment data memory with result in ACC	1	Z
INC [m]	Increment data memory	1(1)	Z
DECA [m]	Decrement data memory with result in ACC	1	Z
DEC [m]	Decrement data memory	1(1)	Z
Rotate			
RRA [m]	Rotate data memory right with result in ACC	1	None
RR [m]	Rotate data memory right	1(1)	None
RRCA [m]	Rotate data memory right through carry with result in ACC	1	C
RRC [m]	Rotate data memory right through carry	1(1)	C
RLA [m]	Rotate data memory left with result in ACC	1	None
RL [m]	Rotate data memory left	1(1)	None
RLCA [m]	Rotate data memory left through carry with result in ACC	1	C
RLC [m]	Rotate data memory left through carry	1(1)	C
Data Move			
MOV A,[m]	Move data memory to ACC	1	None
MOV [m],A	Move ACC to data memory	1(1)	None
MOV A,x	Move immediate data to ACC	1	None
Bit Operation			
CLR [m].i	Clear bit of data memory	1(1)	None
SET [m].i	Set bit of data memory	1(1)	None

Mnemonic	Description	Instruction Cycle	Flag Affected
Branch			
JMP addr	Jump unconditionally	2	None
SZ [m]	Skip if data memory is zero	1(2)	None
SZA [m]	Skip if data memory is zero with data movement to ACC	1(2)	None
SZ [m].i	Skip if bit i of data memory is zero	1(2)	None
SNZ [m].i	Skip if bit i of data memory is not zero	1(2)	None
SIZ [m]	Skip if increment data memory is zero	1(3)	None
SDZ [m]	Skip if decrement data memory is zero	1(3)	None
SIZA [m]	Skip if increment data memory is zero with result in ACC	1(2)	None
SDZA [m]	Skip if decrement data memory is zero with result in ACC	1(2)	None
CALL addr	Subroutine call	2	None
RET	Return from subroutine	2	None
RET A,x	Return from subroutine and load immediate data to ACC	2	None
RETI	Return from interrupt	2	None
Table Read			
TABRDC [m]	Read ROM code (current page) to data memory and TBLH	2(1)	None
TABRDL [m]	Read ROM code (last page) to data memory and TBLH (This instruction is not valid for HT48R05A-1/HT48C05)	2(1)	None
Miscellaneous			
NOP	No operation	1	None
CLR [m]	Clear data memory	1(1)	None
SET [m]	Set data memory	1(1)	None
CLR WDT	Clear Watchdog Timer	1	TO,PDF
CLR WDT1	Pre-clear Watchdog Timer	1	TO(4),PDF(4)
CLR WDT2	Pre-clear Watchdog Timer	1	TO(4),PDF(4)
SWAP [m]	Swap nibbles of data memory	1(1)	None
SWAPA [m]	Swap nibbles of data memory with result in ACC	1	None
HALT	Enter power down mode	1	TO,PDF

Note: x: Immediate data

m: Data memory address

A: Accumulator

i: 0~7 number of bits

addr: Program memory address

√: Flag is affected

–: Flag is not affected

(1): If a loading to the PCL register occurs, the execution cycle of instructions will be delayed for one more cycle (four system clocks).

(2): If a skipping to the next instruction occurs, the execution cycle of instructions will be delayed for one more cycle (four system clocks). Otherwise the original instruction cycle is unchanged.

(3): (1) and (2)

(4): The flags may be affected by the execution status. If the Watchdog Timer is cleared by executing the ″CLR WDT1″ or ″CLR WDT2″ instruction, the TO and PDF are cleared.
Otherwise the TO and PDF flags remain unchanged.

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)

TO	PDF	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)

TO	PDF	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)

TO	PDF	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)

TO	PDF	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)

TO	PDF	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)

TO	PDF	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)

TO	PDF	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)

TO	PDF	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 ← Program Counter+1
Program Counter ← addr

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

CLR [m] Clear data memory

Description The contents of the specified data memory are cleared to 0.

Operation [m] ← 00H

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

CLR [m].i Clear bit of data memory

Description The bit i of the specified data memory is cleared to 0.

Operation [m].i ← 0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

CLR WDT Clear Watchdog Timer

Description The WDT is cleared (clears the WDT). The power down bit (PDF) and time-out bit (TO) are cleared.

Operation WDT ← 00H
PDF and TO ← 0

Affected flag(s)

TO	PDF	OV	Z	AC	C
0	0	—	—	—	—

CLR WDT1 Preclear Watchdog Timer

Description Together with CLR WDT2, clears the WDT. PDF and TO are also cleared. Only execution of this instruction without the other preclear instruction just sets the indicated flag which implies this instruction has been executed and the TO and PDF flags remain unchanged.

Operation WDT ← 00H*
PDF and TO ← 0*

Affected flag(s)

TO	PDF	OV	Z	AC	C
0*	0*	—	—	—	—

CLR WDT2 Preclear Watchdog Timer

Description Together with CLR WDT1, clears the WDT. PDF and TO are also cleared. Only execution of this instruction without the other preclear instruction, sets the indicated flag which implies this instruction has been executed and the TO and PDF flags remain unchanged.

Operation WDT ← 00H*
PDF and TO ← 0*

Affected flag(s)

TO	PDF	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 1 are changed to 0 and vice-versa.

Operation [m] ← $[\overline{m}]$

Affected flag(s)

TO	PDF	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 1 are changed to 0 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)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

DAA [m] Decimal-Adjust accumulator for addition

Description The accumulator value is adjusted to the BCD (Binary Coded 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)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

DEC [m] Decrement data memory

Description Data in the specified data memory is decremented by 1.

Operation [m] ← [m]−1

Affected flag(s)

TO	PDF	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 1, leaving the result in the accumulator. The contents of the data memory remain unchanged.

Operation ACC ← [m]−1

Affected flag(s)

TO	PDF	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 (PDF) is set and the WDT time-out bit (TO) is cleared.

Operation Program Counter ← Program Counter+1
PDF ← 1
TO ← 0

Affected flag(s)

TO	PDF	OV	Z	AC	C
0	1	—	—	—	—

INC [m] Increment data memory

Description Data in the specified data memory is incremented by 1

Operation [m] ← [m]+1

Affected flag(s)

TO	PDF	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 1, leaving the result in the accumulator. The contents of the data memory remain unchanged.

Operation ACC ← [m]+1

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

JMP addr Directly jump

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

Operation Program Counter ←addr

Affected flag(s)

TO	PDF	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)

TO	PDF	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)

TO	PDF	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)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

NOP No operation

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

Operation Program Counter ← Program Counter+1

Affected flag(s)

TO	PDF	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)

TO	PDF	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)

TO	PDF	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)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

RET Return from subroutine

Description The program counter is restored from the stack. This is a 2-cycle instruction.

Operation Program Counter ← Stack

Affected flag(s)

TO	PDF	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 Program Counter ← Stack
ACC ← x

Affected flag(s)

TO	PDF	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.

Operation Program Counter ← Stack
EMI ← 1

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

RL [m] Rotate data memory left

Description The contents of the specified data memory are rotated 1 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)

TO	PDF	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 1 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)

TO	PDF	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 1 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)

TO	PDF	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 1 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)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

RR [m] Rotate data memory right

Description The contents of the specified data memory are rotated 1 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)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

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

Description Data in the specified data memory is rotated 1 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)

TO	PDF	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 1 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)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	√

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

Description Data of the specified data memory and the carry flag are rotated 1 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)

TO	PDF	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)

TO	PDF	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)

TO	PDF	OV	Z	AC	C
—	—	√	√	√	√

SDZ [m] Skip if decrement data memory is 0

Description The contents of the specified data memory are decremented by 1. If the result is 0, the next instruction is skipped. If the result is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SDZA [m] Decrement data memory and place result in ACC, skip if 0

Description The contents of the specified data memory are decremented by 1. If the result is 0, the next instruction is skipped. The result is stored in the accumulator but the data memory remains unchanged. If the result is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SET [m] Set data memory

Description Each bit of the specified data memory is set to 1.

Operation [m] ← FFH

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SET [m]. i Set bit of data memory

Description Bit i of the specified data memory is set to 1.

Operation [m].i ← 1

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SIZ [m] Skip if increment data memory is 0

Description The contents of the specified data memory are incremented by 1. If the result is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SIZA [m] Increment data memory and place result in ACC, skip if 0

Description The contents of the specified data memory are incremented by 1. If the result is 0, the next instruction is skipped and the result is stored in the accumulator. The data memory remains unchanged. If the result is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

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

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SNZ [m].i Skip if bit i of the data memory is not 0

Description If bit i of the specified data memory is not 0, the next instruction is skipped. If bit i of the data memory is not 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

Operation Skip if [m].i≠0

Affected flag(s)

TO	PDF	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)

TO	PDF	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)

TO	PDF	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)

TO	PDF	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 (1 of the data memories) are interchanged.

Operation [m].3~[m].0 ↔ [m].7~[m].4

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SWAPA [m] Swap data memory and 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)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SZ [m] Skip if data memory is 0

Description If the contents of the specified data memory are 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

Operation Skip if [m]=0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SZA [m] Move data memory to ACC, skip if 0

Description The contents of the specified data memory are copied to the accumulator. If the contents is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

Operation Skip if [m]=0

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	—	—	—

SZ [m].i Skip if bit i of the data memory is 0

Description If bit i of the specified data memory is 0, the following instruction, fetched during the current instruction execution, is discarded and a dummy cycle is replaced to get the proper instruction (2 cycles). Otherwise proceed with the next instruction (1 cycle).

Operation Skip if [m].i=0

Affected flag(s)

TO	PDF	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)

TO	PDF	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.
Note that this instruction is not valid for HT48R05A-1/HT48C05

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

Affected flag(s)

TO	PDF	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)

TO	PDF	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 0 flag is affected.

Operation [m] ← ACC ″XOR″ [m]

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

XOR A,x Logical XOR immediate data to the accumulator

Description Data in the accumulator and the specified data perform a bitwise logical Exclusive_OR operation. The result is stored in the accumulator. The 0 flag is affected.

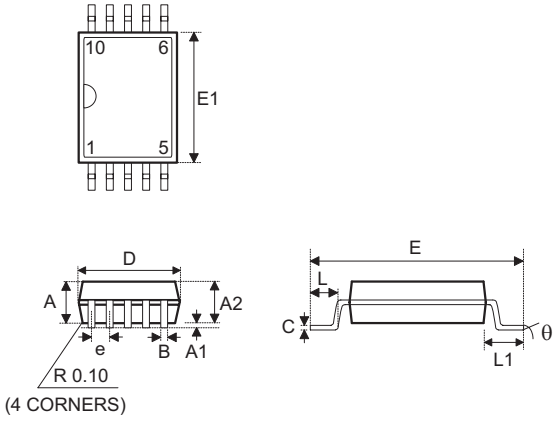
Operation ACC ← ACC ″XOR″ x

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

Package Information

10-pin MSOP Outline Dimensions



Symbol	Dimensions in mm		
	Min.	Nom.	Max.
A	—	—	1.1
A1	0	—	0.15
A2	0.75	—	0.95
B	0.17	—	0.27
C	—	—	0.25
D	—	3	—
E	—	4.9	—
E1	—	3	—
e	—	0.5	—
L	0.4	—	0.8
L1	—	0.95	—
θ	0°	—	8°

Holtek Semiconductor Inc. (Headquarters)
No.3, Creation Rd. II, Science Park, Hsinchu, Taiwan
Tel: 886-3-563-1999
Fax: 886-3-563-1189
http://www.holtek.com.tw

Holtek Semiconductor Inc. (Taipei Sales Office)
4F-2, No. 3-2, YuanQu St., Nankang Software Park, Taipei 115, Taiwan
Tel: 886-2-2655-7070
Fax: 886-2-2655-7373
Fax: 886-2-2655-7383 (International sales hotline)

Holtek Semiconductor Inc. (Shanghai Sales Office)
7th Floor, Building 2, No.889, Yi Shan Rd., Shanghai, China 200233
Tel: 86-21-6485-5560
Fax: 86-21-6485-0313
http://www.holtek.com.cn

Holtek Semiconductor Inc. (Shenzhen Sales Office)
5/F, Unit A, Productivity Building, Cross of Science M 3rd Road and Gaoxin M 2nd Road, Science Park, Nanshan District, Shenzhen, China 518057
Tel: 86-755-8616-9908, 86-755-8616-9308
Fax: 86-755-8616-9533

Holtek Semiconductor Inc. (Beijing Sales Office)
Suite 1721, Jinyu Tower, A129 West Xuan Wu Men Street, Xicheng District, Beijing, China 100031
Tel: 86-10-6641-0030, 86-10-6641-7751, 86-10-6641-7752
Fax: 86-10-6641-0125

Holtek Semiconductor Inc. (Chengdu Sales Office)
709, Building 3, Champagne Plaza, No.97 Dongda Street, Chengdu, Sichuan, China 610016
Tel: 86-28-6653-6590
Fax: 86-28-6653-6591

Holmate Semiconductor, Inc. (North America Sales Office)
46729 Fremont Blvd., Fremont, CA 94538
Tel: 1-510-252-9880
Fax: 1-510-252-9885
http://www.holmate.com

Copyright © 2006 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's products are not authorized for use as critical components in life support devices or systems. 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.