USB Multimedia Keyboard Encoder 8-Bit MCU

Technical Document

- Tools Information
- FAQs
- Application Note

Features

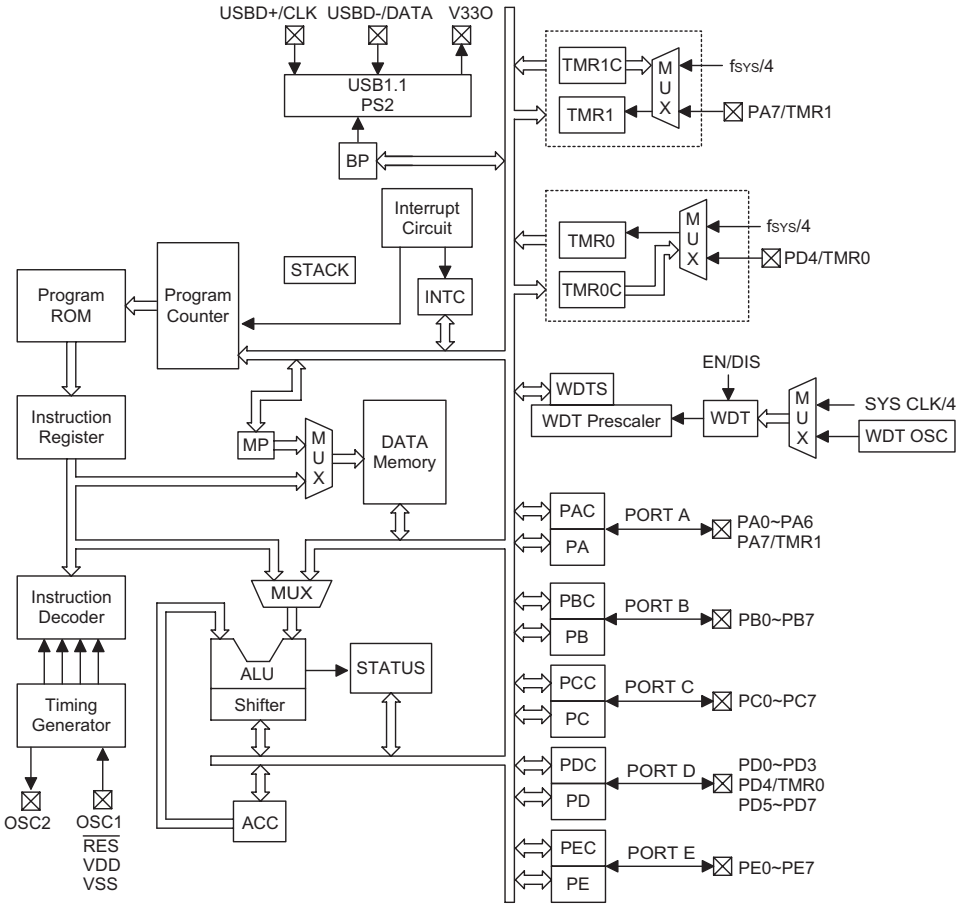
- Operating voltage:
  f_{SYS}=6MHz: 2.2V~5.5V
  f_{SYS}=12MHz: 3.0V~5.5V
- 40 bidirectional I/O lines (max.)
- 8-bit programmable timer/event counter with overflow interrupt (shared with PD4, vector 08H)
- 16-bit programmable timer/event counter and overflow interrupts (shared with PA7, vector 0CH)
- Crystal oscillator (6MHz or 12MHz)
- Watchdog Timer
- PS2 and USB modes supported
- USB 2.0 low speed function
- 4 endpoints supported (endpoint 0 included)
- 6144×16 program memory ROM
- 224×8 data memory RAM
- One internal USB interrupt (vector 04H)
- All I/O ports support wake-up options
- HALT function and wake-up feature reduce power consumption
- 8-level subroutine nesting
- Up to 0.33μs instruction cycle with 12MHz system clock at V_{DD}=5V
- Bit manipulation instruction
- 16-bit table read instruction
- 63 powerful instructions
- All instructions in one or two machine cycles
- Optional 3-battery mode 2.4V LVR/2.6V LVD (±0.1V) by option, Low battery detector with internal bit
- Optional 2-battery mode 2.2V LVR/2.4V LVD (±0.1V) by option, Low battery detector with internal bit
- Operating voltage from 4.0V to 5.5V
  (For Connect USB/PS2 Mode)
- Operating voltage from 2.2V to 3.3V
  (For Pure Cal. Mode)
- 32-pin QFN, 48-pin SSOP packages

General Description

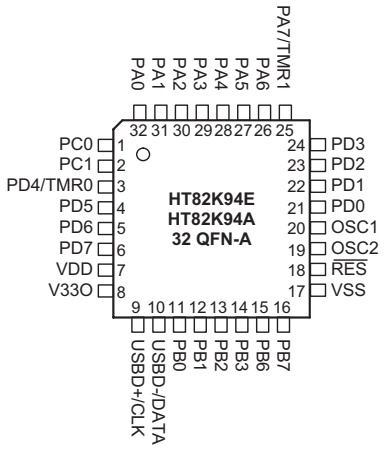
This device is an 8-bit high performance RISC architecture microcontroller designed for USB product applications. It is particularly suitable for use in products such as keyboards and keyboard with calculator. A HALT feature is included to reduce power consumption.

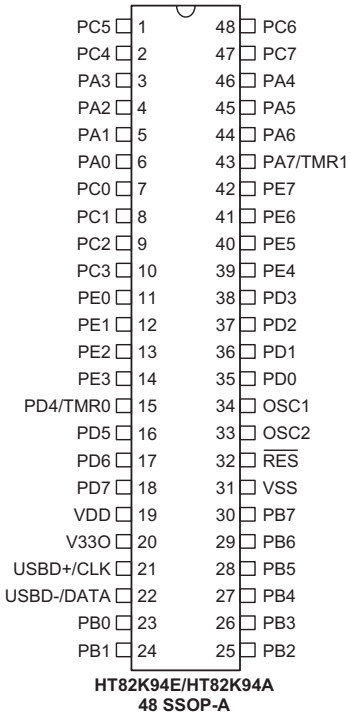
The mask version HT82K94A is fully pin and functionally compatible with the OTP version HT82K94E device.

Block Diagram



Pin Assignment





Pin Description

Pin Name	I/O	ROM Code Option	Description
PA0~PA6 PA7/TMR1	I/O	Pull-high Wake-up CMOS/NMOS/PMOS	Bidirectional 8-bit input/output port. Each bit can be configured as a wake-up input by ROM code option. The input or output mode is controlled by PAC (PA control register). Pull-high resistor options: PA0~PA7 CMOS/NMOS/PMOS options: PA0~PA7 Wake-up options: PA0~PA7 PA7 is pin-shared with TMR1 input, respectively.
PB0~PB7	I/O	Pull-high Wake-up	Bidirectional 8-bit input/output port. Software instructions determine the CMOS output or Schmitt trigger input with pull-high resistor (determined by pull-high options). Wake-up options: PB0~PB7
PC0~PC7	I/O	Pull-high Wake-up	Bidirectional I/O lines. Software instructions determine the CMOS output or Schmitt trigger input with pull-high resistor (determined by pull-high options). Wake-up options: PC0~PC7
PD0~PD3 PD4/TMR0 PD5~PD7	I/O	Pull-high Wake-up	Bidirectional I/O lines. Software instructions determine the CMOS output or Schmitt trigger input with pull-high resistor (determined by pull-high options). Wake-up options: PD0~PD7 PD4 is pin-shared with TMR0 input.
PE0~PE7	I/O	Pull-high Wake-up	Bidirectional I/O lines. Software instructions determine the CMOS output or Schmitt trigger input with pull-high resistor (determined by pull-high options). Wake-up options: PE0~PE7

Pin Name	I/O	ROM Code Option	Description
VSS	—	—	Negative power supply, ground
$\overline{RES}$	I	—	Schmitt trigger reset input. Active low
VDD	—	—	Positive power supply
V33O	O	—	3.3V regulator output
USBD+/CLK	I/O	—	USBD+ or PS2 CLK I/O line USB or PS2 function is controlled by software control register
USBD-/DATA	I/O	—	USBD- or PS2 DATA I/O line USB or PS2 function is controlled by software control register
OSC1 OSC2	I O	—	OSC1, OSC2 are connected to a 6MHz or 12MHz Crystal/resonator (determined by software instructions) for the internal system clock.

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

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

Operating Temperature ..... 0°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	—	f_{SYS}=6MHz	2.2	—	5.5	V
		—	f_{SYS}=12MHz	3.0	—	5.5	V
I_{DD1}	Operating Current (6MHz Crystal)	5V	No load, f_{SYS}=6MHz	—	6.5	12	mA
I_{DD2}	Operating Current (12MHz Crystal)	5V	No load, f_{SYS}=12MHz	—	7.5	16	mA
I_{STB1}	Standby Current	5V	No load, system HALT, USB suspend*	—	—	500	μA
I_{STB2}	Standby Current (WDT Enabled)	5V	No load, system HALT, input/output mode, set SUSPEND2 [1EH]	—	—	15	μA
V_{IL1}	Input Low Voltage for I/O Ports	5V	TTL level	0	—	0.8	V
V_{IH1}	Input High Voltage for I/O Ports	5V	TTL level	2	—	V_{DD}	V
V_{IL2}	Input Low Voltage ($\overline{RES}$)	5V	CMOS level	0	—	$0.4V_{DD}$	V
V_{IH2}	Input High Voltage ($\overline{RES}$)	5V	CMOS level	$0.9V_{DD}$	—	V_{DD}	V
I_{OL1}	I/O Port Sink Current for PA1~PA7, PB, PC, PD	5V	V_{OL}=3.4V	10	15	20	mA
I_{OL2}	I/O Port Sink Current for PA1~PA7, PB, PC, PD	5V	V_{OL}=0.4V	2	4	8	mA
I_{OL3}	I/O Port Sink Current for PA0	5V	V_{OL}=0.4V	7	10	13	mA
I_{OH1}	I/O Port Source Current for PA1~PA7, PB, PC, PD	5V	V_{OH}=3.4V	–2	–4	–8	mA
I_{OH2}	I/O Port Source Current for PA0	5V	V_{OH}=3.4V	–12	–18	–24	mA

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
R_{PH}	Pull-high Resistance for PA, PB, PC, PD	5V	—	25	50	80	kΩ
V_{LVR}	Low Voltage Reset	—	2-battery option	2.1	2.2	2.3	V
			3-battery option	2.3	2.4	2.5	
V_{LVD}	Low Battery Detecting Voltage	—	2-battery option	2.3	2.4	2.5	V
			3-battery option	2.5	2.6	2.7	
V_{V33O}	3.3V Regulator Output	5V	I_{V33O}=−5mA	3.0	3.3	3.6	V

Note: "*" include 15kΩ loading of USBD+, USBD- line in host terminal.

A.C. Characteristics

Ta=25°C

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
f_{SYS}	System Clock (Crystal OSC)	5V	—	6	—	12	MHz
f_{TIMER}	Timer I/P Frequency (TMR)	5V	—	0	—	12	MHz
t_{WDTOSC}	Watchdog Oscillator	5V	—	15	31	70	μs
t_{WDT1}	Watchdog Time-out Period (WDT OSC)	5V	Without WDT prescaler	4	8	16	ms
t_{WDT2}	Watchdog Time-out Period (System Clock)	—	Without WDT prescaler	—	1024	—	t_{SYS}
t_{RES}	External Reset Low Pulse Width	—	—	1	—	—	ms
t_{SST}	System Start-up Timer Period	—	Wake-up from HALT	—	1024	—	t_{SYS}
			Power-up, Watchdog Time-out from normal	—	1024	—	t_{WDTOSC}
t_{INT}	Interrupt Pulse Width	—	—	1	—	—	μs

Functional Description

Execution Flow

The system clock for the microcontroller is derived from a crystal (6 or 12MHz). 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 allows each instruction to be effectively executed in a cycle. If an instruction changes the program counter, two cycles are required to complete the instruction.

Program Counter – PC

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

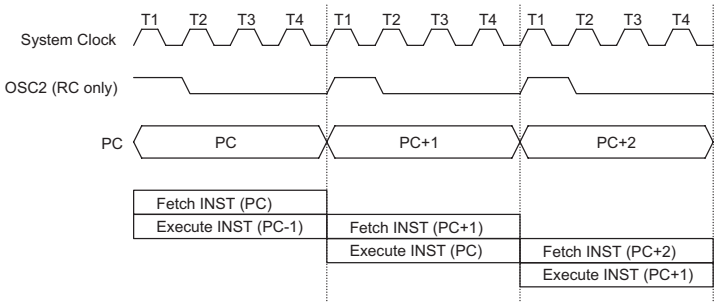
After accessing a program memory word to fetch an instruction code, the contents of the program counter are incremented by one. The program counter then points to the memory word containing the next instruction code.

When executing a jump instruction, conditional skip execution, loading PCL register, subroutine call or return from subroutine, initial reset, internal interrupt, USB interrupt or return from interrupts, 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 to the next instruction.

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

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



Execution Flow

Mode	Program Counter												
	***12**	***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	0
USB Interrupt	0	0	0	0	0	0	0	0	0	0	1	0	0
Timer/Event Counter 0 Overflow	0	0	0	0	0	0	0	0	0	1	0	0	0
Timer/Event Counter 1 Overflow	0	0	0	0	0	0	0	0	0	1	1	0	0
Skip	Program Counter+2												
Loading PCL	*12	*11	*10	*9	*8	@7	@6	@5	@4	@3	@2	@1	@0
Jump, Call Branch	#12	#11	#10	#9	#8	#7	#6	#5	#4	#3	#2	#1	#0
Return from Subroutine	S12	S11	S10	S9	S8	S7	S6	S5	S4	S3	S2	S1	S0

Program Counter

Note: *12~*0: Program counter bits
S12~S0: Stack register bits
#12~#0: Instruction code bits
@7~@0: PCL bits

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 6144×16 bits, addressed by the program counter and table pointer.

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

- Location 000H

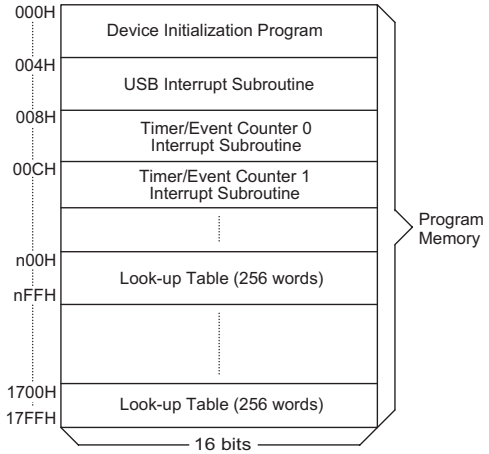
  This area is reserved for program initialization. After chip reset, the program always begins execution at location 000H.

- Location 004H

  This area is reserved for the USB and external interrupt service program. If the USB interrupt is activated, the interrupt is enabled and the stack is not full, the program begins execution at location 004H.

- Location 008H

  This area is reserved for the Timer/Event Counter 0 interrupt service program. If a timer interrupt results from a Timer/Event Counter 0 overflow, and if the interrupt is enabled and the stack is not full, the program begins execution at location 008H.



Note: n ranges from 0 to F

Program Memory

- Location 00CH

  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. There are three method to read the ROM data by two table read instructions: "TABRDC" and "TABRDL", transfer the contents of the lower-order byte to the specified data memory, and the higher-order byte to TBLH (08H).

  The three methods are shown as follows:

  - The instructions "TABRDC [m]" (the current page, one page=256words), where the table locations is defined by TBLP (07H) in the current page. And the ROM code option TBHP is disabled (default).
  - The instructions "TABRDC [m]", where the table locations is defined by registers TBLP (07H) and TBHP (01FH). And the ROM code option TBHP is enabled.
  - The instructions "TABRDL [m]", where the table locations is defined by Registers TBLP (07H) in the last page (1700H~17FFH).

  Only the destination of the lower-order byte in the table is well-defined, the other bits of the table word are transferred to the lower portion of TBLH, and the remaining 1-bit words are read as "0". The Table Higher-order byte register (TBLH) is read only. The table pointer (TBLP, TBHP) is a read/write register (07H, 1FH), which indicates the table location. Before accessing the table, the location must be placed in the TBLP and TBHP (If the OTP option TBHP is disabled, the value in TBHP has no effect). 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. Errors can occur. In other words, 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 should be disabled prior to the table read instruction.

Instruction	Table Location												
	*12	*11	*10	*9	*8	*7	*6	*5	*4	*3	*2	*1	*0
TABRDC [m]	P12	P11	P10	P9	P8	@7	@6	@5	@4	@3	@2	@1	@0
TABRDL [m]	1	1	1	1	1	@7	@6	@5	@4	@3	@2	@1	@0

Table Location

Note: *12~*0: Table location bits
@7~@0: Table pointer bits

P12~P8: Current program counter bits when TBHP is disabled
TBHP register bit3~bit0 when TBHP is enabled

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 on the requirements.

Once TBHP is enabled, the instruction "TABRDC [m]" reads the ROM data as defined by TBLP and TBHP value. Otherwise, the ROM code option TBHP is disabled, the instruction "TABRDC [m]" reads the ROM data as defined by TBLP and the current program counter bits.

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 organized into 8 levels and is neither part of the data nor part of the program space, and is neither readable nor writeable. The activated level is indexed by the stack pointer (SP) and is neither readable nor writeable. At a subroutine call or interrupt acknowledge signal, 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 instruction (RET or RETI), 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 acknowledge signal 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, stack overflow occurs and the first entry will be lost (only the most recent 8 return addresses are stored).

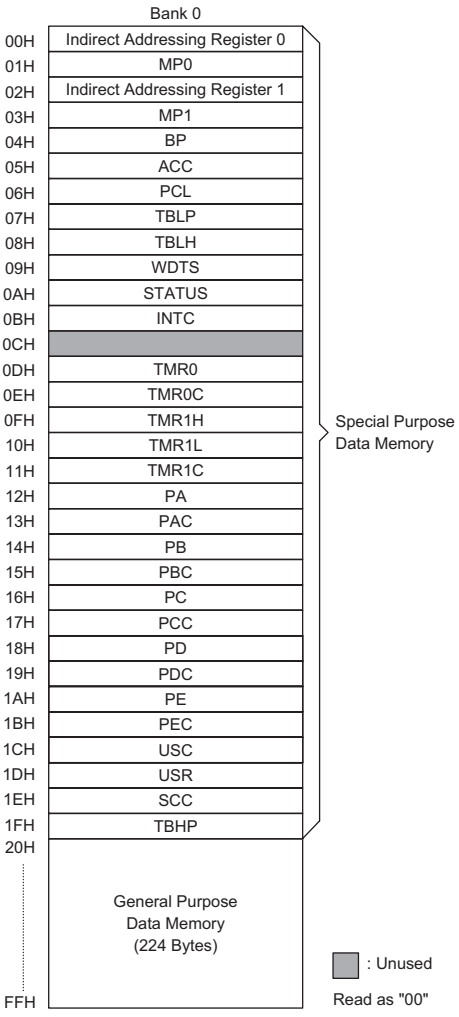
Data Memory – RAM for Bank 0

The data memory is designed with 255×8 bits. The data memory is divided into two functional groups: special function registers and general purpose data memory (224×8). Most are read/write, but some are read only.

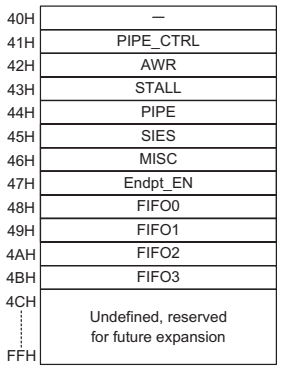
The general purpose data memory, addressed from 20H to FFH, is used for data and control information under instruction commands. All of the data memory areas can handle arithmetic, logic, increment, decrement and rotate operations directly. Except for some dedicated bits, each bit in the data memory can be set and reset by "SET [m].i" and "CLR [m].i". They are also indirectly accessible through memory pointer registers (MP0 or MP1).

Data Memory – RAM for Bank 1

The special function registers used in USB interface are located in RAM bank 1. In order to access the Bank1 register, only the Indirect addressing pointer MP1 can be used and the Bank register BP should be set to "1". The mapping of RAM bank 1 is as shown.



Bank 0 RAM Mapping



Bank 1 RAM Mapping

Indirect Addressing Register

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

The indirect addressing pointer (MP0) always point to Bank0 RAM addresses regardless of the value of the Bank Register (BP).

The indirect addressing pointer (MP1) can access Bank0 or Bank1 RAM data according to the value of BP which is set to ″0″ or ″1″ respectively.

The memory pointer registers (MP0 and MP1) are 8-bit registers.

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 a data operation but also changes the status register.

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 during a system power-up.

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

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.

Interrupt

The device provides internal timer/event counter and USB interrupts. The Interrupt Control Register (INTC;0BH) contains the interrupt control bits to set the enable/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 occur during this interval but only

Bit No.	Label	Function
0	C	C is set if an 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 an 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

the interrupt request flag is recorded. If a certain interrupt requires servicing within the service routine, the EMI bit and the corresponding bit of the INTC 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 (STATUS) are altered by the interrupt service program which corrupts the desired control sequence, the contents should be saved in advance.

USB interrupts are triggered by the following USB events and the related interrupt request flag (USBF; bit 4 of the INTC) will be set.

- The corresponding USB FIFO is accessed from the PC
- The USB suspends signal from the PC
- The USB resumes signal from the PC
- The USB sends Reset signal

When the interrupt is enabled, the stack is not full and the USB interrupt is active, a subroutine call to location 04H will occur. The interrupt request flag (USBF) and EMI bits will be cleared to disable other interrupts.

When the PC Host access the FIFO of the HT82K94E/HT82K94A, the corresponding request bit of the USR is set, and a USB interrupt is triggered. So user can easily decide which FIFO is accessed. When the interrupt has been served, the corresponding bit should be cleared by firmware. When the HT82K94E/HT82K94A receives a USB Suspend signal from the Host PC, the suspend line (bit0 of the USC) of the HT82K94E/HT82K94A is set and a USB interrupt is also triggered.

Also when the HT82K94E/HT82K94A receives a Resume signal from the Host PC, the resume line (bit3 of the USC) of HT82K94E/HT82K94A is set and a USB interrupt is triggered.

Whenever a USB reset signal is detected, the USB interrupt is triggered.

The internal Timer/Event Counter 0 interrupt is initialized by setting the Timer/Event Counter 0 interrupt request flag (; bit 5 of INTC), caused by a timer 0 overflow. When the interrupt is enabled, the stack is not full and the T0F bit is set, a subroutine call to location 08H will occur. The related interrupt request flag (T0F) will be reset and the EMI bit cleared to disable further interrupts.

The internal Timer/Even Counter 1 interrupt is initialized by setting the Timer/Event Counter 1 interrupt request flag (;bit 6 of INTC), caused by a timer 1 overflow. When the interrupt is enabled, the stack is not full and the T1F is set, a subroutine call to location 0CH will occur. The related interrupt request flag (T1F) will be reset and the EMI bit cleared to disable further interrupts.

During the execution of an interrupt subroutine, other interrupt acknowledge signals are held until the ″RETI″ instruction is executed or the EMI bit and the related interrupt control bit are set to ″1″ (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.

Bit No.	Label	Function
0	EMI	Controls the master (global) interrupt (1= enabled; 0= disabled)
1	EUI	Controls the USB 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	USBF	USB interrupt request flag (1= active; 0= inactive)
5	T0F	Internal Timer/Event Counter 0 request flag (1= active; 0= inactive)
6	T1F	Internal Timer/Event Counter 1 request flag (1= active; 0= inactive)
7	—	Unused bit, read as ″0″

INTC (0BH) Register

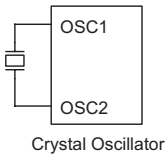
Interrupts, occurring in the interval between the rising edges of two consecutive T2 pulses, will be serviced on the latter of the two T2 pulses, if the corresponding interrupts are enabled. In the case of simultaneous requests the following table shows the priority that is applied. These can be masked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

Interrupt Source	Priority	Vector
USB interrupt	1	04H
Timer/Event Counter 0 overflow	2	08H
Timer/Event Counter 1 overflow	3	0CH

Once the interrupt request flags (TF, USBF) 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 is an oscillator circuits in the microcontroller.



Crystal Oscillator

System Oscillator

This oscillator is designed for system clocks. The HALT mode stops the system oscillator and ignores an external signal to conserve power.

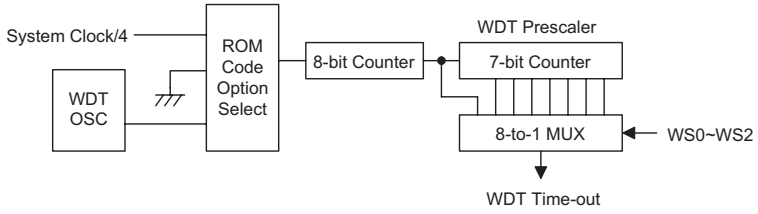
A crystal across OSC1 and OSC2 is needed to provide the feedback and phase shift required for the oscillator. No other external components are required. In stead of a crystal, a resonator can also be connected between OSC1 and OSC2 to get a frequency reference, but two external capacitors in OSC1 and OSC2 are required.

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 within a period of approximately 31μs. The WDT oscillator can be disabled by ROM code option to conserve power.

Watchdog Timer – WDT

The WDT clock source is implemented by a dedicated RC oscillator (WDT oscillator), or instruction clock (system clock divided by 4), determines the ROM code 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 ROM code option. If the Watchdog Timer is disabled, all the executions related to the WDT result in no operation.

Once the internal WDT oscillator (RC oscillator, normally with a period of 31μs/5V) is selected, it is first divided by 256 (8-stage) to get the nominal time-out period of 8ms/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 (bits 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 1s/5V. If the WDT oscillator is disabled, the WDT clock may still come from the instruction clock and operates 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. The high nibble and bit 3 of the WDTS are reserved for user's defined flags, which can only be set to "10000" (WDTS.7~WDTS.3).



Watchdog Timer

If the device operates in a noisy environment, using the on-chip 32kHz RC oscillator (WDT OSC) is strongly recommended, since the HALT will stop the system clock.

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

WDTS (09H) Register

The WDT overflow under normal operation will initialize a "chip reset" and set the status bit "TO". But in the HALT mode, the overflow will initialize a "warm reset" and only the Program Counter and SP are reset to zero. To clear the contents of the WDT (including the WDT prescaler), three methods are employed; external reset (a low level to $\overline{RES}$), 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 ROM code 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 wherein "CLR WDT1" and "CLR WDT2" are chosen (i.e. CLRWDT times is equal to two), these two instructions must be executed to clear the WDT, otherwise, the WDT may reset the chip as a result of time-out.

Power Down Operation – HALT

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

- The system oscillator will be turned off but the WDT oscillator remains 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 I/O ports or a WDT overflow. An external reset causes a device initialization and the WDT overflow performs a "warm reset". After the TO and PDF flags are examined, the cause for chip reset can be determined. The PDF flag is cleared by a 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 remain in their original status.

The I/O ports wake-up and interrupt methods can be considered as a continuation of normal execution. Each bit in the Port A can be independently selected to wake-up the device by option. PB, PC and PD can also be selected to wake-up the device by option. Upon awakening from an I/O port stimulus, the program will resume execution of the next instruction. If it awakens from an interrupt, two sequence may occur. 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 a wake-up. If the wake-up results from an interrupt acknowledge signal, 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 completed.

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 can occur:

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

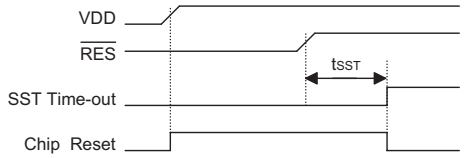
The WDT time-out during HALT is different from other chip reset conditions, since it can perform a "warm reset" that resets only the Program Counter and SP, leaving the other circuits in their original state. Some registers remain unchanged during other reset conditions. Most registers are reset to the "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	$\overline{RES}$ reset during power-up
u	u	$\overline{RES}$ reset during normal operation
0	0	$\overline{RES}$ wake-up HALT
1	u	WDT time-out during normal operation
1	1	WDT wake-up HALT

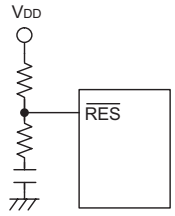
Note: "u" stands for "unchanged"

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

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



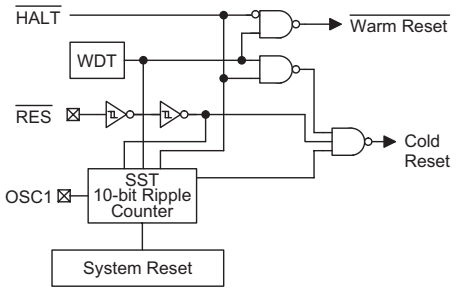
Reset Timing Chart



Reset Circuit

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 Configuration

The status of the registers are summarized in the following table.

Register	Reset (Power On)	WDT Time-out (Normal Operation)	$\overline{RES}$ Reset (Normal Operation)	$\overline{RES}$ Reset (HALT)	WDT Time-Out (HALT)*	USB-Reset (Normal)	USB-Reset (HALT)
TMR0	xxxx xxxx	0000 0000	0000 0000	0000 0000	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR0C	00-0 1000	00-0 1000	00-0 1000	00-0 1000	uu-u uuuu	00-0 1000	00-0 1000
TMR1H	xxxx xxxx	0000 0000	0000 0000	0000 0000	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR1L	xxxx xxxx	0000 0000	0000 0000	0000 0000	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR1C	00-0 1---	00-0 1---	00-0 1---	00-0 1---	uu-u u---	00-0 1---	00-0 1---
Program Counter	000H	000H	000H	000H	000H	000H	000H
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	-xxx xxxx	-uuu uuuu	-uuu uuuu	-uuu uuuu	-uuu uuuu	-uuu uuuu	-uuu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--00 uuuu	--11 uuuu	--uu uuuu	--01 uuuu
INTC	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu	-000 0000	-000 0000
WDTS	1000 0111	1000 0111	1000 0111	1000 0111	uuuu uuuu	1000 0111	1000 0111
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu	1111 1111	1111 1111
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu	1111 1111	1111 1111
PB	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu	1111 1111	1111 1111

Register	Reset (Power On)	WDT Time-out (Normal Operation)	$\overline{RES}$ Reset (Normal Operation)	$\overline{RES}$ Reset (HALT)	WDT Time-Out (HALT)*	USB-Reset (Normal)	USB-Reset (HALT)
PBC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu	1111 1111	1111 1111
PC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu	1111 1111	1111 1111
PCC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu	1111 1111	1111 1111
PD	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu	1111 1111	1111 1111
PDC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu	1111 1111	1111 1111
PE	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu	1111 1111	1111 1111
PEC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu	1111 1111	1111 1111
PIPE_CTRL	0000 1111	uuuu uuuu	0000 1111	0000 1111	uuuu uuuu	0000 1111	0000 1111
AWR	0000 0000	uuuu uuuu	0000 0000	0000 0000	uuuu uuuu	0000 0000	0000 0000
PIPE	0000 0000	uuuu uuuu	0000 0000	0000 0000	uuuu uuuu	0000 0000	0000 0000
STALL	0000 1111	uuuu uuuu	0000 1111	0000 1111	uuuu uuuu	0000 1111	0000 1111
SIES	0000 0000	uuuu uuuu	0000 0000	0000 0000	uuuu uuuu	0000 0000	0000 0000
MISC	0000 0000	uuuu uuuu	0000 0000	0000 0000	uuuu uuuu	0000 0000	0000 0000
Endpt_EN	0000 1111	uuuu uuuu	0000 1111	0000 1111	uuuu uuuu	0000 1111	0000 1111
FIFO0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu	0000 0000	0000 0000
FIFO1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu	0000 0000	0000 0000
FIFO2	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu	0000 0000	0000 0000
FIFO3	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu	0000 0000	0000 0000
USC	11xx 0000	11xx uuuu	11xx 0000	11xx 0000	uuxx uuuu	uu00 0u00	uu00 0u00
USR	0000 0000	uuuu uuuu	0000 0000	0000 0000	uuuu uuuu	u0uu 0000	u0uu 0000
SCC	0000 0010	uuuu uuuu	0000 0010	0000 0010	uuuu uuuu	0uu0 u000	0uu0 u000

Note: "*" stands for "warm reset"
"u" stands for "unchanged"
"x" stands for "unknown"

Timer/Event Counter

Two timer/event counters (TMR0, TMR1) are implemented in the microcontroller. The Timer/Event Counter 0 contains an 8-bit programmable count-up counter and the clock may comes from an external source or from $f_{SYS}/4$.

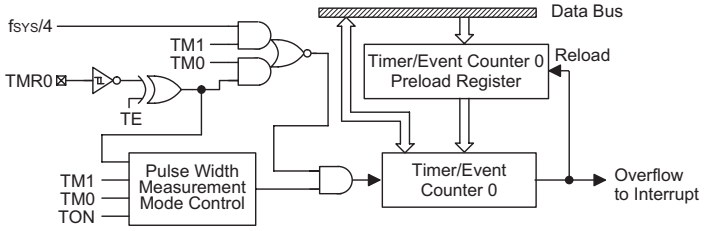
The Timer/Event Counter 1 contains an 16-bit programmable count-up counter and the clock may come from an external source or from the system clock divided by 4.

Bit No.	Label	Function
0~2, 5	—	Unused bit, read as "0"
3	TE	To define the TMR0 active edge of Timer/Event Counter 0 (0=active on low to high; 1=active on high to low)
4	TON	To enable/disable timer 0 counting (0=disabled; 1=enabled)
6 7	TM0 TM1	To define the operating mode 01=Event count mode (external clock) 10=Timer mode (internal clock) 11=Pulse width measurement mode 00=Unused

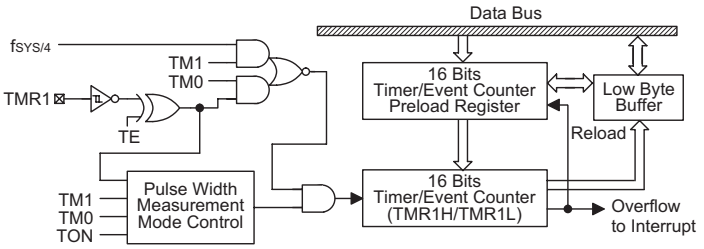
TMR0C (0EH) Register

Bit No.	Label	Function
0~2, 5	—	Unused bit, read as "0"
3	TE	To define the TMR1 active edge of Timer/Event Counter 1 (0=active on low to high; 1=active on high to low)
4	TON	To enable/disable timer 1 counting (0=disabled; 1=enabled)
6 7	TM0 TM1	To define the operating mode 01=Event count mode (external clock) 10=Timer mode (internal clock) 11=Pulse width measurement mode 00=Unused

TMR1C (11H) Register



Timer/Event Counter 0



Timer/Event Counter 1

Using the internal clock source, there is only 1 reference time-base for Timer/Event Counter 0. The internal clock source is coming from $f_{SYS}/4$.

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

Using the internal clock source, there is only 1 reference time-base for Timer/Event Counter 1. The internal clock source is coming from $f_{SYS}/4$. The external clock input allows the user to count external events, measure time intervals or pulse widths.

There are 2 registers related to the Timer/Event Counter 0; TMR0 ([0DH]), TMR0C ([0EH]). Two physical registers are mapped to TMR0 location; writing TMR0 makes the starting value be placed in the Timer/Event Counter 0 preload register and reading TMR0 gets the contents of the Timer/Event Counter 0. The TMR0C is a timer/event counter control register, which defines some options.

There are 3 registers related to Timer/Event Counter 1; TMR1H (0FH), TMR1L (10H), TMR1C (11H). Writing TMR1L will only put the written data to an internal lower-order byte buffer (8 bits) and writing TMR1H will transfer the specified data and the contents of the lower-order byte buffer to TMR1H and TMR1L preload registers, respectively. The Timer/Event Counter 1 preload register is changed by each writing TMR1H operations. Reading TMR1H will latch the contents of TMR1H and TMR1L counters to the destination and the lower-order byte buffer, respectively. Reading the TMR1L will read the contents of the lower-order byte buffer. The TMR1C is the Timer/Event Counter 1 control register, which defines the operating mode, counting enable or disable and active edge.

The TM0, TM1 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 $f_{SYS}/4$ (Timer0/Timer1). 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 $f_{SYS}/4$ (Timer0/Timer1).

In the event count or timer mode, once the Timer/Event Counter 0/1 starts counting, it will count from the current contents in the Timer/Event Counter 0/1 to FFH or FFFFH. Once overflow occurs, the counter is reloaded from the Timer/Event Counter 0/1 preload register and generates the interrupt request flag (T0F/T1F; bit 5/6 of INTC) at the same time.

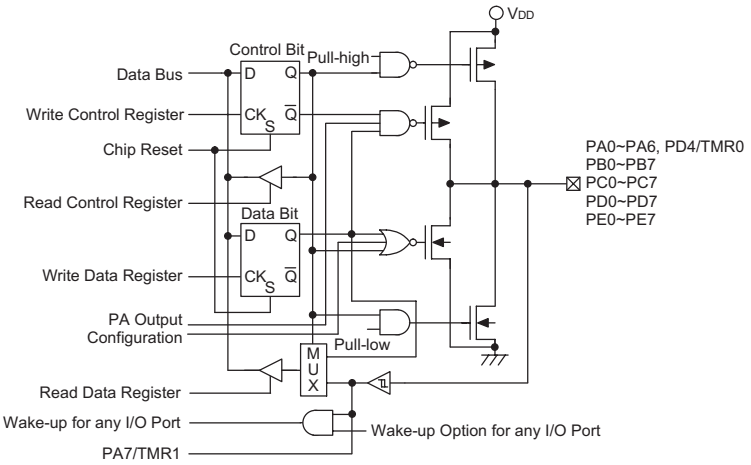
In the pulse width measurement mode with the TON and TE bits equal to one, once the TMR0/TMR1 has received a transient from low to high (or high to low if the TE bits is "0") it will start counting until the TMR0/TMR1 returns to the original level and resets the TON. The measured result will remain in the Timer/Event Counter 0/1 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 function again as long as it receives further transient pulse. Note that, in this operating mode, the Timer/Event Counter 0/1 starts counting not according to the logic level but according to the transient edges. In the case of counter overflows, the counter 0/1 is reloaded from the Timer/Event Counter 0/1 preload register and issues the interrupt request just like the other two modes. To enable the counting operation, the timer ON bit (TON; bit 4 of TMR0C/TMR1C) should be set to 1. In the pulse width measurement mode, the TON will be cleared automatically after the measurement cycle is completed. But in the other two modes the TON can only be reset by instructions. The overflow of the Timer/Event Counter 0/1 is one of the wake-up sources. No matter what the operation mode is, writing a 0 to ET0I/ET1I can disable the corresponding interrupt services.

In the case of Timer/Event Counter 0/1 OFF condition, writing data to the Timer/Event Counter 0/1 preload register will also reload that data to the Timer/Event Counter 0/1. But if the Timer/Event Counter 0/1 is turned on, data written to it will only be kept in the Timer/Event Counter 0/1 preload register. The Timer/Event Counter 0/1 will still operate until overflow occurs (a Timer/Event Counter 0/1 reloading will occur at the same time). When the Timer/Event Counter 0/1 (reading TMR0/TMR1) 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.

Input/Output Ports

There are 40 bidirectional input/output lines in the microcontroller, labeled from PA to PE, which are mapped to the data memory of [12H], [14H], [16H], [18H] and [1AH] respectively. All of these I/O ports can be used for input and 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, 14H, 16H, 18H or 1AH). 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, PBC, PCC, PDC, PEC) to control the input/output configuration. With this control register, CMOS/NMOS/PMOS output or Schmitt trigger input with or without pull-high resistor structures can be reconfigured dynamically under software control. To function as an input, the corresponding latch of the control register must write a "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 except port A which can be



Input/Output Ports

optioned as CMOS/NMOS/PMOS configurations. These control registers are mapped to locations 13H, 15H, 17H and 19H.

After a chip reset, these input/output lines remain at high levels or floating state (depending on the pull-high options). Each bit of these input/output latches can be set or cleared by "SET [m].i" and "CLR [m].i" (m=12H, 14H, 16H or 18H) 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.

Each line of all the I/O ports have the capability of waking up the device.

There are pull-high options available for I/O lines. Once the pull-high option of an I/O line is selected, the I/O line have pull-high resistor. Otherwise, the pull-high resistor is absent. It should be noted that a non-pull-high I/O line operating in input mode will cause a floating state.

It is recommended that unused or not bonded out I/O lines should be set as output pins by software instruction to avoid consuming power under input floating state.

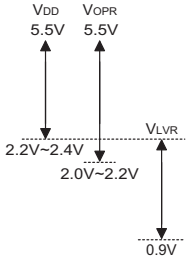
Low Voltage Reset/Detector – LVR/LVD

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

The LVR includes the following specifications:

- For a valid LVR signal, a low voltage i.e. a voltage in the range between 0.9V~V_{LVR} must exist for greater than 1ms. If the low voltage state does not exceed 1ms, the LVR will ignore it and do not perform a reset function.
- The LVR uses the "OR" function with the external $\overline{RES}$ signal to perform chip reset.

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

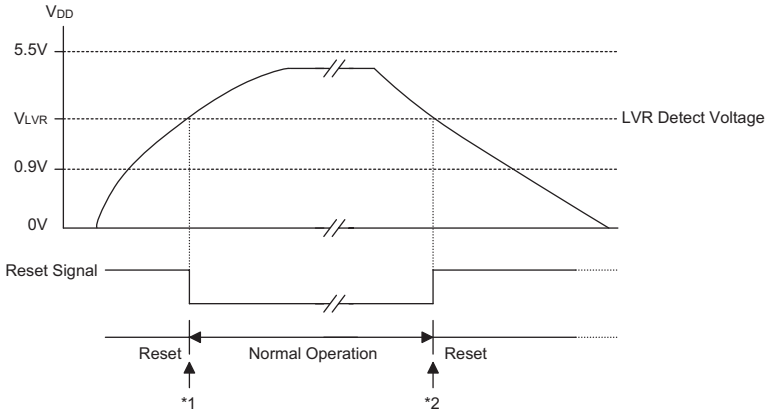


Note: V_{OPR} is the voltage range for proper chip operation at 4MHz system clock.

There is a low voltage detector (LVD) and a low voltage reset circuit (LVR) implemented in the microcontrollers. Where LVR function can be enabled/disabled by options. User can read the LVD detector status (0/1) from MISC.5. The LVR has the same effect or function with the external RES signal which performs a chip reset. Both LVR and LVD functions will disable in the HALT mode.

There are two kind of LVR/LVD definition: 2-battery or 3-battery option.

When 2-battery option is selected: LVR=2.2V; LVD=2.4V. Otherwise, 3-battery option is selected: LVR=2.4V; LVD=2.6V.



Low Voltage Reset

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

*2. Since low voltage has to be maintained for over 1ms in its original state, therefore there's a 1ms delay before entering the reset mode

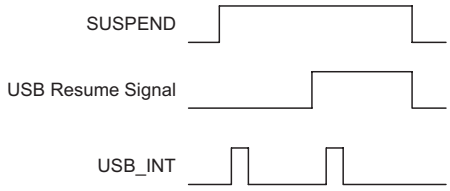
Suspend Wake-Up and Remote Wake-Up

If there is no signal on the USB bus for over 3ms, the HT82K94E/HT82K94A will go into suspend mode. The Suspend line (bit 0 of the USC) will be set to ″1″ and a USB interrupt is triggered to indicate that the HT82K94E/HT82K94A should jump to the suspend state to meet the 500μA USB suspend current spec.

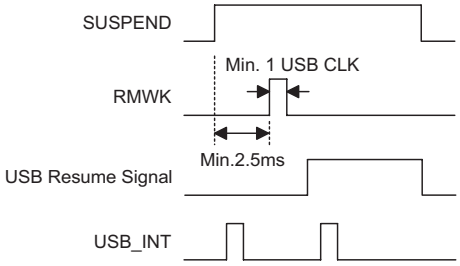
In order to meet the 500μA suspend current, the firmware should disable the USB clock by clearing the USBCKEN (bit3 of the SCC) to ″0″. The suspend current is 400μA.

When the resume signal is sent out by the host, the HT82K94E/HT82K94A will wake-up the MCUby USB interrupt and the Resume line (bit 3 of the USC) is set. In order to make the HT82K94E/HT82K94A function properly, the firmware must set the USBCKEN (bit 3 of the SCC) to ″1″. Since the Resume signal will be cleared before the Idle signal is sent out by the host, the Suspend line (bit 0 of the USC) will be set to ″0″. So when the MCU is detecting the Suspend line (bit0 of USC), the Resume line should be remembered and taken into consideration.

After finishing the resume signal, the suspend line will go inactive and a USB interrupt is triggered. The following is the timing diagram.



The device with remote wake-up function can wake-up the USB Host by sending a wake-up pulse through RMWK (bit 1 of the USC). Once the USB Host receives a wake-up signal from the HT82K94E/HT82K94A, it will send a Resume signal to the device. The timing is as follows:



Configuring the Device as a PS2 Device

The HT82K94E/HT82K94A can be configured as a USB interface or PS2 interface device, by configuring the SPS2 (bit 4 of USR) and SUSB (bit 5 of the USR). If SPS2=1, and SUSB=0, the HT82K94E/HT82K94A is configured as a PS2 interface, pin USBD- is configured as a PS2 Data pin and USBD+ is configured as a PS2 Clk pin. User can easily read or write to the PS2 Data or PS2 Clk pin by accessing the corresponding bit PS2DAI (bit 4 of the USC), PS2CKI (bit 5 of the USC), PS2DAO (bit 6 of the USC) and S2CKO (bit 7 of the USC) respectively.

User should make sure that in order to read the data properly, the corresponding output bit must be set to ″1″. For example, if it is desired to read the PS2 Data by reading PS2DAI, the PS2DAO should set to ″1″. Otherwise it is always read as ″0″.

If SPS2=0, and SUSB=1, the HT82K94E/HT82K94A is configured as a USB interface. Both the USBD- and USBD+ is driven by the SIE of the HT82K94E/HT82K94A. User can only write or read the USB data through the corresponding FIFO.

Both SPS2 and SUSB default is ″0″.

USB Interface

There are eleven registers, including PIPE_CTRL (41H in bank 1), AWR (address + remote wake-up 42H in bank 1), STALL (43H in bank 1), PIPE (44H in bank 1), SIES (45H in bank 1), MISC (46H in bank 1), Endpt_EN (47H in bank 1), FIFO0 (48H in bank 1), FIFO1 (49H in bank 1), FIFO2 (4AH in bank 1) and FIFO3 (4BH in bank 1) used for the USB function. AWR register contains current address and a remote wake up function control bit. The initial value of AWR is ″00H″. The address value extracted from the USB command is not to be loaded into this register until the SETUP stage is completed.

Bit No.	Label	R/W	Function
0	WKEN	W	Remote wake-up enable/disable
7~1	AD6~AD0	W	USB device address

AWR (42H) Register

STALL, PIPE, PIPE_CTRL and Endpt_EN Registers

PIPE register represents whether the endpoint corresponding is accessed by host or not. After ACT_EN signal being sent out, MCU can check which endpoint had been accessed. This register is set only after the time when host access the corresponding endpoint.

STALL register shows whether the endpoint corresponding works or not. As soon as the endpoint work improperly, the bit corresponding must be set.

PIPE_CTRL Register is used for configuring IN (Bit=1) or OUT (Bit=0)Pipe. The default is define IN pipe. Where Bit0 (DATA0) of the PIPE_CTRL Register is used to setting the data toggle of any endpoint (except endpoint0) using data toggles to the value DATA0. Once the user want the any endpoint (except endpoint0) using data toggles to the value DATA0. the user can output a LOW pulse to this bit. The LOW pulse period must at least 10 instruction cycle.

Endpt_EN Register is used to enable or disable the corresponding endpoint (except endpoint 0) Enable Endpoint (Bit=1) or disable Endpoint (Bit=0)

The bitmaps are list as follows :

Register Name	R/W	Register Address	Bit7~Bit4 Reserved	Bit 3	Bit 2	Bit 1	Bit 0	Default Value
PIPE_CTRL	R/W	01000001B	—	Pipe 3	Pipe 2	Pipe 1	Pipe 0	00001111
STALL	R/W	01000011B	—	Pipe 3	Pipe 2	Pipe 1	Pipe 0	00001111
PIPE	R	01000100B	—	Pipe 3	Pipe 2	Pipe 1	Pipe 0	00000000
Endpt_EN	R/W	01000001B	—	Pipe 3	Pipe 2	Pipe 1	Pipe 0	00001111

PIPE_CTRL (41H), STALL (43H), PIPE (44H) and Endpt_EN (47H) Registers

The SIES Register is used to indicate the present signal state which the SIE receives and also defines whether the SIE has to change the device address automatically.

	Bit7	Bit6	Bit5	Bit4	Bit3	Bit2	Bit1	Bit0
Func.	NMI	EOT	CRC_ERR	NAK	IN	OUT	F0_ERR	Adr_set
R/W	R/W	R	R/W	R	R	R/W	R/W	R/W
Reg_ Adr	01000101B							

SIES (45H) Register

Func. Name	R/W	Description
Adr_ set	R/W	This bit is used to configure the SIE to automatically change the device address with the value of the Address+Remote_WakeUp Register (42H). When this bit is set to ″1″ by F/W, the SIE will update the device address with the value of the Address+Remote_WakeUp Register (42H) after the PC Host has successfully read the data from the device by the IN operation. The SIE will clear the bit after updating the device address. Otherwise, when this bit is cleared to ″0″, the SIE will update the device address immediately after an address is written to the Address+Remote_WakeUp Register (42H) Default 0
F0_Err	R/W	This bit is used to indicate that some errors have occurred when accessing the FIFO0. This bit is set by SIE and cleared by F/W. Default 0
Out	R/W	This bit is used to indicate that an OUT token (except for the OUT zero length) has been received. The F/W clear the bit after the OUT data has been read. This bit will also be cleared by the SIE after the next valid SETUP token is received. Default 0
IN	R	This bit is used to indicate that the current signal the USB is receiving from the PC Host is IN token.
NAK	R	This bit is used to indicate that the SIE is transmitting NAK signal to the Host in response to the PC Host IN or OUT token.

Func. Name	R/W	Description
CRC_ERR	R/W	This bit is used to indicate there are CRCerror (bit=1). Firmware must do something to save the device and keep it in good condition. This bit is set by SIE and cleared by F/W.
EOT	R	End of transaction flag, normal status is 1. If suspend='1' line & EOT='0' indicates that something is wrong in the USB Interface. Firmware in-charge must do something to save the device and keep it in good condition.
NMI	R/W	This bit is used to control whether the USB interrupt is output to the MCU in NAK response to the PC Host IN or OUT token. 1: has only USB interrupt, data is transmitted to the PC host or data is received from the PC Host 0: always has USB interrupt if the USB accesses FIFO0 Default 0

SIES Function

MISC register combines a command and status to control desired endpoint FIFO action and to show the status of the desired endpoint FIFO. The MISC will be cleared by USB reset signal.

Bit No.	Label	R/W	Function
0	REQ	R/W	After setting the other status of the desired one in the MISC, endpoint FIFO can be requested by setting this bit to "1". After the job has been done, this bit has to be cleared to "0".
1	TX	R/W	This bit defines the direction of data transferring between MCU and endpoint FIFO. When the TX is set to "1", this means that the MCU wants to write data to the endpoint FIFO. After the job has been done, this bit has to be cleared to "0" before terminating request to represent the end of transferring. For reading action, this bit has to be cleared to "0" to represent that MCU wants to read data from the endpoint FIFO and has to be set to "1" after the job is done.
2	CLEAR	R/W	Clear the requested endpoint FIFO, even if the endpoint FIFO is not ready.
4 3	SELP1 SELP0	R/W	Defines which endpoint FIFO is selected, SELP1,SELP0: 00: endpoint FIFO0 01: endpoint FIFO1 10: endpoint FIFO2 11: endpoint FIFO3
5	SCMD	R/W	Used to show that the data in endpoint FIFO is a SETUP command. This bit has to be cleared by firmware. That is to say, even the MCU is busy, the device will not miss any SETUP commands from the host.
6	READY	R	Read only status bit, this bit is used to indicate that the desired endpoint FIFO is ready to work.
7	LEN0	R/W	Used to indicate that a 0-sized packet is sent from a host to the MCU. This bit should be cleared by firmware.

MISC (46H) Register

The MCU can communicate with the endpoint FIFO by setting the corresponding registers, of which address is listed in the following table. After reading the current data, next data will show after 2μs, used to check the endpoint FIFO status and response to MISC register, if read/write action is still going on.

Registers	R/W	Bank	Address	Bit7~Bit0
FIFO0	R/W	1	48H	Data7~Data0
FIFO1	R/W	1	49H	Data7~Data0
FIFO2	R/W	1	4AH	Data7~Data0
FIFO3	R/W	1	4BH	Data7~Data0

There are some timing constrains and usages illustrated here. By setting the MISC register, MCU can perform reading, writing and clearing actions. There are some examples shown in the following table for endpoint FIFO reading, writing and clearing.

Actions	MISC Setting Flow and Status
Read FIFO0 sequence	00H→01H→delay 2μs, check 41H→read* from FIFO0 register and check not ready (01H)→03H→02H
Write FIFO1 sequence	0AH→0BH→delay 2μs, check 4BH→write* to FIFO1 register and check not ready (0BH)→09H→08H
Check whether FIFO0 can be read or not	00H→01H→delay 2μs, check 41H (ready) or 01H (not ready)→00H
Check whether FIFO1 can be written or not	0AH→0BH→delay 2μs, check 4BH (ready) or 0BH (not ready)→0AH
Read 0-sized packet sequence form FIFO0	00H→01H→delay 2μs, check 81H→read once (01H)→03H→02H
Write 0-sized packet sequence to FIFO1	0AH→0BH→delay 2μs, check 0BH→0FH→0DH→08H

Note: *: There are 2μs existing between 2 reading action or between 2 writing action

USB/PS2 Status and Control Register

The register is used to indicate there are USB suspend, USB resume and USB reset signal in USB bus. Also user can output a high pulse in RMWK bit to wake-up the PC for USB remote function.

Bit No.	Label	R/W	Function
0	SUSP	R	Read only, USB suspend indication. When this bit is set to ″1″ (set by SIE), it indicates the USB bus enters suspend mode. The USB interrupt is also triggered on any changes of this bit.
1	RMWK	W	USB remote wake-up command. It is set by MCU to force the USB host leaving the suspend mode. When this bit is set to ″1″, 2μs delay for clearing this bit to ″0″ is needed to insure the RMWK command is accepted by SIE.
2	URST	R/W	USB reset indication. This bit is set/cleared by USB SIE. This bit is used to detect which bus (PS2 or USB) is attached. When the URST is set to ″1″, this indicates that a USB reset has occurred (the attached bus is USB) and a USB interrupt will be initialized.
3	RESUME	R	USB resume indication. When the USB leaves the suspend mode, this bit is set to ″1″ (set by SIE). This bit will appear 20ms waiting for the MCU to detect. When the RESUME is set by the SIE, an interrupt will be generated to wake-up the MCU. In order to detect the suspend state, the MCU should set the USBCKEN and clear SUSP2 (in SCC register) to enable the SIE detecting function. The RESUME will be cleared while the SUSP is going ″0″. When the MCU is detecting the SUSP, the RESUME (wakes-up the MCU) should be remembered and taken into consideration.
4	PS2DAI	R	Read only, USBD-/DATA input
5	PS2CKI	R	Read only, USBD+/CLK input
6	PS2DAO	W	Data for driving the USBD-/DATA pin (Default=″1″)
7	PS2CKO	W	Data for driving the USBD+/CLK pin (Default=″1″)

USC (1CH) Register

USB Endpoint Interrupt Status Register

The register is used to indicate which endpoint is accessed or has external interrupt PA4/$\overline{EXT}$ is activated and to select the serial bus (PS2 or USB). The endpoint request flags (EP0IF, EP1IF, EP2IF, EP3IF and EXTIF) are used to indicate which endpoints are accessed. If an endpoint is accessed, the related endpoint request flag will be set to ″1″ and the USB interrupt will occur (if the USB interrupt is enabled and the stack is not full). When the active endpoint request flag is served, the endpoint request flag has to be cleared to ″0″.

Where USB_flag bit is only a bit for firmware to store the USB-mode data. This bit only clear to zero after power-on reset.

Bit No.	Label	R/W	Function
0	EP0IF	R/W	When this bit is set to ″1″ (set by the SIE), it indicates the endpoint 0 is accessed and a USB interrupt will occur. When the interrupt has been served, this bit should be cleared by firmware.
1	EP1IF	R/W	When this bit is set to ″1″ (set by the SIE), it indicates the endpoint 1 is accessed and a USB interrupt will occur. When the interrupt has been served, this bit should be cleared by firmware.
2	EP2IF	R/W	When this bit is set to ″1″ (set by the SIE), it indicates the endpoint 2 is accessed and a USB interrupt will occur. When the interrupt has been served, this bit should be cleared by firmware.
3	EP3IF	R/W	When set to ″1″, indicated endpoint 3 interrupt event . Must wait MCU to process interrupt event, then clear this bit by firmware. This bit must be 0, then next interrupt event will be process. Default value is 0.
4	SPS2	R/W	The PS2 function is selected when this bit is set to ″1″. (Default=″0″)
5	SUSB	R/W	The USB function is selected when this bit is set to ″1″. (Default=″0″)
6	—	—	Undefined, should be cleared to ″0″
7	USB_flag	R/W	This flag is used to show the MCU is in USB mode. (Bit=1) This bit is R/W by FW and will be cleared to ″0″ after power-on reset. (Default=″0″)

USR (1DH) Register

System Clock Control Register

This register is designed to control the system clock and make the device to meet USB 500μA suspend current spec. as well as a LVD indicator.

Since the device can operate at 6MHz or 12MHz in USB mode, so in order to make SIE work properly, there should has a SYSCLK bit to indicate what system frequency is working.

The USBCKEN bit is used to turn-off or turn-on the SIE system clock to meet the USB 500μA suspend current. For normal operation, this bit must be 1. Otherwise, the SIE cannot detect the USB signal.

PS2_flag bit is only a bit for firmware to store the PS2 mode data. This bit only clear to zero by hardware after power-on reset. SUSPEND2 bit is used to second suspend mode.

Bit No.	Label	R/W	Function
0, 2	—	—	Undefined, should be cleared to ″0″
1	OSC_ctrl	R/W	1: High driver of oscillator circuit for low voltage 0: normal driver of oscillator circuit
3	USBCKEN	R/W	USB clock control bit. When this bit is set to ″1″, it indicates that the USB clock is enabled. Otherwise, the USB clock is turned-off. (Default=″0″)
4	SUSPEND2	R/W	This bit is used to reduce power consumption in the suspend mode. In the normal mode this bit must be cleared to zero(Default=″0″). In the HALT mode this bit should be set high to reduce power consumption and LVR with no function. In the USB mode this bit cannot be set high.
5	PS2_flag	R/W	This flag is used to show the MCU is under PS2 mode. (Bit=1) This bit is R/W by FW and will be cleared to ″0″ after power-on reset. (Default=″0″)
6	SYSCLK	R/W	This bit is used to specify the system oscillator frequency used by the MCU. If a 6MHz crystal oscillator or resonator is used, this bit should be set to ″1″. If a 12MHz crystal oscillator or resonator is used, this bit should be cleared to ″0″ (default).
7	LVD	R/W	1: battery voltage low 0: battery voltage high

SCC (1EH) Register

Table High Byte Pointer for Current Table Read TBHP (Address 0X1F)

Register	Bits	Labels	Read/Write	Option	Functions
(0X1F)	4~0	PGC4~PGC0	R/W	—	Store current table read bit12~bit8 data

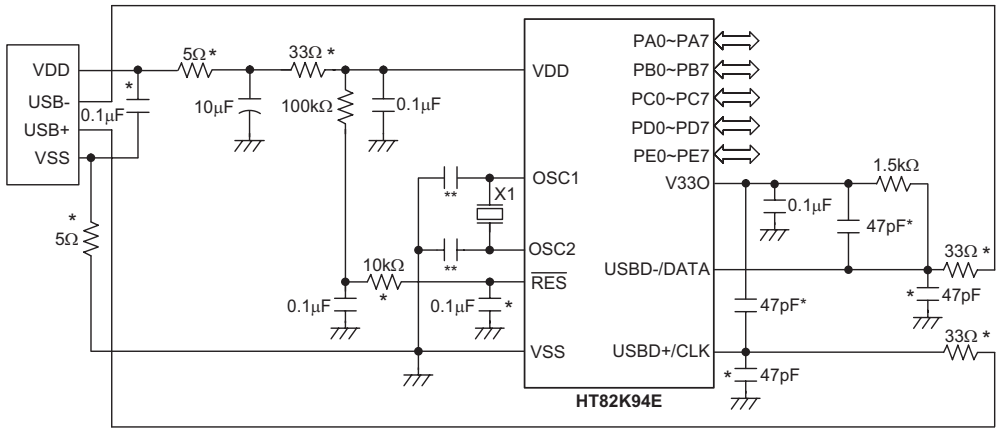
Options

The following table shows all kinds of option in the microcontroller. All of the options must be defined to ensure proper system functioning.

No.	Option
1	Chip lock bit (by bit)
2	PA0~PA7 pull-high resistor enabled or disabled (by bit)
3	PB0~PB7 pull-high resistor enabled or disabled (by nibble)
4	PC0~PC7 pull-high resistor enabled or disabled (by nibble)
5	PD0~PD7 pull-high resistor enabled or disabled (by nibble)
6	PE0~PE7 pull-high resistor enabled or disabled (by nibble)
7	LVR enable or disable
8	WDT enable or disable
9	WDT clock source: $f_{SYS}/4$ or WDTOSC
10	″CLRWDT″ instruction(s): 1 or 2
11	PA0~PA7 output structures: CMOS/NMOS open-drain/PMOS open-drain (by bit)
12	PA0~PA7 wake-up enabled or disabled (by bit)
13	PB0~PB7 wake-up enabled or disabled (by nibble)
14	PC0~PC7 wake-up enabled or disabled (by nibble)
15	PD0~PD7 wake-up enabled or disabled (by nibble)
16	TBHP enable or disable (default disable)
17	LVR/LVD kind: 2-battery or 3-battery

Application Circuits

Crystal or Ceramic Resonator for Multiple I/O Applications – HT82K94E



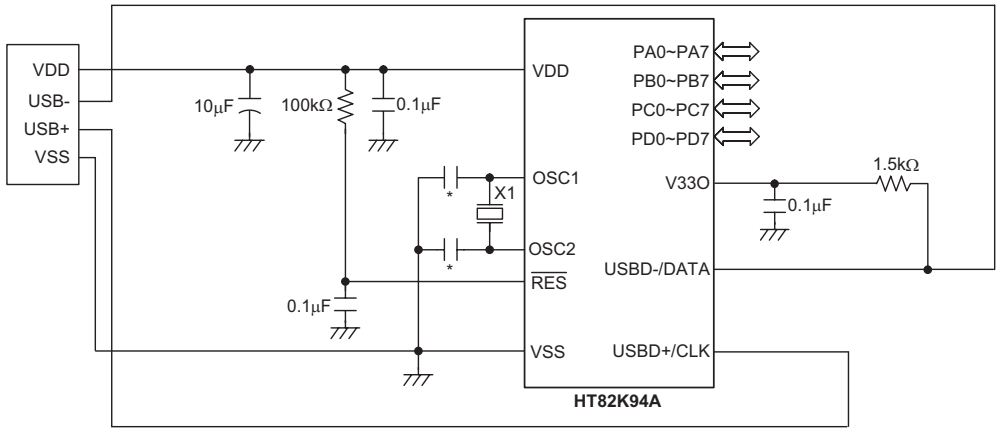
Note: The resistance and capacitance for reset circuit should be designed in such a way as to ensure that the VDD is stable and remains within a valid operating voltage range before bringing $\overline{RES}$ high.

X1 can use 6MHz or 12MHz, X1 as close OSC1 & OSC2 as possible.

Components with * are used for EMC issue.

Components with ** are used for resonator only. If necessary.

Crystal or Ceramic Resonator for Multiple I/O Applications – HT82K94A



Note: X1 can use 6MHz or 12MHz, X1 as close OSC1 & OSC2 as possible.

Components with ** are used for resonator only. If necessary.

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](5)	Read ROM code (locate by TBLP and TBHP) to data memory and TBLH	2(1)	None
TABRDC [m](6)	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	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.

(5): ″ROM code TBHP option″ is enabled

(6): ″ROM code TBHP option″ is disabled

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 (locate by TBLP and TBHP) to TBLH and data memory (ROM code TBHP is enabled)

Description The low byte of ROM code addressed by the table pointer (TBLP and TBHP) 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
—	—	—	—	—	—

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

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.

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.

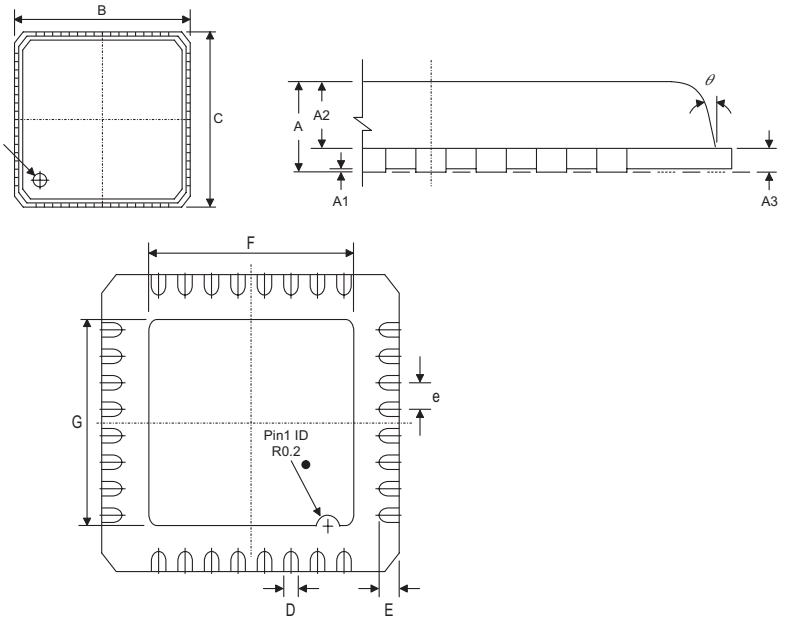
Operation ACC ← ACC ″XOR″ x

Affected flag(s)

TO	PDF	OV	Z	AC	C
—	—	—	√	—	—

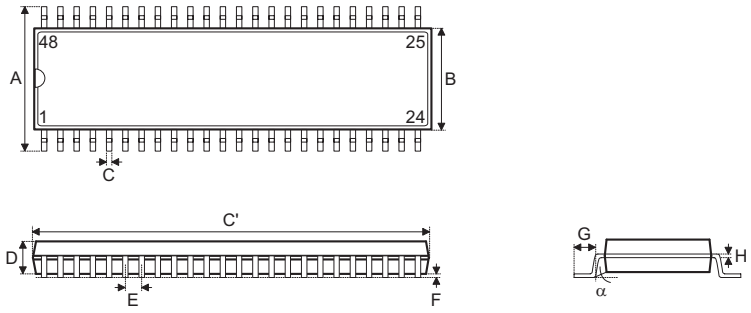
Package Information

32-pin (5×5mm) SAW Type QFN Outline Dimensions



Symbol	Dimensions in mm.		
	Min.	Nom.	Max.
A	0.7	—	0.8
A1	0	—	0.05
A3	—	0.2	—
b	0.18	—	0.3
D	—	5	—
E	—	5	—
e	—	0.5	—
D2	1.25	—	3.25
E2	1.25	—	3.25
L	0.3	—	0.5
K	—	—	—

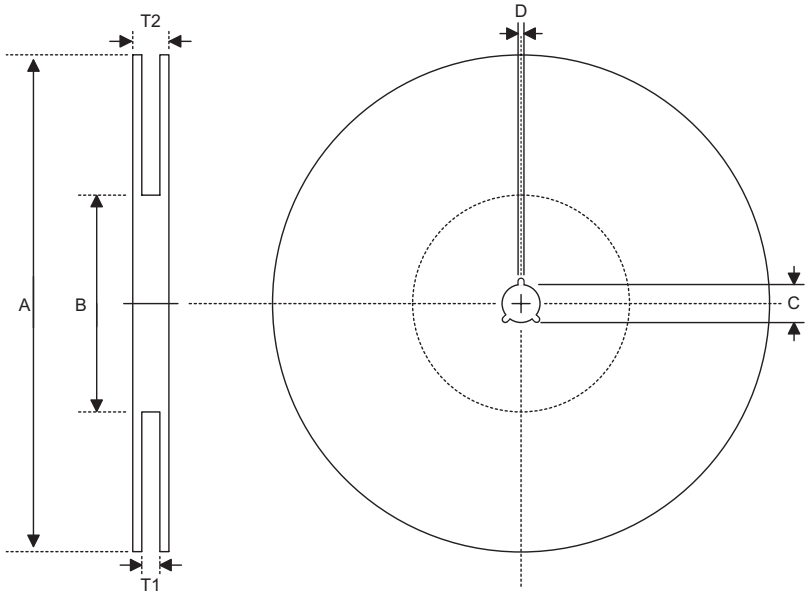
48-pin SSOP (300mil) Outline Dimensions



Symbol	Dimensions in mil		
	Min.	Nom.	Max.
A	395	—	420
B	291	—	299
C	8	—	12
C′	613	—	637
D	85	—	99
E	—	25	—
F	4	—	10
G	25	—	35
H	4	—	12
α	0°	—	8°

Product Tape and Reel Specifications

Reel Dimensions



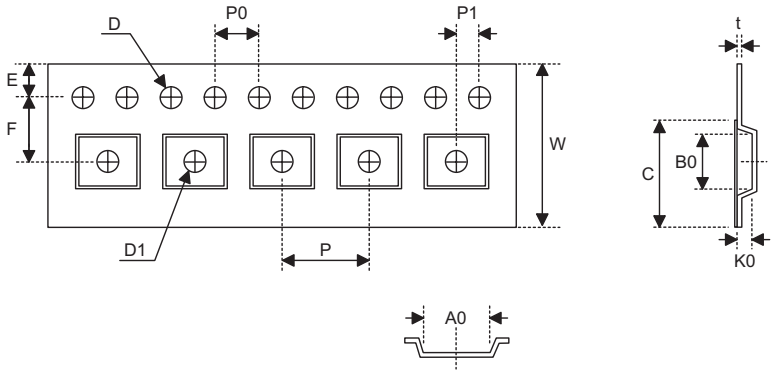
SAW QFN 32 (5×5mm)

Symbol	Description	Dimensions in mm
A	Reel Outer Diameter	330±1
B	Reel Inner Diameter	100±0.1
C	Spindle Hole Diameter	13+0.5 −0.2
D	Key Slit Width	2±0.5
T1	Space Between Flange	12.5+0.3 −0.2
T2	Reel Thickness	—

SSOP 48W

Symbol	Description	Dimensions in mm
A	Reel Outer Diameter	330±1
B	Reel Inner Diameter	100±0.1
C	Spindle Hole Diameter	13+0.5 −0.2
D	Key Slit Width	2±0.5
T1	Space Between Flange	32.2+0.3 −0.2
T2	Reel Thickness	38.2±0.2

Carrier Tape Dimensions



SAW QFN 32 (5×5mm)

Symbol	Description	Dimensions in mm
W	Carrier Tape Width	12±0.3
P	Cavity Pitch	8±0.1
E	Perforation Position	1.75±0.1
F	Cavity to Perforation (Width Direction)	5.5±0.05
D	Perforation Diameter	1.5+0.1
D1	Cavity Hole Diameter	1.5+0.25
P0	Perforation Pitch	4±0.1
P1	Cavity to Perforation (Length Direction)	2±0.05
A0	Cavity Length	5.25±0.1
B0	Cavity Width	5.25±0.1
K0	Cavity Depth	1.1±0.1
t	Carrier Tape Thickness	0.3±0.05
C	Cover Tape Width	—

SSOP 48W

Symbol	Description	Dimensions in mm
W	Carrier Tape Width	32±0.3
P	Cavity Pitch	16±0.1
E	Perforation Position	1.75±0.1
F	Cavity to Perforation (Width Direction)	14.2±0.1
D	Perforation Diameter	2 Min.
D1	Cavity Hole Diameter	1.5+0.25
P0	Perforation Pitch	4±0.1
P1	Cavity to Perforation (Length Direction)	2±0.1
A0	Cavity Length	12±0.1
B0	Cavity Width	16.2±0.1
K1	Cavity Depth	2.4±0.1
K2	Cavity Depth	3.2±0.1
t	Carrier Tape Thickness	0.35±0.05
C	Cover Tape Width	25.5

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)
5F, Unit A, Productivity Building, Gaoxin M 2nd, Middle Zone Of High-Tech Industrial Park, ShenZhen, China 518057
Tel: 86-755-8616-9908, 86-755-8616-9308
Fax: 86-755-8616-9722

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

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