FCC ID: IAJBE139

Technical Description :

The brief circuit description is listed as follows :
For Reader:
- L1, L2 and associated circuit act as Antennas.
- Q1, Q2 and associated circuit act as Switching Circuit.
- Q9, Q10, Q11 and associated circuit act as RF Power Amplifier.
- X2, Q12 and associated circuit act as 13.56 MHz Oscillator and Receiver.
- U3 acts as Singal Amplifier.
- U1 acts as Voice Synthesizer 8-bit MCU.
- Q7, Q8 and associated circuit act as Audio Amplifier.
- Q4, Q5 and associated circuit act as Motor Driver.
- S1~S10 and S12A~S16 act as Control Keys.

For Tag:
- L4 acts as Antenna.
- U4 act as RFID Transponder.

Antenna Used :

Loop antennas have been used.

HT86XXX
Voice Synthesizer 8-Bit MCU

Features

- Operating voltage: 2.4V~5.2V
- System clock: 4MHz~8MHz
- Crystal or RC oscillator for system clock
- 23 I/O pins with 4 shared pins included
- 8K×16-bit program ROM
- 208×8-bit RAM
- One external interrupt input
- Three 16-bit programmable timer counter and overflow interrupts
- 12-bit high quality D/A output by transistor or HT82V733
- Built-in voice ROM in various capacity
- One optional 32768Hz crystal oscillator for RTC time base (8-bit counter with 3-bit prescaler)
- Watchdog Timer
- 8-level subroutine nesting
- HALT function and wake-up feature reduce power consumption
- Up to 1μs (0.5μs) instruction cycle with 4MHz (8MHz) system clock
- Support 16-bit table read instruction (TBLP, TBHP)
- 63 powerful and efficient instructions
- HT86072/144/192/384: 28-pin SOP, 100-pin QFP package
- HT86576/768: 32-pin SOP, 100-pin QFP package

Applications

- Intelligent educational leisure products
- Alert and warning systems
- High end leisure product controllers
- Sound effect generators

General Description

The HT86XXX series are 8-bit high performance microcontroller with voice synthesizer and tone generator. The HT86XXX is designed for applications on multiple I/Os with sound effects, such as voice and melody. It can provide various sampling rates and beats, tone levels, tempos for speech synthesizer and melody generator. It has a single built-in high quality, D/A output. There is an external interrupt which can be triggered with falling edge pulse or falling/rising edge pulse.

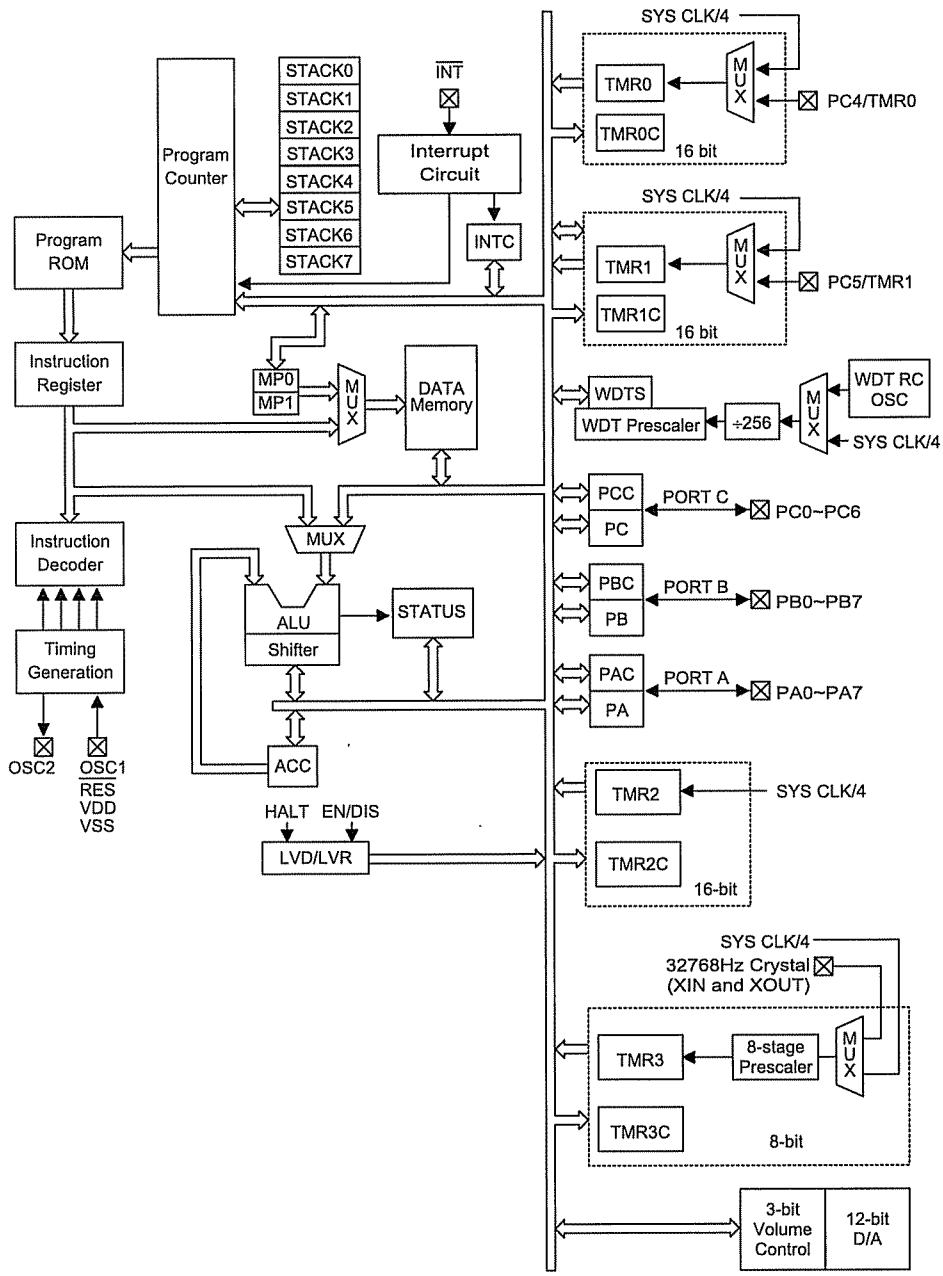
The HT86XXX is excellent for versatile voice and sound effect product applications. The efficient MCU instructions allow users to program the powerful custom applications. The system frequency of HT86XXX can be up to 8MHz under 2.4V and include a HALT function to reduce power consumption.

Selection Table

Body	HT86072	HT86144	HT86192	HT86384	HT86576	HT86768
Voice ROM size	1536K-bit	3072K-bit	4096K-bit	8192K-bit	12288K-bit	16392K-bit
Voice length	72 sec	144 sec	192 sec	384 sec	576 sec	768 sec

Note: * Voice length is estimated by 21K-bit data rate

Block Diagram



Functional Description

Execution Flow

The system clock for the HT86XXX series is derived from either a crystal or an RC oscillator. It is internally divided into four non-overlapping clocks. One instruction cycle consists of four system clock cycles.

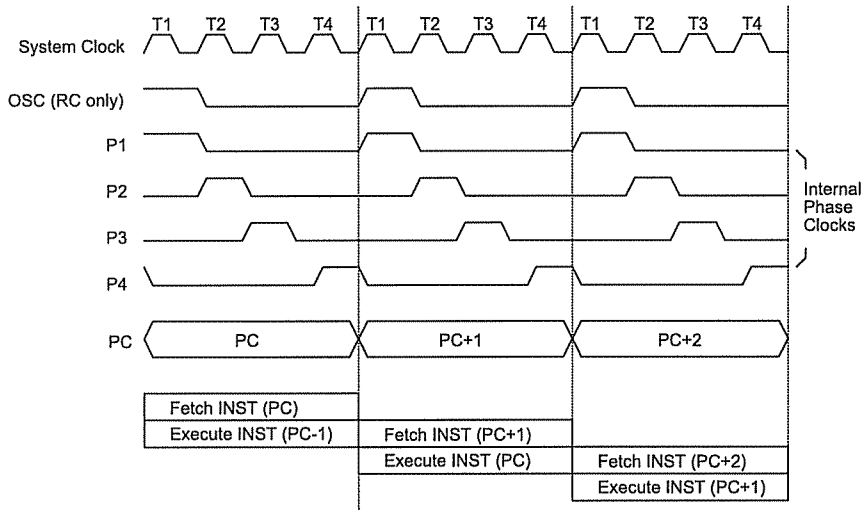
Instruction fetching and execution are pipelined in such a way that a fetch takes one instruction cycle while decoding and execution takes the next instruction cycle. However, the pipelining scheme causes each instruction to effectively execute within one cycle. If an instruction changes the program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

Program Counter – PC

The 13-bit program counter (PC) controls the sequence in which the instructions stored in program ROM are executed.

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

When executing a jump instruction, conditional skip execution, loading PCL register, subroutine call, initial reset, internal interrupt, external interrupt or return from



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
External or Serial Input 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
Timer Counter 2 Overflow	0	0	0	0	0	0	0	0	1	0	0	0	0
Timer Counter 3 Overflow	0	0	0	0	0	0	0	0	1	0	1	0	0
Skip	PC+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
#12~#0: Instruction code bits
S12~S0: Stack register bits
@7~@0: PCL bits

subroutine, the PC manipulates the program transfer by loading the address corresponding to each instruction.

The conditional skip is activated by instruction. Once the condition is met, the next instruction, fetched during the current instruction execution, is discarded and a dummy cycle takes its place while the correct instruction is obtained.

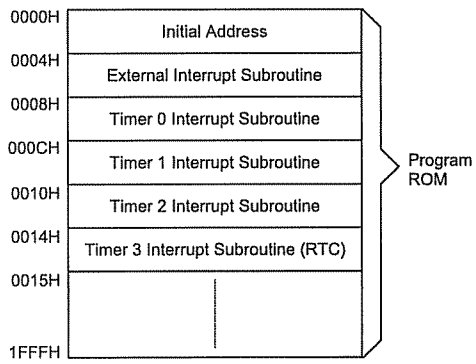
The lower byte of the program counter (PCL) is a read/write register (06H). Moving data into the PCL performs a short jump. The destination must be within 256 locations.

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

Program Memory – ROM

The program memory stores the program instructions that are to be executed. It also includes data, table and interrupt entries, addressed by the program counter along with the table pointer. The program memory size for HT86XXX is 8192×16 bits. Certain locations in the program memory are reserved for special usage:

- Location 000H
  This area is reserved for program initialization. The program always begins execution at location 000H each time the system is reset.
- Location 004H
  This area is reserved for the external interrupt service program. If the $\overline{INT}$ input pin is activated, and the interrupt is enabled and the stack is not full, the program will jump to location 004H and begins execution.



Program Memory

- Location 008H
  This area is reserved for the 16-bit 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 will jump to location 008H and begins execution.
- Location 00CH
  This area is reserved for the 16-bit Timer/Event Counter 1 interrupt service program. If a timer interrupt results from a Timer/Event Counter 1 overflow, and if the interrupt is enabled and the stack is not full, the program will jump to location 00CH and begins execution.
- Location 010H
  This area is reserved for the 16-bit Timer Counter 2 interrupt service program. If a timer interrupt results from a Timer Counter 2 overflow, and if the interrupt is enabled and the stack is not full, the program will jump to location 010H and begins execution.
- Location 014H
  This area is reserved for the 8-bit Timer Counter 3 interrupt service program. If a timer interrupt results from a Timer Counter 3 overflow, and if the interrupt is enabled and the stack is not full, the program will jump to location 014H and begins execution.

Table location

Any location in the ROM space can be used as look up tables. The instructions TABRDC [m] (used for any bank) and TABRDL [m] (only used for last page of program ROM) transfer the contents of the lower-order byte to the specified data memory [m], and the higher-order byte to TBLH (08H). Only the destination of the lower-order byte in the table is well-defined. The higher-order bytes of the table word are transferred to the TBLH. The table higher-order byte register (TBLH) is read only.

The table pointer (TBHP, TBLP) is a read/write register, which indicates the table location. Because TBHP is unknown after power on reset, TBHP must be set specified.

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: Current program ROM table
P12~P8: Write P12~P8 to TBHP pointer register
@7~@0: Write @7~@0 to TBLP pointer register

Stack Register – Stack

The stack register is a special part of the memory used to save the contents of the program counter (PC). This stack is organized into eight levels. It is neither part of the data nor part of the program space, and cannot be read or written to. Its activated level is indexed by a stack pointer (SP) and cannot be read or written to. At a subroutine call or interrupt acknowledgment, the contents of the program counter are pushed onto the stack.

The program counter is restored to its previous value from the stack at the end of subroutine or interrupt routine, which is signaled by return instruction (RET or RETI). After a chip resets, SP will point to the top of the stack.

The interrupt request flag will be recorded but the acknowledgment will be inhibited when the stack is full and a non-masked interrupt takes place. After the stack pointer is decremented (by RET or RETI), the interrupt request will be serviced. This feature prevents stack overflow and allows programmers 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 is lost.

Data Memory – RAM

The data memory is designed with 208×8 bits. The data memory is further divided into two functional groups, namely, special function registers (00H~2AH) and general purpose user data memory (30H~FFH). Although most of them can be read or be written to, some are read only.

The special function registers include an indirect addressing register (R0:00H), memory pointer register (MP0:01H), accumulator (ACC:05H), program counter lower-order byte register (PCL:06H), table pointer (TBLP:07H), table higher-order byte register (TBLH:08H), status register (STATUS:0AH), interrupt control register 0 (INTC:0BH), Timer/Event Counter 0 (TMR0H:0CH,TMR0L:0DH), Timer/Event Counter 0 control register (TMR0C:0EH), Timer/Event Counter 1 (TMR1H:0FH, TMR1L:10H), Timer/Event Counter 1 control register (TMR1C:11H), I/O registers (PA:12H,PB:14H,PC:16H), I/O control registers (PAC:13H,PBC:15H,PCC:17H), voice ROM address latch0[23:0] (LATCH0H:18H, LATCH0M:19H, LATCH0L:1AH), voice ROM address latch1[23:0] (LATCH1H:1BH, LATCH1M:1CH, LATCH1L:1DH), interrupt control register 1 (INTCH:1EH), table pointer higher-order byte register (TBHP:1FH), Timer Counter 2 (TMR2H:20H, TMR2L:21H), Timer Counter 2 control register (TMR2C:22H), Timer Counter 3 (TMR3L:24H), Timer Counter 3 control register (TMR3C:25H), voice control register (VOICEC:26H), DAC output (DAH:27H,DAL:28H), volume control register (VOL:29H), voice ROM latch data register (LATCHD:2AH).

The general purpose data memory, addressed from 30H~FFH, is used for data and control information under instruction commands.

The areas in the RAM can directly handle the arithmetic, logic, increment, decrement, and rotate operations. Except some dedicated bits, each bit in the RAM can be set and reset by "SET [m].i" and "CLR [m].i". They are also indirectly accessible through the memory pointer register 0 (MP0:01H) or the Memory Pointer register 1 (MP1:03H).

Address	RAM Mapping	Read/Write	Description
00H	R0	R/W	Indirect addressing register 0
01H	MP0	R/W	Memory pointer 0
02H	R1	R/W	Indirect addressing register 1
03H	MP1	R/W	Memory pointer 1
04H	Unused		
05H	ACC	R/W	Accumulator
06H	PCL	R/W	Program counter lower-order byte address
07H	TBLP	R/W	Table pointer lower-order byte address
08H	TBLH	R	Table higher-order byte content register
09H	WDTS	R/W	Watchdog Timer option setting register
0AH	STATUS	R/W	Status register
0BH	INTC	R/W	Interrupt control register 0
0CH	TMR0H	R/W	Timer/Event counter 0 higher-byte register
0DH	TMR0L	R/W	Timer/Event counter 0 lower-byte register
0EH	TMR0C	R/W	Timer/Event counter 0 control register

Address	RAM Mapping	Read/Write	Description
0FH	TMR1H	R/W	Timer/Event counter 1 higher-byte register
10H	TMR1L	R/W	Timer/Event counter 1 lower-byte register
11H	TMR1C	R/W	Timer/Event counter 1 control register
12H	PA	R/W	Port A I/O data register
13H	PAC	R/W	Port A I/O control register
14H	PB	R/W	Port B I/O data register
15H	PBC	R/W	Port B I/O control register
16H	PC	R/W	Port C I/O data register
17H	PCC	R/W	Port C I/O control register
18H	LATCH0H	R/W	Voice ROM address latch 0 [A23~A16]
19H	LATCH0M	R/W	Voice ROM address latch 0 [A15~A8]
1AH	LATCH0L	R/W	Voice ROM address latch 0 [A7~A0]
1BH	LATCH1H	R/W	Voice ROM address latch 1 [A23~A16]
1CH	LATCH1M	R/W	Voice ROM address latch 1 [A15~A8]
1DH	LATCH1L	R/W	Voice ROM address latch 1 [A7~A0]
1EH	INTCH	R/W	Interrupt control register 1
1FH	TBHP	R/W	Table pointer higher-order byte register
20H	TMR2H	R/W	Timer Counter 2 higher-byte register
21H	TMR2L	R/W	Timer Counter 2 lower-byte register
22H	TMR2C	R/W	Timer Counter 2 control register
23H	Unused		
24H	TMR3L	R/W	Timer Counter 3 lower-byte register
25H	TMR3C	R/W	Timer Counter 3 control register
26H	VOICEC	R/W	Voice control register
27H	DAL	R/W, higher-nibble available only	DAC output data D3~D0 to DAL7~DAL4
28H	DAH	R/W	DAC output data D11~D4 to DAH7~DAH0
29H	VOL	R/W, higher-nibble available only	Volume control register, and volume controlled by VOL7~VOL5
2AH	LATCHD	R	Voice ROM data register
2BH~2FH	Unused		
30H~FFH	User data RAM	R/W	User data RAM

Indirect Addressing Register

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

The function of data movement between two indirect addressing registers is not supported. The memory pointer registers, MP0 and MP1, are both 8-bit registers used to access the RAM by combining the corresponding indirect addressing registers.

Accumulator – ACC (05H)

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

Arithmetic and Logic Unit – ALU

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

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

Status Register – STATUS (0AH)

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

Except the TO and PD flags, bits in the status register can be altered by instructions similar to other registers. Data written into the status register does not alter the TO or PD flags. Operations related to the status register, however, may yield different results from those intended. The TO and PD flags can only be changed by a Watchdog Timer overflow, chip power-up, or clearing the Watchdog Timer and executing the "HALT" instruction. The Z, OV, AC, and C flags reflect the status of the latest operations.

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

Interrupts

The HT86XXX provides an external interrupt, three 16-bit programmable timer interrupts, and an 8-bit programmable timer interrupt. The Interrupt Control registers (INTC:0BH, INTCH:1EH) contain the interrupt control bits to set to enable/disable and the interrupt request flags.

Once an interrupt subroutine is serviced, all other interrupts will be blocked (by clearing the EMI bit). This scheme may prevent any further interrupt nesting. Other interrupt requests may happen during this interval but only the interrupt request flag is recorded. If a certain interrupt needs servicing within the service routine, the EMI bit and the corresponding INTC/INTCH bit 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.

Labels	Bits	Function
C	0	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.
AC	1	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.
Z	2	Z is set if the result of an arithmetic or logical operation is zero; otherwise Z is cleared.
OV	3	OV is set if an operation results in a carry into the highest-order bit but not a carry out of the highest-order bit, or vice versa; otherwise OV is cleared.
PD	4	PD is cleared by system power-up or executing the "CLR WDT" instruction. PD is set by executing the "HALT" instruction.
TO	5	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 Register

As an interrupt is serviced, a control transfer occurs by pushing the program counter onto the stack and then branching to subroutines at the specified location(s) in the program memory. Only the program counter is pushed onto the stack. The programmer must save the contents of the register or status register (STATUS) in advance if they are altered by an interrupt service program which corrupts the desired control sequence.

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

The internal Timer/Event Counter 0 interrupt is initialized by setting the Timer/Event counter 0 interrupt request flag (T0F:bit 5 of INTC), caused by a Timer/Event Counter 0 overflow. When the interrupt is enabled, and 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/Event Counter 1 interrupt is initialized by setting the Timer/Event Counter 1 interrupt request flag (T1F:bit 6 of INTC), caused by a Timer/Event Counter 1 overflow. When the interrupt is enabled, and the stack is not full and the T1F bit 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.

The internal Timer Counter 2 interrupt is initialized by setting the Timer Counter 2 interrupt request flag (T2F:bit 0 of INTCH), caused by a Timer Counter 2 overflow. When the interrupt is enabled, and the stack is not full and the T2F bit is set, a subroutine call to location 10H will occur. The related interrupt request flag (T2F) will be reset and the EMI bit cleared to disable further interrupts.

The internal Timer Counter 3 interrupt is initialized by setting the Timer Counter 3 interrupt request flag (T3F:bit 1 of INTCH), caused by a Timer Counter 3 overflow. When the interrupt is enabled, and the stack is not full and the T3F bit is set, a subroutine call to location 14H will occur. The related interrupt request flag (T3F) will be reset and the EMI bit cleared to disable further interrupts.

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

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

The Timer/Event Counter 0/1 interrupt request flag (T0F/T1F) which enables Timer/Event Counter 0/1 control bit (ET0I/ET1I), the Timer Counter 2/3 interrupt request flag (T2F/T3F) which enables Timer Counter 2/3 control bit (ET2I/ET3I), and external interrupt request flag (EIF) which enables external interrupt control bit (EEI) form the interrupt control register (INTC:0BH and INTCH:1EH). EMI, EEI, ET0I, ET1I, ET2I, and ET3I are used to control the enabling/disabling of interrupts. These bits prevent the requested interrupt begin serviced. Once the interrupt request flags (T0F, T1F, T2F, T3F, EIF) are set, they will remain in the INTC/INTCH register until the interrupts are serviced or cleared by a software instruction.

It is recommended that application programs do not use CALL subroutines within an 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 the interrupt enable is not well controlled, once a CALL subroutine if used in the interrupt subroutine will corrupt the original control sequence.

Register	Bit No.	Label	Function
INTC (0BH)	0	EMI	Controls the master (global) interrupt (1= enabled; 0= disabled)
	1	EEI	Controls the external interrupt (1= enabled; 0= disabled)
	2	ET0I	Controls the timer 0 interrupt (1= enabled; 0= disabled)
	3	ET1I	Controls the timer 1 interrupt (1= enabled; 0= disabled)
	4	EIF	External interrupt request flag (1= active; 0= inactive)
	5	T0F	Timer 0 request flag (1= active; 0= inactive)
	6	T1F	Timer 1 request flag (1= active; 0= inactive)
	7	—	Unused bit, read as "0"

INTC0 Register

Register	Bit No.	Label	Function
INTCH (1EH)	0	ET2I	Controls the timer 2 interrupt (1= enabled; 0= disabled)
	1	ET3I	Controls the timer 3 interrupt (1= enabled; 0= disabled)
	2, 3	—	Unused bit, read as "0"
	4	T2F	Timer 2 interrupt request flag (1= active; 0= inactive)
	5	T3F	Timer 3 interrupt request flag (1= active; 0= inactive)
	6, 7	—	Unused bit, read as "0"

INTC1 Register

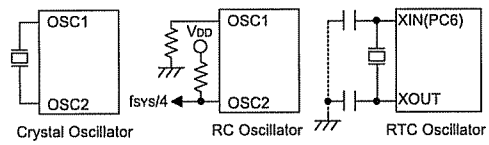
Interrupt Source	Priority	Vector
External Interrupt	1	04H
Timer/Event Counter 0 Overflow	2	08H
Timer/Event Counter 1 Overflow	3	0CH
Timer Counter 2 Overflow	4	10H
Timer Counter 3 Overflow	5	14H

Oscillator Configuration

The HT86XXX provides two types of oscillator circuit for the system clock, i.e., RC oscillator and crystal oscillator. No matter what type of oscillator, the signal is used for the system clock. The HALT mode stops the system oscillator and ignores external signal to conserve power. If the RC oscillator is used, an external resistor between OSC1 and VSS is required, and the range of the resistance should be from 30kΩ to 680kΩ. The system clock, divided by 4, is available on OSC2 with pull-high resistor, which can be used to synchronize external logic. The RC oscillator provides the most cost effective solution. However, the frequency of the oscillation may vary with VDD, temperature, and the chip itself due to process variations. It is therefore not suitable for timing sensitive operations where accurate oscillator frequency is desired.

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

There is another oscillator circuit designed for Timer3's clock source as the RTC time base which is determined by mask option. If the mask option determines that Timer3's clock source is from a 32kHz crystal, then a 32kHz crystal should be connected to XIN and XOUT.



System Oscillator

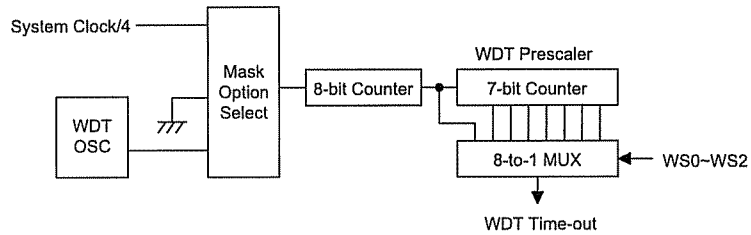
Watchdog Timer – WDT

The WDT clock source is implemented by a dedicated RC oscillator (WDT oscillator) or instruction clock (system clock divided by 4), decided by mask options. This timer is designed to prevent a software malfunction or sequence jumping to an unknown location with unpredictable results. The Watchdog Timer can be disabled by mask 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 with period 78μs normally) is selected, it is first divided by 256 (8-stages) to get the nominal time-out period of approximately 20 ms. This time-out period may vary with temperature, VDD and process variations. By invoking the WDT prescaler, longer time-out period can be realized. Writing data to WS2, WS1, WS0 (bit 2,1,0 of WDTS(09H)) can give different time-out period.

If WS2, WS1, WS0 all equal to 1, the division ratio is up to 1:128, and the maximum time-out period is 2.6 seconds.

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



Watchdog Timer

The WDT overflow under normal operation will initialize a "chip reset" and set the status bit "TO". Whereas in the HALT mode, the overflow will initialize a "warm reset" only the PC and SP are reset to zero. To clear the contents of the WDT (including the WDT prescaler), three methods are adopted; external reset (external reset (a low level to $\overline{RES}$), software instructions, or a HALT instruction. The software instruction is "CLR WDT" and execution of the "CLR WDT" instruction will clear the WDT.

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 Register

Power Down – HALT

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

The system oscillator will be turned off but the WDT oscillator keeps running (if the WDT oscillator is selected).

- The contents of the on chip RAM and registers remain unchanged.
- WDT and WDT prescaler will be cleared and recount again.
- All I/O ports maintain their their original status.
- The PD flag is set and the TO flag is cleared.

The system can leave the HALT mode by means of an external reset, an interrupt, an external falling edge signal on port A or a WDT overflow. An external reset causes a device initialization and the WDT overflow performs a "warm reset". By examining the TO and PD flags, the reason for the chip reset can be determined. The PD flag is cleared when the system powers-up or executes the "CLR WDT" instruction, and is set when the "HALT" instruction is executed. The TO flag is set if a WDT time-out occurs, and causes a wake-up that only resets the PC and SP. The other maintain their original status.

The port A wake-up and interrupt methods can be considered as a continuation of normal execution. Each bit in port A can be independently selected to wake up the device by a mask option. Awakening from an I/O port stimulus, the program will resume execution of the next instruction. If awakening from an interrupt, two sequences may happen. If the related interrupt is disabled or the interrupt is enabled by 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.

Once a wake-up event occurs, it takes 1024 system clock period to resume normal operation. In other words, a dummy cycle period will be inserted after a wake-up. If the wake-up results from an interrupt acknowledge, the actual interrupt subroutine will be delayed by one more cycle. If the wake-up results in next instruction execution, this will be executed immediately after a dummy period is finished. 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. To minimize power consumption, all I/O pins should be carefully managed before entering the HALT status.

Reset

There are 3 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

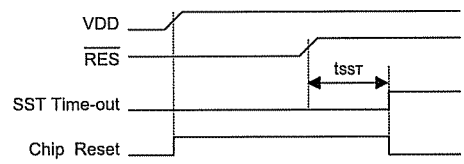
The WDT time-out during HALT is different from other chip reset conditions, since it can perform a "warm reset" that resets only the PC and SP, leaving the other circuits in their original state. Some registers remain unchanged during any other reset conditions. Most registers are reset to their "initial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

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

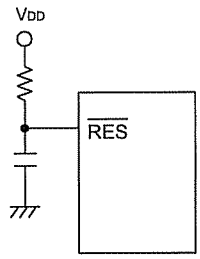
Note: "u" stands for "unchanged"

To guarantee that the system oscillator has started and stabilized, the SST (System Start-up Timer) provides an extra-delay of 1024 system clock pulses after a system power up or when awakening from a HALT state.

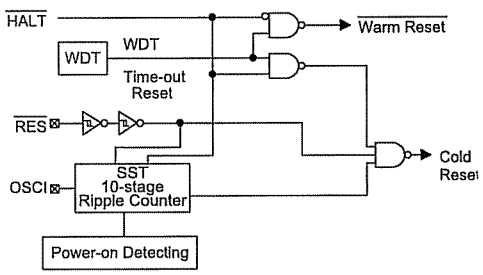
When a system power up occurs, the SST delay is added during the reset period. But when the reset comes from the $\overline{RES}$ pin, the SST delay is disabled. Any wake-up from HALT will enable the SST delay.



Reset Timing Chart



Reset Circuit



Reset Configuration

The function unit chip reset status are shown below.

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

Timer/Event Counter 0/1

There are four timer counters are implemented in the HT86XXX. The Timer/Event Counter 0 and 1 contain 16-bit programmable count-up counters whose clock may come from an external source or the system clock divided by 4 (T1). Using the internal instruction clock (T1), there is only one reference time base. The external clock input allows the user to count external events, measure time intervals or pulse width, or to generate an accurate time base.

There are three registers related to Timer/Event Counter 0; TMR0H (0CH), TMR0L (0DH), TMR0C (0EH). Writing to TMR0L only writes the data into a low byte buffer. Writing to TMR0H will write the data and the contents of the low byte buffer into the Timer/Event Counter 0 preload register (16-bit) simultaneously. The Timer/Event Counter 0 preload register is changed only by a write to TMR0H operation. Writing to TMR0L will keep the Timer/Event Counter 0 preload register unchanged.

Reading TMR0H will also latch the TMR0L into the low byte buffer to avoid false timing problems. Reading the TMR0L only returns the value from the low byte buffer which may be a previously loaded value. In other words, the low byte of Timer/Event Counter 0 cannot be read directly. It must read the TMR0H first to ensure that the low byte contents of Timer/Event Counter 0 are latched into the buffer.

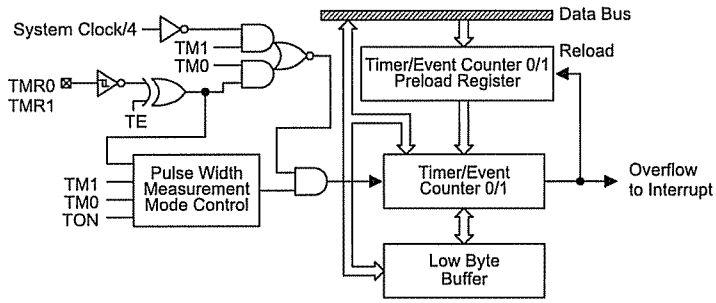
There are three registers related to the Timer/Event Counter 1; TMR1H (0FH), TMR1L (10H), TMR1C (11H). The Timer/Event Counter 1 operates in the same manner as Timer/Event Counter 0.

Label	Bits	Function
—	0~2	Unused bit, read as "0"
TE	3	To define the TMR0/TMR1 active edge of timer/event counter (0=active on low to high; 1=active on high to low)
TON	4	To enable/disable timer counting (0=disabled; 1=enabled)
—	5	Unused bit, read as "0"
TM0, TM1	6 7	To define the operating mode (TMR1, TMR0) 01=Event count mode (external clock) 10=Timer mode (internal clock) 11=Pulse width measurement mode 00=Unused

TMR0C/TMR1C Register

Label	Bits	Function
—	0~2	Unused bit, read as "0"
TE	3	To define the TMR0/TMR1 active edge of timer/event counter (0=active on low to high; 1=active on high to low)
TON	4	To enable/disable timer counting (0=disabled; 1=enabled)
—	5	Unused bit, read as "0"
TM0, TM1	6 7	To define the operating mode (TMR1, TMR0) 01=Unused 10=Timer mode (internal clock) 11=Unused 00=Unused

TMR2C Register



Timer/Event Counter 0/1

The TMR0C is the Timer/Event Counter 0 control register, which defines the Timer/Event Counter 0 options. The Timer/Event Counter 1 has the same options as the Timer/Event Counter 0 and is defined by TMR1C.

The timer/event counter control registers define 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 implies that the clock source comes from an external (TMR0/TMR1 is connected to PC4/PC5) pin. The timer mode functions as a normal timer with the clock source coming from the instruction clock. The pulse width measurement mode can be used to count the high or low level duration of an external signal (TMR0/TMR1). The counting method is based on the instruction clock.

In the event count or timer mode, once the timer/event counter starts counting, it will count from the current contents in the timer/event counter to FFFFH. Once an overflow occurs, the counter is reloaded from the timer/event counter preload register and generates a corresponding 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 bit is 0) it will start counting until the TMR0/TMR1 returns to the original level and resets TON. The measured result will remain in the timer/event counter even if the activated transient occurs again. In other words, only one cycle measurement can be done. When TON is set again, the cycle measurement will function again as long as it receives further transient pulses. Note that, in this operating mode, the timer/event counter starts counting not according to the logic level but according to the transient edges. In the case of counter overflows, the counter is reloaded from the timer/event counter preload register and issues the interrupt request just like in the other two modes.

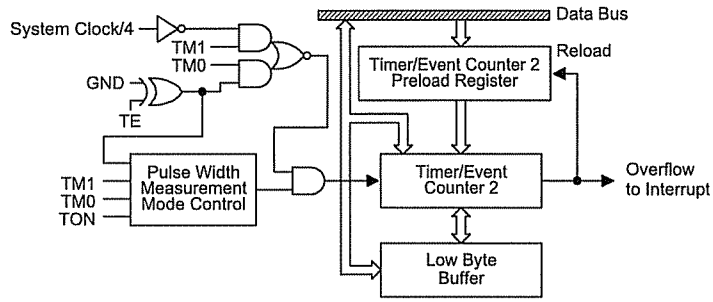
To enable the counting operation, the Timer ON bit (TON; bit 4 of TMR0C/TMR1C) should be set to 1. In the pulse width measurement mode, TON will be cleared automatically after the measurement cycle is complete. But in the other two modes TON can only be reset by instruction. The overflow of the timer/event counter is one of the wake-up sources. No matter what the operation mode is, writing a 0 to ET0I/ET1I can disable the corresponding interrupt service.

In the case of a Timer/Event Counter OFF condition, writing data to the timer/event counter preload register will also reload that data to the timer/event counter. But if the timer/event counter is turned on, data written to the timer/event counter will only be kept in the timer/event counter preload register. The timer/event counter will continue to operate until an overflow occurs.

When the timer/event counter (reading TMR0H/TMR1H) is read, the clock will be blocked to avoid errors. As this may result in a counting error, this must be taken into consideration by the programmer.

Timer Counter 2

The timer counter TMR2 is also a 16-bit programmable count-up counter. It operates in the same manner as Timer/Event Counter 0/1, but the clock source of TMR2 is from only internal instruction cycle (T1). Therefore only (TM1,TM0)=(1,0) is allowable.



Timer Counter 2

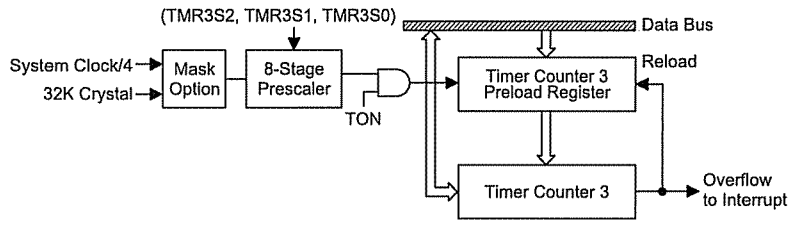
Timer Counter 3 (RTC Time Base)

The timer counter TMR3 is an 8-bit programmable count-up counter. Its counting is as the same manner as Timer Event Counter 0/1 and Timer Counter 2, but the clock source of TMR3 can be from internal instruction cycle (T1) or external 32kHz crystal which is connected to XIN and XOUT. The TMR3's clock source is determined by mask option. If the 32kHz crystal is enabled, then TMR3's clock source is 32kHz which is from XIN and XOUT. If the 32kHz crystal is disabled, then TMR3's clock source is internal T1.

The TMR3 is internal clock source only, i.e. (TM1,TM0)=(1,0). There is a 3-bit prescaler (TMR3S2,TMR3S1,TMR3S0) which defines different division ratio of TMR3's clock source.

Label	Bits	Function
TMR3S2, TMR3S1, TMR3S0	0~2	To define the operating clock source (TMR3S2, TMR3S1, TMR3S0) 000: clock source/2 001: clock source/4 010: clock source/8 011: clock source/16 100: clock source/32 101: clock source/64 110: clock source/128 111: clock source/256
TE	3	To define the TMR3 active edge of timer/event counter (0=active on low to high; 1=active on high to low)
TON	4	To enable/disable timer counting (0=disabled; 1=enabled)
—	5	Unused bit, read as "0"
TM0, TM1	6 7	To define the operating mode (TM1, TM0) 01=Unused 10=Timer mode (internal clock) 11=Unused 00=Unused

TMR3 Register



Timer Counter 3

The registers states are summarized in the following table.

Register	Reset (Power On)	WDT Time-out (Normal Operation)	RES Reset (Normal Operation)	RES Reset (HALT)	WDT Time-out (HALT)
PC	0000H	0000H	0000H	0000H	0000H
MP0	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
MP1	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
ACC	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLP	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TBLH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
WDTS	0000 0111	0000 0111	0000 0111	0000 0111	uuuu uuuu
STATUS	--00 xxxx	--1u uuuu	--uu uuuu	--01 uuuu	--11 uuuu
INTC	-000 0000	-000 0000	-000 0000	-000 0000	-uuu uuuu
TMR0H	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR0L	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR0C	00-0 1---	00-0 1---	00-0 1---	00-0 1---	uu-u u---
TMR1H	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR1L	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR1C	00-0 1---	00-0 1---	00-0 1---	00-0 1---	uu-u u---
PA	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PAC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PB	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PBC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
PCC	1111 1111	1111 1111	1111 1111	1111 1111	uuuu uuuu
TMR2H	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR2L	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR2C	00-0 1---	00-0 1---	00-0 1---	00-0 1---	uu-u u---
TMR3L	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
TMR3C	00-0 1xxx	00-0 1uuu	00-0 1uuu	00-0 1uuu	uu-u uuuu
INTCH	-000 ---0	-000 ---0	-000 ---0	-000 ---0	-uuu ---u
TBHP	---x xxxx	---u uuuu	---u uuuu	---u uuuu	---u uuuu
DAL	xxxx ----	uuuu ----	uuuu ----	uuuu ----	uuuu ----
DAH	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
VOL	xxx- ----	uuu- ----	uuu- ----	uuu- ----	uuu- ----
VOICEC	0--0 -00-	u--u -uu-	u--u -uu-	u--u -uu-	u--u -uu-
LATCH0H	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
LATCH0M	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
LATCH0L	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
LATCH1H	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
LATCH1M	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
LATCH1L	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu
LATCHD	xxxx xxxx	uuuu uuuu	uuuu uuuu	uuuu uuuu	uuuu uuuu

Note: "u" means "unchanged"
"x" means "unknown"
"–" means "undefined"

Input/Output Ports

There are 23 bidirectional input/output lines in the microcontroller, labeled from PA to PC, which are mapped to the data memory of [12H], [14H], and [16H] 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 or 16H). 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) to control the input/output configuration. With this control register, CMOS output or Schmitt trigger input with or without pull-high resistor structures can be reconfigured dynamically (i.e. on-the-fly) under software control. To function as an input, the corresponding latch of the control register must write "1". The input source also depends on the control register. If the control register bit is "1", the input will read the pad state. If the control register bit is "0", the contents of the latches will move to the internal bus. The latter is possible in the "read-modify-write" instruction.

For output function, CMOS is the only configuration. These control registers are mapped to locations 13H, 15H, and 17H. Bit 7 which is mapped to location [17H] is always written as "1".

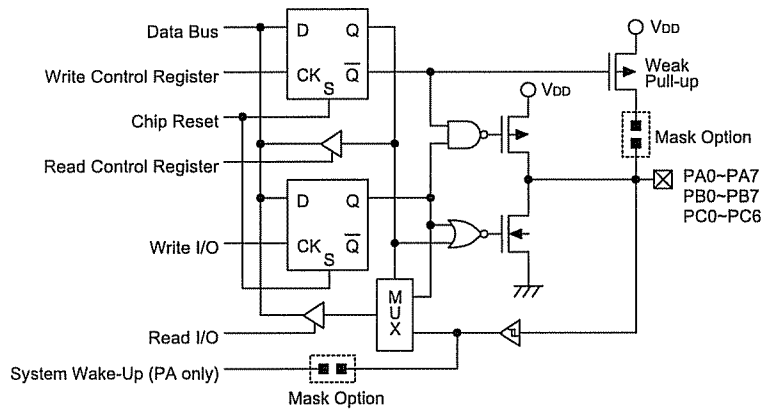
After a chip reset, these input/output lines remain at high levels or floating state (dependent on pull-high options).

Each bit of these input/output latches can be set or cleared by "SET [m].i" and "CLR [m].i" (m=12H, 14H, 16H) 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 port A has the capability of waking-up the device. The wake-up capability of port A is determined by mask option. There is a pull-high option available for all I/O lines. Once the pull-high option is selected, all I/O lines have pull-high resistors. Otherwise, the pull-high resistors are absent. It should be noted that a non-pull-high I/O line operating in input mode will cause a floating state.

By some different mask options, there are 3 shared pins (PC.4, PC.5, and PC.6) in PC. They can be normal I/O pins or for special functions. The PC.4 is the external clock source of timer/event counter TMR0 if TMR0 is set to external clock mode, and the PC.5 is the external clock source of timer/event counter TMR1 if TMR1 is set to external clock mode. PC6 is pin-shared with XIN. The XIN and XOUT can be connected to a 32kHz crystal as the clock source of the timer counter TMR3 if the mask option is set to enable 32kHz (RTC) crystal.



Input/Output Ports

Audio Output and Volume Control – DAL, DAH, VOL

The HT86XXX provides one 12-bit current type DAC device for driving external 8Ω speaker through an external NPN transistor. The programmer must write the voice data to register DAL (27H) and DAH (28H). The 12-bit audio output will be written to the higher nibble of DAL and the whole byte of DAH, and the DAL3~0 is always read as 0H. There are 8 scales of volume controllable level that are provided for the current type DAC output. The programmer can change the volume by only writing the volume control data to the higher-nibble of the VOL (29H), and the lower-nibble of VOL (29H) is always read as 0H.

Voice Control Register – VOICEC (26H)

The voice control register controls the voice ROM circuit and DAC circuit, selects voice ROM latch counter, and controls 32kHz crystal to start in speed-up mode or not. If the DAC circuit is not enabled, any DAH/DAL output is invalid. Writing a "1" to DAC bit is to enable DAC circuit, and writing a "0" to DAC bit is to disable DAC circuit. If the voice ROM circuit is not enabled, then voice ROM data cannot be accessed at all. Writing a "1" to VROMC bit is to enable the voice ROM circuit, and writing a "0" to VROMC bit is to disable the voice ROM circuit. The bit 4 (LATCHC) is to determine what voice ROM address latch counter will be adopted as voice ROM address latch counter. The bit 7 (FAST) is to determine how to activate 32kHz crystal of TMR3's clock source.

Label	Bits	Function
—	0	Unused bit, read as "0"
DAC	1	Enable/disable DAC circuit (0= disable DAC circuit; 1= enable DAC circuit) The DAC circuit is not affected by the HALT instruction. The software controls bit DAC (VoiceC.1) whether to enable/disable.
VROMC	2	Enable/disable voice ROM circuit (0= disable voice ROM circuit; 1= enable voice ROM circuit)
—	3	Unused bit, read as "0"
LATCHC	4	Select voice ROM counter (0= voice ROM address latch 0; 1= voice ROM address latch 1)
—	5, 6	Unused bit, read as "0"
FAST	7	Enable/disable speed-up 32kHz crystal. Default to 0. (0= speed-up 32kHz crystal; 1= non-speed-up 32kHz crystal)

Voice ROM Data Address Latch Counter

LATCH0H(18H)/LATCH0M(19H)/LATCH0L(1AH), LATCH1H(1BH)/LATCH1M(1CH)/LATCH1L(1DH) and voice ROM data register(2AH)

The voice ROM data address latch counter is the handshaking between the microcontroller and voice ROM, where the voice codes are stored. One 8-bit of voice ROM data will be addressed by setting 24-bit address latch counter LATCH0H/LATCH0M/LATCH0L or LATCH1H/LATCH1M/LATCH1L. After the 8-bit voice ROM data is addressed, a few instruction cycles (4μs at least) will be cost to latch the voice ROM data, then the microcontroller can read the voice data from LATCHD(2AH).

Example: Read an 8-bit voice ROM data which is located at address 000007H by address latch 0

```
set     [26H].2         ; Enable voice ROM circuit
clr     [26H].4         ; Select voice ROM address
                        ; latch counter 0
mov     A, 07H          ;
mov     LATCH0L, A      ; Set LATCH0L to 07H
mov     A, 00H          ;
mov     LATCH0M, A      ; Set LATCH0M to 00H
mov     A, 00H          ;
mov     LATCH0H, A      ; Set LATCH0H to 00H
call    Delay Time      ; Delay a short period of time
mov     A, LATCHD       ; Get voice data at 000007H
```

Mask Option

Mask Option	Description
PA Wake-up	Enable/disable PA wake-up function
Watchdog Timer (WDT)	Enable/disable WDT function One or two CLR instruction WDT clock source is from WDTOSC or T1
External INT Trigger Edge	External INT is triggered on falling edge only, or is triggered on falling and rising edge.
Timer 3 Clock Source	Timer3's clock source is from T1, or is from the external 32kHz crystal which is connected to XIN and XOUT.
External Timer 0/1 Clock Source	Enable/disable external timer of timer 0 and timer 1, share with PC4 and PC5.
PA Pull-high	Enable/disable PA pull-high
PB Pull-high	Enable/disable PB pull-high
PC Pull-high	Enable/disable PC pull-high

f_{OSC} – R_{OSC} Table (V_{DD}=3V)

f_{OSC}	R_{OSC}
4MHz±10%	100kΩ
6MHz±10%	75kΩ
8MHz±10%	62kΩ

HT82V732
60mA Audio Power Amp

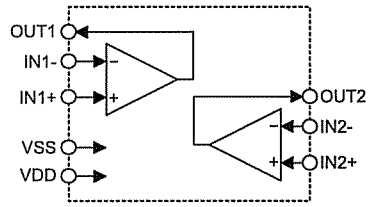
Features

- Single power supply voltage: 5V
- Low power consumption
- Low distortion
- Low clock Jitter sensitivity
- High SNR ratio range
- Wide temperature range
- 8-pin SOP package

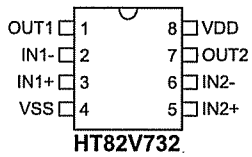
General Description

HT82V732 is a class AB stereo earphone driver designed for portable digital audio application. It provides 8-SOP package. Pin assignments and application circuit is compatible with TDA1308 which is suitable for effective low cost applications. HT82V732 is ideal for portable digital audio equipment, CD ROM/DVD ROM and DISCMAN system.

Block Diagram



Pin Assignment



HT82V732
–8 SOP-A

Pin Description

Pin No.	Pin Name	I/O	Description
1	OUT1	O	Output
2	IN1–	I	Inverting input
3	IN1+	I	Non-inverting input
4	VSS	—	Negative power supply, ground
5	IN2+	I	Non-inverting input
6	IN2–	I	Inverting input
7	OUT2	O	Output
8	VDD	—	Positive power supply

HT672A

13.56MHz RFID Transponder

Features

- Low low operating current (15μA @ V_{DD}=2V)
- Wide range operating voltage
- Battery less RF transponder
- Data transmission in read-only operation
- Max. of 64-bits customer programmable data
- 16-bits CRC error detection code
- OTP data memory
- 13.56MHz carrier frequency
- Output data baud rate: 5kbps (Typ.)
- PWM/ASK modulation
- Built-in voltage limiter

Applications

- Interactive toys
- Security system
- Access control
- Anti-counterfeit for commercial product
- Material management
- Animal management
- Personnel working time record
- Car park monitoring system

General Description

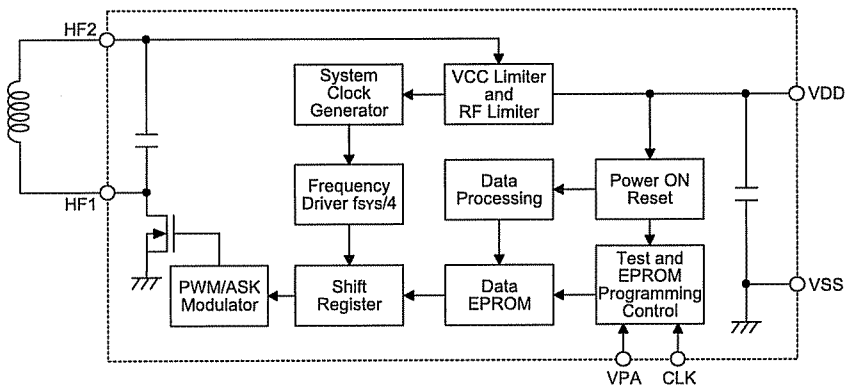
The HT672A is an RF transponder IC with 13.56MHz RF carrier, which provides a low cost battery less transponder solution when combined with an external inductor. The inductor and internal capacitor form an LC tank which induce voltage from the radiated 13.56MHz carrier signal generated from the reader antenna. HT672A has a built-in low power RC oscillator which is activated if the induced carrier field strength is high enough to supply the operating current and the response signal (pre-programmed in the OTP memory) is serially transmitted out. The response data is transmitted using PWM/ASK modulation. Modulation of 13.56MHz is accomplished by damping the LC tank with a fixed baud rate.

The transmission information is stored in a 96 bits one time programmable memory OTP, with a 16-bit CRC code (up to 64 bits reserved for customer). The effective detection range for a small sized antenna is 2cm~10cm which is dependent on antenna format & reader design. The larger the antenna loop used the longer the detection range. It is advisable to use larger antenna to attain a 15 cm detection range.

Implementing Holtek's advanced OTP and low power technology, HT672A offers a very cost effective solution for RF contact less detection system.

A code area of 64-bits (max.) wide is provided so customers can program the device using the specified programmer supplied by Holtek. The pre-programmed ICs are also available upon customer's request.

Block Diagram



Electrical Characteristics

Symbol	Parameter	Test Conditions		Min.	Typ.	Max.	Unit
		V_{DD}	Conditions				
V_{DD}	Operating Voltage	—	V_{DD} pad voltage	2	3	4	V
I_{dd}	Operating Current	2V	—	—	15	—	μA
		3V	Voltage limiter not started	—	600	—	μA
R_m	Modulation Resistance	5V	—	—	320	—	Ω
V_{LCL}	LC Input Limiter Voltage	—	—	—	6.5	—	V
B_R	Output Data Baud Rate	3V	V_{DD} vs V_{SS}	2.5	5	7.5	Kbps

Functional Description

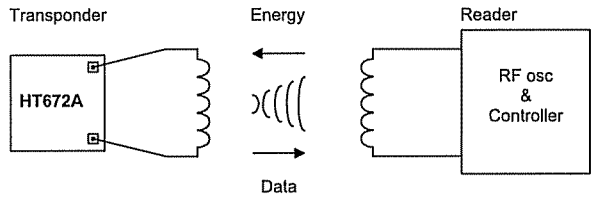
Operation Concept

The reader transmits a 13.56MHz carrier signal from its antenna, the LC tank on the transponder side converts the carrier energy to voltage form and supply to the transponder chip with an internal pump circuit. If the induced energy is high enough, the pumped voltage reaches the break-in voltage of the internal RC-oscillator, the transponder is actuated to transmit its internal data serially by means of damping the LC tank.

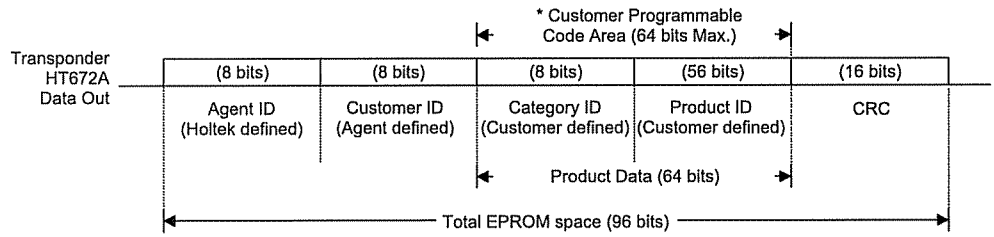
The reader receives the transponder's data by means of detecting the energy variation on its own antenna, and recognize the information with a microcontroller.

The HT672A has a built-in internal Voltage Limiter to prevent excess power supply and RF levels induced by the LC tank from damaging the device or causing the device to function abnormally.

A total of 96 bits of OTP memory space is provided, from which 64 bits wide are customer programmable, which can be programmed using the specified programmer supplied by Holtek. The pre-programmed ICs are also available upon customer's request.



Timing & Code Package



Code Package

A total of 96 bits information can be stored in the HT672A, from which 64 bits are customer programmable.

Agent ID: This 8-bit wide code is not customer programmable and is supplied together with the data writer after registering to Holtek. The writer generates the code automatically.

Customer ID: This area is for the Agent, for example used to store current number of customer.

Category ID: Can be used to store the application field information code.

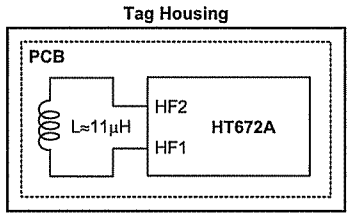
Product ID: Storing the contents of the user ID number or data.

Data CRC: A 16 bits of CRC code is generated automatically by the writer.

Application Circuits

Tag

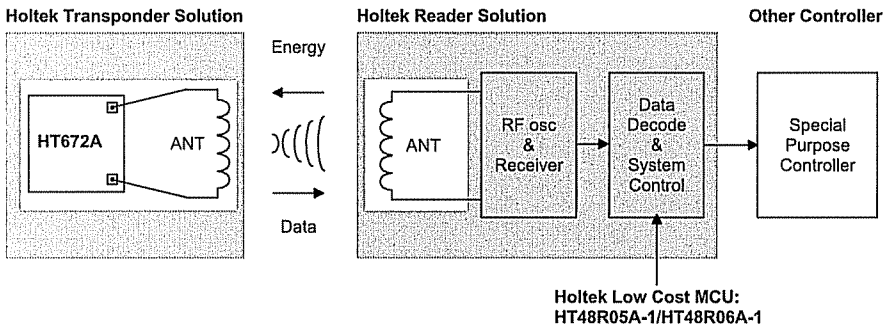
A tag consists of a PCB (or Mylar film) with printed coil, HT672A and a housing. The housing can be of various shapes.



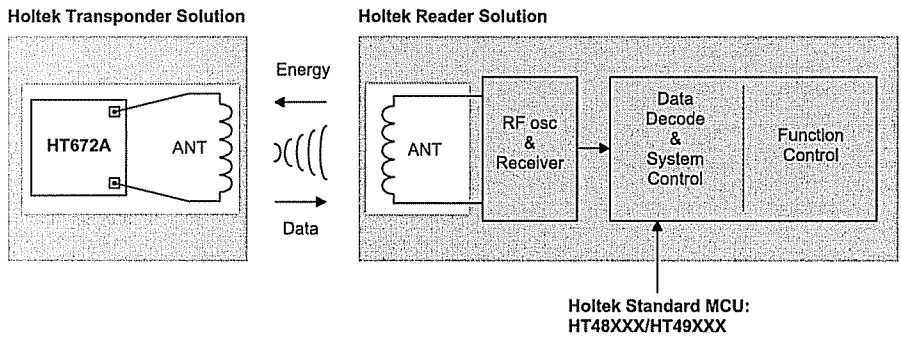
Note: The value of the antenna inductance is 11μH, however the optimum value will be changed slightly due to the variation of the internal resonance capacitor (10pF typically) during process.

For more application information about the reader, refer to Holtek's 13.56MHz RF ID reader data.

2-chip Solution



1-chip Solution (I)



1-chip Solution (II)