

# 透过 HT48R10-1 BZ 口输出 32768Hz Crystal 的使用

文件编码：HA0099S

## 使用说明

将 32K 晶振直接挂在 PA.0 和 PA.1 上，PA.0 作为输入口，记作 OSCI，PA.1 作为输出口，记作 OSCO，其作用为读取 OSCI 的值并将其取反输出。程序运行后将会从 PB.0 口输出一个频率为 32K 的方波，同时在 PA.1 口也可以看到与 PB.0 同步的波形。

C\_TMRC, C\_TMR 和 C\_INTC 的值分别为在程序中需写入 TMRC, TMR 及 INTC 的值，C\_INIT\_L 和 C\_INIT\_H 为使 32K 晶振起振所需的循环次数的低位及高位。R\_INIT 为记忆 C\_INIT\_L 值的缓存器。

子程序 INI\_TIAL 作用为设置 IO 口的输入输出状态，对 TMR 及 INT 进行初始设置。子程序 INI\_32K 作用为实时追踪 OSCI 的输入状态并及时改变 OSCO 的输出状态，使 32K 晶振能起振并达到稳定状态。子程序 SBR\_OSCI\_L 作用为等待 OSCI 输入变为低准位，随后将 OSCO 取反输出，同时将 OSCO 输出的值从 PB.0 同步输出。子程序 SBR\_OSCI\_H 作用为等待 OSCI 输入变为高准位，随后将 OSCO 取反输出，同时将 OSCO 输出的值从 PB.0 同步输出。TIMER 的中断服务程序 ISR\_TIMER 的作用为等待 OSCI 状态的翻转然后改变 OSCO 及 PB.0 的输出。

在 32K 晶振稳定起振后，微控制器每隔 28 个指令周期才会进入一次 TIMER 中断，这 28 个指令周期可以运行其它的程序。由于每个 32K 的晶振所振出的频率略有不同，如果使用中发现 32K 晶振短暂振荡后停振，可以将 C\_TMR 改为一个较大的值直到能使 32K 晶振稳定振荡。注意：改变 C\_TMR 的值会影响微控制器进入中断的时间。



## 程序范例

```

;-----
;;Client:
;;Program Name:BZ to 32K
;;HT_ICE Version:
;;MCU Body:ht48r10-1
;;Power Supply & VDD:
;;MCU Frequency:8MHz
;;Finished Date:2005.8.9
;;Programmer:Liu Tao

;;Main Function:output 32768Hz crystal from BZ.
;;Mask Option:
;Wdt: Disable
;Clrwdt: One Clear Instruction
;Wake_Up Pa0-7:non Wake Up
;Pull-high PA:NON-PULL-HIGH
;Osc: Crystal
;Vdd :5.00V
;Fsys: 8MHz
;time clock source:system clock
;BZ/BZB:all disable
;LVR:disable
;-----
include ht48r10a-1.inc
;-----
;-----
DATA SECTION 'DATA'
C_TMRC EQU 80H ;TMRC's value
C_TMR EQU (256-28) ;TMR's value
C_INTC EQU 05H ;INTC's value
C_INIT_L EQU 0FFH ;INIT's low value
C_INIT_H EQU 0FFH ;INIT's high value

P_BZ EQU PB.0 ;BZ PORT
P_nBZ EQU PB.1 ;BZ PORT
P_BZC EQU PBC.0 ;BZ PORT's control
P_nBZC EQU PBC.1 ;BZ PORT's control
P_OSCI EQU PA.0 ;32k IN or OUT pin
P_OSCO EQU PA.1
P_OSCIC EQU PAC.0 ;32k IN or OUT control
P_OSCOC EQU PAC.1

R_INIT DB ? ;start osc's time
;-----

```

```

;;-----
CODE .SECTION at 00H 'CODE'
NOP
JMP MAIN
ORG 04H
RETI
ORG 08H ;TMR interrupt service
CLR TMRC.4 ;timer off
JMP ISR_TIMER
;;----- MAIN -----
MAIN:
CALL INI_TIAL ;initial I/O TMR INT & REM
CALL INI_32K ;initial 32K output
SET TMRC.4 ;timer on
JMP $
;;-----

;;----- Initial block -----
;;STACK: 1
;;ROM: 11
;;RAM: 0
;;WDT: DISABLE
;;TIMER: 0
;;INTERRUPT:0
;;PORT: PB.0,PB.1
;;MAXRUNTIME: 12 CYCLE(8MHZ)
;;INPUT/OUTPUT: NONE
;;-----
INI_TIAL:
CLR P_BZC
CLR P_nBZC
CLR P_BZ
CLR P_nBZ ;initial BZ port
MOV A,C_TMR
MOV TMR,A ;write timer's value
MOV A,C_TMRC
MOV TMRC,A ;set timer
MOV A,C_INTC
MOV INTC,A ;set int
CLR P_OSCO
CLR P_OSCOC
SET P_OSCIC ;initial OSC port
MOV A,C_INIT_L
MOV R_INIT,A ;initial start loop time
RET

```

```

;;----- Initial 32K output -----
;;STACK: 2
;;ROM: 10
;;RAM: 1
;;TIMER: 0
;;INTERRUPT:0
;;PORT: Pa.0,Pa.1,Pb.0
;;MAXRUNTIME:
;;-----
INI_32K:
    MOV    A,C_INIT_H
    SNZ    P_OSCI
    SET    P_OSCO           ;OSCI input low then OSCO output high
INI_LOOP:
    CALL   SBR_OSCI_L       ;wait for OSCI input low and then set
                                ;OSCO high
    CALL   SBR_OSCI_H       ;wait for OSCI input high and then set
                                ;OSCO low
                                ;32k initial over?
    SDZ    R_INIT
    JMP    INI_LOOP
    SDZ    ACC
    JMP    INI_LOOP
    RET
;;----- Wait for OSCI input low ----
;;STACK: 1
;;ROM: 5
;;RAM: 1
;;TIMER: 0
;;INTERRUPT:0
;;PORT: Pa.0,Pa.1,Pb.0
;;MAXRUNTIME:
;;-----
SBR_OSCI_L:
    SZ     P_OSCI           ;wait OSCI low
    JMP    $-1
    SET    P_OSCO           ;set OSCO high
    SET    P_BZ             ;set BZ out high
    RET
;;----- Wait for OSCI input hige ----
;;STACK: 1
;;ROM: 5
;;RAM: 1
;;TIMER: 0
;;INTERRUPT:0
;;PORT: Pa.0,Pa.1,Pb.0
;;MAXRUNTIME:

```

```

;-----
SBR_OSCI_H:
    SNZ    P_OSCI          ;wait OSCI high
    JMP    $-1
    CLR    P_OSCO         ;set OSCO low
    CLR    P_BZ           ;set BZ out low
    RET
;----- Timer's interrupt service -----
;STACK: 2
;ROM: 7
;RAM: 0
;TIMER: 0
;INTERRUPT:0
;PORT:  Pa.0,Pa.1,Pb.0
;MAXRUNTIME:
;-----
ISR_TIMER:
    SNZ    P_OSCI          ;judge OSCI input low or high
    JMP    ISR_NEXT
    CALL   SBR_OSCI_L
    JMP    ISR_END        ;jump ISR end
ISR_NEXT:
    CALL   SBR_OSCI_H
ISR_END:
    SET    TMRC.4         ;timer on
    RETI

```