

Using the I/O ports in the HT48 MCU Series

D/N : HA0021E

Introduction

This application note uses the HT48R10A-1 device to explain how to use the I/O ports in the I/O series as well as mentioning any special points, which should be noted. Attention is paid to issues such as using the basic input and output function, the $\overline{\text{BZ}}$ /BZ outputs as well as explaining the bit control operation.

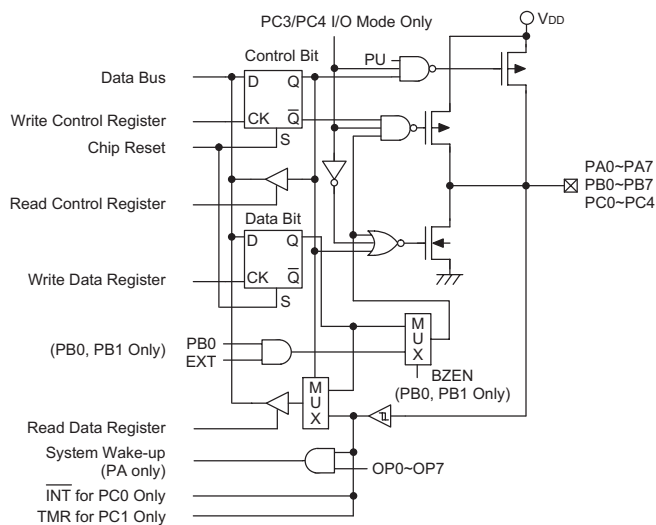
Description

The HT48R10A-1 is an 8-bit high function RISC microcontroller device. As a device in the I/O series of MCUs, and having 21 bi-directional I/O lines, it is particularly suitable for products requiring I/O control. Among its I/O ports, PA is an 8-bit port with each pin of the port being configurable as either an input or output, and each pin also being configurable to have a wake-up function. Port PB is an 8-bit Schmitt trigger I/O port. Port PC is a 5-bit I/O port. In this I/O series of devices, each port has a corresponding I/O data register, the data in which will influence the corresponding port. In addition each port also has a corresponding port control register, the function of which is to setup which pins on the port should be setup as an input or as an output. In addition to having an I/O function some pins also have other shared functions, these pins are listed in Table 1. Among these shared function pins, the OSC1 and OSC2 pins definitions have different schemes according to the device chosen.

Each I/O pin can be configured as CMOS output or as a Schmitt trigger input. If configured as an input, the pin can also be configured to have an internal pull-high resistor connected.

Pin	Shared Pin Function
PB.0	Buzzer BZ output pin
PB 1	Buzzer \overline{BZ} output pin
PC.0	External \overline{INT} input
PC.1	External TMR control pin
PC.3	Used as OSC1 or normal I/O if internal RC system clock is used
PC.4	Used as OSC2 or normal I/O if internal RC system clock is used

Table 1. I/O shared function pins



Note: EXT=BZ for PB0 only, EXT= \overline{BZ} for PB1 only, control=PB0 data register

Fig. 1 Input/Output pins

The PA~PC ports have corresponding data registers located at locations [12H], [14H] and [16H] in the data memory, all of which can be bit controlled. Their corresponding port control registers PAC~PCC are located at locations 13H, 15H and 17H in the data memory, and are used to setup the input or output status of the port pins. By writing a 1 or "0" to the corresponding bit, the pin can be configured as an input or output respectively. When the pin is setup as an input, the input port data is not latched. Therefore, for example, when executing the instruction "MOV A, [M]" (M=12H, 14H, 16H) the input data on the input pin must be ready before the T2 rising edge. When the pin is setup as an output, because the output data is stored in an output data latch, the output data is always maintained. After a reset the I/O lines will be in a high or floating input condition. It is recommended that any unused I/O pins are setup as outputs.

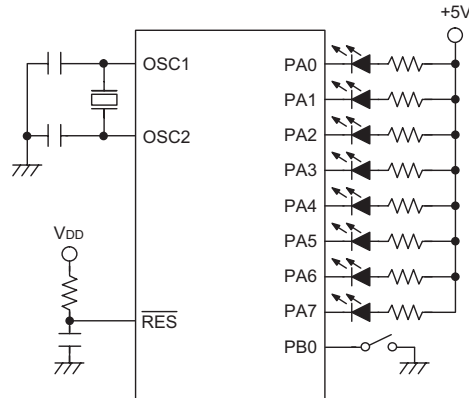
Program Example

Example 1

```

;Basic input output example (ledflash.asm) :
;Author: Huang Shan Yun
;Objective: Using the PAC output data latch
;Configuration Options: PA and PB with pull-high resistors

```



;Hardware description: Each PA output is connected to an LED cathode,
the LED anode is connected via a 470Ω resistor to 5V.

```

INCLUDE HT48R10A-1.INC
    data .section 'data'
        delay_count1 db ?
        delay_count2 db ?
        displaytemp db ?
    code .section 'code'
        org 00h
main:
    mov a,0feh          ;display initial value
    mov displaytemp,a
start:
    clr pac            ;PA setup as outputs
    set pbc.0         ;PB.0 setup as inputs
key_in:
    mov a,pb          ;Determine if the key has been pressed
    sz acc.0
    jmp key_in
    rl displaytemp    ;rotate left one bit
    mov a,displaytemp
    mov pa,a
    call delay        ;delay
    jmp start

```

```

delay proc
    mov     a,0ffh
    mov     delay_count1,a
    mov     delay_count2,a
    wait:
    sdz     delay_count2
    jmp     wait
    sdz     delay_count1
    jmp     wait
    ret
delay     endp
end

```

- Program Description

The LEDs will illuminate in turn with each switch press.

Example 2

```

;Program Name : Using BZ/ $\overline{\text{BZ}}$  (BZ1.ASM)
;Author: Huang Shan Yun
;Objective: Introduction to using PB.0 and PB.1 Buzzer Outputs BZ/ $\overline{\text{BZ}}$ 
;Configuration Options:select BZ/ $\overline{\text{BZ}}$  function
include ht48R10A-1.inc
code .section at 0h 'code'
org   00h
    jmp  start
;-----
data .section 'data'
    count1  db ?
    count2  db ?
;-----
main .section 'code'
org   20h
start:
    mov     a,00h           ;Setup PB.0 and PB.1 as outputs
    mov     pbC,a
    ;CLR    PB.0           ;This instruction is used to switch off the
                           ;buzzer
    mov     a,0DAh         ;Use the TIMER initial value to setup the
                           ;audio frequency

    mov     tmr,a
    mov     a,90h
    mov     tmrc,a
    set     pb.0
    jmp     $
end

```

- Program Description

After PB0 and PB1 are setup as BZ and $\overline{\text{BZ}}$ outputs, the output signal will be governed by the timer controlled PFD signal.

Note: The function of PB0/PB1 is governed by the following table:

PB0 I/O	I	I	I	I	O	O	O	O	O	O	O	O
PB1 I/O	I	O	O	O	I	I	I	O	O	O	O	O
PB0 Mode	X	X	X	X	C	B	B	C	B	B	B	B
PB1 Mode	X	C	B	B	X	X	X	C	C	C	B	B
PB0 Data	X	X	0	1	D	0	1	D ₀	0	1	0	1
PB1 Data	X	D	X	X	X	X	X	D ₁	D	D	X	X
PB0 Pad Status	I	I	I	I	D	0	B	D ₀	0	B	0	B
PB1 Pad Status	I	D	0	B	I	I	I	D ₁	D	D	0	B

Note: "I" indicates that it needs to be setup as an input, "O" indicates that it needs to be setup as an output, "D, D₀, D₁" indicate data, "B" indicates setup as buzzer, "X" indicates any condition, and "C" indicates CMOS output.

Example 3

```

;Program Name: Using the Read-Write-Modify command (rmw.asm)
;Author: Huang Shan Yun
;Objective: using the "Read-Write-Modify" instruction
Program statements:
INCLUDE HT48R10A-1.INC
    data .section 'data'
    code .section 'code'
    org 00h
    jmp start
start:
    clr pac.7          ;setup pa.7 as an output
    clr pa.7          ;Set output low
    clr pa.0 ;(1)     ;first execute a "read-write-modify"
                    ;instruction, pin pa.7 is not affected
    set pac.7         ;input state, pa.7 floating but because
                    ;of the pull-high resistor, pin pa.7
                    ;will go high. The output data latch will not
                    ;be affected and will remain at "0".
    clr pac.7         ;place the output latch data onto the output
                    ;pin pa.7, the value will return low as before

    set pac.7         ;now define again the pin as an input
    set pa.1 ;(2)     ;execute again a read-write-modify
                    ;a value of "1" will be read for pa.7 now the
                    ;output data latch already has a high value
    clr pac.7         ;now a "read-write-modify" instruction will
                    ;place the pa.7 output in a high condition

    jmp $

```

- Program Description

“Read-Write-Modify” Instruction:

When the MCU executes a “read-write-modify” instruction, it will first read the condition of all 8-pins, the data of the requested pin will then be modified, after which the data will be written to the output port. The instructions “set (pn).i”, “clr (pn).i”, “cpl” (pn) and “cpla” (pn) etc are all statements of the read-write-modify type. When an I/O has been setup as an output, the output data will be stored in an output data latch. If now a read output port instruction is executed, for example using the instruction, “mov a, pa”, if the pin is an output, the data will be read from the output data latch and not from the actual output pin. However, if the pin is an input, the actual data on the input pin will be read.

When a pin has been setup as an input, if a “read-write-modify” instruction is executed, then it is possible that the output data latch could be affected. Here, if the program now returns the pin to an output state, then the original logical state of the output may be changed. Therefore, it is recommended that in the program, if any pin is setup as an output, it is immediately followed by a statement which sets its actual data value. In addition, any unused pins should be setup as outputs. By writing a “0” to PAC, the pins can be prevented from floating to a high level as shown in the flowing statements:

```

...
CLR PAC.2           ;Unused pins setup as outputs
CLR PAC.3
...
CLR PAC.1   ;PA.1=O/P
CLR PA.1    ;PA.1 O/P LOW ;follows immediately
...

```

- Explanation

Follow the program comments to see what is happening, taking special note of the changing pin conditions. The program shows that if a certain pin is setup as an output, then if the other pins have a “read-write-modify” instruction executed, then the pin will not be affected. The program also indicates that if a pin has been setup as an input, then if a “read-write-modify” instruction is executed on other pins, it can influence the status of this pin if it is returned to an output condition.