



● 产品名称

MK7XXXX系列芯片

● 标题

如何使用MK7XXXX系列芯片的IO唤醒功能

● 简介

当前市场上许多电子产品是用电池来供电的，因此提高电池的使用寿命已经成为所有 IC 设计厂家共同面对的一项课题。对此 MKT 为自己的芯片提供了 SLEEP 模式功能。在 SLEEP 模式中，由于系统时钟停止振荡，所以芯片耗电非常低，芯片在进入了 SLEEP 模式之后，可以通过两种途径唤醒到正常工作模式，这两种途径分别为 IO 唤醒和 WDT 唤醒。下面我们就以 MK7A11P 的 IO 唤醒试验为例，看看 MKT 的芯片是如何通过 IO 来唤醒的。

本次试验电路的线路图见图 1，基本功能要求如下：

- 当有按键按下时，该按键对应的键码将通过 LED 显示出来（见表 1）
- 没有按键按下后 5S 重新进入 SLEEP
- 进入 SLEEP 前所有 LED 将熄灭

按键	键码	LED4	LED3	LED2	LED1
SW1	0001	灭	灭	灭	亮
SW2	0010	灭	灭	亮	灭
SW3	0011	灭	灭	亮	亮
SW4	0100	灭	亮	灭	灭
SW5	0101	灭	亮	灭	亮
SW6	0110	灭	亮	亮	灭
SW7	0111	灭	亮	亮	亮
SW8	1000	亮	灭	灭	灭

表 1 按键、键码及 LED 的关系

Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
WAKEUP	EN7	EN6	EN5	EN4	EN3	EN2	EN1	EN0

表 2 PortB 端口唤醒使能寄存器

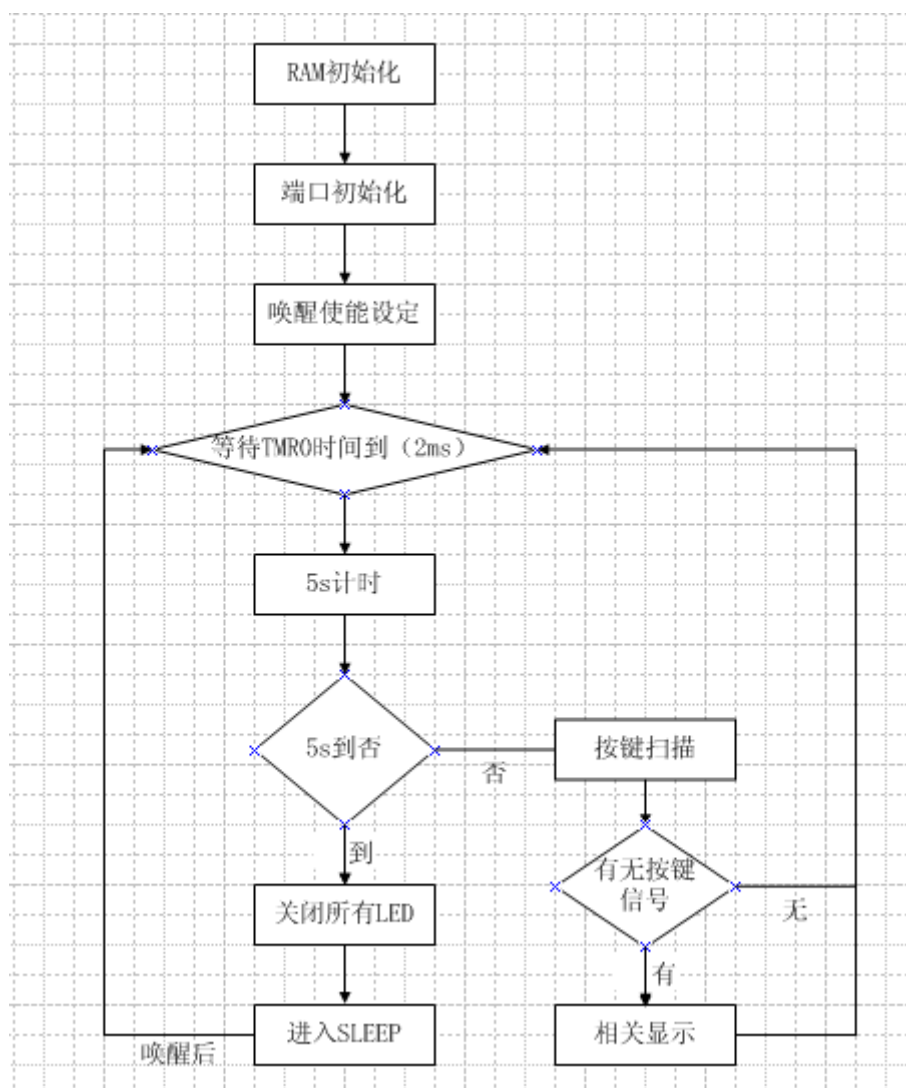


Register	Bit 7	Bit 6	Bit 5	Bit 4	Bit 3	Bit 2	Bit 1	Bit 0
PB_PUP	UB7	UB6	UB5	UB4	--	UB2	UB1	UB0

表 3 PortB 端口上拉使能寄存器

MK7A11P 芯片 PortB 端口的唤醒和上拉使能寄存器分别见表 2 和表 3。
由于 PB3 无内部上拉电阻，所以要外加一个 20K 欧姆的上拉电阻。

● 程序流程图



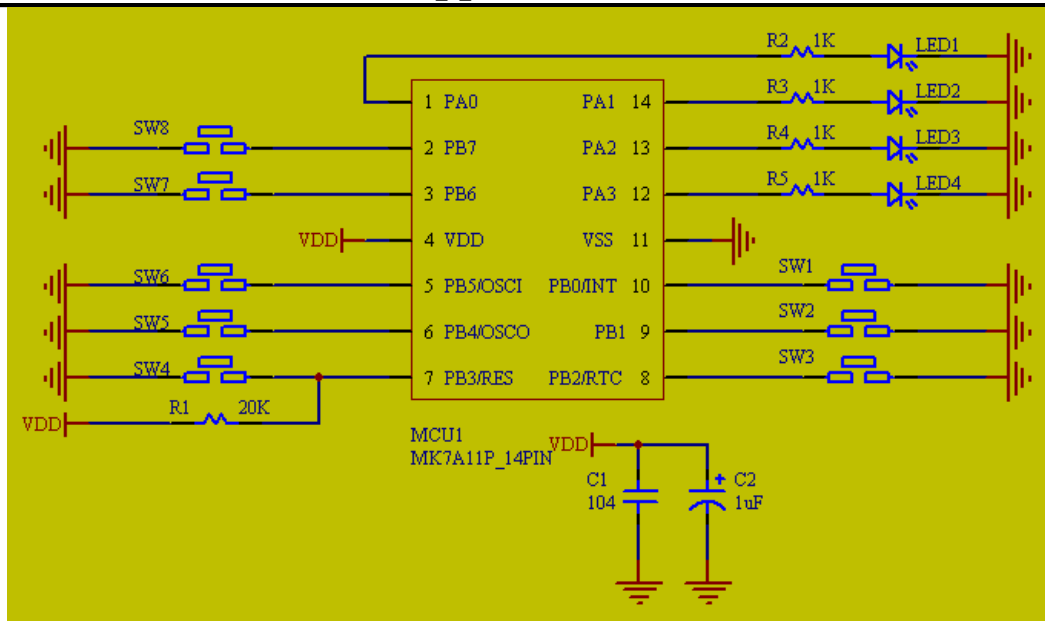


图 1 MK7A11P 的 IO 唤醒试验线路图

● DEMO 程序

➤ 汇编程序文档

```
-----  
#include "mk7a11p_hw.inc" ;编译该文档需包含"mk7a11p_hw.inc"文件  
-----  
;芯片型号 (mk7a11p)  
-----  
;配置寄存器设置说明 (CONFIG)  
;1-----FOSC=RC ;LS,NS,HS,RC  
;2-----INRC=ON ;ON,OFF  
;3-----CPT=OFF ;ON,OFF  
;4-----WDTE=Disable ;Enable,Disable  
;5-----LV=Low Vol Reset ON ;Low Vol Reset ON,Low Vol Reset OFF  
;6-----RESET=...input... ;...input...,...reset...  
-----  
signal_new equ 0x20 ;存储当前的按键信号  
signal_old equ 0x21 ;存储上次的按键信号  
key_r equ 0x22 ;检测信号的次数  
flag equ 0x23 ;标志  
tm_r0 equ 0x24 ;5s 计时
```



```
tm_r1      equ      0x25      ;5s 计时
;-----
#define     flag_key    flag,0    ;按键未松开标志
#define     flag_t      flag,1    ;重新计时标志
;-----
          org      0x3ff      ;mk7a11p 的复位向量地址定义
          lgoto     main      ;跳转到主程序入口
;-----
          org      0x100      ;主程序入口地址定义
main
          movla     0x20
          movam     fsr      ;将 0x20 送入 fsr 寄存器
;-----
clear_ram
;利用 indf 和 fsr 来进行间接寻址
;对 0x20-0x2f 的 RAM 进行 clear
clr        indf
mov        fsr,a
andla     b'00111111';将无关的数据虑除
xorla     0x2f
btsc      status,z
lgoto     $+3
inc        fsr,m
lgoto     clear_ram
clr        fsr      ;使用 fsr 时要注意 bank 的归位
;-----
;PortA 端口方向及状态设定
movla     b'11110000'
movam     pa_pdm
iodir     porta
clr        porta
;-----
;PortB 端口方向及状态设定
movla     b'11111111'
movam     pb_pup      ;上拉使能
iodir     portb
clr        portb
clr        pb_pod
clr        pb_pdm
;-----
```



```
;PortB 端口 8 个 IO 唤醒使能
movla      b'11111111'
movam      wake_up
;-----
;配置 TMR0,预分频比为 1:8
;TMR0 初始值为 250
movla      b'11000010'
select
movla      .250
movam      tmr0
;-----
loop                                ;程序循环入口
;程序循环周期控制处 (T=2ms)
clrwdt
movla      .1
xor         tmr0,a
btss       status,z
lgoto      $-4
;-----
time_count
btss       flag_t
lgoto      $+4                    ;转 5s 计时
bc         flag_t
clr        tm_r0
clr        tm_r1
;-----
;5s 计时
inc        tm_r0,m
movla      .100                    ;100x2=200ms
xor        tm_r0,a
btss       status,z
lgoto      scan_key
clr        tm_r0
;-----
inc        tm_r1,m
movla      .25                    ;200x25=5s
xor        tm_r1,a
btss       status,z
lgoto      scan_key
clr        tm_r1
```



into_sleep

```
;-----  
clr          key_r  
clr          flag  
;-----  
;进入睡眠之前关掉所有 LED  
clr          porta  
;-----  
;下面这几项由于已经设置，所以不用再设  
;movla      b'11111111'  
;movam      pb_pup      ;PortB 上拉使能  
;iodir      portb      ;PortB 输入  
;movam      wake_up     ;PortB 唤醒使能  
;nop  
;nop  
;nop  
;-----  
;为避免耗电流过大  
;进入睡眠之前送 ff 给 tmr0  
;并且 PB3/RESETB 作输入时不能悬空  
movla      0xff  
movam      tmr0  
;-----  
;进入睡眠之前保存 IO 的状态  
mov        portb,a  
nop  
nop  
nop  
sleep  
nop  
nop  
nop  
lgoto      loop
```

```
;-----  
scan_key          ;          按键扫描处  
;获取 PortB 端口信号,并与上次  
;信号比较,相同有效,否则无效  
;同一行连续检测到有效信号达  
;20 次,被认为信号稳定转信号  
;处理， 并进行行切换
```



```
com      portb,a
movam    signal_new
xor      signal_old,a
btss     status,z
clr      key_r          ;信号无效清除 key_r
;-----
;保存当前检测的信号
mov      signal_new,a
movam    signal_old
;-----
;连续有效信号次数累计
inc      key_r,m
movla    .10
xor      key_r,a
btss     status,z
lgoto    loop          ;未到次数转程序循环入口
;-----
;按键信号判断
movla    0x01
xor      signal_new,a
btsc     status,z
lgoto    check_key     ;判断按键是否处理过
movla    0x02
xor      signal_new,a
btsc     status,z
lgoto    check_key     ;判断按键是否处理过
movla    0x04
xor      signal_new,a
btsc     status,z
lgoto    check_key     ;判断按键是否处理过
movla    0x08
xor      signal_new,a
btsc     status,z
lgoto    check_key     ;判断按键是否处理过
movla    0x10
xor      signal_new,a
btsc     status,z
lgoto    check_key     ;判断按键是否处理过
movla    0x20
xor      signal_new,a
```



```
btsc      status,z
lgoto     check_key      ;判断按键是否处理过
movla     0x40
xor       signal_new,a
btsc      status,z
lgoto     check_key      ;判断按键是否处理过
movla     0x80
xor       signal_new,a
btsc      status,z
lgoto     check_key      ;判断按键是否处理过
;-----
;无按键信号出现
bc        flag_key      ;清按键处理过标志
lgoto     loop           ;无按键信号转程序循环入口
;-----
check_key
;判断按键是否处理过
btsc      flag_key
lgoto     loop           ;按键处理过不再进处理
;-----
;按键信号处理
bs        flag_key      ;置按键处理过标志
bs        flag_t        ;置重新计时标志
;-----
movla     .15
btsc      signal_new,0
movla     .1
btsc      signal_new,1
movla     .2
btsc      signal_new,2
movla     .3
btsc      signal_new,3
movla     .4
btsc      signal_new,4
movla     .5
btsc      signal_new,5
movla     .6
btsc      signal_new,6
movla     .7
btsc      signal_new,7
```



```
movla    .8
movam    porta
lgoto    loop
```

```
-----
end
```

➤ mk7a11p_hw.inc 文档

```
-----Define special register(Define SFR)-----
```

```
indf      equ      0x00
tmr0      equ      0x01
pc        equ      0x02
status    equ      0x03
fsr       equ      0x04
porta     equ      0x05      ;porta(0-3)
portb     equ      0x06      ;portb(0-7)
```

```
-----
```

```
irqm      equ      0x09
irqf      equ      0x0a
```

```
-----
```

```
pa_pdm    equ      0x0b
pb_pup    equ      0x0c
pb_pdm    equ      0x0d
pb_pod    equ      0x0e
wake_up   equ      0x0f
```

```
-----Define [status Register] special bit-----
```

```
c         equ      0
dc        equ      1
z         equ      2
pd        equ      3
to        equ      4
```

```
-----Define [irqm Register] special bit-----
```

```
tm0m      equ      0
extm      equ      1
intm      equ      7
```

```
-----Define [irqf Register] special bit-----
```

```
tm0f      equ      0
extf      equ      1
```

```
-----Define [pa_pdm Register] special bit-----
```

```
da0       equ      0
da1       equ      1
da2       equ      2
```



da3	equ	3
;-----Define [pb_pup Register] special bit-----		
ub0	equ	0
ub1	equ	1
ub2	equ	2
ub4	equ	4
ub5	equ	5
ub6	equ	6
ub7	equ	7
;-----Define [pb_pdm Register] special bit-----		
db0	equ	0
db1	equ	1
db2	equ	2
inte	equ	6
rtce	equ	7
;-----Define [pb_pod Register] special bit-----		
ob0	equ	0
ob1	equ	1
ob2	equ	2
ob4	equ	4
ob5	equ	5
ob6	equ	6
ob7	equ	7
;-----Define [wake_up Register] special bit-----		
en0	equ	0
en1	equ	1
en2	equ	2
en3	equ	3
en4	equ	4
en5	equ	5
en6	equ	6
en7	equ	7
;-----		
