SinoMCU

MC32P8141

内部高频校准内部低频说明

广东晟矽微电子有限公司

Guangdong SinoMCU Microelectronics Co., Ltd.

目录

一、简介

1. 应用环境：

RTC 定时，对于成本、空间要求，不能外接 32768HZ 晶振；使用内部低频误差较大；

2. 软件实现：

采用内部高频在上电时校准内部低频计数器；定时器 0 时钟配置为内部低频，启动定时器 0 计数，延时 1S，读取定时器 0 的计数器，关闭定时器 0 计数，得到 1S 定时器 0 计数值；按照此计数值重新装载定时器 0，利用定时器分频、装载值移位，实现不同定时；

二、仿真环境配置

可以根据实际应用调整，主频改变时，需要修改 1S 延时时间参数；

WDTM	WDT始终关闭
RSTEN	P04为输入/输出脚
LVRSLP	**LVR低功耗模式下关闭**
VLVRS	**100：2.4V**
WDTT	1024ms
P16DRS	P16下拉电阻复位后有效
P17URS	P17下拉电阻复位后有效
SMTVS	**0.8VDD/0.2VDD**
FCPUS	**FCPU=FHOSC/32**

三、软件介绍

1. 清 RAM、IO 初始化：

上电时，RAM 处于随机不确定状态，需要将 RAM 全部清 0，非必须操作，客户在一些应用场合可以不用清 RAM，根据实际需求来选择；

IO 全部设置为输出低；

```
void CLR_RAM(void)
{
__asm
    movai   0x7F
    movra   FSR0
    clrr    INDF0
    DJZR    FSR0
    goto    $ -2
    clrr    INDF0
__endasm;
}
void IO_Init(void)
{
    IOP0 = 0x00; //io口数据位
    OEP0 = 0xff; //io口方向 1:out  0:in
    PUP0 = 0x00; //io口上拉电阻   1:enable  0:disable
    PDP0 = 0x00; //io口下拉电阻   1:enable  0:disable

    IOP1 = 0x00; //io口数据位
    OEP1 = 0xff; //io口方向 1:out  0:in
    PUP1 = 0x00; //io口上拉电阻   1:enable  0:disable
    PDP1 = 0x00; //io口下拉电阻   1:enable  0:disable

    P0ADCR = 0x00; //端口数模控制   1:模拟  0:数字
    P1ADCR = 0x00; //端口数模控制   1:模拟  0:数字
}
void Sys_Init(void)
{
    GIE = 0;
    CLR_RAM();
    IO_Init();
    GIE = 1;
}
```

2. 延时 1S 函数：

延时 1S，利用三个变量累加延时，主频修改后，应调整 DELAY_NUM_H、DELAY_NUM_M、DELAY_NUM_L 三个宏的参数；延时函数可自行编写；

```
/*************************************************
;  *    @Function Name        : Delay_1S
;  *    @Description          : 延时1s
;  *    @IN_Parameter         :
;  *    @Return parameter     :
;  *************************************************/
void Delay_1S(void)
{
    uint8_t cnth;
    uint8_t cntm;
    uint8_t cntl;

    while (1)
    {
        cntl++;
        if (cntl == 0)
        {
            cntm++;
            if (cntm == 0)
            {
                cnth++;
            }
        }
        if ((cnth >= DELAY_NUM_H) && (cntm >= DELAY_NUM_M)
            && (cntl >= DELAY_NUM_L))
        {
            break;
        }
    }
}
```

3. 低频校准函数：

配置定时器 0 时钟为 LIRC（内部低频），使能定时器 0 计数，延时 1S，读取定时器 0 计数器，关闭定时器 0 计数，得到 1S 定时器 0 计数值；按照此计数值重新装载定时器 0，利用定时器分频、装载值移位，实现不同定时；

```
126  void LIRC_ADJ(void)
127  {
128      //T0
129      T0IF = 0;
130      T0EN = 0;
131
132      T0CR = 0x10; //FLOSC 不分频
133      T0LOADH = 0xff;
134      T0LOADL = 0xff;
135      T0IE=0; //关闭定时器0中断
136      T0EN = 1;       //使能定时器0
137
138      P17D = 1;       //延时时间测试
139      Delay_1S();
140      P17D = 0;       //延时时间测试
141
142      u8_templ = 0xff - T0CNTL;
143      u8_temph = 0xff - T0CNTH;
144      u8_templ = 0xff - T0CNTL;
145      u8_temph = 0xff - T0CNTH;    //为了稳定，读取两次
146      T0EN = 0;       //关闭定时器
147      T0IF = 0;
148      //如果需要定时2S  选择2分频
149      //如果需要定时0.5S  LOAD值右移，移位后装载值不能为0
150      T0CR = 0x10; //FLOSC 不分频 1S
151      T0LOADH = u8_temph;
152      T0LOADL = u8_templ;
153      //T0LOADH = u8_temph>>1;
154      //T0LOADL = u8_templ>>1;
155      T0EN = 1;   //使能定时器0
156      T0IE = 1;   //使能定时器中断
157  }
```

4.main、中断服务函数：

main：上电调用初始化函数，对 RAM、IO 初始化，使能内部低频工作，等待低频稳定，校准低频；

中断服务函数：现场保护，定时器 0 中断后，P17 翻转；

```
158  void main(void)
159  {
160      Sys_Init();
161      LFEN = 1; //低频时钟工作
162      while (!STBL); //等待低频稳定
163      LIRC_ADJ();     //低频校准
164      while (1)
165      {
166      }
167  }
168
169  void int_isr(void) __interrupt
170  {
171  __asm
172      movra   _abuf
173      swapar  _PFLAG
174      movra   _statusbuf
175  __endasm;
176      if ((T0IE) && (T0IF))       //T0中断
177      {
178          T0IF = 0;
179          //P17翻转
180      __asm
181          MOVAI 10000000B
182          XORRA IOP1
183      __endasm;
184      }
185  __asm
186      swapar  _statusbuf
187      movra   _PFLAG
188      swapr   _abuf
189      swapar  _abuf
190  __endasm;
191  }
```

版本修订记录

版本号	修订日期	修订内容
V1.0	2020-05-26	新建