FT67/64/61F0Ax Application Note

[rev0.1]

1. 文档更改历史

版本	日期	内容
0.1	2021-1-8	初版

2. 详细说明

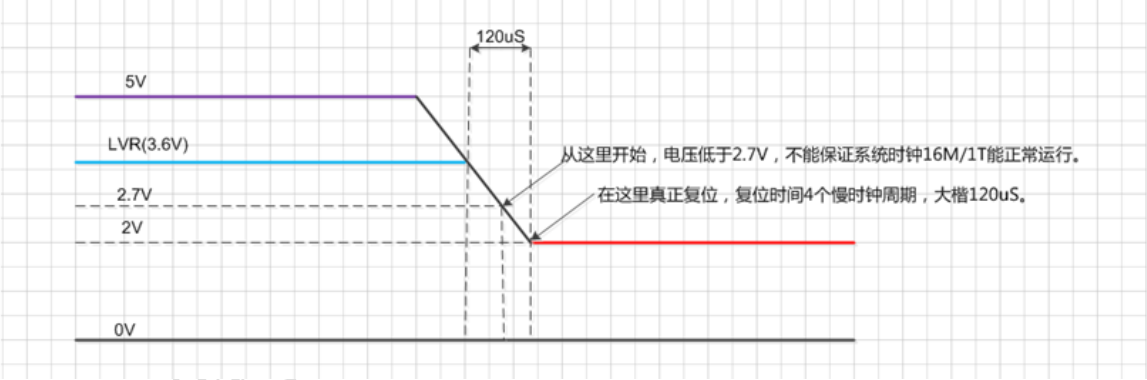
2.1 低电压检测的两种方式

2.1.1. 使用 LVR

使用 LVR 时，有以下 3 种限制需要考虑。

2.1.1.1 LVR 事件到开始复位有 3 ~ 4 慢时钟延时

低电压复位（LVR）反应的时间是 3 ~ 4 个慢时钟周期（120us），当 VDD 掉电速度比较快时，在 LVR 真正起作用之前 VDD 可能已经低于最小工作电压（如下图中的 2.7V），这时 MCU 将可能跑错：



2.1.1.2 执行 CLRWDT 指令时，发生 LVR 复位，系统无法工作

原因分析：
在执行 CLRWDT 指令时，发生 LVR 复位，由于 CLRWDT 指令一直将 WDT 复位，此时需要 WDT 进行上电 4ms 计时， WDT 无法进行计数，系统一直处于等待 WDT 计时状态。

解决方法：
使用写 WDTCON 或者写 MISC0 寄存器方式替代 CLRWDT 指令来清零看门狗。

2.1.1.3 执行 SLEEP 指令时，发生 LVR 复位，系统无法工作

原因分析：
在执行 SLEEP 指令时，发生 LVR 复位，由于 SLEEP 指令一直将 WDT 复位，此时需要 WDT

进行上电 4ms 计时， WDT 无法进行计数，系统一直处于等待 WDT 计时状态。

解决方法：
使用软件方式使能 LVR 复位，在进入 SLEEP 时，关闭 LVR 复位功能，退出 SLEEP 后，再打开 LVR 复位功能。具体示例程序如下所示：

```
MOVLB 03H
BCR LVDCON0,7 ;禁止 LVR 功能
NOP
NOP
SLEEP
….
```

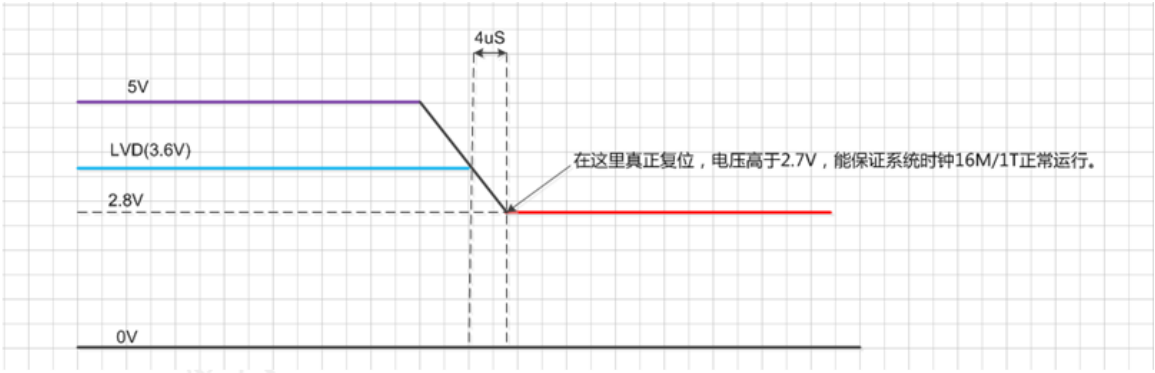
如果在睡眠下，还需要用到电压检测功能，可以使用 LVD（及软复位方式）代替 LVR 功能。

2.1.2. 使用 LVD

- 若电路中电源掉电速度较快时，而 LVR 检测过慢，为了防止供电异常时应用出错，我们建议用 LVD 软件复位，LVD 软件复位程序如下:

```
void LvdInit()
{
    LVDCON0 = 0x96;   //LVD 设置为 3.6V
    LVDCON1 = 0x00;   //不使用 LVD 去抖
    LVDIE = 1;
    LVDIF = 1;        //写 1 清零
}
void interrupt ISR(void)        //中断中添加 LVD 复位程序并放在第一顺序位置
{
    if(LVDIF == 1)
    {
        LVDIF = 1;
        #asm
            RESET               //软件复位
        #endasm
    }
}
```

LVD 应用示意图如下：



综上所述，建议低电压检测使用 LVD 模块；如果非得要使用 LVR，则建议把 LVR 阈值尽可能设高，且注意 CLRWDT 及 SLEEP 指令的使用。

2.2 IAP 程序需要注意以下三点

- IAP 程序和 APP, IAP 程序跳 APP 程序前面没有加“PCLATH=XX ”的赋值语句，在跳转过程中会变程序翻页处理，需设置“PCLATH=XX ”。

```
//例程
    PCLATH = 0x06;          //设置 PC 指针高位
    asm("goto 0x604");      //应用程序
```

- IAP 程序编译后程序不是按连续的空间存放，有可能会放到超出的范围。需增加伪指令将其固定到固定位置。

```
#asm
    ORG  0Xxx   //固定地址
    NOP
#endasm
    //需要固定位置的程序
```

- 进行 FLASH/EEDATA 操作时，解锁 FLASH/EEDATA 的时序被打断。程序中要将此段用汇编指令处理防止被优化。

```
void Unlock_Flash()      // Flash 解锁时序不能被打断
{   #asm
    MOVLW   0X03
    MOVWF   _BSR
    MOVLW   0X55
    MOVWF   _EECON2
```

```
    MOVLW   0XAA
    MOVWF   _EECON2
    BSF     _EECON1,1           //WR = 1;
    NOP
    NOP
#endasm
}
void Flashwrite(uchar FlashAddr,uchar Data)
{
    GIE = 0;                    //写数据必须关闭中断
    NOP;NOP;
    while(GIE){GIE=0;};         //等待 GIE 为 0
    EEADRL = FlashAddr;         //Flash 的地址
    EEDATL = Data;              //要写 Flash 的数据

    CFGS = 0;
    EEPGD = 1;
    FERAE  = 0;
    WREN = 1;                    //写使能

    Unlock_Flash()               //Flash 解锁时序不能修改
    GIE = 1;
    while(WR);                   //等待 Flash 写入完成
    WREN = 0;
}
```