# mini2440裸机试炼之——看门狗中断和复位操作

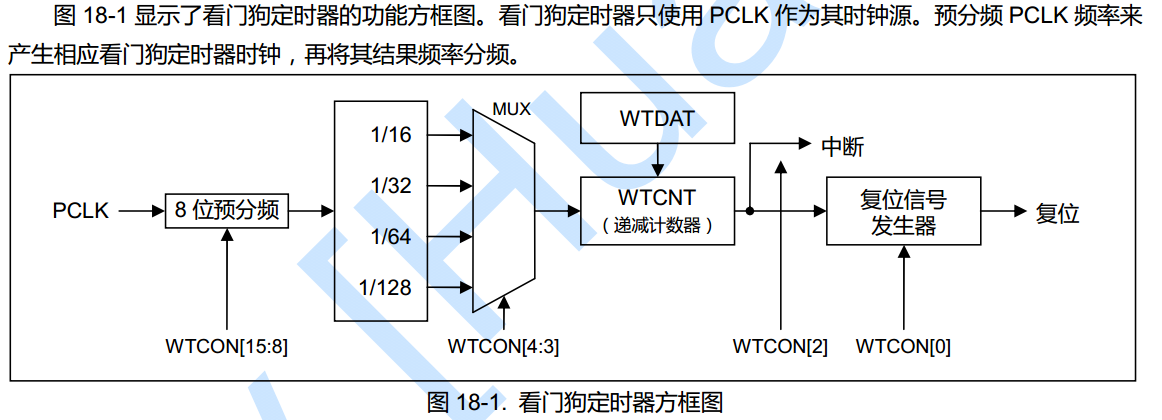
## 看门狗的工作原理：

设本系统程序完整运行一周期的时间是Tp，看门狗的定时周期为Ti，Ti>Tp，在程序正常运行时，定时器就不会溢出，若由于干扰等原因使系统不能在Tp时刻修改定时器的记数值，定时器将在Ti时刻溢出，引发系统复位，使系统得以重新运行，从而起到监控的作用。

## 看门狗具有两个功能：

1. 当做常规时钟，可以产生中断；
2. 当看门狗定时器使用，当计数器WTCNT为0时，产生复位；

## 看门狗的功能方框图:



我程序中PCLK为50MHz，预分频值=77，时钟除数因子选128，

时钟周期t\_watchdog=1/[PCLK/(预分频值+1)/时钟除数因子]

=1/[50\*1000000/(77+1)/128]=0.0002

t\_watchdog表示的是看门狗计数计时器WTCNT每减少1所用的时间，WTCNT又相当于一个节拍的作用，当WTCNT为0时，如果看门狗控制寄存器WTCON[0] 开启复位功能，则复位；如果看门狗控制寄存器WTCON[0] 禁止复位，开启中断，则中断操作，并数据（WTDAT）寄存器重新赋值到计数（WTCNT）寄存器内，循环中断操作（这里的中断就类似RTC的闹钟中断）。

所谓的喂狗：就是使WTCNT在为0之前重新赋值，不使之发生中断或者复位。

引用赵老师的一段话：

s3c2440的看门狗定时器不仅可以引起系统复位，还可以引起一般的中断，因此s3c2440的看门狗定时器可以当作一般的定时器使用。

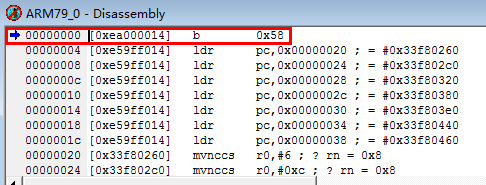
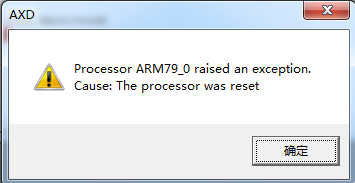
## 看门狗中断结果：

每四秒一个中断，LED点亮并串口提示

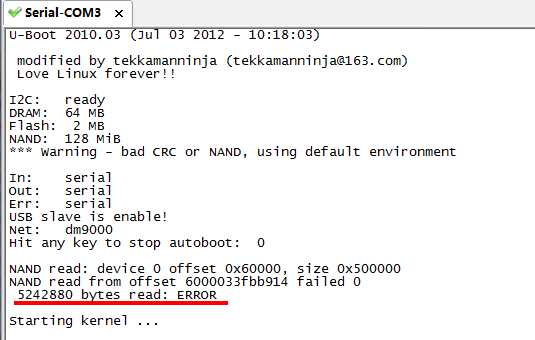


## 看门狗复位结果:

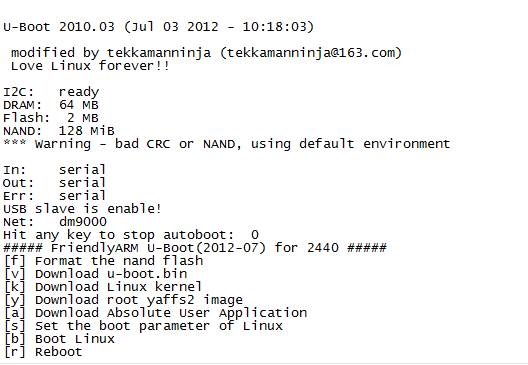
AXD调试运行到看门狗计数器rWTCNT为零时,进入复位,CPU停止工作,也就不产生中断。



复位关闭axd后,u-boot串口初始化提示(没能及时在第一次蜂鸣器响按下按键停止autoboot):



复位关闭axd后,u-boot串口初始化提示(及时在第一次蜂鸣器响按下按键停止autoboot):



这里就是复位到用uboot初始化开发板的时候,但是不明白为什么必须要在第一次蜂鸣器就按下按键才能进到uboot

## 附：

**硬件环境：J-link v8、mini2440、J-link转接板、串口转USB线**

**软件环境：windows7（32位）、开发板uboot（NandFlash）、J-link驱动（J-Link ARM V4.10i）、SecureCRT、ADS1.2**

**代码块**

|  |
| --- |
| //====================================================================  // 实现功能：  // 看门狗中断和复位操作  // 当WTCON[0] 开启复位功能; 实现复位  // 当WTCON[0] 禁止复位，WTCON[2]开启中断; 实现中断  // by：梁惠涌  //====================================================================  #include "2440addr.h"  #include "def.h"  #include "2440lib.h"    static U8 led\_flag**=**0**;**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  看门狗初始化函数  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  void watchdog\_init**(){**  rWTCON**=((**77**<<**8**)|(**1**<<**5**)|(**3**<<**3**)|(**1**<<**2**));** //预分频值=77;看门狗定时器使能;时钟除数因子=128 ;看门狗中断使能;  //时钟周期t\_watchdog=1/[PCLK/(预分频值+1)/时钟除数因子]=1/[50\*1000000/(77+1)/128]=0.0002  //t\_watchdog表示的是看门狗计数计时器WTCNT每减少1所用的时间  rWTDAT**=**20000**;** //WTDAT看门狗数据寄存器,看门狗定时器重载的计数值  rWTCNT**=**20000**;** //WTCNT看门狗计数寄存器,看门狗定时器当前计数值  **}**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  看门狗中断函数  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  void \_\_irq watchdog\_isr**(){**  rSUBSRCPND **|=**1**<<**13**;** //清除次级中断挂起状态  rSRCPND**|=**0x1**<<**9**;** //清除中断挂起状态  rINTPND**|=**0x1**<<**9**;**    //中断LED点亮  led\_flag**++;**  Uart\_Printf**(**"\n -----看门狗中断，点亮LED %d\n"**,**led\_flag**);**  rGPBCON**=**0x015400**;**  **switch(**led\_flag**){**  **case** 1**:** rGPBDAT**=**0xe**<<**5 **;break;**  **case** 2**:** rGPBDAT**=**0xd**<<**5 **;break;**  **case** 3**:** rGPBDAT**=**0xb**<<**5 **;break;**  **case** 4**:** rGPBDAT**=**0x7**<<**5 **;break;**  **}**  **if(**led\_flag**==**4**)** led\_flag**=**0**;**    rSUBSRCPND **&=(~**0x1**<<**13**);**  rSRCPND **&=(~**0x1**<<**9**);** //开中断  rINTPND **&=(~**0x1**<<**9**);**  **}**  /\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*  看门狗子函数  \*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*\*/  void watchdog**(){**    watchdog\_init**();** //初始化看门狗寄存器    //选择是否开启复位  //rWTCON |= (1<<0);//开启复位  rWTCON **|=** **(**0**<<**0**);**//禁止复位    rINTSUBMSK **&=~(**1**<<**13**);**//开看门狗次级中断  rINTMSK **&=~(**1**<<**9**);** //开看门狗中断  pISR\_WDT\_AC97**=(**unsigned**)**watchdog\_isr**;**    **while(**1**){**  **}**  **}** |