**优微电子科技**

**CC2530 基础实验**

**系统时钟配置**

# 实验介绍

C2530设备有一个内部系统时（或称作主时钟）。该系统时钟的源既可以用16MHzRC振荡器（内部RC），也可以采用32MHz晶体振荡器（外部晶振）。时钟的控制可以通过配置相关的寄存器以达到对时钟的选择控制。

# 实验目的

1. 通过本实验了解CC2530 的时钟寄存器配置；
2. 通过本实验了解CC2530 的时钟功能的运用。

# **三、**实验设备

## 3.1、硬件设备

（1）ZIGBEE开发板

（2）SmartRF04EB仿真器

## 3.2、软件环境

（1）IAR Embedded Workbench for 8051集成开发环境。

# **四、**实验原理

CC2530在正常运行的时候需要一个高频时钟信号和一个低频时钟信号。

高频时钟信号有两个来源，分别是芯片本身有的一个16MHzRC振荡电路和外接的32MHz石英晶振，主要供给CPU，保证程序正常运行。

低频时钟信号也有两个来源，分别是芯片本身有的一个32kHzRC振荡电路和外接的32.768MHz石英晶振，主要供给看门狗、睡眠定时器等片上外设。

默认上电时，有内部的两个RC振荡电路提供高频和低频时钟信号。当使用串口功能时尤其是无线通信，始终必须用32MHz石英晶振提供高频信号，这是因为32MHz石英晶振可以提供一个更为精准的时钟。

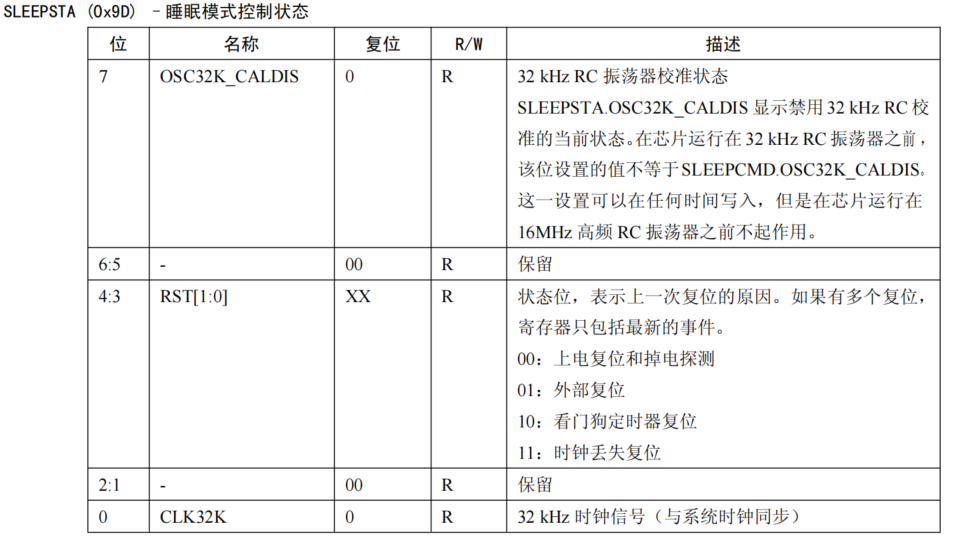
在CC2530中，两个高频时钟源可以同时起振产生高频时钟信号，而某一时刻只能有一个低频时钟源起振并且供给CC2530。

系统高频时钟源从16MHz切换到32MHHz的步骤：

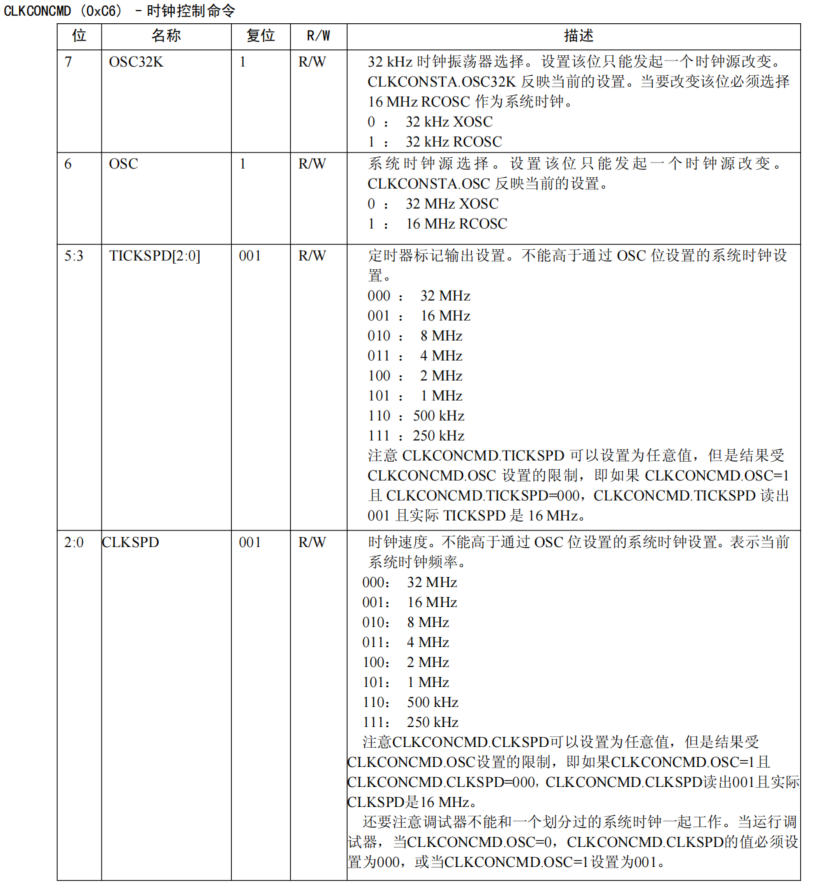
1. 通过配置SLEEPCMD寄存器，让两个高频时钟源起振。用于控制CPU，保证程序正常运行。当SLEEPCMD寄存器第2位为0时，表示两个时钟源都起振。



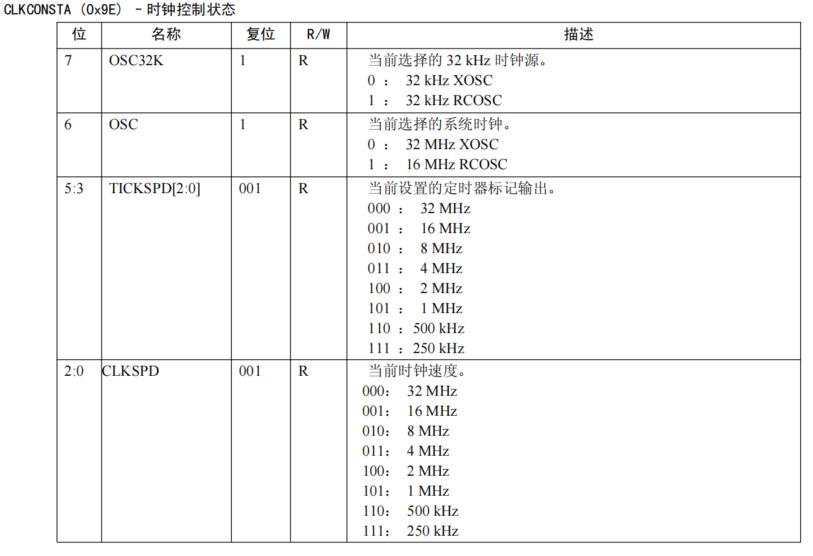
1. 等待时钟源振荡稳定。通过观察SLEEPSTA寄存器第6位是否为1，为1时表示32MHz石英晶振提供的时钟源稳定。



1. 进行延时大于63us。
2. 配置不分频输出。当CLKCONCMD寄存器后三位写入000时表示不分频输出，即32MHz输出；当写入001时表示二分频输出，即16MHz输出。



1. 选中目标高频时钟源作为系统主时钟。当CLKCONCMD寄存器第6位写入0时表示设置32MHz石英晶振提供的时钟源作为系统主时钟。
2. 确认当前工作的系统时钟是否为目标时钟。如果观察到CLKCONSTA寄存器的第6位为0时表示已设置32MHz石英晶振提供的时钟源作为系统主时钟。



# **五、**实验演示

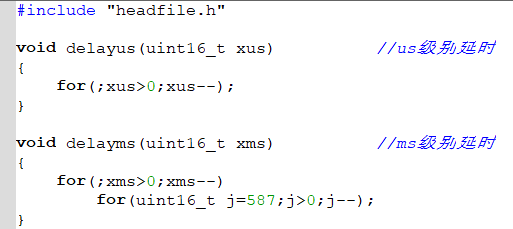
首先分别创建delay.c和delay.h文件用于存放我们的延时函数，创建完把我们前面写的延时函数也放到delay.c文件。再简单的写一个us级别的延时函数：

void delayus(uint16\_t xus) //us级别延时

{

for(;xus>0;xus--);

}



还有delay.h文件：

#ifndef \_\_DELAY\_H\_\_

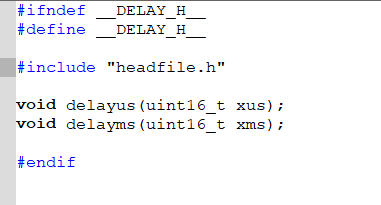
#define \_\_DELAY\_H\_\_

#include "headfile.h"

void delayus(uint16\_t xus);

void delayms(uint16\_t xms);

#endif



接下来是时钟转换。也是分别创建clk32m.c和clk32m.h文件。

整合起来：

void clk32m\_init(void)

{

SLEEPCMD &= 0xFB; //1111 1011 两个高频时钟源同时起振

while((SLEEPSTA & 0x40) == 0); //0100 0000 判断32M晶振是否起振稳定

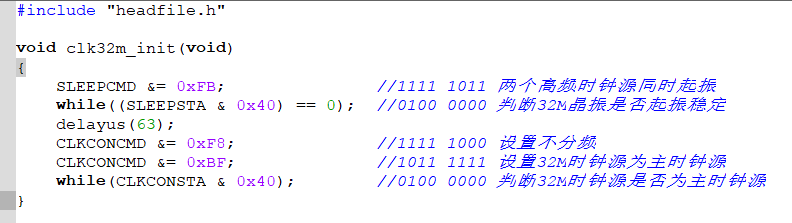
delayus(63);

CLKCONCMD &= 0xF8; //1111 1000 设置不分频

CLKCONCMD &= 0xBF; //1011 1111 设置32M时钟源为主时钟源

while(CLKCONSTA & 0x40); //0100 0000 判断32M时钟源是否为主时钟源

}



然后是补充clk32m.h文件：

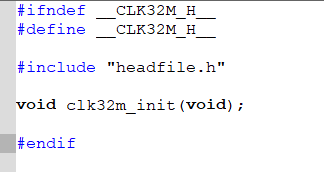
#ifndef \_\_CLK32M\_H\_\_

#define \_\_CLK32M\_H\_\_

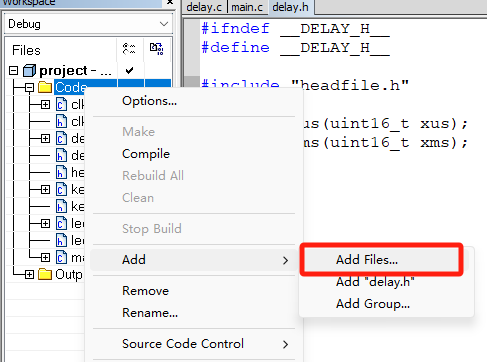
#include "headfile.h"

void clk32m\_init(void);

#endif



最后是把delay.c、delay.h、clk32m.c、clk32m.h文件给加入到工程的Code文件里就可以了。



现在我们来检验一下。

当不转换时，我们编写程序使LED在0.5s闪烁：

void main(void)

{

led\_init();

while(1)

{

led\_proc(1,1);

delayms(500);

led\_proc(1,0);

delayms(500);

}

}

观察现象后加入我们的时钟源转换函数：

void main(void)

{

led\_init();

clk32m\_init();

while(1)

{

led\_proc(1,1);

delayms(500);

led\_proc(1,0);

delayms(500);

}

}

可以明显观察到，LED的闪烁时间加快了一倍，这时候就说明时钟源已成功切换。