

32 位微控制器

HC32F120 系列的通用定时器 ICG

适用对象

系列	产品型号
HC32F120	HC32F120H8TA
	HC32F120F8TA
	HC32F120H6TA
	HC32F120F6TA

目 录

1	摘要	3
2	ICG 简介	3
2.1	主要特性	3
3	HC32F120 系列的 ICG	4
3.1	NMI 管脚中断	4
3.1.1	ICG 配置	4
3.1.2	注意事项	4
3.2	HRC 时钟频率	5
3.2.1	ICG 配置	5
3.2.2	CMU 配置	5
3.2.3	注意事项	5
3.3	SWDT 硬件启动	6
3.3.1	SWDT 特性	6
3.3.2	硬件启动	6
3.3.3	注意事项	7
3.4	LVD 低电压检测	8
3.4.1	ICG 配置	8
3.4.2	注意事项	8
3.5	寄存器介绍	9
4	样例代码	10
4.1	代码介绍	10
4.2	工作流程	13
4.3	代码运行	14
5	总结	15
6	版本信息 & 联系方式	16

1 摘要

本篇应用笔记主要介绍 HC32F120 系列初始化配置（ICG）模块，并简要说明 ICG 的硬件启动 SWDT 功能如何实现。

2 ICG 简介

芯片复位解除后，硬件电路会读取主闪存地址 0x000000C0H~0x000000DBH（其中 0x000000C8~0x000000DB 为预留功能地址，该 20bytes 地址的值需要用户设定全 0xFF，以保证芯片动作正常）把数据加载到初始化配置寄存器，用户需要编程或擦除扇区 0 来修改初始化配置寄存器。

寄存器复位值后初始值由用户 FLASH 地址数据确定。

2.1 主要特性

- NMI 管脚中断
- HRC 时钟频率
- SWDT 硬件启动
- LVD 低电压检测

3 HC32F120 系列的 ICG

3.1 NMI 管脚中断

NMI 管脚中断支持芯片启动配置（即 ICG 配置）和使用后再配置（即 INTC 配置），此处主要讲解 ICG 配置。

3.1.1 ICG 配置

通过配置 ICG 寄存器 ICG1.NMITRG 位，选择 NMI 触发边沿；设定 ICG1.NMIFCLK 位，选择数字滤波器采样时钟；设定 ICG1.NMIFEN 位，使能数字滤波器；设定 ICG1.NMIEN 位，使能 NMI 管脚中断。

NMI 管脚中断还需配置 INTC 中的 NMIER 寄存器，开启 NMI 管脚作为 NMI 的中断源；NMI 中断事件请求状态可以通过查询标志寄存器（NMIFR）来确定。

3.1.2 注意事项

- ICG 设定只针对外部 NMI 管脚。
- ICG 设定后，寄存器设定无效。
- 一旦 INTC 中的 NMIER 相应位被设定为“1”后，将不能被更改，除非用 RESET 来复位。
- 在 NMI 中断处理退出前确认所有的标志位都为“0”。

3.2 HRC 时钟频率

系统复位后，CPU 时钟源为 HRC（频率由 ICG 的设定值决定），HRC 时钟信号由内部高速振荡器生成，可直接用作系统时钟。

3.2.1 ICG 配置

HRC 的频率可由 ICG0.HRCFREQS 配置成 48MHz、32MHz、24MHz、16MHz、12MHz、8MHz、6MHz、4MHz、3MHz、2MHz、1.5MHz、1MHz。

3.2.2 CMU 配置

在复位解除后，CPU 一定以内部高速振荡时钟开始运行，可通过 CMU_HRCCR 控制寄存器中的 HRCSTP 位打开或关闭，配置 CMU_HRCCR 需开启功能保护控制，参见：hc32f120_clk.h 中的 CLK_REG_WRITE_ENABLE()。

可通过 CMU_HRCCFGR.HRCFREQS[2:0] 寄存器软件设定频率，配置 CMU_HRCCFGR 需解除写保护，参见：hc32f120_clk.h 中的 CLK_HRCCFGR_WRITE_ENBALE()。

3.2.3 注意事项

- HRC 选作系统时钟源时，禁止 CMU_HRCCR.HRCSTP 写“1”停止 HRC 时钟。
- CMU_HRCCFGR 的 HRCFREQS[3] 只能通过 ICG0.HRCFREQS[3] 设定。
- CMU_HRCCFGR 的 HRCFREQS[2:0] 可通过 ICG0.HRCFREQS[2:0] 设定，也可软件改写。

3.3 SWDT 硬件启动

SWDT 是一种计数时钟源为内部低速 RC（32.768KHz）的专用看门狗计数器，采用 16 位递减计数器，支持窗口功能，在计数开始前可预设窗口区间，计数值位于窗口区间时，可刷新计数器，计数重新开始。

看门狗计数器的启动方式有两种：硬件启动方式和软件启动方式，此处主要讲解硬件启动方式（ICG 配置方式）。

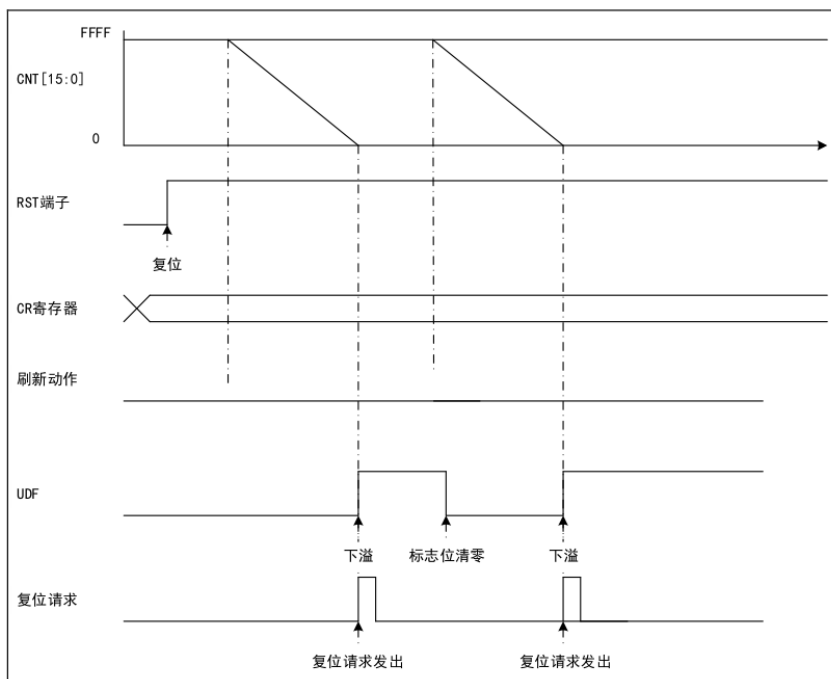
3.3.1 SWDT 特性

- 计数时钟：SWDTCLK 的 1/16/32/64/128/256/2048 分频
- 最长溢出时间：1.14hour
- 窗口功能：可设定窗口区间，定义刷新动作的允许区间
- 启动方式
 - 硬件启动
 - 软件启动
- 停止条件
 - 复位中
 - 下溢，刷新错误发生时
- 中断/复位条件
 - 计数下溢
 - 刷新错误

3.3.2 硬件启动

硬件启动方式时，在复位期间将 ICG0 寄存器里的 SWDT 相关设定（计数时钟、窗口设定值、计数周期等）载入到 SWDT 的模块中，复位之后、计数器按照设定自动开始计数；下溢或刷新错误发生时，复位或中断请求输出后自动开始重新计数。

ICG0 寄存器的位 0(SWDTAUTS)为 0 时，为硬件启动方式；选择硬件启动方式时，SWDT_CR 寄存器的相关设定信息无效，通过设定 ICG0 的 SWDTITS 位，来决定产生中断请求还是复位请求；下图为硬件启动方式的动作示例。



SWDT_RR 寄存器中先写 0x0123、再写 0x3210 完成一次刷新动作，SWDT 的计数器就重新开始计数；刷新错误标志位和计数下溢标志位在复位请求和中断请求的情况下都会保持。

3.3.3 注意事项

- 标志位清零：先读“1”再写“0”。
- 刷新错误或者计数下溢标志位为置位时，硬件启动模式看门狗计数不停止。

3.4 LVD 低电压检测

低电压检测单元(LVD)提供了上电复位(POR)、掉电复位(PDR)、电压检测 (LVD)等功能，其中 POR、PDR 通过检测 VDD 电压，控制芯片复位动作；LVD 通过检测 VCC 电压或者外部输入比较，根据寄存器设定设置使芯片产生复位或者中断。

LVD 支持芯片启动配置（即 ICG 配置）和使用后再配置（即 PWC 配置），此处主要讲解 ICG 配置。

3.4.1 ICG 配置

电压检测(LVD)阈值通过 ICG1 进行配置，LVD 检测电压可以从 14 个阈值中选择，通过配置 ICG1 的 LVDDIS 和 LVDLVL 位，可以选择 LVD 有效及阈值；电压检测电路通过 ICG1 的 IRDIS 和 IRS 位，选择产生复位或者中断功能。

电源电压经过各个检测电路的阈值电压点时，可将该事件编程配置成复位/中断(可屏蔽/不可屏蔽)。

3.4.2 注意事项

- 在停止模式中使用 LVD 电路时，必须将数字滤波器无效。
- LVD 中断的程序运行时，PWR_LVDCSR 中的 DETF 需要清零后才能再次发生中断。

3.5 寄存器介绍

初始化配置 ICG 模块的寄存器如下表所示，若需了解具体细节，请参考用户手册：

寄存器简称	寄存器功能
ICG0	初始化配置寄存器 0
ICG1	初始化配置寄存器 1
ICG2	初始化配置寄存器 2
ICG3	初始化配置寄存器 3
ICG4	初始化配置寄存器 4
ICG5	初始化配置寄存器 5
ICG6	初始化配置寄存器 6

4 样例代码

4.1 代码介绍

用户可以根据上述的工作流程编写自己的代码来学习验证该模块，也可以直接通过华大半导体的网站下载到 HC32F120 系列 MCU 的设备驱动库（Device Driver Library, DDL）来体验 ICG 的硬件启动 SWDT 功能。

以下部分主要基于 DDL 的 ICG 模块的硬件启动 SWDT 功能样例 `icg_swdt_interrupt_hw_startup` 代码，简要介绍 ICG 的硬件启动 SWDT 使用方法：

1) 打开 `hc32f120_icg.h` 文件，配置 ICG 的 SWDT 功能：

```
/* Enable or disable SWDT hardware start */
#define ICG0_SWDT_HARDWARE_START    ICG_FUNCTION_ON

/* SWDT register config */
#define ICG0_SWDT_AUTS                ICG_SWDT_AFTER_RESET_AUTOSTART
#define ICG0_SWDT_ITS                ICG_SWDT_TRIG_EVENT_INT
#define ICG0_SWDT_PERI                ICG_SWDT_COUNTER_CYCLE_256
#define ICG0_SWDT_CKS                ICG_SWDT_CLOCK_DIV128
#define ICG0_SWDT_WDPT                ICG_SWDT_RANGE_100PCT
#define ICG0_SWDT_SLTPOFF             ICG_SWDT_LPW_MODE_COUNT_CONTINUE
```

2) 配置 LED：

```
/* Configure structure initialization */
GPIO_StructInit(&stcGpioInit);

/* LED Port/Pin initialization */
stcGpioInit.u16PinMode = PIN_MODE_OUT;
GPIO_Init(LED_R_PORT, LED_R_PIN, &stcGpioInit);
GPIO_Init(LED_G_PORT, LED_G_PIN, &stcGpioInit);
LED_R_OFF();
LED_G_OFF();
```

3) 配置 SWDT 中断功能：

```
/* NVIC configure of SWDT */
stcIrqRegister.enIntSrc = INT_SWDT_NMIUNDF;
stcIrqRegister.enIRQn = Int008_IRQn;
stcIrqRegister.pfnCallback = SWDT_IrqCallback;
u8Ret = INTC_IrqRegistration(&stcIrqRegister);
if (Ok != u8Ret)
{
    // check parameter
    while (1);
}
```

```

/* Clear pending */
NVIC_ClearPendingIRQ(stcIrqRegister.enIRQn);
/* Set priority */
NVIC_SetPriority(stcIrqRegister.enIRQn, DDL_IRQ_PRIORITY_DEFAULT);
/* Enable NVIC */
NVIC_EnableIRQ(stcIrqRegister.enIRQn);

/* Enable stop mode wakeup */
INTC_WakeupSrcCmd(INTC_WUPENR_SWDTWUEN, Enable);

```

4) 配置外部按键功能:

```

/* configure structure initialization */
GPIO_StructInit(&stcGpioInit);
EXINT_StructInit(&stcExIntInit);

/* External interrupt Ch.6 initialize */
stcGpioInit.u16ExInt = PIN_EXINT_ON;
GPIO_Init(SW1_PORT, SW1_PIN, &stcGpioInit);

/* EXINT Channel 6 (SW1) configure */
stcExIntInit.u16ExIntCh = EXINT_CH06;
stcExIntInit.u8ExIntFE = EXINT_FILTER_OFF;
stcExIntInit.u8ExIntLvl = EXINT_TRIGGER_FALLING;
EXINT_Init(&stcExIntInit);

/* Clear pending */
NVIC_ClearPendingIRQ(EXINT06_IRQn);
/* Set priority */
NVIC_SetPriority(EXINT06_IRQn, DDL_IRQ_PRIORITY_DEFAULT);
/* Enable NVIC */
NVIC_EnableIRQ(EXINT06_IRQn);

/* Enable stop mode wakeup */
INTC_WakeupSrcCmd(INTC_WUPENR_EIRQWUEN_6, Enable);

```

5) 按键次数切换工作模式:

```

/* Sleep mode */
if (1u == u8ExIntCnt)
{
    PWC_EnterSleepMode();
}
/* Stop mode */
else if (2u == u8ExIntCnt)
{
    PWC_EnterStopMode();
}

```

6) 按键中断改变工作模式 Flag:

```

if (Set == EXINT_GetExIntSrc(EXINT_CH06))
{

```

```

u8ExIntCnt++;
if (u8ExIntCnt >= 3u)
{
    u8ExIntCnt = 0u;
}
LED_R_OFF();
LED_G_OFF();
EXINT_ClrSrc(EXINT_CH06);
}

```

- 7) 设置约 1 秒触发一次 SWDT 计数溢出中断，通过 SW1 循环切换工作模式（正常模式、睡眠模式、停止模式），正常模式 LED_R 闪烁，睡眠模式 LED_G 闪烁，停止模式 LED_R 和 LED_G 同时闪烁。

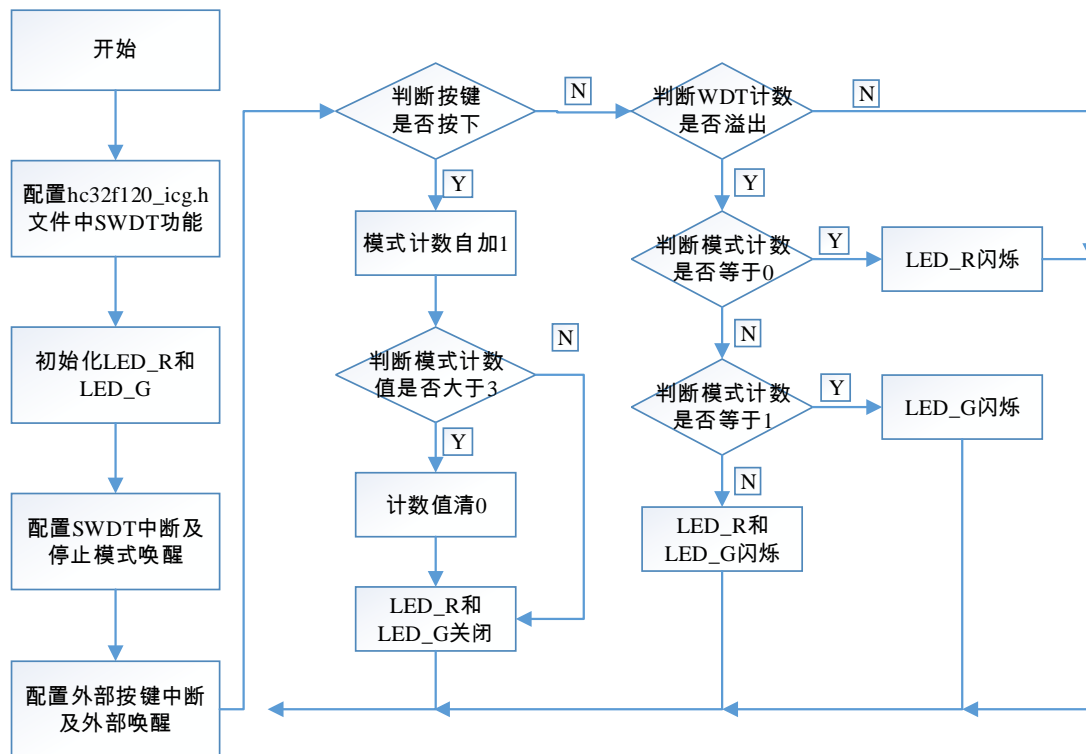
```

enFlagSta = SWDT_GetFlag(SWDT_FLAG_COUNT_UNDERFLOW);
/* SWDT underflow interrupt */
if (Set == enFlagSta)
{
    SWDT_ClearFlag(SWDT_FLAG_COUNT_UNDERFLOW);
    /* Normal mode */
    if (0u == u8ExIntCnt)
    {
        LED_R_TOGGLE();
    }
    /* Sleep mode */
    else if (1u == u8ExIntCnt)
    {
        LED_G_TOGGLE();
    }
    /* Stop mode */
    else
    {
        LED_R_TOGGLE();
        LED_G_TOGGLE();
    }
}
}

```

4.2 工作流程

样例代码中 ICG 操作流程如下图所示：



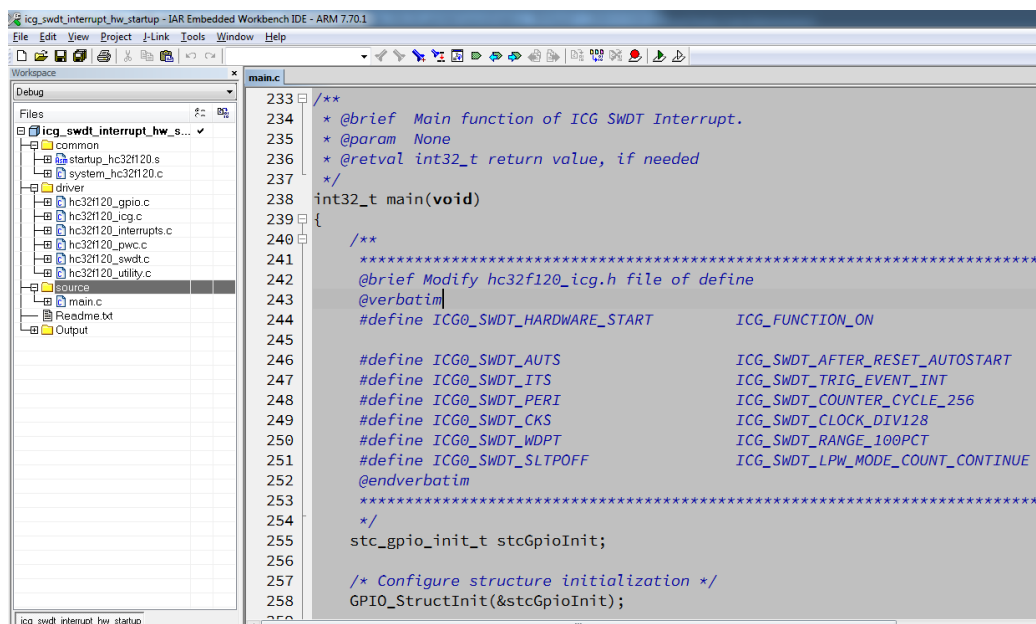
4.3 代码运行



用户可以通过华大半导体的网站下载到 DDL 的样例代码（icg_hrc_osc_hw_startup、icg_lvd_interrupt_hw_startup、icg_lvd_reset_hw_startup、icg_nmi_pin_hw_startup、icg_swtdt_interrupt_hw_startup、icg_swtdt_reset_hw_startup），并配合评估用板（比如‘STK_HC32F120_LQFP44_080_V11’）运行相关代码学习使用 ICG 模块。

以下部分主要介绍如何在‘STK_HC32F120_LQFP44_080_V11’评估板上，通过 IAR EWARM 编译、运行 icg_swtdt_interrupt_hw_startup 样例代码并观察结果：

- 确认安装正确的 IAR EWARM v7.7 工具（请从 IAR 官方网站下载相应的安装包，并参考用户手册进行安装）。
- 获取‘STK_HC32F120_LQFP44_080_V11’评估板。
- 从华大半导体网站下载 HC32F120 DDL 代码。
- 下载并运行 icg\icg_swtdt_interrupt_hw_startup\ 中的项目文件：

1) 打开 icg_swtdt_interrupt_hw_startup\ 项目，并打开‘main.c’如下视图：



- 2) 点击  重新编译整个项目；
- 3) 点击  将代码下载到评估板上，关闭 IDE 下载界面；
- 4) 复位按键复位芯片，进入正常模式 LED_R 闪烁；
- 5) 按下 SW1 切换工作状态到睡眠模式，LED_G 闪烁；

- 6) 再次按下 SW1 切换工作状态到停止模式，LED_R 和 LED_G 闪烁；
- 7) 再次按下 SW1 切换工作状态到正常模式，LED_R 闪烁。

5 总结

以上章节简要介绍 HC32F120 系列的 ICG 寄存器、功能模式。演示了如何操作 ICG 的硬件启动 SWDT 样例代码，在开发中用户可以根据自己的实际需要使用 ICG 模块。

6 版本信息 & 联系方式

日期	版本	修改记录
2019/10/31	Rev1.0	初版发布



如果您在购买与使用过程中有任何意见或建议，请随时与我们联系。

Email: mcu@hdsc.com.cn

网址: www.hdsc.com.cn

通信地址: 上海市浦东新区中科路 1867 号 A 座 10 层

邮编: 201203

