

## 32 位微控制器

# HC32F460 系列的通用定时器 TIMERA

适用对象

F 系列	HC32F460
------	----------

# 目 录

<b>1</b>	<b>摘要 .....</b>	<b>3</b>
<b>2</b>	<b>TIMERA 简介 .....</b>	<b>3</b>
2.1	主要特性.....	3
2.2	基本框图.....	4
<b>3</b>	<b>HC32F460 系列的 TIMERA .....</b>	<b>5</b>
3.1	波形模式.....	5
3.1.1	锯齿波模式.....	5
3.1.2	三角波模式.....	5
3.2	基本功能.....	6
3.2.1	基本计数 .....	6
3.2.2	捕获输入 .....	6
3.2.3	比较输出 .....	7
3.3	正交编码.....	8
3.3.1	位置计数模式 .....	8
3.3.2	公转计数模式 .....	9
3.4	特殊功能.....	10
3.4.1	数字滤波 .....	10
3.4.2	缓存功能 .....	11
3.4.3	同步启动 .....	12
3.5	寄存器介绍.....	13
<b>4</b>	<b>样例代码 .....</b>	<b>14</b>
4.1	代码介绍.....	14
4.2	工作流程.....	17
4.3	代码运行.....	18
<b>5</b>	<b>总结 .....</b>	<b>19</b>
<b>6</b>	<b>版本信息 &amp; 联系方式 .....</b>	<b>20</b>

## 1 摘要

本篇应用笔记主要介绍 HC32F460 系列通用定时器（TIMERA）模块，并简要说明通过 TIMERA 的正交编码计数功能如何实现 3 相正交编码计数。

## 2 TIMERA 简介

通用定时器（TIMERA）是一个具有 16 位计数宽度、8 路 PWM 输出的定时器，该定时器支持三角波和锯齿波两种波形模式，可生成各种 PWM 波形，支持计数器同步启动，比较基准值寄存器支持缓存功能，支持 2 相正交编码计数和 3 相正交编码计数。

本系列产品搭载 6 个单元 TIMERA，最大可实现 48 路 PWM 输出。

### 2.1 主要特性

- 波形模式：锯齿波、三角波
- 递加、递减计数方向
- 基准值缓存功能
- 正交编码计数
- 同步启动计数器
- 比较匹配事件输出
- 比较匹配中断
- 周期匹配中断

端口名	方向	功能
TIMA_<t>_PWMm	输入或输出	捕获输入事件端口或 PWM 输出端口（m=1~8）
TIMA_<t>_CLKA	输入	捕获输入事件端口或 PWM 输出端口（m=1~8）
TIMA_<t>_CLKB		
TIMA_<t>_TRIG	输入	硬件触发启动、停止、清零事件输入端口

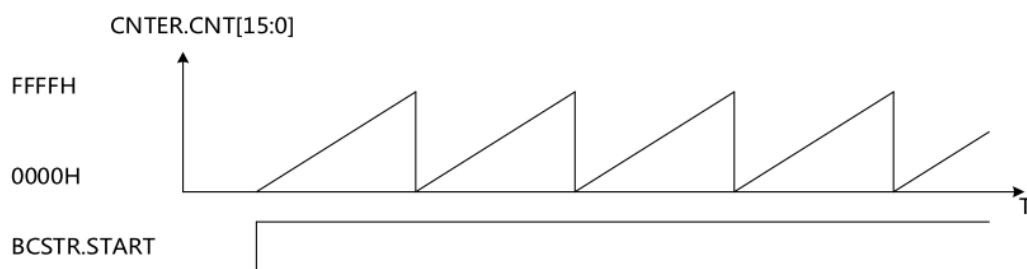
## 3 HC32F460 系列的 TIMERA

### 3.1 波形模式

TIMERA 有 2 种基本计数波形模式，锯齿波模式和三角波模式。

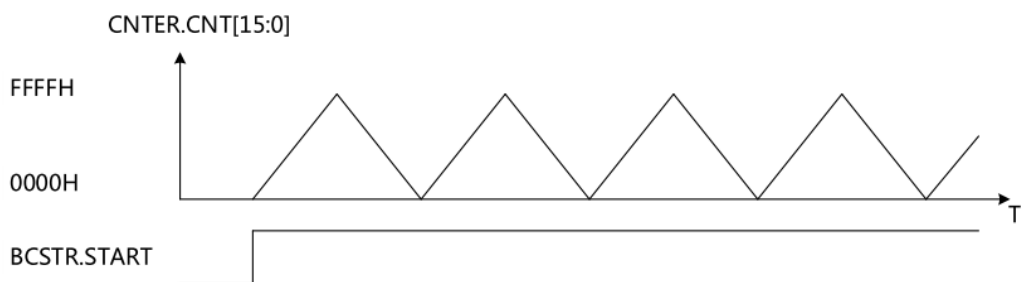
#### 3.1.1 锯齿波模式

计数器设置为锯齿波模式，基本波形如下图：



#### 3.1.2 三角波模式

计数器设置为三角波模式，基本波形如下图：



## 3.2 基本功能

### 3.2.1 基本计数

每个 TIMERA 单元可以设定一个基准计数值，根据配置的模式可以选择向上计数、向下计数，计数溢出时会可以配置相应的中断触发，每个通道可设定在计数值和基准值相等时产生计数比较匹配事件。

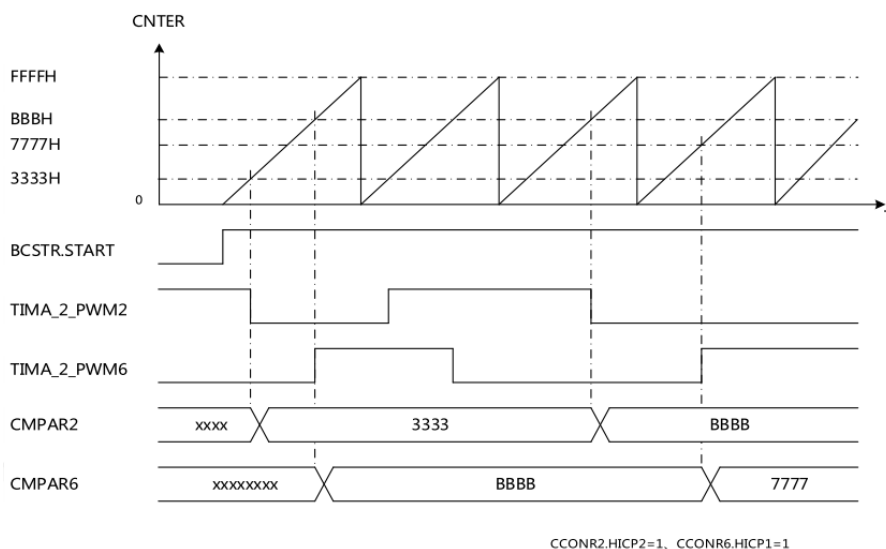
锯齿波模式递加计数至上溢点、锯齿波模式递减计数至下溢点、三角波模式计数至谷点或峰点，控制状态寄存器（BCSTR）的 OVFF 或 UDFE 位会被置为 1。

### 3.2.2 捕获输入

各个 TIMERA 单元的每个 PWM 输出通道都具有捕获输入功能，用于保存捕获到的计数值，设定捕获控制寄存器（CCONRn）的 CCONR.CAPMD 位为 1，捕获输入功能变为有效，各个单元的 TIMA\_<t>\_CLKA、TIMA\_<t>\_CLKB、TIMA\_<t>\_TRIG、TIMA\_<t>\_PWMn（捕获输入功能时）端口输入都有数字滤波功能。

当捕获输入条件有效时，当前的计数值就被保存到相应的寄存器（CMPARn）中（n=1~8），捕获输入条件可以选择内部捕获动作触发事件（通过 HTSSR1 寄存器选择）、

TIMA\_<t>\_PWMn 端口输入等，具体的条件选择可通过捕获控制寄存器（CCONRn）的 HICP 位来设定（n=1~8），下图为单元 2 的捕获输入动作示例：

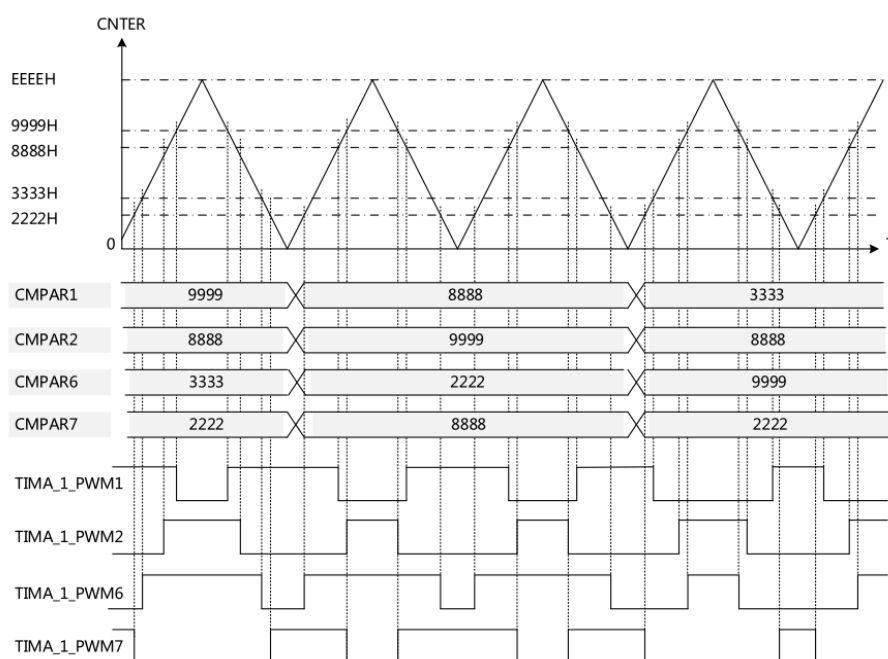


### 3.2.3 比较输出

各个 TIMERA 单元内部均含有 8 个通道的比较输出 (TIMA\_<t>\_PWMn)，可在计数值与比较基准值比较匹配时输出指定的电平，CMPARn 寄存器分别对应 TIMA\_<t>\_PWMn 输出端口的计数比较基准值；当定时器的计数值和 CMPARn 相等时，TIMA\_<t>\_PWMn 端口输出指定的电平 (n=1~8)。

TIMERA 内部的 8 路输出 TIMA\_<t>\_PWMn，每路输出都可以通过端口控制寄存器

(PCONRn) 的相关控制位实现不同的输出波形 (n=1~8)，下图为单元 1 三角波模式下，通道 1、2、6、7 的 PWM 输出波形示例：



### 3.3 正交编码

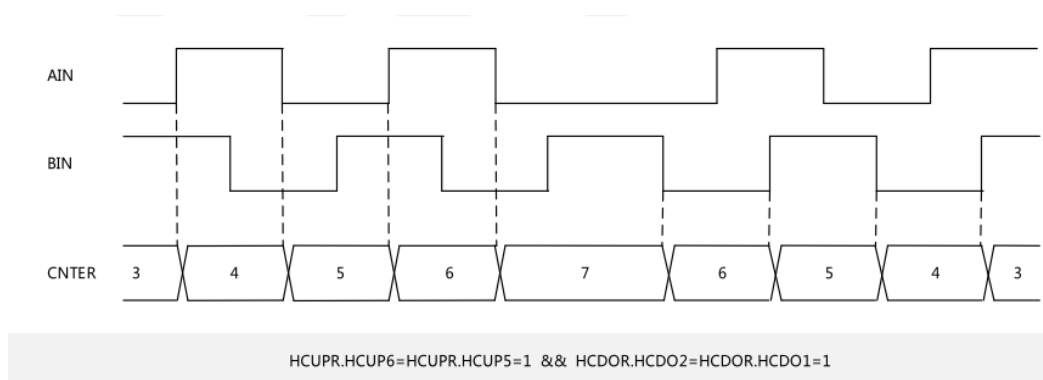
将 TIMA\_<t>\_CLKA 输入看作 AIN 输入、TIMA\_<t>\_CLKB 输入看作 BIN 输入、TIMA\_<t>\_TRIG 输入看作 ZIN 输入，TIMERa 就可以实现三路输入的正交编码计数。

每个单元的 AIN、BIN 单独动作可以实现位置计数模式；两个单元的 AIN、BIN、ZIN 组合动作可以实现公转计数模式，其中用于位置计数的单元称之为位置计数单元、用于公转计数的单元称之为公转计数单元。公转计数模式时，每两个单元间互相组合（单元 1、2 组合；单元 3、4 组合；单元 5、6 组合），组合内位置计数单元和公转计数单元可以任意指定。

#### 3.3.1 位置计数模式

正交编码位置计数模式，是指根据 AIN、BIN 的输入实现基本计数功能、相位差计数功能和方向计数功能。

其中相位差计数是指根据 AIN 和 BIN 的相位关系进行计数。根据设定的不同，可以实现 1 倍计数、2 倍计数、4 倍计数等，如下图为相位差 2 倍计数：



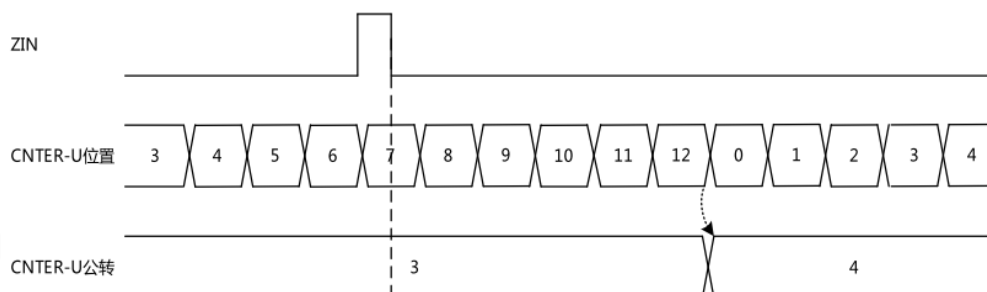


### 3.3.2 公转计数模式

正交编码公转计数模式，是指在 AIN、BIN 计数的基础上，加入 ZIN 的输入事件以实现对公转圈数等的判断，公转计数模式时根据公转定时器的计数方式，可实现 Z 相计数功能、位置溢出计数功能和混合计数功能。

其中位置溢出计数是指位置计数单元计数发生上溢或下溢时，产生一个溢出事件，从而触发公转计数单元的定时器进行一次计数（在该计数方式时 ZIN 的输入不进行公转计数单元的计数动作和位置计数单元的清零动作）。公转计数单元的硬件递加（递减）事件选择寄存器

（HCUPR 或 HCDOR）的递加（递减）事件 bit12~11 位使能，位置计数单元的溢出事件就可以触发公转计数单元实现一次计数，如下图所示：

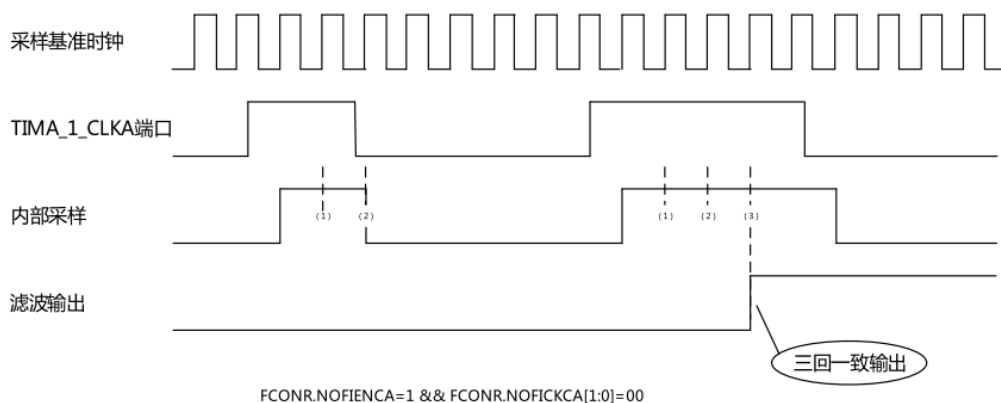


## 3.4 特殊功能

### 3.4.1 数字滤波

各个单元的 TIMA\_<t>\_CLKA、TIMA\_<t>\_CLKB、TIMA\_<t>\_TRIG、TIMA\_<t>\_PWMn（捕获输入功能时）端口输入都有数字滤波功能，各端口的滤波功能的使能和滤波时钟的选择可通过设定滤波控制寄存器（FCONR）和捕获控制寄存器（CCONRn）的对应位来实现（n=1~8）。

下图为单元 1 的 CLKA 端口滤波动作：

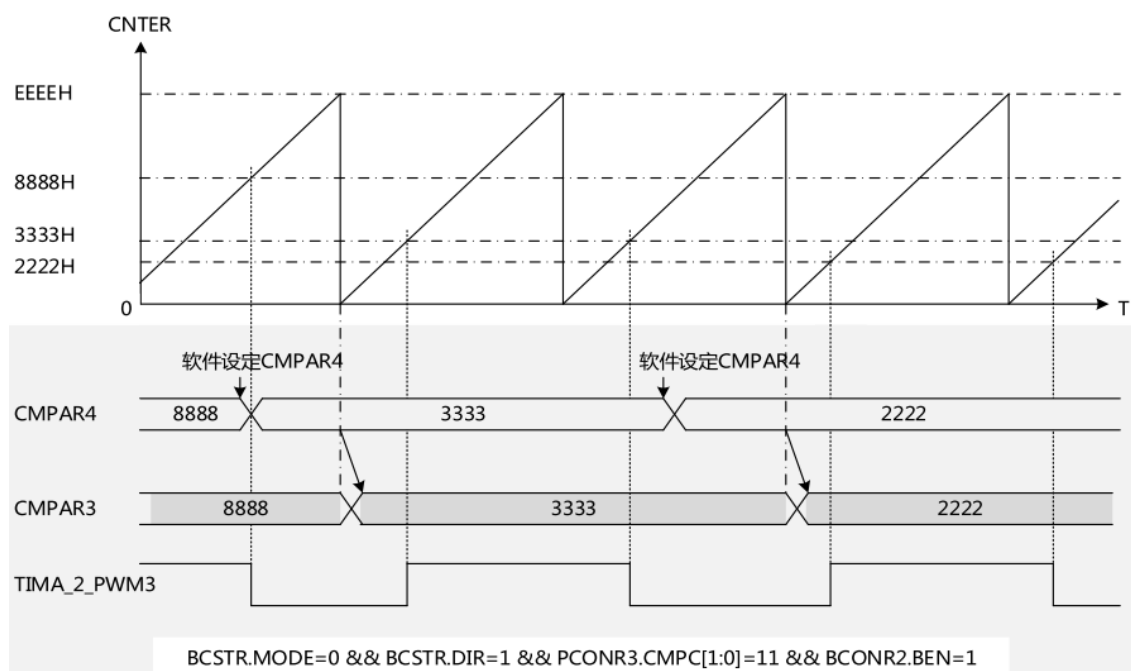


### 3.4.2 缓存功能

TIMERA 的共计 8 个比较基准寄存器 (CMPARn) 可以成对实现缓存功能 (n=1~8)，即 CMPAR2 作为 CMPAR1 的缓存基准值、CMPAR8 作为 CMPAR7 的缓存基准值，缓存控制寄存器 (BCONRm) 分别实现对四组缓存功能的控制 (m=1~4)。

当缓存控制寄存器 (BCONRm) 的 BEN 位被置位时，缓存功能变为有效 (m=1~4)，此时计数器计数到特定时间点时就发生一次缓存传送 (CMPAR8/6/4/2->CMPAR7/5/3/1)，该“特定时间点”有几种情况，详见参考手册。

下图为单元 2 锯齿波模式的缓存传送：

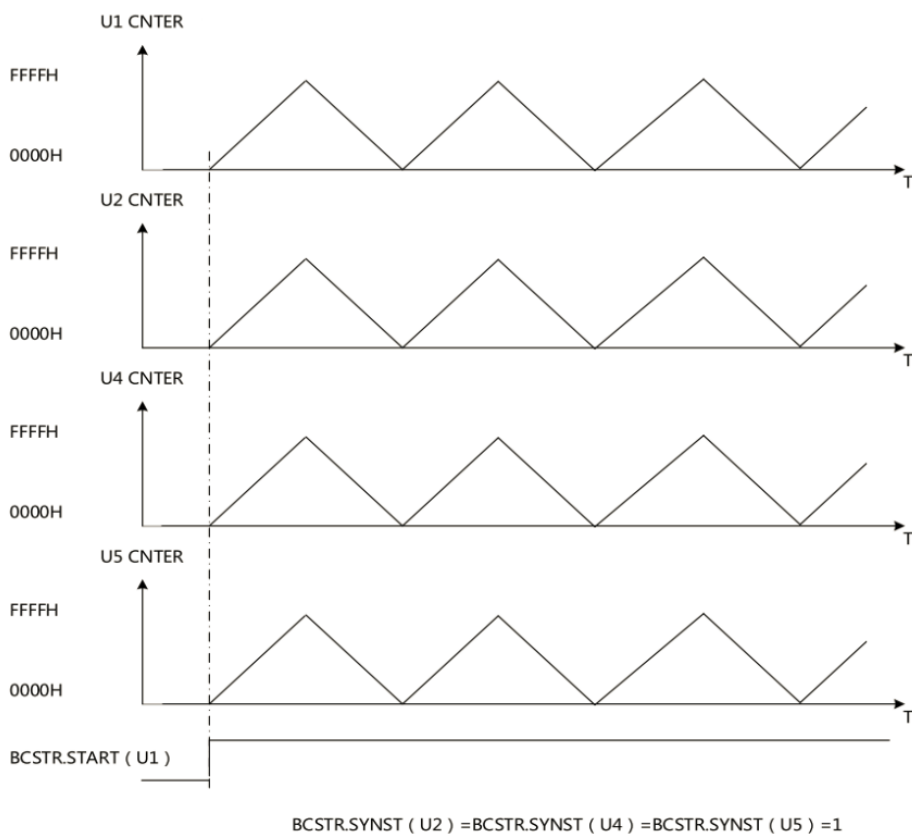


### 3.4.3 同步启动

TIMERA 可以实现软件同步启动或硬件同步启动，单元 2~单元 6 可以选择与单元 1 实现同步启动，当单元 2~单元 6 中的 BCSTR.SYNST 位设定为 1 时，对应单元与单元 1 的同步启动功能有效，此时，若软件设定单元 1 的 BCSTR.START 位为 1，被同步单元（单元 2~单元 6）的计数器开始软件同步计数。

若硬件设定单元 1 的 HCONR.HSTA1~0 中任意位为 1，且单元 1 的对应硬件事件发生时，被同步单元（单元 2~单元 6）的计数器开始硬件同步计数，在选择硬件同步计数启动功能时，被同步单元（单元 2~单元 6）的 HCONR.HSTA1~0 的对应位也必须设定为有效。

下图为设定单元 2、单元 4、单元 5 的 BCSTR.SYNST=1 时的软件同步启动：



### 3.5 寄存器介绍

通用定时器 TIMERA 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	备注
TMRA_CNTER	通用计数值寄存器	
TMRA_PERAR	周期基准值寄存器	
TMRA_CMPARn	比较基准值寄存器	n=1~8
TMRA_BCSTR	控制状态寄存器	
TMRA_ICONR	中断控制寄存器	
TMRA_ECONR	事件控制寄存器	
TMRA_FCONR	滤波控制寄存器	
TMRA_STFLR	状态标志寄存器	
TMRA_BCONRn	缓存控制寄存器	n=1~4
TMRA_CCONRn	捕获控制寄存器	n=1~8
TMRA_PCONRn	端口控制寄存器	n=1~8
TMRA_HCONR	硬件触发事件选择寄存器	
TMRA_HCUPR	硬件递加事件选择寄存器	
TMRA_HCDOR	硬件递减事件选择寄存器	
TMRA_HTSSRn	内部触发事件选择寄存器	n=0~1

## 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 HC32F460 系列 MCU 的设备驱动库（Device Driver Library, DDL）来体验 TIMERA 的正交编码计数功能。

以下部分主要基于 DDL 的 TIMERA 模块的正交编码计数功能样例

timera\_position\_overflow\_count 代码，简要介绍 TIMERA 的正交编码计数使用方法：

- 1) 开启 TIMERA 时钟：

```
/* Initialize I/O */  
Led_Init();
```

- 2) 配置 TIMERA 正交编码计数功能使用的 IO：

```
/* Initialize I/O */  
/* Configuration TIMERA coding pin */  
PORT_SetFunc(TIMERA_UNIT1_CLKA_PORT, TIMERA_UNIT1_CLKA_PIN,  
TIMERA_UNIT1_CLKA_FUNC, Disable);  
PORT_SetFunc(TIMERA_UNIT1_CLKB_PORT, TIMERA_UNIT1_CLKB_PIN,  
TIMERA_UNIT1_CLKB_FUNC, Disable);
```

- 3) 初始化 TIMERA 单元 1 和单元 2 基本计数功能：

```
/* Configuration timera unit 1 structure */  
stcTimeraInit.enCntMode = TimeraCountModeSawtoothWave;  
stcTimeraInit.enSyncStartupEn = Disable;  
stcTimeraInit.u16PeriodVal = 1000;  
TIMERA_BaseInit(TIMERA_UNIT1, &stcTimeraInit);  
TIMERA_IrqCmd(TIMERA_UNIT1, TimeraIrqOverflow, Enable);  
  
/* Configure timera unit 2 structure */  
stcTimeraInit.u16PeriodVal = 6;  
TIMERA_BaseInit(TIMERA_UNIT2, &stcTimeraInit);  
TIMERA_IrqCmd(TIMERA_UNIT2, TimeraIrqOverflow, Enable);
```

- 4) 初始化 TIMERA 单元 1 和单元 2 正交编码计数功能：

```
/* Configure coding count structure */  
stcTimeraCondInit.enIncClkBHighAndClkARisingEn = Enable;  
stcTimeraCondInit.enClkAFilterEn = Enable;  
stcTimeraCondInit.enClkAClkDiv = TimeraFilterPclkDiv4;  
stcTimeraCondInit.enClkBFilterEn = Enable;
```

```
stcTimeraCondngInit.enClkBCLKDiv = TimeraFilterPclkDiv4;
TIMERA_OrthogonalCodingInit(TIMERA_UNIT1, &stcTimeraCondngInit);

/* Configure position overflow count structure */
MEM_ZERO_STRUCT(stcTimeraCondngInit);
stcTimeraCondngInit.enIncAnotherUnitOverflowEn = Enable;
TIMERA_OrthogonalCodingInit(TIMERA_UNIT2, &stcTimeraCondngInit);
```

5) 初始化 TIMERA 单元 1 和单元 2 溢出中断功能:

```
/* Configure count overflow interrupt of timera unit 1 */
stcIrqRegiConf.enIntSrc = TIMERA_UNIT1_OVERFLOW_INT;
stcIrqRegiConf.enIRQn = Int006_IRQn;
stcIrqRegiConf.pfnCallback = TimeraUnit1Over_IrqCallback;
enIrqRegistration(&stcIrqRegiConf);
NVIC_ClearPendingIRQ(stcIrqRegiConf.enIRQn);
NVIC_SetPriority(stcIrqRegiConf.enIRQn, DDL_IRQ_PRIORITY_15);
NVIC_EnableIRQ(stcIrqRegiConf.enIRQn);

/* Configure count overflow interrupt of timera unit 2 */
stcIrqRegiConf.enIntSrc = TIMERA_UNIT2_OVERFLOW_INT;
stcIrqRegiConf.enIRQn = Int007_IRQn;
stcIrqRegiConf.pfnCallback = TimeraUnit2Over_IrqCallback;
enIrqRegistration(&stcIrqRegiConf);
NVIC_ClearPendingIRQ(stcIrqRegiConf.enIRQn);
NVIC_SetPriority(stcIrqRegiConf.enIRQn, DDL_IRQ_PRIORITY_15);
NVIC_EnableIRQ(stcIrqRegiConf.enIRQn);
```

6) 启动 TIMERA 单元 1 和单元 2 计数:

```
/* Timera unit 1 and unit 2 startup */
TIMERA_Cmd(TIMERA_UNIT1, Enable);
TIMERA_Cmd(TIMERA_UNIT2, Enable);
```

7) 配置函数发生器, 通道 A、B 输出频率 (1000HZ)、占空比相同, 将函数发生器输出通道连接到 PE9 和 PE11 引脚, 修改函数发生器相位配置, B 通道的相位超前 A 通道 1/4 周期, 观察 LED0 闪烁; LED0 状态切换 6 次则触发 LED1 状态切换。

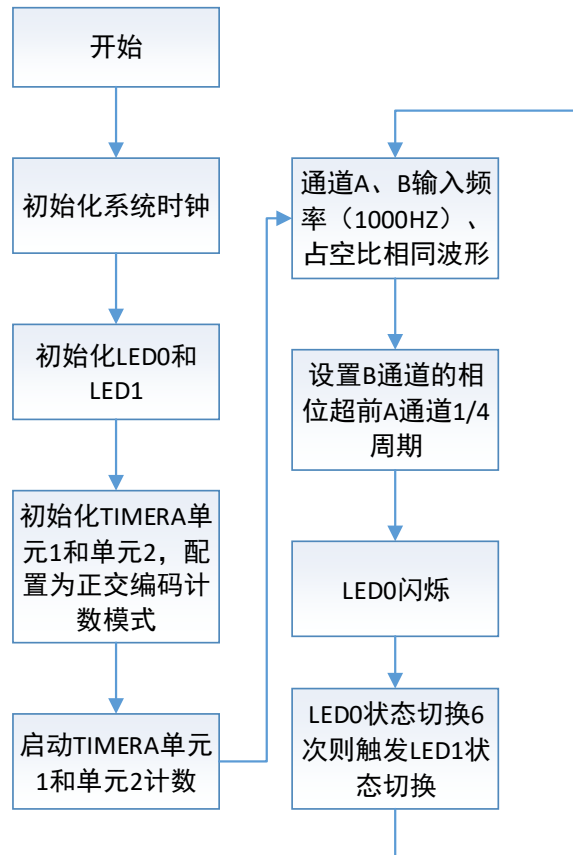
```
/**
*****
** \brief Timera unit 1 count overflow callback function
**
** \param [in] None
**
** \retval None
*****/
void TimeraUnit1Over_IrqCallback(void)
{
    LED0_TOGGLE();
    TIMERA_ClearFlag(TIMERA_UNIT1, TimeraFlagOverflow);
}
```

```
/**
*****
** \brief Timera unit 2 count overflow callback function
**
** \param [in] None
**
** \retval None
*****/
void TimeraUnit2Over_IrqCallback(void)
{
    LED1_TOGGLE();
    TIMERA_ClearFlag(TIMERA_UNIT2, TimeraFlagOverflow);
}
```



## 4.2 工作流程

样例代码中 TIMERA 操作流程如下图所示：



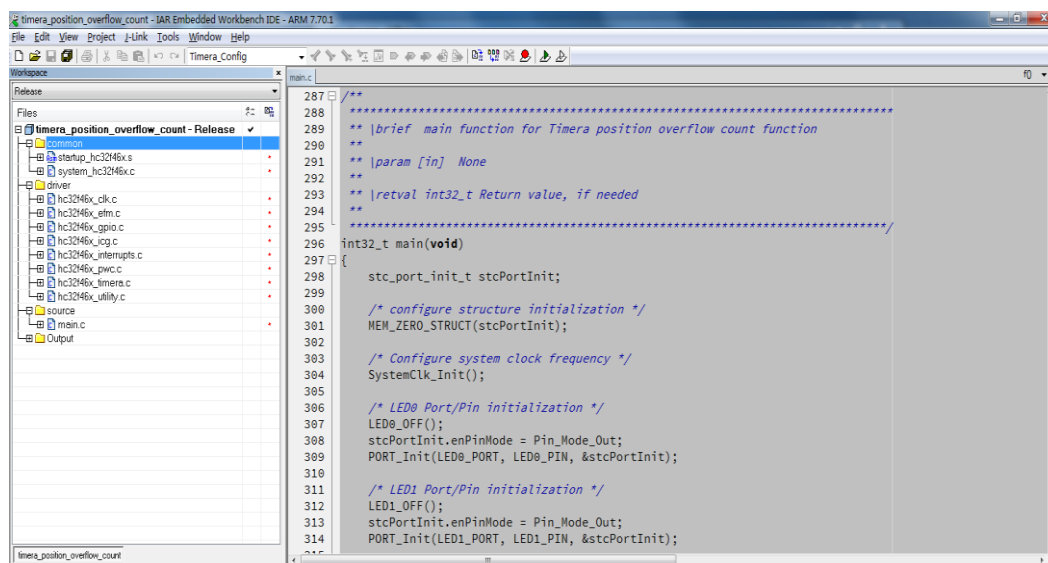
## 4.3 代码运行



用户可以通过华大半导体的网站下载到 DDL 的样例代码（timera\_sawtooth\_wave\_base\_timer、timera\_sawtooth\_wave\_capture\_input、timera\_triangular\_wave\_compare\_output、timera\_position\_overflow\_count），并配合评估用板（比如‘EV-HC32F460-LQFP100-050-V1.1’）运行相关代码学习使用 TIMERA 模块。

以下部分主要介绍如何在‘EV-HC32F460-LQFP100-050-V1.1’评估板上，通过 IAR EWARM 编译、运行 timera\_position\_overflow\_count 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‘EV-HC32F460-LQFP100-050-V1.1’评估板。
- 从华大半导体网站下载 HC32F460 DDL 代码。
- 下载并运行 timera\timera\_position\_overflow\_count\中的项目文件：

1) 打开 timera\_position\_overflow\_count\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，全速运行；
- 4) 配置函数发生器，通道 A、B 输出频率（1000HZ）、占空比相同，将函数发生器输出通道连接到 PE9 和 PE11 引脚；
- 5) 修改函数发生器相位配置，B 通道的相位超前 A 通道 1/4 周期，观察 LED0 闪烁；

6) 观察测试板，LED0 状态切换 6 次则触发 LED1 状态切换。

## 5 总结

以上章节简要介绍 HC32F460 系列的 TIMERA 寄存器、功能模式。演示了如何操作 TIMERA 正交编码计数样例代码，在开发中用户可以根据自己的实际需要使用 TIMERA 模块。

## 6 版本信息 & 联系方式

日期	版本	修改记录
2019/3/15	Rev1.0	初版发布
2020/8/26	Rev1.1	更新支持型号



---

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](mailto:mcu@hdsc.com.cn)

网址: <http://www.hdsc.com.cn/mcu.htm>

通信地址: 上海市浦东新区中科路 1867 号 A 座 10 层

邮编: 201203

---

