

32 位 微控制器

HC32F460 系列的通用定时器 TIMER0

适用对象

系列	产品型号
HC32F460	HC32F460JEU A
	HC32F460JETA
	HC32F460KEUA
	HC32F460KETA
	HC32F460PETB

目 录

1	摘要	3
2	TIMER0 简介	3
3	HC32F460 系列的 TIMER0	4
3.1	系统框图	4
3.2	功能说明	5
3.2.1	时钟源选择	5
3.2.2	基本计数功能	5
3.2.3	硬件触发动作	5
3.2.4	中断及事件输出	6
3.3	注意事项	7
3.4	寄存器说明	8
4	样例代码	9
4.1	代码介绍	9
4.2	代码运行	11
5	总结	12
6	版本信息 & 联系方式	13

1 摘要

本篇应用笔记主要介绍 HC32F460 系列芯片的通用定时器（TIMER0）模块，并通过展示 BaseTimer 样例代码简要说明如何使用 TIMER0 模块。

2 TIMER0 简介

HC32F460 系列的通用定时器（TIMER0）模块是一个可以实现同步计数、异步计数两种方式的基本定时器。定时器内含 2 个通道，可以在计数期间产生比较匹配事件。该事件可以触发中断、也可以作为事件输出来控制其它模块等。本系列芯片搭载 2 个独立的 TIMER0 单元，TMR01 和 TMR02。

TIMER0 主要特性：

- 同步计数方式异步计数方式可选
- 中断输出或事件输出
- 两个通道共用一个内部硬件触发源

3 HC32F460 系列的 TIMER0

3.1 系统框图

一个单元 TIMER0 的系统框图如下所示，每个单元由 CH_A、CH_B 两个通道定时器组成。

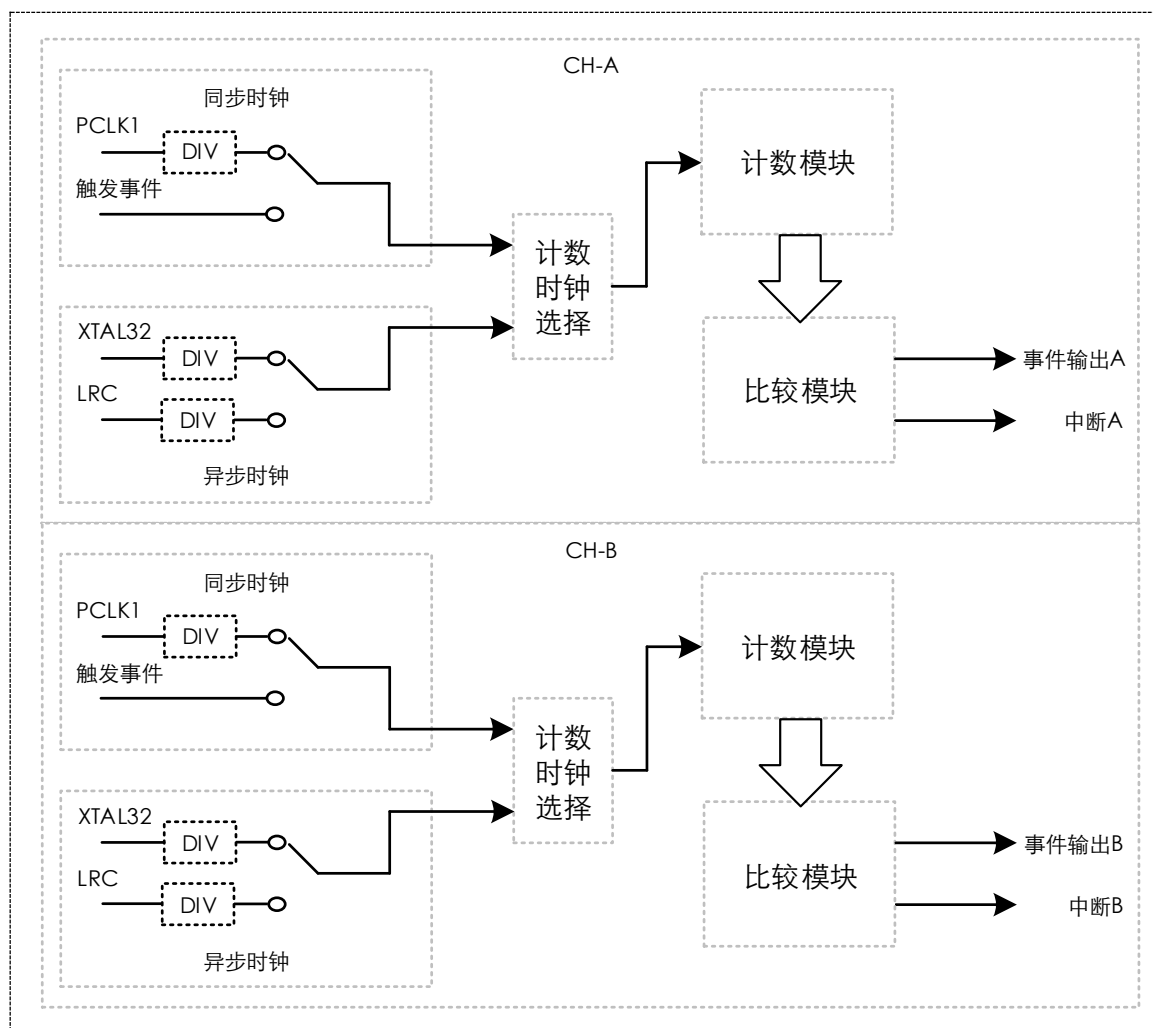


图 1 TIMER0 系统框图

3.2 功能说明

3.2.1 时钟源选择

通道 A 和通道 B 的计数单元可以分别选择独立的时钟源。时钟源分为两类：同步时钟源和异步时钟源。时钟源寄存器配置详情如下所示：

同步计数	BCONR.SYNS=0	BCONR.SYNCLK=0	PCLK1 及 PCLK1 的 2、4、8、16、32、64、128、256、512、1024 分频，分频系数由 BCONR.CKDIV[3:0]配置。
		BCONR.SYNCLK=1	内部硬件触发事件输入
异步计数	BCONR.SYNS=1	BCONR.ASYNCLK=0	LRC 时钟源输入及其 2、4、8、16、32、64、128、256、512、1024 分频作为输入时钟，分频系数由 BCONR.CKDIV[3:0]配置。
		BCONR.ASYNCLK=1	XTAL32 时钟源输入及其 2、4、8、16、32、64、128、256、512、1024 分频作为输入时钟，分频系数由 BCONR.CKDIV[3:0]配置。

3.2.2 基本计数功能

TIMER0 单元内每个通道可以独立设定基准计数值，在计数值和基准计数值匹配时产生比较匹配事件。详细介绍请参考本系列芯片用户手册相关章节。

3.2.3 硬件触发动作

TIMER0 单元内 2 个通道有一个共用的内部硬件触发源，触发源可以触发定时器计数、启动、停止、清零以及捕获输入动作。硬件触发源通过寄存器 HTSSR 进行选择，具体说明请参考本系列芯片用户手册 INTC 章节。

如采用外部中断事件作为触发源对 TIMER0 的单元 1 的 CH_B 进行事件触发时，需要的配置步骤如下：

- 1) 配置产生事件的外部中断 ExtiCh03
- 2) 配置 TIMER0 单元 1 的 CH_B 为硬件触发模式，硬件触发使能，触发动作为 start
- 3) 配置寄存器 TMR0_HTSSR 选择硬件触发源为 EVT_PORT_EIRQ3。
- 4) 配置 TIMER0 中断使能
- 5) 使能 TIMER0 单元 1 计数功能

完成以上配置后，通过外部中断事件可以触发产生 TIEMR0 的中断。

3.2.4 中断及事件输出

每个单元含有 2 个中断输出，分别是通道 A 和通道 B 中断输出。如通道 A 的比较匹配中断和输入捕获中断共用一个中断输出。

TIMER0 功能的事件输出与中断输出一一对应，如下所示。

单元号	中断输出	事件输出
TIMER0 单元 1 (TMR01)	INT_TMR01_GCMA INT_TMR01_GCMB	EVT_TMR01_GCMA EVT_TMR01_GCMB
TIMER0 单元 2 (TMR02)	INT_TMR02_GCMA INT_TMR02_GCMB	EVT_TMR02_GCMA EVT_TMR02_GCMB

3.3 注意事项

使用 TIMER0 模块时，需要注意以下几点：

- 1) 在异步计数方式时，寄存器读操作必须在计数停止状态下实现。
- 2) 寄存器标志位 BCONR.CSTA 和 BCONR.CSTB 的写入必须读出寄存器标志位确定写成功才能进行后续动作。
- 3) 单元 1 通道 A 的比较匹配中断 TMR_U1_GCMA 仅在异步计数方式时可用。

3.4 寄存器说明

以下为 TIMER0 模块的寄存器列表，详细寄存器说明请查看本系列芯片用户手册相关章节。

BASE ADDR: 0x40024000 (TMR01) 0x40024400 (TMR02)

寄存器名	符号	偏移量	位宽	复位值
计数值寄存器	TMR0_CNTAR	0x0000h	32	0x00000000h
计数值寄存器	TMR0_CNTBR	0x0004h	32	0x00000000h
基准值寄存器	TMR0_CMPAR	0x0008h	32	0x0000FFFFh
基准值寄存器	TMR0_CMPBR	0x000ch	32	0x0000FFFFh
基本控制寄存器	TMR0_BCONR	0x0010h	32	0x00000000h
触发选择寄存器	TMR0_HTSSR	(0x40010840h)	32	0x000001FFh
状态标志寄存器	TMR0_STFLR	0x0014h	32	0x00000000h

4 样例代码

4.1 代码介绍

用户可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设备驱动库（Device Driver Library, DDL）的样例代码并使用其中的 TIMER0 的样例进行验证。

以下部分简要介绍本 AN 基于 DDL 的 TIMER0 模块样例 BaseTimer 代码所涉及的各项配置。

1) 系统时钟、测试 LED 端口初始化

```
SysClkIni();  
/*initiallize LED port*/  
stcPortInit.enPinMode = Pin_Mode_Out;  
stcPortInit.enExInt = Enable;  
stcPortInit.enPullUp = Enable;  
/* LED0 and LED1 Port/Pin initialization */  
PORT_Init(LED0_PORT, LED0_PIN, &stcPortInit);  
PORT_Init(LED1_PORT, LED1_PIN, &stcPortInit);  
/* Get pclk1 */  
CLK_GetClockFreq(&stcClkTmp);  
u32Pclk1 = stcClkTmp.pclk1Freq;
```

2) TIMER0 CH_A 外设使能及基本计数器功能配置

```
/* Timer0 peripheral enable */  
ENABLE_TMR0();  
/*config register for channel A */  
stcTimerCfg.Tim0_CounterMode = Tim0_Async;  
stcTimerCfg.Tim0_AsyncClockSource = Tim0_XTAL32;  
stcTimerCfg.Tim0_ClockDivision = Tim0_ClkDiv4;  
stcTimerCfg.Tim0_CmpValue = 32000/4 - 1;  
TIMER0_BaseInit(TMR_UNIT, Tim0_ChannelA, &stcTimerCfg);
```

3) TIMER0 CH_A 中断功能配置及使能

```
/* Enable channel A interrupt */  
TIMER0_IntCmd(TMR_UNIT, Tim0_ChannelA, Enable);  
/* Register TMR_INI_GCMA Int to Vect.No.001 */  
stcIrqRegiConf.enIRQn = Int001_IRQn;  
/* Select Event interrupt function */  
stcIrqRegiConf.enIntSrc = TMR_INI_GCMA;  
/* Callback function */  
stcIrqRegiConf.pfnCallback = Timer0A_CallBack;  
/* Registration IRQ */  
enIrqRegistration(&stcIrqRegiConf);
```

```
/* Clear Pending */  
NVIC_ClearPendingIRQ(stcIrqRegiConf.enIRQn);  
/* Set priority */  
NVIC_SetPriority(stcIrqRegiConf.enIRQn, DDL_IRQ_PRIORITY_15);  
/* Enable NVIC */  
NVIC_EnableIRQ(stcIrqRegiConf.enIRQn);
```

4) TIMER0 CH_B 计数功能及中断配置

略

5) 计数功能使能

```
/*start timer0*/  
TIMER0_Cmd(TMR_UNIT,Tim0_ChannelA,Enable);  
TIMER0_Cmd(TMR_UNIT,Tim0_ChannelB,Enable);
```

6) 测试代码，计数值读取

```
while(1)  
{  
    /* Read counter register of channelB*/  
    u16cnt = TIMER0_GetCntReg(TMR_UNIT,Tim0_ChannelB);  
    u16cmp = TIMER0_GetCntReg(TMR_UNIT,Tim0_ChannelB);  
  
    /* Read counter register of channel A, need stop counter function for asynchronous mode*/  
    TIMER0_Cmd(TMR_UNIT,Tim0_ChannelA,Disable);  
    u16cnt = TIMER0_GetCntReg(TMR_UNIT,Tim0_ChannelA);  
    u16cmp = TIMER0_GetCntReg(TMR_UNIT,Tim0_ChannelA);  
    TIMER0_Cmd(TMR_UNIT,Tim0_ChannelA,Enable);  
  
    u32tmp = 0xFFFFF;  
    while(u32tmp--);  
}
```

7) 中断服务程序

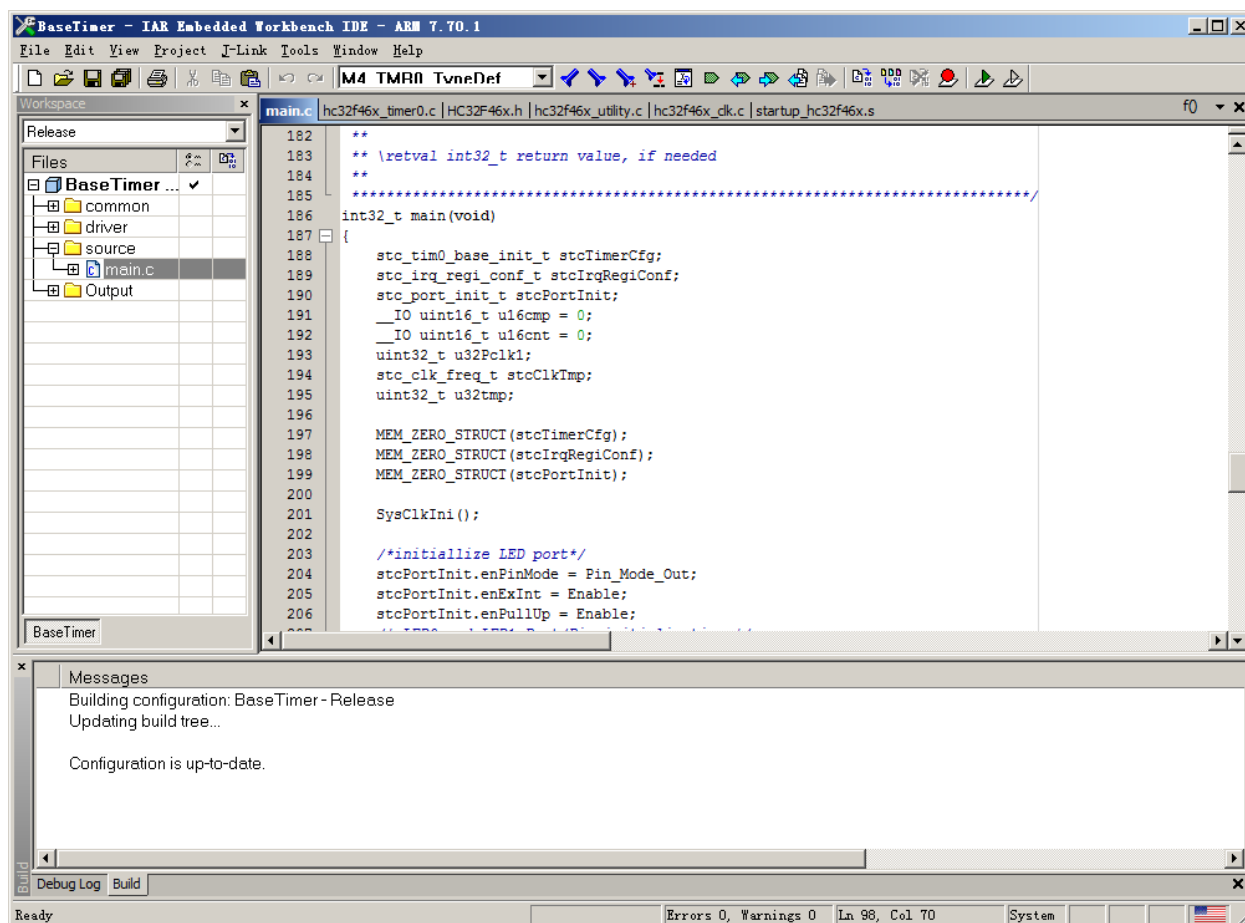
```
void Timer0A_CallBack(void)  
{  
    LED0_TOGGLE();  
}  
  
void Timer0B_CallBack(void)  
{  
    LED1_TOGGLE();  
}
```



4.2 代码运行

用户可以通过华大半导体的网站下载到 HC32F460 的 DDL 的样例代码（BaseTimer），并配合评估用板（EV-HC32F460-LQFP100-050-V1.1）运行相关代码学习使用 TIMERO 模块。

以下部分主要介绍如何在评估板上运行 BaseTimer 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）。
- 从华大半导体网站下载 HC32F460 DDL 代码。
- 下载并运行 BaseTimer\中的工程文件：
 - 1) 打开 BaseTimer\工程，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) 观测 LED 的亮灭变化，此时评估板上的红灯和绿灯以不同的频率闪烁，表示 TIMER0 的 CH_A 和 CH_B 的计数及中断功能正常运行。

5 总结

以上章节简要介绍了 HC32F460 系列的 TIMER0，说明了 TIMER0 模块的寄存器及部分操作流程，并且演示了如何使用 TIMER0 样例代码，在实际开发中用户可以根据自己的需要配置和使用 TIMER0 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



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: <http://www.hdsc.com.cn/mcu.htm>

通信地址: 上海市张江高科园区碧波路 572 弄 39 号

邮编: 201203

