

硬核处理器系统 (HPS) 提供 4 个、32-bit 的与 level 4 (L4) 外设总线相连接的通用计时器。当 32-bit 二进制倒计时计时器达到零时，计时器有选择性地生成一个中断。计时器是 Synopsys® DesignWare® APB 计时器 (DW_apb_timers) 外设的实例。



微处理器单元 (MPU) 子系统提供额外的计时器。要了解关于 MPU 中计时器的更多信息，请参考 *Cyclone V 器件手册* 第 3 卷的 *Cortex A9 MPU System* 章节。

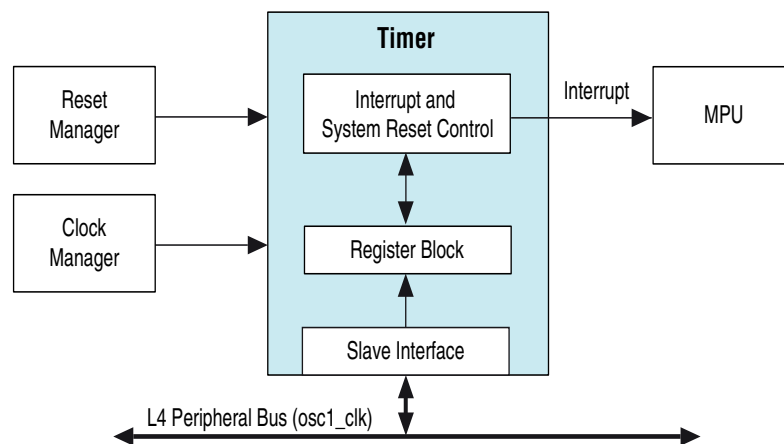
计时器的功能

- 支持中断生成
- 支持自由运行模式
- 支持用户定义的计数模式

计时器结构图和系统集成

图 23-1 显示了计时器的结构图。每个计时器包括用于控制和状态寄存器 (CSR) 访问的从接口、寄存器模块和达到零时生成中断的可编程 32-bit 递减计数器。计时器在时钟管理器驱动的单时钟域上进行操作。

图 23-1. 计时器结构图



© 2012 Altera Corporation. All rights reserved. ALTERA, ARRIA, CYCLONE, HARDCOPY, MAX, MEGACORE, NIOS, QUARTUS and STRATIX words and logos are trademarks of Altera Corporation and registered in the U.S. Patent and Trademark Office and in other countries. All other words and logos identified as trademarks or service marks are the property of their respective holders as described at www.altera.com/common/legal.html. Altera warrants performance of its semiconductor products to current specifications in accordance with Altera's standard warranty, but reserves the right to make changes to any products and services at any time without notice. Altera assumes no responsibility or liability arising out of the application or use of any information, product, or service described herein except as expressly agreed to in writing by Altera. Altera customers are advised to obtain the latest version of device specifications before relying on any published information and before placing orders for products or services.

Portions © 2011 Synopsys, Inc. Used with permission. All rights reserved. Synopsys & DesignWare are registered trademarks of Synopsys, Inc. All documentation is provided "as is" and without any warranty. Synopsys expressly disclaims any and all warranties, express, implied, or otherwise, including the implied warranties of merchantability, fitness for a particular purpose, and non-infringement, and any warranties arising out of a course of dealing or usage of trade.

†Paragraphs marked with the dagger (†) symbol are Synopsys Proprietary. Used with permission.



计时器的功能说明

32-bit 计数器从编程的值倒数并且当计数达到零时生成一个中断。计时器具有连接到系统时钟信号或外部时钟源的一个独立时钟输入。[†]

计时器支持以下操作模式：

- 自由运行模式 — 从最大值 (0xFFFFFFFF) 递减。达到零时重新加载最大值。
- 用户定义的计数模式 — 生成一个周期性中断。从 timer1 的加载计数寄存器 (timerloadcount) 加载的用户定义的计数值递减。达到零时重新加载用户定义的计数值。

计时器的初始值（从此值开始倒数）通过 timerloadcount 寄存器被加载到计时器。以下事件会导致计时器从 timerloadcount 寄存器加载初始计数值：[†]

- 复位或禁用后计时器被使能
- 计时器倒数至 0

时钟

表 23 - 1 显示了与计时器相关的时钟信号和连接。

表 23 - 1. 计时器时钟特征

计时器	系统时钟	注释
OSC1 计时器 0	oscl_clk	—
OSC1 计时器 1		
SP 计时器 0	l4_sp_clk	如果时钟频率更改，那么必须禁用计时器
SP 计时器 1		

SP 计时器 0 和 SP 计时器 1 必须在 l4_sp_clk 被更改为另一个频率之前被禁用。一旦时钟频率更改生效，您可以重新使能计时器。不可以更改 OSC1 计时器 0 和 OSC1 计时器 1 的频率。



要了解关于时钟性能的更多信息，请参考 *Cyclone V 器件手册* 第 3 卷的 *Clock Manager* 章节。

复位

计时器由冷或暖复位进行复位。复位计时器产生以下结果：

1. 计时器被禁用
2. 中断被使能
3. 计时器进入自由运行模式
4. 计时器的计数加载寄存器值被设置为零

中断

timer1 中断状态 (timerlintstat) 和中断的 timer1 的中断结束 (timerleoi) 寄存器处理中断。timerlintstat 寄存器使您能够读取中断的状态。timerleoi 寄存器的读取会返回 0 并且 清除中断。

timer1 控制寄存器 (timer1controlreg) 包含 timer1 中断屏蔽位 (timer1_interrupt_mask)，以便屏蔽中断。在自由运行和用户定义计数的操作模式中，当计时器计数值达到零并且控制寄存器的中断屏蔽位变为高电平时，计时器生成中断信号。


如果计时器中断被设置，那么当禁用计时器时，它被清零。

计时器编程模型

该部分介绍计时器的编程模型。

初始化

要初始化计时器，请执行以下步骤：

1. 通过 timer1controlreg 寄存器初始化计时器：
 - a. 通过在 timer1controlreg 寄存器的 timer1 使能位 (timer1_enable) 中写入 0 来禁用计时器。
 写入 timer1 加载计数寄存器 (timer1loadcount) 之前，必须通过写入 0 到 timer1controlreg 寄存器的 timer1_enable 位来禁用计时器，以便避免潜在的同步问题。
 - b. 编程计时器模式 — 用户定义的计数或自由运行 — 通过分别写入 0 或 1 到 timer1controlreg 寄存器的 timer1 模式位 (timer1_mode)。
 - c. 通过分别写入 1 或 0 到 timer1controlreg 寄存器的 timer1_interrupt_mask 位，以设置中断屏蔽为屏蔽的或未屏蔽的。
2. 将计时器计数器值加载到 timer1loadcount 寄存器。
3. 通过写入 1 到 timer1controlreg 寄存器的 timer1_enable 位使能计时器。

使能或禁用计时器

要使能计时器，写入 1 到 timer1controlreg 寄存器的 timer1_enable 位。要禁用计时器，写入 0 到 timer1_enable 位。

当计时器跳转到使能状态时，timer1loadcount 寄存器的当前值被加载到计时器计数器。

当计时器使能位被清 0 时，计时器时钟域中的计时器计数器和任何相关的寄存器被异步复位。

加载计时器倒计数值

当计时器计数器在复位或禁用后被使能时，计数值从 timer1loadcount 寄存器加载；该操作发生在自由运行和用户定义的计数模式中。

当计时器倒计数为 0 时，根据计时器操作模式，它加载以下两个值的其中之一：

- 用户定义的计数模式 — 计时器加载 timer1loadcount 寄存器的当前值。如果想要一个固定的，计时的中断，请使用该模式。通过写入 1 到 timer1controlreg 寄存器的 timer1_mode 位来指定该模式。

- 自由运行模式 — 计时器加载最大值 (0xFFFFFFFF)。另一个中断发生之前，计时器最大计数值支持最大时间量以重新编程或禁用计时器。如果您想要一个单计时中断，那么使用该模式。通过写入 0 到 timerlcontrolreg 寄存器的 timerl_mode 位使能该模式。

执行中断

该部分讨论各种中断情况以及如何执行它们。

清除中断

通过读取 timerleoi 寄存器或通过禁用计时器，您可以清除一个激活的计时器中断。当使能计时器时，它的中断保持置位直到通过读取 timerleoi 寄存器将其清除。

如果在计时器达到 0 的同时清除中断，那么中断保持置位。这是因为设置计时器中断优先于清除中断。


检查中断状态

您可以查询计时器的中断状态，而无需清除它的中断。要检查中断状态，请读取 timerlintstat 寄存器。

屏蔽中断

计时器中断可以使用 timerlcontrolreg 寄存器而被屏蔽。要屏蔽一个中断，请写入 1 到 timerlcontrolreg 寄存器的 timerl_interrupt_mask 位。


计时器地址映射和寄存器定义

 地址映射和寄存器定义位于该手册卷附带的 [hps.html](#) 文件中。点击链接以打开文件。

要查看模块说明和基地址，请找到并且点击以下任何模块实例的链接：

- `oscltimer0`
- `oscltimer1`
- `sptimer0`
- `sptimer1`

然后要查看寄存器和域说明，找到并且点击寄存器名称。寄存器地址是相对于每个模块实例的基地址的偏移。

 所有模块的基地址也在 *Cyclone V 器件手册* 第 3 卷的 *Introduction to the Hard Processor System* 章节列出。

文档修订历史

表 23 - 2 显示了该文档的修订历史。

表 23 - 2. 文档修订历史

日期	版本	修订内容
2012 年 11 月	1.2	少量文本编辑。
2012 年 5 月	1.1	添加了编程模型和地址映射以及寄存器定义部分。
2012 年 1 月	1.0	首次发布

