CoreTimer

Product Summary

Intended Use

Intended for Use in an Advanced Microcontroller Bus Architecture (AMBA)–Based Subsystem to Provide Timing Functionality

Key Features

- Optimized for Use with CoreMP7
- Configurable 16-Bit or 32-Bit Timer
- Runs from the Advanced Peripheral Bus (APB) Clock (PCLK) – No Additional Clock Required
- Prescale Provides Clock Division by up to 1,024
- Continuous or One-Shot Mode
- Interrupt Generation
- Supplied in SysBASIC Core Bundle

Benefits

- Configurable Programmable Timer Functionality for AMBA-Based Systems.
- Automatically Stitched in CoreConsole
- Compatible with AMBA and CoreMP7

Supported Families

- ARM®-Enabled ProASIC®3 and ProASIC3E
- ARM-Enabled Fusion

Synthesis and Simulation support

- Supported in the Actel Libero® Integrated Design Environment (IDE)

Verification and Compliance

- Compliant with AMBA

Contents

Introduction 1
Functional Description 2
Connecting CoreTimer in CoreConsole 3
CoreTimer Configurable Options 3
Programmer's Model 3
Resource Requirements 5
Ordering Information 5
Datasheet Categories 6

Introduction

The CoreTimer module is an APB slave that provides access to an interrupt-generating, programmable decrementing counter. Figure 1 shows a top-level block diagram of CoreTimer.

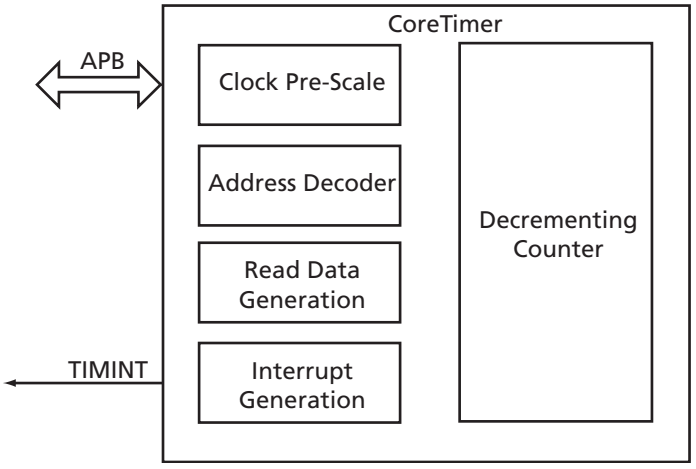

Figure 1 • **CoreTimer Block Diagram**

Functional Description

The width of the decrementing counter in the CoreTimer module can be statically configured as either 16 or 32 bits. Programmable registers provide a means to dynamically control the operation of the timer. If the interrupt is enabled, an interrupt is generated when the decrementing counter reaches zero.

There are two modes of operation available for CoreTimer: continuous mode and one-shot timer mode.

Continuous Mode

This is the default mode. When zero is reached, the counter is reloaded with the start value, which is stored in a programmable register, and continues to count down. If the interrupt is enabled, this mode can be used to generate an interrupt on a constant interval.

One-Shot Timer Mode

The counter decrements from its high value and halts on reaching zero. The timer must be reprogrammed to begin counting down again. This can be achieved by either clearing the timer mode bit in the control register or writing a new value to the load register.

Operation

The timer is loaded by writing to the load register and then, if enabled, counts down to zero. When the counter is already running, writing to the load register will cause the counter to immediately restart at the new value.

When zero is reached, an interrupt is generated if the interrupt is enabled. The interrupt can be cleared by writing to the interrupt clear register. If one-shot mode is selected, the counter halts on reaching zero until one-shot mode is deselected or a new load value is written. Otherwise, after reaching zero, the timer reloads the count value from the load register and continues to decrement. In this mode, the counter effectively generates a periodic interrupt.

Continuous or one-shot mode is selected by the timer mode bit in the timer control register. At any point, the current counter value can be read from the current value register.

The counter is enabled by a bit in the control register. At reset, the counter is disabled, the interrupt is cleared, and the load register is set to zero. The mode is set to continuous, and the prescale value is set to divide the clock by two.

A prescale unit is used to provide a clock enable pulse for the decrementing counter. The prescaler is driven by the APB clock (PCLK) and can be programmed via the timer control register to provide an enable pulse every 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1,024 periods of PCLK.

Interrupt Generation and Clearing

An interrupt is generated when the counter reaches zero and is only cleared when the interrupt clear register, TimerIntClr, is written to. A register holds the value until the interrupt is cleared.

The interrupt can be masked by writing '0' to the interrupt enable bit in the control register. Both the raw interrupt status (prior to masking) and the final interrupt status (after masking) can be read from status registers.

Clocking

The counter in CoreTimer is clocked with PCLK, but a clock enable signal produced by the prescaler is used to enable the counter to operate from a lower effective frequency than that at which PCLK is running.

The interval between clock enable pulses can be adjusted via the prescale field in the timer control register. It is possible to generate a clock enable pulse every 2, 4, 8, 16, 32, 64, 128, 256, 512, or 1,024 periods of PCLK.

Connecting CoreTimer in CoreConsole

Table 1 lists the ports present on the CoreTimer module and describes how to connect these in CoreConsole.

Table 1 • **CoreTimer Connections**

Connection	CoreConsole Label	Description
Required Connections		
APB slave interface	APBslave	Connect this interface to any available slave slot on the APB.
PCLK	PCLK	APB clock signal. Normally connected to the HCLK output of the MP7Bridge.
PRESETn	PRESETn	Active low APB reset input. Normally connected to the HRESETn output of the MP7Bridge.
Optional Connections		
Timer interrupt	TIMINT	Interrupt output for timer. This signal indicates that an interrupt has been generated by the counter having decremented to zero. The polarity of this signal is controlled by the **Interrupt active level** configurable option. This is normally connected to one of the interrupt source (ICINTSOURCEx) inputs of the Interrupt Controller.

CoreTimer Configurable Options

The configurable options for CoreTimer are outlined in Table 2.

Table 2 • **CoreTimer Configurable Options**

Configurable Option	Default Setting	Description
Counter width	32 bits	Sets the width of the counter in CoreTimer. Possible settings are "32 bit" and "16 bit."
Interrupt active level	Low	Selects active low or active high TIMINT interrupt: Low = active low interrupt (default) High = active high interrupt

Programmer's Model

The CoreTimer registers are shown in Table 3.

Table 3 • **CoreTimer Registers**

Offset	Type	Width	Reset Value	Name	Description
0x00	Read/Write	16 or 32	0x0000 or 0x00000000	TimerLoad	Load value
0x04	Read	16 or 32	0xFFFF or 0xFFFFFFFF	TimerValue	Current value
0x08	Read/Write	3	0x0	TimerControl	Control register
0x0C	Read/Write	4	0x0	TimerPrescale	Clock prescale setting
0x10	Write	–	–	TimerIntClr	Interrupt clear
0x14	Read	1	0x0	TimerRIS	Raw interrupt status
0x18	Read	1	0x0	TimerMIS	Masked interrupt status

Load Register – TimerLoad

This register contains the value from which the counter is to decrement. When this register is written to, the counter is loaded with the value written and begins to decrement if the timer is enabled.

The counter will be reloaded with the value in the load register when the current count reaches zero and continuous mode is enabled.

The load register is either 16 or 32 bits wide, depending on how CoreTimer is configured.

Current Value Register – TimerValue

This register gives the current value of the decrementing counter.

The current value register is either 16 or 32 bits wide, depending on how CoreTimer is configured.

Timer Control Register – TimerControl

Table 4 gives the bit assignments for the TimerControl register.

Table 4 • **Bit Assignments for the TimerControl Register**

Bit(s)	Name	Type	Function
31:3	–	–	Unused; reads zero.
2	Timer Mode	Read/Write	Selects timer operation mode: 0 = Continuous operation (default) 1 = One-shot count
1	Interrupt Enable	Read/Write	Interrupt enable bit: 0 = Timer interrupt disabled (default) 1 = Timer interrupt enabled
0	Timer Enable	Read/Write	Enable bit for timer: 0 = Timer disabled (default) 1 = Timer enabled

Prescale Setting Register – TimerPrescale

This register contains a single four-bit field that determines the effective clock rate for the timer counter, based on PCLK. Table 5 gives the bit assignments for the TimerPrescale register.

Table 5 • **Bit Assignments for the TimerPrescale Register**

Bits	Name	Type	Function
31:4	–	–	Unused; reads zero.
3:0	Prescale	Read/Write	Prescale field. Determines effective clock rate for counter based on PCLK: 0000 = divide by 2 (default) 0001 = divide by 4 0010 = divide by 8 0011 = divide by 16 0100 = divide by 32 0101 = divide by 64 0110 = divide by 128 0111 = divide by 256 1000 = divide by 512 1001 = divide by 1,024 Others = divide by 1,024

Interrupt Clear Register – TimerIntClr

Any write to this register will clear (deassert) the TIMINT interrupt output from the counter. Any data may be written.

Raw Interrupt Status Register – TimerRIS

This register indicates the raw interrupt status from the counter. This value is ANDed with the timer interrupt enable bit from the control register to create the masked interrupt, which is passed to the interrupt output pin. Table 6 gives the bit assignments for the TimerRIS register.

Table 6 • **Bit Assignments for the TimerRIS Register**

Bit(s)	Name	Type	Function
31:1	–	–	Unused; reads zero.
0	Raw Timer Interrupt	Read	Raw interrupt status from the counter: 0 = Raw interrupt not pending 1 = Raw Interrupt pending

Interrupt Status Register – TimerMIS

This register indicates the masked interrupt status from the counter. This value is the logical AND of the raw interrupt status with the timer interrupt enable bit from the control register and is the same value that is passed to the interrupt output pin. Table 7 gives the bit assignments for the TimerMIS register.

Table 7 • **Bit Assignments for the TimerMIS Register**

Bit(s)	Name	Type	Function
31:1	–	–	Unused; reads zero.
0	Timer Interrupt	Read	Enabled interrupt status from the counter: 0 = Interrupt not pending 1 = Interrupt pending

Resource Requirements

The utilization for CoreTimer in a ProASIC3E device is as follows:

Configured for 16-bit counter:	310 tiles
Configured for 32-bit counter:	535 tiles

Ordering Information

CoreTimer is included in the SysBASIC core bundle that is supplied with the Actel CoreConsole IP Deployment Platform tool. The obfuscated RTL version of SysBASIC (SysBASIC-OC) is available for free with CoreConsole. The source RTL version of SysBASIC (SysBASIC-RM) can be ordered through your local Actel sales representative. CoreTimer cannot be ordered separately from the SysBASIC core bundle.

Datasheet Categories

In order to provide the latest information to designers, some datasheets are published before data has been fully characterized. Datasheets are designated as "Product Brief," "Advanced," and "Production." The definitions of these categories are as follows:

Product Brief

The product brief is a summarized version of an advanced or production datasheet containing general product information. This brief summarizes specific device and family information for unreleased products.

Advanced

This datasheet version contains initial estimated information based on simulation, other products, devices, or speed grades. This information can be used as estimates, but not for production.

Unmarked (production)

This datasheet version contains information that is considered to be final.

www.actel.com

Actel Corporation

2061 Stierlin Court
Mountain View, CA
94043-4655 USA
Phone 650.318.4200
Fax 650.318.4600

Actel Europe Ltd.

Dunlop House, Riverside Way
Camberley, Surrey GU15 3YL
United Kingdom
Phone +44 (0) 1276 401 450
Fax +44 (0) 1276 401 490

Actel Japan
www.jp.actel.com

EXOS Ebisu Bldg. 4F
1-24-14 Ebisu Shibuya-ku
Tokyo 150 Japan
Phone +81.03.3445.7671
Fax +81.03.3445.7668

Actel Hong Kong
www.actel.com.cn

Suite 2114, Two Pacific Place
88 Queensway, Admiralty
Hong Kong
Phone +852 2185 6460
Fax +852 2185 6488

51700069-0/04.06