TIMER MODULE

Version 1.2.1

INICORE INC.
5600 Mowry School Road
Suite 180
Newark, CA 94560
t: 510 445 1529 *f:* 510 656 0995 *e:* info@inicore.com
www.inicore.com

Table of Contents

1 INTRODUCTION....4

1.1 Features....4

1.2 Deliverables....4

1.3 Block Diagram....5

1.4 Description of Main Building Blocks....5

1.4.1 Pre-Scaler....5

1.4.2 Bus Interfaces....5

1.4.3 Timer A/B....6

2 SIGNAL DESCRIPTIONS....7

2.1 Symbol....7

2.2 General Inputs....7

2.3 Local Bus....8

2.4 Other Ports....8

3 PROGRAMMING MODEL....9

3.1 Memory Mapping....9

3.2 Register Description....10

3.2.1 Operation....11

4 TIMING DIAGRAM....12

4.1 Local Bus Interface....12

Table of Figures

Figure 1: Block Diagram TIMERmodule.....5
Figure 2: Block Diagram Counter/Timer A.....6
Figure 3: Symbol with I/Os.....7
Figure 4: Local Bus Interface Timing.....12

1 Introduction

The TIMERmodule is a versatile counter/timer and can be used as a watchdog, a timer or a counter for external events.

Apart of a small asynchronous prescaler, the design is fully synchronous and implemented in technology independent VHDL and Verilog RTL code.

1.1 Features

Following special features are available:

- Clock prescaler for external high-frequency sources
- Internal or external clock source
- Supports single-shot, free running and counter mode
- Provides two 16-bit counters/timers with individual 16-bit prescaler
- Two interrupt sources, one for each counter/timer
- Individual counter capture commands
- Synchronous bus interface
  - Zero wait-states
  - Supports system bus' such as AMBA APB version 2.0
- Technology independent

1.2 Deliverables

- VHDL or Verilog RTL source code
- Simulation testbench
- Timing constraints file
- Synthesis script
- User Guide

1.3 Block Diagram

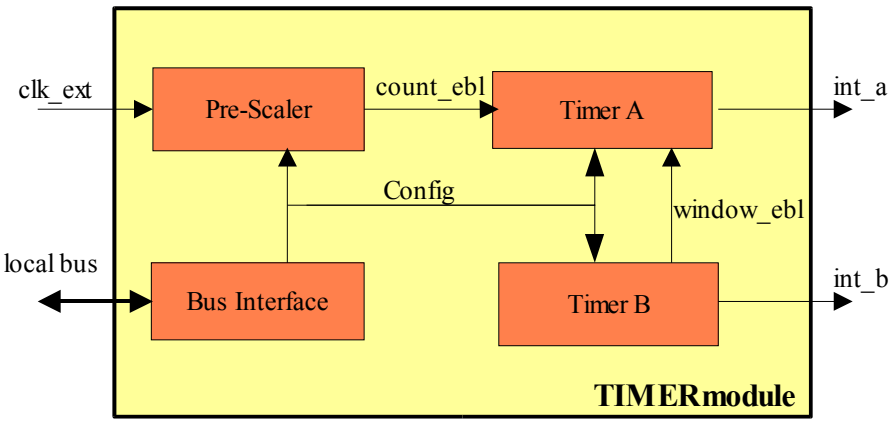

Figure 1: Block Diagram TIMERmodule

1.4 Description of Main Building Blocks

1.4.1 Pre-Scaler

An asynchronous divider is provided to downscale external high frequency clocks. The division factor is programmable in the range from 2, 4, 8, .. 256. This prescaler can be bypassed or alternatively, the system clock can be used.

1.4.2 Bus Interfaces

A synchronous bus interface is provided to facilitate system embedding. The interface timing is provided in chapter 4.

1.4.3 Timer A/B

The timer block contains a prescaler stage and the main counter/timer. The structure of timer block A is shown in the block diagram below:

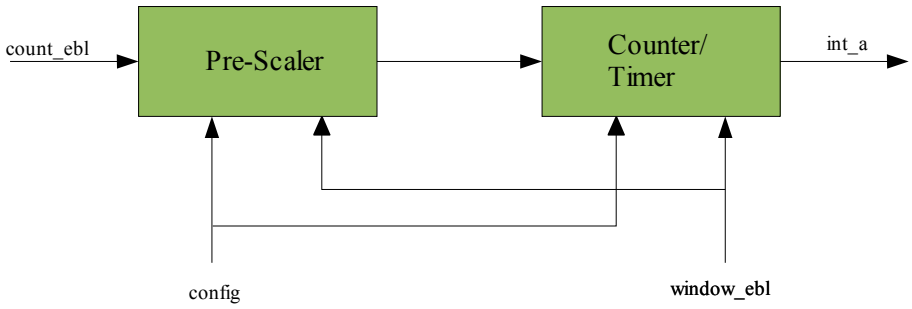

Figure 2: Block Diagram Counter/Timer A

The timer block has three different operating modes:

- free running
  The timer counts down to zero, issues an interrupt and restarts from the initial value.
- one-shot
  Upon starting the timer, it loads the initial value and counts down to zero. Once the terminal count is reached it stops and an interrupt is issued.
- counter
  In the counter mode, a time window is provided by counter b in which events are counted.

This prescaler and the counter/timer are clocked with the system clock 'clk'. The external clock (divided by 1/2/4/8/16/.../256) synchronized to the system clock is used as the count event. Alternatively the system clock can be used as well.

The system can always examine the state of the counters by issuing a capture command.

The difference between counter A and B is very small: counter B has no window_ebl input and doesn't support the 'ext_clk' prescaler.

2 Signal Descriptions

The following paragraph lists the input and output ports of the TIMERmodule. Please refer to the chapter 4 for a definition of the interface timings.

2.1 Symbol

The figure below shows all interfaces of the TIMERmodule core.

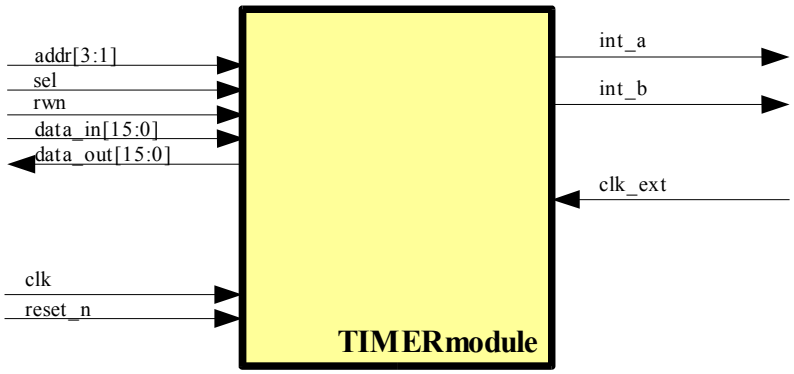

Figure 3: Symbol with I/Os

2.2 General Inputs

The module is initialized with one asynchronous, active low reset input. All registers are clocked with the system clock.

Pin Name	Type	Description
clk	in	System clock
reset_n	in	Asynchronous system reset, active low

2.3 Local Bus

A synchronous zero wait-states system bus interface provides access to all status and configuration registers. The timing diagram is shown in chapter 4.

Pin Name	*Type*	*Description*
addr[3:1]	in	Address bus input
data_write[15:0]	in	Data bus input
data_read[15:0]	out	Data bus output
sel	in	Module chip select, active high
rwn	in	Read/write control signal '0': Write '1': Read

2.4 Other Ports

The TIMERmodule provides two local interrupt sources. They can be, either drive an external output, control a system watchdog or they can be connected to a local interrupt controller.

Pin Name	*Type*	*Description*
int_a	out	Local timer A interrupt event
int_b	out	Local timer B interrupt event
clk_ext	in	External clock source

3 Programming Model

3.1 Memory Mapping

The table below shows the entire memory mapping of the TIMERmodule. A detailed description of the functionalities of these registers can be found in paragraph 3.2.

Addr	*Name*	*R/W*	*Description*
+0x0	tm_control	R/W	Control/Status Register
+0x2	tm_command	W	Command Register
+0x4	tm_count_a	W	Timer A: Count
		R	Timer A: Count Capture Register
+0x6	tm_prescale_a	W	Timer A: Prescaler
		R	Timer A: Prescaler Capture Register
+0x8	tm_count_b	W	Timer B: Count
		R	Timer B: Count Capture Register
+0xA	tm_prescale_b	W	Timer B: Prescaler
		R	Timer B: Prescaler Capture Register
+0xC	tm_config	r/W	Configuration Register

The following acronyms are used in the table above:

r: Data readback

R: Data read

W: Data write

3.2 Register Description

Addr	Name	R/W	Description
+0x0	tm_control	R/W	Control/Status Register [3]: Timer B running (read only) '0': Timer expired '1': Running [2]: Timer A running (read only) '0': Timer expired '1': Running [1]: Timer B enable '0': Timer disabled '1': Timer enabled [0]: Timer A enable '0': Timer disabled '1': Timer enabled
+0x2	tm_command	W	Command Register [1]: Capture Timer B '0': no operation '1': capture timer [0]: Capture Timer A '0': no operation '1': capture timer
+0x4	tm_count_a	W	Timer A: Count Initial and load value of the counter/timer.
		R	Timer A: Count Capture Register Captured counter value.
+0x6	tm_prescaler_a	W	Timer A: Prescaler Initial and load value of the counter/timer.
		R	Timer A: Prescaler Capture Register Captured prescaler value.
+0x8	tm_count_b	W	Timer B: Count Initial and load value of the counter/timer.
		R	Timer B: Count Capture Register

Addr	Name	R/W	Description
+0xA	tm_prescaler_b	W	Timer B: Prescaler Initial and load value of the counter/timer
		R	Timer B: Prescaler Capture Register Captured prescaler value.
+0xC	tm_config	r/W	Configuration Register [1:0]: Counter A mode '00': free running '01': one-shot '1x': counter [3:2]: Counter B mode '00': free running '01': one-shot '1x': counter [7:4]: Ext Clock Prescaler '0000': by-pass '0001': divide by 2 '0010': divide by 4 ... '01111': divide by 128 '1xxx': divide by 256 [8]: Counter A clock source '0': System Clock '1': External clock: clk_ext

3.2.1 Operation

For proper operation, following procedure should be observed:

Use following procedure to change the configuration of the TIMERmodule:

1. Disable the timer: timer_ebl = '0'
2. Set the new configuration
3. Enable the timer: timer_ebl = '1'

Changing the configuration while the TIMERmodule is enabled results in unpredictable results and is not recommended.

4 Timing Diagram

Timing numbers depend on the chosen device, synthesis options, place&route constraints. Typical numbers will be provided.

4.1 Local Bus Interface

The microprocessor interface is designed to operate on a full synchronous bus with zero wait-states. The internal registers are selected using cs_n, wr_n or rd_n respectively and addr. The selected register is written on the rising edge of the system clock. data_out is asynchronously generated.

Following figures shows the interface timing diagram.

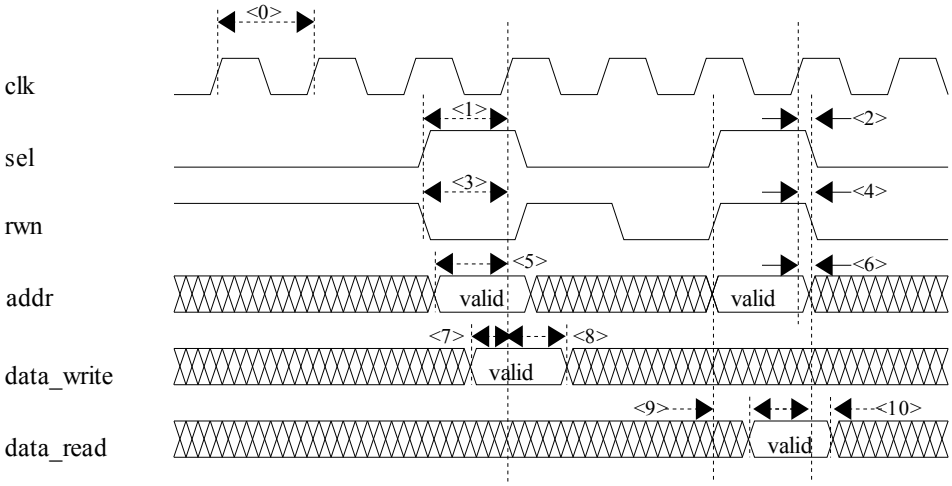

Figure 4: Local Bus Interface Timing

The timing numbers are technology dependent and can only be given once a target technology is selected.

Time nr	t_{min} *[ns]*	t_{typ} *[ns]*	t_{max} *[ns]*	*Comment*
<0>	tbd			Clock period
<1>		tbd		sel setup time
<2>		0		sel hold time
<3>		tbd		rwn setup time
<4>		0		rwn hold time
<5>		tbd		addr setup time
<6>		0		addr hold time
<7>		tbd		data_write setup time
<8>		0		data_write hold time
<9>		tbd		data_read valid after sel, rwn, addr valid
<10>		0		data_read hold time after sel, rwn, addr invalid

About Inicore

- FPGA and ASIC Design
- Easy-to-use IP Cores
- System-on-Chip Solutions
- Consulting Services
- ASIC to FPGA Migration
- Obsolete ASIC Replacements

Inicore is an experienced system design house providing FPGA / ASIC and SoC design services. The company's expertise in architecture, intellectual property, methodology and tool handling provides a complete design environment that helps customers shorten their design cycle and speed time to market. Our offering covers feasibility study, concept analysis, architecture definition, code generation and implementation. When ready, we deliver you a FPGA or take your design to an ASIC provider, whatever is more suitable for your unique solution.

Customer Advantages

We offer one-stop shopping for everything from the specifications to the chip or module solution. Our experience and fast turnaround time reduces your development costs and increases your returns from the market. Your system is not limited by the level of expertise and standard chip solutions you happen to have in-house. Achieve market success by differentiating and optimizing your product. Reusability builds the basis for further developments in the ever-decreasing product life cycle.

Visit us @ www.inicore.com