OMAP5912 Multimedia Processor
Timers
Reference Guide

Literature Number: SPRU759B
October 2004

IMPORTANT NOTICE

Texas Instruments Incorporated and its subsidiaries (TI) reserve the right to make corrections, modifications, enhancements, improvements, and other changes to its products and services at any time and to discontinue any product or service without notice. Customers should obtain the latest relevant information before placing orders and should verify that such information is current and complete. All products are sold subject to TI's terms and conditions of sale supplied at the time of order acknowledgment.

TI warrants performance of its hardware products to the specifications applicable at the time of sale in accordance with TI's standard warranty. Testing and other quality control techniques are used to the extent TI deems necessary to support this warranty. Except where mandated by government requirements, testing of all parameters of each product is not necessarily performed.

TI assumes no liability for applications assistance or customer product design. Customers are responsible for their products and applications using TI components. To minimize the risks associated with customer products and applications, customers should provide adequate design and operating safeguards.

TI does not warrant or represent that any license, either express or implied, is granted under any TI patent right, copyright, mask work right, or other TI intellectual property right relating to any combination, machine, or process in which TI products or services are used. Information published by TI regarding third-party products or services does not constitute a license from TI to use such products or services or a warranty or endorsement thereof. Use of such information may require a license from a third party under the patents or other intellectual property of the third party, or a license from TI under the patents or other intellectual property of TI.

Reproduction of information in TI data books or data sheets is permissible only if reproduction is without alteration and is accompanied by all associated warranties, conditions, limitations, and notices. Reproduction of this information with alteration is an unfair and deceptive business practice. TI is not responsible or liable for such altered documentation.

Resale of TI products or services with statements different from or beyond the parameters stated by TI for that product or service voids all express and any implied warranties for the associated TI product or service and is an unfair and deceptive business practice. TI is not responsible or liable for any such statements.

Following are URLs where you can obtain information on other Texas Instruments products and application solutions:

Products		**Applications**	
Amplifiers	amplifier.ti.com	Audio	www.ti.com/audio
Data Converters	dataconverter.ti.com	Automotive	www.ti.com/automotive
DSP	dsp.ti.com	Broadband	www.ti.com/broadband
Interface	interface.ti.com	Digital Control	www.ti.com/digitalcontrol
Logic	logic.ti.com	Military	www.ti.com/military
Power Mgmt	power.ti.com	Optical Networking	www.ti.com/opticalnetwork
Microcontrollers	microcontroller.ti.com	Security	www.ti.com/security
		Telephony	www.ti.com/telephony
		Video & Imaging	www.ti.com/video
		Wireless	www.ti.com/wireless

Mailing Address: Texas Instruments
Post Office Box 655303 Dallas, Texas 75265

Preface

Read This First

About This Manual

This document describes various timers of the OMAP5912 multimedia processor.

Notational Conventions

This document uses the following conventions.

- Hexadecimal numbers are shown with the suffix h. For example, the following number is 40 hexadecimal (decimal 64): 40h.

Related Documentation From Texas Instruments

Documentation that describes the OMAP5912 device, related peripherals, and other technical collateral, is available in the OMAP5912 Product Folder on TI's website: www.ti.com/omap5912.

Trademarks

OMAP and the OMAP symbol are trademarks of Texas Instruments.

Contents

Figures

Tables

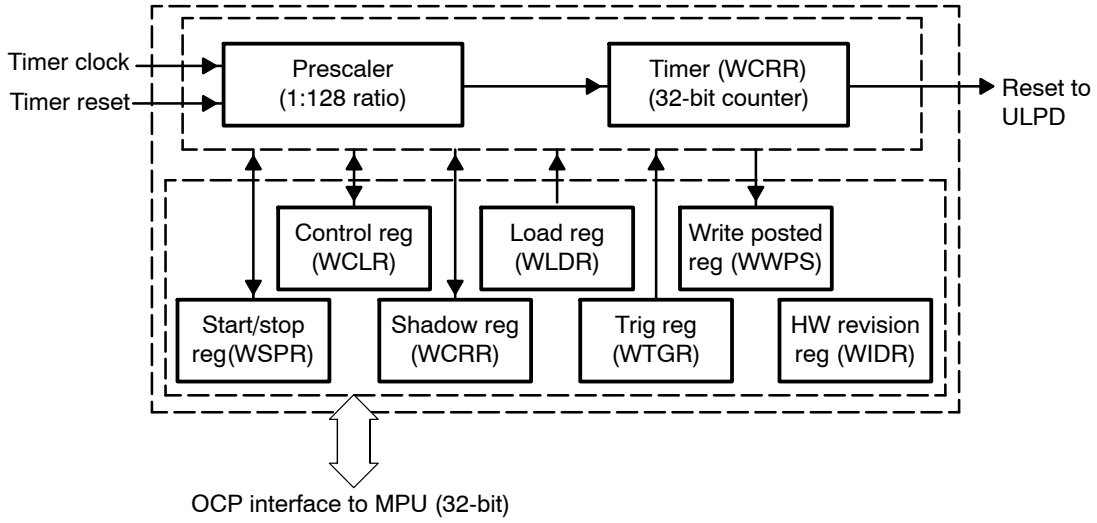
Timers

This document describes various timers of the OMAP5912 multimedia processor.

1 32-Bit Watchdog Timer General Overview

The 32-bit watchdog timer (see Figure 1) is an upward counter that generates a reset to the ultra-low power device (ULPD) module upon an overflow condition. This counter can be accessed, loaded, and cleared by accessing registers through a 16-/32-bit open-core protocol (OCP) interface.

Figure 1. 32-Bit Watchdog



The 32-bit watchdog is an upward 32-bit counter coupled with a prescaler stage. The prescaler ratio can be set between 1 and 128 clock ratios by accessing the PTV field and the PRE field of the watchdog control register (WCLR). Timer value can be accessed on-the-fly by reading the watchdog counter register (WCRR), modified by accessing the watchdog load register (WLDR) (no on-the-fly update), or reloaded by following a specific reload sequence on the watchdog trigger register (WTGR). A specific start/stop sequence applied to the watchdog start/stop register (WSPR) can start/stop the watchdog.

1.1 Posted and Nonposted Writes

There are two clock domains in the watchdog: the timer clock and the interface clock. Table 1 lists these clocks and their frequencies.

Table 1. Watchdog Timer Clock Domains

Device	Timer Clock	Interface Clock
32-kHz watchdog	CLK32K_IN: 32 kHz	MPU peripheral clock: 96 MHz†

† The MPU peripheral clock is typically at half the frequency of the MPU clock frequency.

- The counter and its related registers (WCLR, WCRR, WLDR, WTGR and WSPR registers) are clocked by the timer clock.
- The read/write interface (designated as the OCP interface) which manages the data flow to and from the MPU is clocked by the interface clock. Timer registers WDIR, WD_SYSCONFIG, WD_SYSSTATUS and WWPS are also clocked by the interface clock

For the 32-kHz watchdog timer, any write to a register located in the timer clock domain is done in posted mode.

In effect, the write is acknowledged after it has been resynchronized to the 32-kHz clock. You can perform concurrent writes to the timer clock domain registers and verify that they were acknowledged by reading the status bit in the write posted status register (WWPS). In the case where a register associated write posted status bit is set, a write from the MPU is not acknowledged (see Table 2).

Table 2. Write Posted Status Register (WWPS)

Bit	Name	Function
4	W_PEND_WSPR	When equal to 1, a write is pending to the WSPR register.
3	W_PEND_WTGR	When equal to 1, a write is pending to the WTGR register.
2	W_PEND_WLDR	When equal to 1, a write is pending to the WLDR register.
1	W_PEND_WCRR	When equal to 1, a write is pending to the WCRR register.
0	W_PEND_WCLR	When equal to 1, a write is pending to the WCLR register.

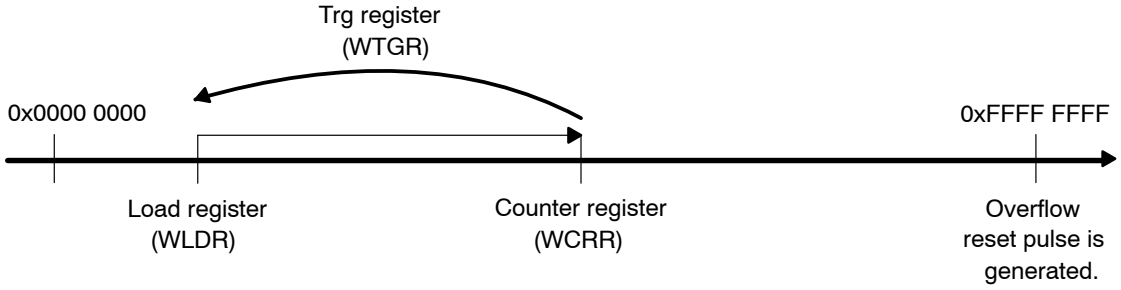
1.2 Reset Context

After reset release, the 32-bit watchdog is on. To get the reset values, software must read the PTV field of WCLR and the 32-bit register to catch the static configuration of the module.

1.3 Overflow/Reset Generation

When the watchdog counter register (WTCR) reaches overflow, an active-low pulse is generated to the ULPD. After reset generation, the counter is automatically reloaded with the watchdog load register (WLDR) and the prescaler counter is reset. The prescaler ratio remains unchanged. Then, after the reset pulse has been generated, the prescaler counter and timer counter are incremented again. Figure 2 shows the 32-bit watchdog functional overview.

Figure 2. 32-Bit Watchdog- General Functional Overview



1.4 Triggering New Reload

To reload the timer counter and reset the prescaler counter values without reaching overflow, a reload command can be executed by accessing the watchdog trigger register (WTGR) using a specific reload sequence.

The specific reload sequence is performed when the written value on the watchdog trigger register (WTGR) is different from its previous value. In this case, reload is executed in the same way as overflow autoreload, without a reset pulse generation. The timer is loaded with the watchdog load register (WLDR) value and the prescaler counter is reset.

1.5 Prescaler Value/Timer Reset Frequency

The 32-bit watchdog is composed of a prescaler stage and a timer counter. The prescaler stage is clocked with the timer clock and acts as a clock divider for the timer counter stage. The ratio can be managed by accessing the ratio definition field of the control register (PTV of the watchdog control register (WCLR)) and enabled with the PRE field of WCLR. Table 3 provides the divisor ratios.

Table 3. Prescaler Clock Ratios

PRE Bit (WCLR)	PTV Bits (WCLR)	Divisor (PS)
0	X	1
1	0	1
1	1	2
1	2	4
1	3	8
1	4	16
1	5	32
1	6	64
1	7	128

The reset period can be estimated by using:

Reset Period = (0xFFFF FFFF – WLDR + 1) x Timer Clock Period x Clock Divider (PS),

where Timer Clock = 1/Timer clock frequency and PS = $2^{(PTV)}$.

For example, if we consider a timer clock input of 32 kHz with a prescaler ratio value of 0x1 (clock divided by 2) and PRE field = 1 (clock divider enabled), the reset period is as shown in Table 4.

Table 4. Reset Period Examples

WLDR Value	Reset Period
0x0000 0000	74 h 56 min
0xFFFF 0000	4 s
0xFFFF FFF0	1 ms
0xFFFF FFFF	62.5 µs

1.6 Accessing Watchdog Registers

All registers are 32-bit wide, accessible via OCP interface with 16-bit or 32-bit OCP access (read/write).

1.7 Modifying WCRR, WLDR, and Prescaler Ratios

To modify the timer counter value (WCRR), prescaler ratio (PTV of WCLR), or load value (WLDR), the 32-bit watchdog must be disabled by using a specific start/stop sequence (WSPR).

In this case, the load register value and prescaler ratio are updated registers, but new values are taken into account only after a new overflow context or a new trigger command (WTGR).

1.8 WCRR Access Restriction

Because the WCRR register is directly related to timer counter value and is updated on the timer clock, a 32-bit shadow register is implemented to read a coherent value of the WCRR.

The shadow register is updated by an 32-bit read command or an 16-bit LSB read command. To be sure that a coherent value is read inside WCRR, use 32-bit reads, or, in case of $2 \times$ 16-bit reads, be sure the least significant 16-bit read is performed first (followed by the most significant 16-bit read).

1.9 Start/Stop Command

To start/stop the 32-bit watchdog, access must be made through the start/stop register (WSPR) using a specific sequence.

To disable the 32-bit watchdog, follow this sequence:

1) Write 0xXXXX AAAA in WSPR.
2) Write 0xXXXX 5555 in WSPR.

To enable the 32-bit watchdog, follow this sequence:

1) Write 0xXXXX BBBB in WSPR.
2) Write 0xXXXX 4444 in WSPR.

Other write sequences on WSPR do not have any effect on the start/stop feature of the module.

1.10 Watchdog Module Under Emulation

During emulation mode, the 32-bit watchdog can/cannot continue to run according to the value of the EMUFREE bit of the system configuration register (WD_SYSCONFIG).

If EMUFREE is 1, timer execution is not stopped in emulation mode and a reset pulse is still generated when overflow is reached.

If EMUFREE is 0, counters (prescaler/timer) are frozen and increment starts again after exit from emulation mode.

1.11 Watchdog Timer Registers

Address of register: Start address + Address offset
Registers width: 32 bits (can be accessed as 32 bits or as 2 x 16 bits – OCP access)

Table 5 lists the 32-bit watchdog timer registers. Table 6 through Table 14 describe the register bits.

Table 5. Watchdog Timer Registers

Base Address = 0xFFFE B000			
Name	**Description**	**R/W**	**Offset**
WIDR	Watchdog HW revision (ID)	R	0x00
RESERVED†	Reserved	-	0x04
RESERVED†	Reserved	-	0x08
RESERVED†	Reserved	-	0x0C
WD_SYSCONFIG	Watchdog system configuration	R/W	0x10
WD_SYSSTATUS	Watchdog status	R	0x14
RESERVED†	Reserved	-	0x18
RESERVED†	Reserved	-	0x1C
RESERVED†	Reserved	-	0x20
WCLR	Watchdog control	R/W	0x24
WCRR	Watchdog counter	R/W	0x28
WLDR	Watchdog load	R/W	0x2C
WTGR	Watchdog trigger	R/W	0x30

† To keep mapping compatibility between 32-bit watchdog and dual-mode timer modules.

Table 5. Watchdog Timer Registers (Continued)

Base Address = 0xFFFE B000			
Name	**Description**	**R/W**	**Offset**
WWPS	Watchdog write pending	R	0x34
RESERVED†	Reserved	-	0x38
RESERVED†	Reserved	-	0x3C
RESERVED†	Reserved	-	0x40
RESERVED†	Reserved	-	0x44
WSPR	Watchdog start/stop	R/W	0x48

† To keep mapping compatibility between 32-bit watchdog and dual-mode timer modules.

Table 6. Watchdog HW Revision (ID) Register (WIDR)

Base Address = 0xFFFE B000, Offset = 0x00				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:8	RESERVED	Reserved	R	0x000000
7:0	WD_REV	Module HW revision number indicates the revision number of the current timer module. This value is fixed in hardware. Least significant 4 bits indicate a minor revision. Most significant 4 bits indicate a major revision. A reset has no effect on the value returned.	R	0x10

This read-only, 32-bit register (accessible in 16-bit mode) contains the hardware revision number of the module. A write to this register has no effect.

Table 7. Watchdog System Configuration Register (WD_SYSCONFIG)

Base Address = 0xFFFE B000, Offset = 0x10				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:6	RESERVED	Write 0s for future compatibility Read returns 0	R/W	0x0000000
5	EMUFREE	Enable sensitivity to suspend signal (emulation) 0: The module freezes its internal logic upon suspend assertion. 1: The module ignores the suspend input.	R/W	0
4:2	RESERVED	Write 0s for future compatibility Read returns 0	R/W	0x0
1	SOFTRESET	Software reset. Set this bit to 1 to trigger a module reset. This bit is automatically reset by hardware. During read, it always returns 0. 0: Normal mode 1: The module is reset.	R/W	0
0	AUTOIDLE	Internal OCP gating strategy 0: Interface clock is free-running. 1: Interface clock gating is active.	R/W	0

Table 8. Watchdog System Status Register (WD_SYSSTATUS)

Base Address = 0xFFFE B000, Offset = 0x14				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:8	RESERVED	Reserved	R	0x0000000
7:1	RESERVED	Reserved	R	0x00
0	RESETDONE	Internal reset monitoring 0: Internal module reset is ongoing. 1: Reset completed	R	0

This register provides system status information about the module.

Table 9. Watchdog Control Register (WCLR)

Base Address = 0xFFFE B000, Offset = 0x24				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:6	RESERVED	Reserved	R/W	0x00000
5	PRE	0: The timer clock input pin clocks the counter. 1: The divided input pin clocks the counter.	R/W	1
4:2	PTV	Prescaler ratio value:	R/W	0x0
0:1	RESERVED	Reserved	R/W	0x0

This 32-bit register is accessible in 16-bit mode.

Table 10. Watchdog Counter Register (WCRR)

Base Address = 0xFFFE B000, Offset = 0x28				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:0	TIME_COUNTER	Value of timer counter	R/W	0xFFF0 0000

This 32-bit register is accessible in 16-bit mode.

The WCRR register is a 32-bit register (16-bit addressable). The MPU can perform either a 32-bit access or two 16-bit accesses to the register, whereas the DSP performs two consecutive 16-bit transactions. Because the interface clock is completely asynchronous with the timer clock, some synchronization is done to ensure that the WCRR value is not read while it is incremented.

In 16-bit-mode, the following sequence must be followed to read the WCRR register properly:

1) Read the lower 16 bits of the WCRR register (offset = 0x08). When the WCRR is read, the lower 16 bits (LSBs) are read, and the upper 16 bits (MSW) of the WCRR register are stored in a temporary register (shadow register).

2) Read the upper 16 bits of the WCRR register (offset = 0x0A). The value of the upper 16 bits (MSW), which has been stored in the temporary register, is read.

To read the value of WCRR correctly, the first OCP read access must be to the lower 16-bits (that is, offset = 0x08), followed by OCP read access to the upper 16-bits (that is, offset = 0x0A).

WCRR value can be modified by writing on WCRR register and only if watchdog has been disabled (using WSPR register).

Table 11. Watchdog Load Register (WLDR)

Base Address = 0xFFFE B000, Offset = 0x2C				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:0	TIME_LOAD	Timer counter value loaded on overflow in autoreload mode or on WTGR write access: 32-kHz watchdog	R/W	0xFFF0 0000

This 32-bit register is accessible in 16-bit mode.

This register specifies the load value of the timer counter. This load value is effective (loaded inside WCRR) after an overflow context or by triggering a new reload via WTGR register.

Table 12. Watchdog Trigger Register (WTGR)

Base Address = 0xFFFE B000, Offset = 0x30				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:0	TTGR_VALUE	Trigger value. Triggers a reload of the watchdog module by writing a value different from the previous value inside WTGR	R/W	0x0000 0000

This 32-bit register is accessible in 16-bit mode.

A write to the watchdog trigger register (WTGR) reloads the value contained in the WLDR register value to the watchdog counter register (WCRR).

This command is performed by writing a value inside WTGR different from the previous value inside WTGR.

Table 13. Watchdog Write Pending Register (WWPS)

Base Address = 0xFFFE B000, Offset = 0x34				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:5	RESERVED	Reserved		0x000000
4	W_PEND_WSPR	When equal to one, a write is pending to the WSPR register.	R	0
3	W_PEND_WTGR	When equal to one, a write is pending to the WTGR register.	R	0
2	W_PEND_WLDR	When equal to one, a write is pending to the WLDR register.	R	0

Table 13. Watchdog Write Pending Register (WWPS) (Continued)

Base Address = 0xFFFE B000, Offset = 0x34				
Bit	**Name**	**Function**	**R/W**	**Reset**
1	W_PEND_WCRR	When equal to one, a write is pending to the WCRR register.	R	0
0	W_PEND_WCLR	When equal to one, a write is pending to the WCLR register.	R	0

This 32-bit register is accessible in 16-bit mode.

In write-posting mode, the software must read the pending write status bits (watchdog write-posted status register bits [4:0]) to ensure that the following write access is not discarded because of an ongoing write synchronization process. These bits are automatically cleared by internal logic when the write to the corresponding register is acknowledged.

Table 14. Watchdog Start/Stop Register (WSPR)

Base Address = 0xFFFE B000, Offset = 0x48				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:0	WSPR_VALUE	WSPR value switches on/off watchdog module with specific protocol.	R/W	0x0000 0000

This 32-bit register is accessible in 16-bit mode.

The 32-bit watchdog (timer counter + prescaler counter) can be started/stopped by accessing WSPR with a specific start/stop sequence. To disable the 32-bit watchdog, follow this sequence:

- ❏ Write 0x XXXX AAAA in WSPR.
- ❏ Write 0x XXXX 5555 in WSPR.

To enable the 32-bit watchdog, follow this sequence:

- ❏ Write 0x XXXX BBBB in WSPR
- ❏ Write 0x XXXX 4444 in WSPR.

A read on this register (WSPR) returns the last data written in WSPR.

2 32-kHz Watchdog Timer

The software can access the 32-kHz watchdog at any time.

All writes to the watchdog register file are write posted, so the watchdog timer write command is granted before the actual write in the timer clock domain is performed.

This mode allows software to perform concurrent writes on 32-bit watchdog registers and to manage them at software level by reading the status of posted writes (by reading status bit of the write posted status register (WWPS)).

- ❏ The prescaler value is forced to 1 at power up reset.
- ❏ The WLDR value is forced to 0xFFF00000 at power up reset.
- ❏ The timer input clock is 32 kHz.

Table 15. Time-Out for 32-kHz Watchdog Timer

Clock Frequency (kHz)	**32**
PTV	0
WLDR value	0xFFF0 0000
Reset period(s)	33
Default Configuration at Power-Up Reset	
PTV	0
WLDR value	0xFFF0 0000

The time-out value generates a reset that is logged into the ULPD reset status register (Bit [3]).

After power-up reset, the time-out period can be changed by programming either the prescaler (PTV) or the timer-load value (WLDR).

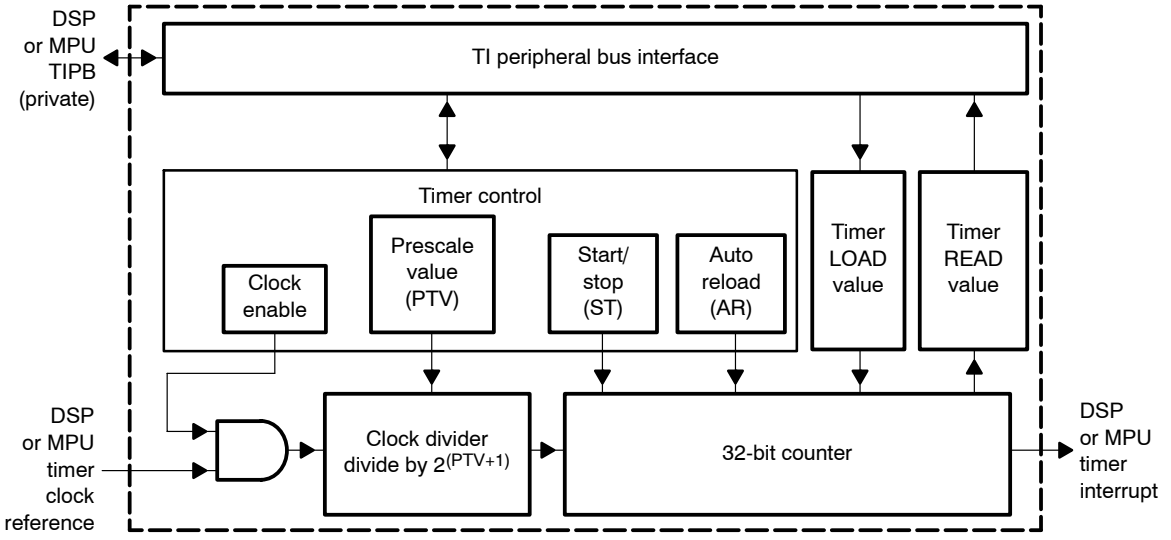
3 OMAP3.2 Operating System Timer

There are six timers for the OS and general-purpose housekeeping functions inside the hardware engine. Three are in the MPU subsystem (controlled by the MPU), and the other three are controlled by the DSP._Figure 3 shows the timer block diagram.

The functionality for all timers is identical, except the MPU timer interface to the 32-bit MPU private TI peripheral bus (TIPB) and the DSP timer interface to the 16-bit DSP private TIPB.

The timer is configured either in autoreload or in one-shot mode with on-the-fly read capability. The timer generates an interrupt to the MPU or DSP when the value is equal to zero.

Figure 3. Timer Block Diagram



3.1 DSP and MPU OS Timer Start and Stop

An OS timer can be started by setting the ST bit of the DSP_CNTL_TIMER or the MPU_CNTL_TIMER register. At start, the timers are loaded with the values programmed to the DSP_LOAD_TIMER_HI, DSP_LOAD_TIMER_LO, and MPU_LOAD_TIMER registers. The timer is stopped by resetting the ST bit. When stopped, the timer value is frozen at the current value.

3.2 DSP and MPU OS Timer

If the autoreload (AR) bit is set in DSP_CNTL_TIMER or MPU_CNTL_TIMER, a new value (from the load register) is loaded into the timer when it passes through zero.

If the AR bit is reset, the timer decrements from the loaded value to zero and then stops.

3.3 Reading OS Timer Values

The timer values can be read from the respective DSP or MPU read timer registers either on-the-fly or after the timer is stopped.

- For the MPU, MPU_READ_TIMER is a 32-bit register and can be read directly.
- By contrast, the data width of the DSP is only 16 bits. To read the value of the DSP OS timers correctly, the first access must be to the upper 16 bits.

  When the upper read occurs, the lower 16 bits are simultaneously stored in a temporary register. An access to the lower 16 bits provides the content of this temporary register as the read result. A read from DSP_READ_TIMER_HI followed by a read from DSP_READ_TIMER_LO provides the correct DSP OS timer read result.

3.4 DSP and MPU OS Timer Input Clocks

This section describes clocking for the DSP and MPU OS timers.

3.4.1 Configuration of the Input Clock for DSP 32-Bit OS Timers

The input clock reference for the DSP timer modules is controlled from the clock generation and reset module. Programming the TIMXO bit of the DSP prescaler selection register DSP_CKCTL in the clock module selects between two possible clock sources:

- Programming the TIMXO bit to 0 selects the main input reference clock as the clock reference for the DSP timers.
- Programming the TIMXO bit to 1 selects the output from the DPLL1 module as the source for the DSP 32-bit OS timers.

The default is referenced from DPLL1. When operating from DPLL1, it is recommended to stop the DSP OS timer before programming a change to the DPLL1 frequency divisor. See *Clock Generation and Reset Management* in the Multimedia Processor OMAP 3.2 Subsystem Reference Guide (SPRU749) for details on DSP_CKCTL and on programming the DPLL1 module.

3.4.2 Configuration of the Input Clock for MPU 32-Bit OS Timers

The input clock reference for the MPU timer modules is controlled from the clock generation and reset module. Programming the ARM_TIMXO bit of the MPU prescaler selection register ARM_CKCTL: in the clock module selects between two possible clock sources.

- Programming the ARM_TIMXO bit to 0 selects the main input reference clock as the clock reference for the MPU timers.
- Programming the ARM_TIMXO bit to 1 selects the output from the DPLL1 module as the source for the MPU 32-bit OS timers.

The default is referenced from DPLL1. When operating from DPLL1, it is recommended to stop the MPU OS timer before programming a change to the DPLL1 frequency divisor. See *Clock Generation and Reset Management* in the Multimedia Processor OMAP 3.2 Subsystem Reference Guide (SPRU749) for details of ARM_CKCTL: and on programming the DPLL1 module.

3.4.3 Input Clock Enable

Input reference clocks for the DSP and MPU OS timer modules are ANDed with a clock enable signal to gate the clock from timer internal logic. The clock is enabled to the timers when the CLOCK_ENABLE bit is set in DSP_CNTL_TIMER and in MPU_CNTL_TIMER.

3.5 Timer Interrupts

An interrupt occurs when the corresponding timer decrements to zero.

DSP and MPU OS timer interrupt periods are defined by:

1) The programmed value for the DSP and MPU OS timer input clock references (see Section 3.4.1 and Section 3.4.2*).*
2) The value of the prescaler bit field, PTV, in DSP_CNTL_TIMER and in MPU_CNTL_TIMER. Table 16 lists valid entries for PTV.
3) The value of the load registers, DSP_LOAD_TIMER_HI, DSP_LOAD_TIMER_LO, and MPU_LOAD_TIMER.

Table 16. Valid Prescaler Values for OS Timer Configuration

PTV Bit Field in DSP and MPU OS Timer Control Register	Input Clock Divisor
000	2
001	4
010	8
011	16
100	32
101	64
110	128
111	256

The following equations are used to determine the timer interrupt periods.

- For the DSP OS timer:

$$T_{DSP\_interrupt} = T_{DSP\_ref\_clk} \times (< DSP\_LOAD\_TIMER\_HI, LO > + 1) \times 2^{(PTV+1)}$$

- For the MPU OS timer

$$T_{MPU\_interrupt} = T_{MPU\_ref\_clk} \times (< MPU\_LOAD\_TIMER > + 1) \times 2^{(PTV+1)}$$

Where $T_{DSP\_ref\_clk}$ and $T_{MPU\_ref\_clk}$ are clock periods of the DSP and MPU OS timer input clocks.

Based on these equations, examples for various MPU OS timer interrupt periods are calculated in Table 17 for a hypothetical MPU OS timer input clock reference frequency of 100 MHz ($T_{MPU\_ref\_clk}$ = 10 ns).

Table 17. Example MPU OS Timer Interrupt Periods

MPU_LOAD_TIMER	PTV = 000	PTV = 111
0x0 (granularity)	20 ns	2.56 µs
0xFFFFFFFF(max period)	85.9 s	10995 s (3 hr 3 min 15 s)

3.6 Timer Programming

To activate the DSP and MPU OS timers, perform the following sequence:

1) Program the load register.

   The content of the load register is loaded into the timer read register when the ST bit is set.

2) Write to DSP_CNTL_TIMER or to MPU_CNTL_TIMER to configure and start the timer.

   - Set the PTV value as the dividing factor for the timer clock.
   - Set the autoreload feature with the AR bit.
   - Set the ST bit to start the timer.

3) You can stop the timer at any time by resetting the ST bit (bit 0) of the DSP_CNTL_TIMER or MPU_CNTL_TIMER.

Note:

Only the ST or CLOCK_ENABLE bits of DSP_CNTL_TIMER or MPU_CNTL_TIMER can be written while the timer is running. Undefined results occur if the prescaler (PTV) or autoreload (AR) bits in the control register or timer load registers are written while the timer is running.

3.7 OS Timer Registers

The following sections describe the DSP and MPU operating system timer registers.

Note:

The register descriptions and base address offsets given here apply identically to each of the three DSP or three MPU OS timers. See the Applications Processor Data Manual (SPRS231), to determine the base address for each timer.

3.7.1 DSP 32-Bit OS Timers

Accesses to the DSP 32-bit OS timer registers are controlled by the DSP private TIPB. Table 18 lists the DSP OS timer registers. Table 19 through Table 23 provide register bit descriptions.

Table 18. DSP OS Timer Registers

Base Address = 0xE100 5000, 0xE100 5800, 0xE100 6000			
Name	**Description**	**R/W**	**Offset**
DSP_CNTL_TIMER	DSP control timer	R/W	0x00
DSP_LOAD_TIMER_LO	DSP load timer value, 1/2 LSW	W	0x04
DSP_LOAD_TIMER_HI	DSP load timer value, 1/2 MSW	W	0x06
DSP_READ_TIMER_LO	DSP read timer value, 1/2 LSW	R	0x08
DSP_READ_TIMER_HI	DSP read timer value, 1/2 MSW	R	0x0A

Table 19. DSP Control Timer Register (DSP_CNTL_TIMER)

Base Address = 0xE100 5000, 0xE100 5800, 0xE100 6000, Offset = 0x00				
Bit	**Name**	**Function**	**R/W**	**Reset**
15:7	RESERVED			0x000
6	FREE	Specifies what action the timer takes upon receiving a suspend indication from the TIPB bridge. (For example, suspend is indicated if processor execution has been suspended at an emulation breakpoint.) 0: Stop counting on suspend indication (even when ST=1). 1: A suspend indication has no effect on the count.	R/W	0
5	CLOCK_ENABLE	Enables input reference clock to the DSP OS timer module 0: Disable 1: Enable	R/W	0
4:2	PTV	Prescale timer input reference clock 000: Divide input reference clock by 2 001: Divide input reference clock by 4 010: Divide input reference clock by 8 011: Divide input reference clock by 59120: Divide input reference clock by 32 101: Divide input reference clock by 64 110: Divide input reference clock by 128 111: Divide input reference clock by 256	R/W	000

Table 19. DSP Control Timer Register (DSP_CNTL_TIMER) (Continued)

Base Address = 0xE100 5000, 0xE100 5800, 0xE100 6000, Offset = 0x00				
1	AR	0: One-shot mode	R/W	0
		1: Autoreload mode		
0	ST	0: Stop timer value decrement	R/W	0
		1: Start timer value decrement		
		If one-shot mode is selected (AR = 0), this bit is automatically reset by internal logic when the timer value is equal to 0.		

Table 20. DSP Load Timer Value, 1/2 LSW (DSP_LOAD_TIMER_LO)

Base Address = 0xE100 5000, 0xE100 5800, 0xE100 6000, Offset = 0x04				
Bit	**Name**	**Function**	**R/W**	**Reset**
15:0	DSP_LOAD_TIMER_HI	This value is loaded when the timer passes through 0 or when it starts.	W	Undefined

Table 21. DSP Load Timer Value, 1/2 MSW (DSP_LOAD_TIMER_HI)

Base Address = 0xE100 5000, 0xE100 5800, 0xE100 6000, Offset = 0x06				
Bit	**Name**	**Function**	**R/W**	**Reset**
15:0	DSP_LOAD_TIMER_LO	This value is loaded when the timer passes through 0 or when it starts.	W	Undefined

Table 22. DSP Read Timer Value, 1/2 LSW (DSP_READ_TIMER_LO)

Base Address = 0xE100 5000, 0xE100 5800, 0xE100 6000, Offset = 0x08				
Bit	**Name**	**Function**	**R/W**	**Reset**
15:0	DSP_READ_TIMER_LO	Value of the timer bits (15:0): To read a correct value for DSP 32-bit OS timer, the upper 16 bits (from DSP_READ_TIMER_HI) must read prior to reading this register.	R	Undefined

Table 23. DSP Read Timer Value, 1/2 MSW (DSP_READ_TIMER_HI)

Base Address = 0xE100 5000, 0xE100 5800, 0xE100 6000, Offset = 0x0A				
Bit	**Name**	**Function**	**R/W**	**Reset**
15:0	DSP_READ_ TIMER_HI	Value of the timer bits (31:16): To read correct value for DSP 32-bit OS timer, this register must be read first, followed by lower 16 bits of DSP_READ_TIMER_LO.	R	Undefined

3.7.2 MPU 32-Bit OS Timers

Accesses to the MPU 32-bit OS timer configuration registers are controlled by the MPU private TIPB.

Table 24 lists the MPU OS timer registers. Table 25 through Table 27 provide register bit descriptions.

Table 24. MPU OS Timer Registers

Base Address = 0xFFFE C500, 0xFFFE C600, 0xFFFE C700			
Name	**Description**	**R/W**	**Offset**
MPU_CNTL_TIMER	MPU control timer	R/W	0x00
MPU_LOAD_TIMER	MPU load timer	W	0x04
MPU_READ_TIMER	MPU read timer	W	0x08

Table 25. MPU Control Timer Register (MPU_CNTL_TIMER)

Base Address = 0xFFFE C500, 0xFFFE C600, 0xFFFE C700, Offset = 0x00				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:7	RESERVED			0x0000000
6	FREE	Specifies what action the timer takes upon receiving a suspend indication from the TIPB bridge. (For example, suspend is indicated if processor execution has been suspended at an emulation breakpoint.) 0: Stop counting on suspend indication (even when ST = 1). 1: Suspend indication has no effect on the count.	R/W	0

Table 25. MPU Control Timer Register (MPU_CNTL_TIMER) (Continued)

Base Address = 0xFFFE C500, 0xFFFE C600, 0xFFFE C700, Offset = 0x00				
Bit	**Name**	**Function**	**R/W**	**Reset**
5	CLOCK_ENABLE	Enable input reference clock to the MPU OS timer module	R/W	0
4:2	PTV	Prescale timer input reference clock	R/W	000
1	AR	0: One-shot mode 1: Autoreload mode	R/W	0
0	ST	0: Stop timer value decrement 1: Start timer value decrement If one-shot mode is selected (AR = 0), this bit is automatically reset by internal logic when the timer value is equal to 0.	R/W	0

Table 26. MPU Load Timer Register Value (MPU_LOAD_TIMER)

Base Address = 0xFFFE C500, 0xFFFE C600, 0xFFFE C700, Offset = 0x04				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:0	MPU_LOAD_TIMER	This value is loaded when the timer passes through 0 or when it starts.	W	UD

Table 27. MPU Read Timer Register Value (MPU_READ_TIMER)

Base Address = 0xFFFE C500, 0xFFFE C600, 0xFFFE C700, Offset = 0x08				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:0	MPU_READ_TIMER	Value of the timer	R	UD

4 Dual-Mode Timer

This peripheral is a 32-bit timer with the following features:

- Counter timer with compare and capture modes
- Autoreload mode
- Start-stop mode
- Programmable divider clock source
- 16-/32-bit addressing
- On the fly read/write registers
- Interrupts generated on overflow, compare, and capture
- Interrupt enable
- Wake-up enable
- Write posted mode
- Dedicated input trigger for capture mode and dedicated output trigger/PWM signal

The timer module contains a free-running upward counter with autoreload capability on overflow. The timer counter can be read and written on the fly (while counting). The timer module includes compare logic to allow interrupt event on programmable counter matching value. A dedicated output signal can be pulsed or toggled on overflow and match event. This offers timing stamp trigger signal or PWM (pulse width modulation) signal sources. A dedicated input signal can be used to trigger automatic timer counter capture and interrupt event, on programmable input signal transition type. A programmable clock divider (prescaler) allows reduction of the timer input clock frequency. All internal timer interrupt sources are merged into one module interrupt line and one wake-up line. Each internal interrupt source can be independently enabled/disabled with a dedicated bit of the TIER register for the interrupt features and a dedicated bit of TWER for the wake-up.

For each timer implemented in OMAP5912, there are three possible clock sources:

- The 32-kHz clock
- The system clock
- An external clock source

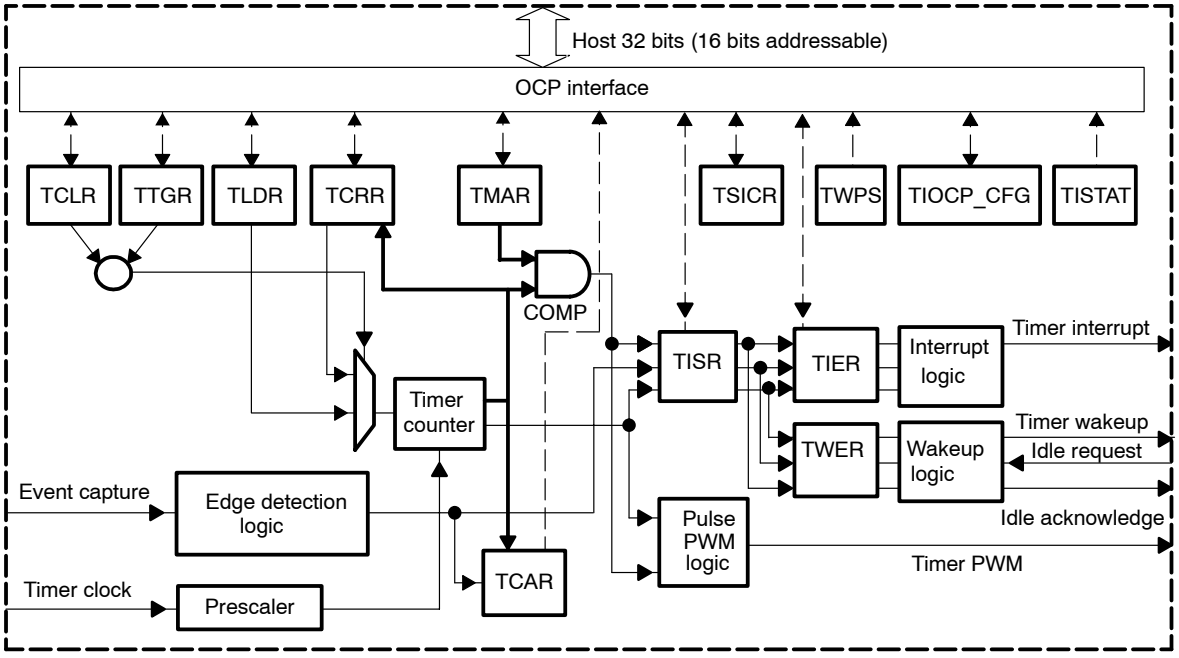
The selection is done in configuration. Refer to the MOD_CONF_CTRL_1 description for additional information.

Table 28 lists the access types for various clock selections, and Figure 4 shows the dual-mode timer.

Table 28. Access Types (Posted Mode vs Nonposted Mode) for Various Clock Selections

Timer Clock Selection	CONF_MOD_GPTIMER_CLK_SEL = 00		CONF_MOD_GPTIM-ER_CLK_SEL = 10	CONF_MOD_GPTIM-ER_CLK_SEL = 01
Timer clock frequency	System clock = 12 MHz	System clock = 19.2 MHz	External timer clock = 20 MHz	32 kHz
Access type	Posted mode	Nonposted mode	Nonposted mode	Posted mode
Action to be taken by software before next write to timer	Software must read the pending write status bits (timer write posted status register bits [4:0]) to ensure that following write access is not discarded because of ongoing write synchronization process.	None	None	Software must read the pending write status bits (timer write posted status register bits [4:0]) to ensure that following write access is not discarded because of ongoing write synchronization process.

Figure 4. Dual-Mode Timer Block Diagram



4.1 Description

The general-purpose timer is an upward counter. It supports three functional modes:

- Timer mode
- Capture mode
- Compare mode

By default, the capture and compare modes are disabled after core reset.

4.2 Mode Functionality

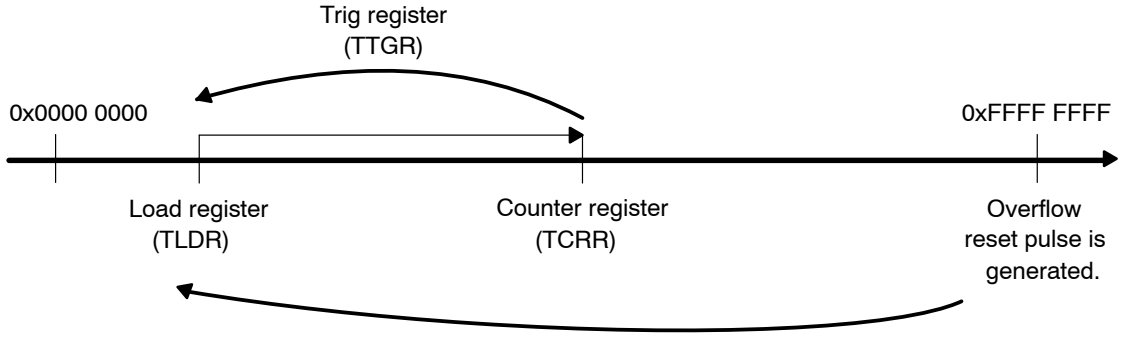
The timer is an upward counter that can be started and stopped at any time through the timer control register (TCLR ST bit). The timer counter register (TCRR) can be loaded when stopped or on the fly (while counting). TCRR can be loaded directly by a TCRR write access with the new timer value. TCRR can also be loaded with the value held in the timer load register TLDR by a trig register (TTGR) write access. The TCRR loading is done regardless of the TTGR written value. The timer counter register TCRR value can be read when stopped or captured on the fly by a TCRR read access. The timer is stopped and the counter value set to 0 when the modules reset is asserted. The timer is maintained in stop after reset is released. When the timer is stopped, TCRR is frozen and it can be restarted from the frozen value, unless TCRR has been reloaded with a new value.

In one-shot mode (TCLR AR bit =0), the counter is stopped after counting overflow (counter value remains at zero).

When autoreload mode is enabled (TCLR AR bit =1), the TCRR is reloaded with the timer load register (TLDR) value after a counting overflow. An interrupt can be issued on overflow if the overflow interrupt enable bit is set in the timer interrupt enable register (TIER OVF_IT_ENA bit =1). A dedicated output pin (timer PWM) can be programmed through TCLR (TRG and PT bits) to generate one positive pulse (prescaler duration) or to invert the current value (toggle mode) when an overflow occurs.

Figure 5 shows the TCRR timing value.

Figure 5. TCRR Timing Value



4.3 Capture Mode Functionality

The timer value in TCRR can be captured and saved in TCAR when a transition is detected on the module input pin (event capture).

Rising edge, falling edge, or both can be selected in TCLR (TCM) to trigger the timer counter capture. The module sets the TISR (TCAR_IT_FLAG bit) when an active edge is detected; at the same time the counter value TCRR is stored in the timer capture register TCAR. If there are several consecutive capture events, the first one is saved in TCAR.

The edge detection logic is reset when either the TCAR_IT_FLAG is cleared by writing a 1 to it, or when the TCM bits in the TCLR register transition from the no-capture mode detection to any other mode. The timer functional clock (input to prescaler) is used to sample the input pin (event capture). Input negative or positive pulses can be detected when pulse time is greater than the functional clock period. An interrupt is issued on edge detection if the capture interrupt enable bit is set in the timer interrupt enable register TIER (TCAR_IT_ENA bit).

4.4 Compare Mode Functionality

When the compare enable TCLR (CE bit) is set to 1, the timer value (TCRR) is continuously compared to the value held in the timer match register (TMAR). TMAR value can be loaded at any time (timer counting or stopped). When the TCRR and the TMAR values match, an interrupt is issued if the TIER (MAT_IT_ENA bit) is set.

The dedicated output pin (timer PWM) can be programmed through TCLR (TRG and PT bits) to generate one positive pulse (timer clock duration) or to invert the current value (toggle mode) when an overflow and a match occur.

4.5 Prescaler Functionality

A prescaler can be used to divide the timer counter input clock frequency. The prescaler is enabled when TCLR bit 5 is set (PRE). The 2^n division ratio value (PTV) can be configured in the TCLR register.

The prescaler counter is reset when the timer counter is stopped or reloaded on the fly.

Table 29 lists the prescaler/timer reload values versus contexts.

Table 29. Prescaler/Timer Reload Values Versus Contexts

Contexts	Prescaler Counter	Timer Counter
Overflow (when autoreload on)	Reset	TLDR
TCRR write	Reset	TCRR
TTGR write	Reset	TLDR
Stop	Reset	Frozen

4.6 Pulse-Width Modulation

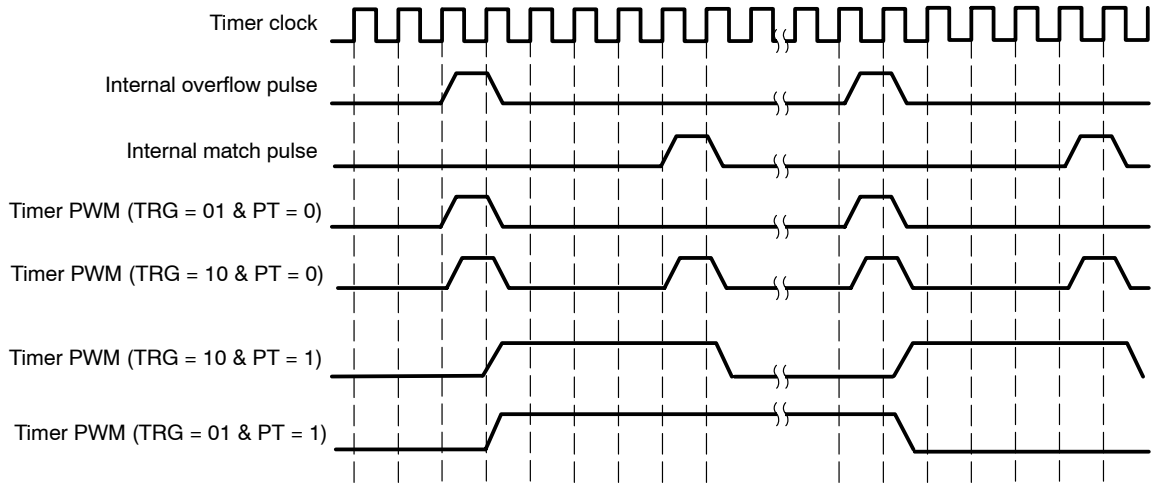
The timer can be configured to provide a programmable pulse-width modulation (timer PWM) output. The timer PWM output pin can be configured to toggle on an event. TCLR (TRG bits) determine on which register value the timer PWM pin toggles. Either overflow or match can be used to toggle the timer PWM pin when a compare condition occurs. The TCLR (SCPWM bit) can only be programmed to set or clear the timer PWM output signal while the counter is stopped or the trigger is off. This allows the output pin to be set to a known state before modulation starts. The modulation is synchronously stopped when TRG bit is cleared and overflow occurred. This allows the output pin to be set to a known state when modulation is stopped.

In Figure 6, the internal overflow pulse is set each time the (0xFFFF FFFF – TLDR +1) value is reached, and the internal match pulse is set when the counter reaches TMAR register value. According to TCLR (TRG and PT bits) value, the timer provides pulse or PWM on the output pin (timer PWM).

In Figure 6 TCLR (SCPWM bit) is set to 0.

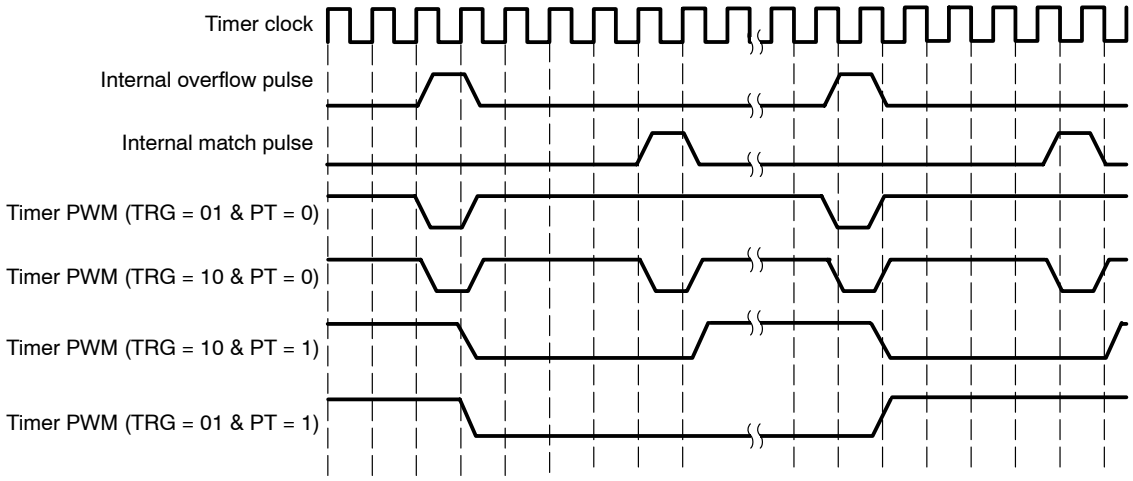
The TLDR and TMAR registers must keep values smaller than the overflow value (0xFFFFFFFF) with at least two units. In case the PWM trigger events are both overflow and match, the difference between the values kept in TMAR register and the value in TLDR must be at least two units. When match event is used, the compare mode TCLR (CE) must be set.

Figure 6. Timing Diagram of Pulse-Width Modulation With SCPWM Bit = 0



In Figure 7 TCLR (SCPWM bit) is set to 1.

Figure 7. Timing Diagram of Pulse-Width Modulation With SCPWM Bit = 1



4.7 Timer Interrupt Control

The timer can issue an overflow interrupt, a timer match interrupt, and a timer capture interrupt. Each internal interrupt source can be independently enabled/disabled in the interrupt enable register TIER. When the interrupt event has been issued, the associated interrupt status bit is set in the timer status register (TISR). The pending interrupt event is reset when the set status bit is overwritten by a 1. Reading the interrupt status register and writing the value back allows fast interrupt acknowledge.

4.8 Sleep Mode Request and Acknowledge

Upon a sleep mode request issued by the host processor, the timer module goes into sleep mode depending on the IDLEMODE field of the system configuration register (see TIOCP_CFG).

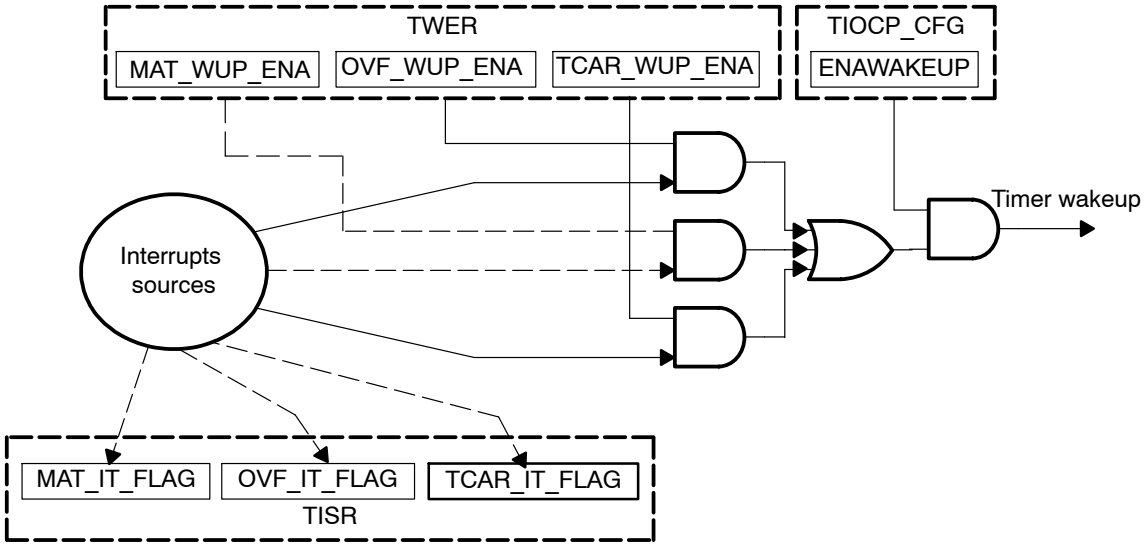
If the IDLEMODE field sets no-idle mode, the timer does not go into sleep mode and the idle acknowledge signal is never asserted.

If the IDLEMODE field sets force-idle mode, the timer goes into sleep mode independently of the internal module state, and the idle acknowledge signal is unconditionally asserted.

If the IDLEMODE field sets smart-idle mode, the timer module evaluates its internal capability to have the interface clock switched off. Once there is no more internal activity (no pending interrupt sources: match, overflow, or timer capture events), the idle acknowledge signal is asserted and the timer enters into sleep mode, ready to issue a wake-up request. This wake-up request is sent only if the field ENAWAKEUP of TIOCP_CFG enables the timer wake-up capability.

Figure 8 shows the wake-up request generation.

Figure 8. Wake-Up Request Generation



4.8.1 Wake-up Line Release

When the host processor receives a wake-up request issued by the timer peripheral, the interface clock is reactivated, the host processor deactivates the idle request signal, the timer deactivates the idle acknowledge signal, and the host then can read the corresponding bit in TISR to find out which interrupt source has triggered the wake-up request. After acknowledging the wake-up request, the processor resets the status bit and releases the interrupt line by writing a 1 in the corresponding bit of the TISR register.

4.9 Timer Counting Rate

The dual-mode timer is composed of a prescaler stage and a timer counter. The prescaler stage is clocked with the timer clock and acts as a clock divider for the timer counter stage.

The timer rate is defined by:

- Value of the prescaler fields (PRE and PTV of TCLR register)
- Value loaded into the timer load register (TLDR)

Table 30 lists prescaler clock ratios values.

Table 30. Prescaler Clock Ratios Values

PRE	PTV	Divisor (PS)
0	X	1
1	0	2
1	1	4
1	2	8
1	3	16
1	4	32
1	5	64
1	6	128
1	7	256

timer rate = (0xFFFF FFFF – TLDR + 1) x timer clock period x clock divider (PS)

With timer clock period = 1/ timer clock frequency and PS = $2^{(PTV + 1)}$.

For example, with a timer clock input of 32 kHz and a PRE field equal to 0, the timer output period is as shown in Table 31.

Table 31. Value and Corresponding Interrupt Period

TLDR	Interrupt Period
0x0000 0000	37 h
0xFFFF 0000	2 s
0xFFFF FFF0	500 µs
0xFFFF FFFF	31.25 µs

4.10 Dual-Mode Timer Under Emulation

During emulation mode, the dual-mode timer continues to run according to the value of the EMUFREE bit of the timer OCP configuration register (TIOCP_CFG).

If EMUFREE is 1, timer execution is not stopped in emulation mode and the interrupt assertion is still generated when overflow is reached.

If EMUFREE is 0, the prescaler and timer are frozen and both resume on exit from emulation mode. The asynchronous input pin is internally synchronized on two timer-clock rising edges.

4.11 Accessing Registers

All registers are 32 bits wide, accessible via OCP interface with 16-bit or 32-bit OCP access (read/write).

4.12 Programming Timer Registers

The host uses the OCP bus protocol to write the TLDR, TCRR, TIER, TISR, TCLR, TIOCP_CFG, TWER, TTGR, TSICR, and TMAR registers synchronously with the timer interface clock.

Because the timer registers are 32-bit wide, 16-bit wide access mode requires two consecutive write operations (16 LSBs followed by 16 MSBs).

4.13 Reading Timer Registers

In 16-bit access mode, reading the LSBs from the timer counter register (TCRR) captures the current timer counter value. This must be followed by reading the 16 MSBs.

4.14 Writing Timer Registers

4.14.1 Software Write Posting Synchronization Mode

This mode is used if TSCR (POSTED bit) is set to 1 in the timer control register.

It uses a posted write scheme for updating any internal register. This means that the write transaction is immediately acknowledged on the OCP interface, although the effective write operation occurs later, because of a resynchronization in the timer clock domain. The advantage is that neither the interconnect nor the CPU that requested the write transaction is stalled. For each register, a status bit is provided that is set if there is a pending write access to the register.

In this mode, it is mandatory that the CPU check this status bit prior to any write access. In case a write is attempted to a register with a previous access pending, the previous access is discarded without notice.

There is one status bit per register, accessible in the timer write-posted status register.

The timer module updates the timer counter register value synchronously with the OCP clock. Consequently, any read access to the timer counter register does not add any resynchronization latency; the current value is always available.

If a write access is pending for a register, reading from this register does not yield a correct result. Software synchronization must be used to avoid incorrect results.

The drawback of this automatic update mechanism is that it assumes a given relationship between the timer interface frequency and the timer clock frequency:

Functional frequency range: freq(timer clock) ≤ freq(host peripheral clock)/4

4.14.2 Software Non-Write Posting Synchronization Mode

This mode is used if TSCR (POSTED bit) is set to 0 in the timer control register.

This mode uses a non-posted write scheme for updating any internal register. This means that the write transaction is not acknowledged on the OCP interface until the effective write operation occurs, after the resynchronization in the timer clock domain. The drawback is that both the interconnect and the CPU are stalled during this period.

- The CPU cannot serve interrupts, increasing the interrupt latency.
- An interconnect including time-out logic to detect erroneous transactions can generate an unwanted system abort event.

This stall period can be quantified as follows:

- T (stall) = 3 MPU peripheral clocks + 5 timer clocks

The same full resynchronization scheme is used for a read transaction. The same stall period applies.

A register read following a write to the same register is always coherent.

This mode is functional regardless of the ratio between the OCP interface frequency and the functional clock frequency.

4.14.3 Write Mode Selection

The choice between the synchronization modes must consider the frequency ratio and the stall periods that can be supported by the system without affecting the global performance.

4.15 Dual-Mode Timer Registers

Table 32 lists the 32-bit (or 2 x 16-bit) dual-mode timer registers. Table 33 through Table 46 describe the individual register bits. The registers are accessible in 16-bit mode and use little-endian addressing.

Table 32. Dual-Mode Timer Registers

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7				
Name	**Description**	**R/W**	**Offset LSB**	**Offset MSB**
TIDR	Timer identification	R	0x00	0x02
RESERVED	Reserved		0x04	0x06
RESERVED	Reserved		0x08	0x0A
RESERVED	Reserved		0x0C	0x0E
TIOCP_CFG	Timer OCP configuration	R/W	0x10	0x12
TISTAT	Timer system status	R	0x14	0x16
TISR	Timer status	R/W	0x18	0x1A
TIER	Timer interrupt enable	R/W	0x1C	0x1E
TWER	Timer wake-up enable	R/W	0x20	0x22
TCLR	Timer control	R/W	0x24	0x26
TCRR	Timer counter	R/W	0x28	0x2A
TLDR	Timer load	R/W	0x2C	0x2E
TTGR	Timer trigger	R/W	0x30	0x32
TWPS	Timer write pending status	R	0x34	0x36
TMAR	Timer match	R/W	0x38	0x3A
TCAR	Timer capture	R	0x3C	0x3E
TSICR	Timer synchronization interface control	R/W	0x40	0x42

Table 33. Timer Identification Register (TIDR)

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7, Offset = 0x00 (LSB), 0x02 (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:8	RESERVED			0x000000
7:0	TID_REV	Module HW revision number of the current timer module: value set by hardware. Four LSBs of TID_REV indicate a minor revision. Four MSBs of TID_REV indicate a major revision. A reset has no effect on value returned	R	HW ID revision

Table 34. Timer OCP Configuration Register (TIOCP_CFG)

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7, Offset = 0x10 (LSB), 0x12 (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:6	RESERVED			0x000 0000
5	EMUFREE	0: The timer is frozen in emulation mode. 1: The timer runs free.	R/W	0
4:3	IdleMode	Power management, request/acknowledge control 00: Force idle. An idle request is acknowledged unconditionally. 01: No idle. An idle request is never acknowledged. 10: Smart idle. Acknowledgement to an idle request is given based on the internal activity of the timer (see Section 4.8, *Sleep Mode Request and Acknowledge*). 11: Reserved	R/W	0x0
2	ENAWAKEUP	Wake-up feature control 0: Wake-up is disabled. 1: Wake-up is enabled.	R/W	0

Table 34. Timer OCP Configuration Register (TIOCP_CFG) (Continued)

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7, Offset = 0x10 (LSB), 0x12 (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
1	SOFTRESET	Software reset. Set this bit to 1 to trigger a module reset. The bit is automatically reset by hardware. During reads, it always returns 0. 0: Normal mode 1: Reset the OCP and the functional domain	R/W	0
0	AUTOIDLE	Interface clocks gating strategy 0: Interface clock is free-running. 1: Automatic interface gating strategy is applied, based on the interface activity.	R/W	0

This register allows controlling various parameters of the OCP interface.

Table 35. Timer System Status Register (TISTAT)

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7, Offset = 0x14 (LSB), 0x16 (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:1	RESERVED		R	0x000 0000
0	RESETDONE	Internal global reset monitoring: 0: Reset is ongoing 1: Reset completed	R	1

This register monitors the internal global reset status. This status bit is set to one when all clock domains have been reset. This status can be monitored by the software to check if the module is ready-to-use following a reset (either hardware or software reset).

Table 36. Timer Status Register (TISR)

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7, Offset = 0x18 (LSB), 0x1A (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:3	RESERVED		R	0x000 0000
2	TCAR_IT_FLAG	0: No capture interrupt request 1: Capture interrupt request	R/W	0
1	OVF_IT_FLAG	0: No overflow interrupt request 1: Overflow interrupt pending	R/W	0
0	MAT_IT_FLAG	0: No compare interrupt request 1: Compare interrupt pending	R/W	0

The timer status register is used to determine which of the timer events requested an interrupt. Bit 0 corresponds to the compare result of TCRR and TMAR, and is set when the compare register matches the counter value. Bit 1 corresponds to the TCRR overflow; and bit 2 indicates that an external pulse transition of the correct polarity is detected on the external event capture pin. If the value is 1, then that timer event is requesting the interrupt. If the user wants to reset the status bit, then a 1 must be written to the appropriate bit. Writing a 1 to the bit TCAR_IT_FLAG resets the edge detection logic. However, the user cannot generate an interrupt by writing a 1 to the timer status register bits. If the user writes a 0 to a bit in the timer status register bits, the value remains unchanged.

Table 37. Timer Interrupt Enable Register (TIER)

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7, Offset = 0x1C (LSB), 0x1E (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:3	RESERVED		R	0x000 0000
2	TCAR_IT_ENA	Capture interrupt enable 0: Interrupt disabled 1: Interrupt enabled	R/W	0
1	OVF_IT_ENA	Overflow interrupt enable 0: Interrupt disabled 1: Interrupt enabled	R/W	0
0	MAT_IT_ENA	Match interrupt enable 0: Interrupt disabled 1: Interrupt enabled	R/W	0

The timer interrupt enable register allows the user to enable certain timer events for generating an interrupt request. Bit 0 determines whether or not the compare register flag MAT_IT_FLAG can generate an interrupt. Bit 1 determines whether or not the overflow counter flag OVF_IT_FLAG can generate an interrupt. Bit 2 determines whether the edge detection flag TCAR_IT_FLAG can generate an interrupt.

Table 38. Timer Wake-Up Enable Register (TWER)

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7, Offset = 0x20 (LSB), 0x22 (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:3	RESERVED		R/W	0x0000 0000
2	TCAR_WUP_ENA	0: Wake-up generation is disabled. 1: Wake-up generation is enabled.	R/W	0
1	OVF_WUP_ENA	0: Wake-up generation is disabled. 1: Wake-up generation is enabled.	R/W	0
0	MAT_WUP_ENA	0: Wake-up generation is disabled. 1: Wake-up generation is enabled.	R/W	0

The timer wake-up enable register (TWER) allows the user to mask the expected source of a wake-up event that generates a wake-up request. The TWER is programmed synchronously with the interface clock before any idle mode request coming from the host processor.

Table 39. Timer Control Register (TCLR)

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7, Offset = 0x24 (LSB), 0x26 (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:13	RESERVED			0x00000
12	PT	Pulse or toggle mode on timer PWM output pin 0: Pulse 1: Toggle	R	0
11:10	TRG	Trigger output mode on timer PWM output pin 00: No trigger 01: Trigger on overflow 10: Trigger on overflow and match 11: Reserved	R/W	0x0
9:8	TCM	Transition capture mode on event capture input pin 00: No capture 01: Capture on low to high transition 10: Capture on high to low transition 11: Capture on both edge transition	R/W	0x0
7	SCPWM	This bit must be set or cleared while the timer is stopped or the trigger is off. 1: Set the timer PWM output pin and select negative pulse for pulse mode. 0: Clear the timer PWM output pin and select positive pulse for pulse mode.	R/W	0
6	CE	1: Compare mode is enabled. 0: Compare mode is disabled.	R/W	0
5	PRE	Prescaler enable 0: The timer clock input pin clocks the counter. 1: The divided input pin clocks the counter.	R/W	0
4:2	PTV	Prescale clock timer value	R/W	0x0

Table 39. Timer Control Register (TCLR) (Continued)

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7, Offset = 0x24 (LSB), 0x26 (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
1	AR	1: Autoreload timer 0: One-shot timer	R/W	0
0	ST	1: Start timer 0: Stop timer: Only the counter is frozen In case of one-shot mode selected (AR = 0), this bit is automatically reset by internal logic when the counter overflows.	R/W	0

Table 40. Timer Counter Register (TCRR)

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7, Offset = 0x28 (LSB), 0x2A (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:0	TIME_COUNTER	Value of timer counter	R/W	0x000 0000

The TCRR register is a 32-bit register (16-bit addressable). Therefore, the MPU can perform one 32-bit access or two 16-bit accesses to the register while the DSP performs two consecutive 16-bit transactions. Because the timer interface clock is completely asynchronous with the timer clock, some synchronization is done to ensure that the TCRR value is not read while it is being incremented.

In 16-bit mode, the following sequence must be followed to read the TCRR register properly:

1) Read the lower 16-bits of the TCRR register (offset = 0x28). When the TCRR is read, the lower 16-bit LSB are read, and the upper 16-bits of the TCRR MSB register are stored in a temporary register.

2) Read the upper 16 bits of the TCRR register (offset = 0x2A). During this read, the value of the upper 16-bit MSB that has been stored in the temporary register is read.

Therefore, to read the value of TCRR correctly, the first OCP read access must be to the lower 16-bits (that is, offset = 0x28), followed by OCP read access to the upper 16-bits (that is, offset= 0x2A).

Table 41. Timer Load Register (TLDR)

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7, Offset = 0x2C (LSB), 0x2E (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:0	TIME_VALUE	Timer counter value loaded on overflow in autoreload mode or on TTGR write access	R/W	0x0000 0000

Table 42. Timer Trigger Register (TTGR)

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7, Offset = 0x30 (LSB), 0x32 (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:0	TTGR_VALUE	Writing in the TTGR register, TCRR is loaded from TLDR and prescaler counter is cleared. Reload is done regardless the AR field value of TCLR register.	R/W	0xFFFF FFFF

The read value of this register is always 0xFFFF FFFF.

Table 43. Timer Write Pending Status Register (TWPS)

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7, Offset = 0x34 (LSB), 0x36 (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:5	RESERVED		R	0x000 0000
4	W_PEND_TMAR	When equal to one, a write is pending to the TMAR register.	R	0
3	W_PEND_TTGR	When equal to one, a write is pending to the TTGR register.	R	0
2	W_PEND_TLDR	When equal to one, a write is pending to the TLDR register.	R	0
1	W_PEND_TCRR	When equal to one, a write is pending to the TCRR register.	R	0
0	W_PEND_TCLR	When equal to one, a write is pending to the TCLR register.	R	0

In posting mode, the software must read the pending write status bits (timer write posted status register bits [4:0]) to ensure that following write access is not discarded because of an ongoing write synchronization process. These bits are automatically cleared by internal logic when the write to the corresponding register is acknowledged. The TISR and TIER registers interface only with the timer interface clock, so they do not need a write pending status bit.

Table 44. Timer Match Register (TMAR)

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7, Offset = 0x38 (LSB), 0x3A (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:0	COMPARE VALUE	Value to be compared to the timer counter	R/W	0x000 0000

The compare logic consists of a 32-bit wide, read/write data TMAR register and logic to compare the counter current value with the value stored in the TMAR register.

Table 45. Timer Capture Register (TCAR)

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7, Offset = 0x3C (LSB), 0x3E (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:0	CAPTURED VALUE	Timer counter value captured on an external event trigger	R	0x0000 0000

When the appropriate transition (rising, falling, or both) is detected in the edge detection logic, the current counter value is stored to the TCAR register.

Table 46. Timer Synchronization Interface Control Register (TSICR)

Base Address = 0xFFFB 1400 (n * 0x800, MPU), E101 1400 (n * 0x800, DSP); n = 0...7, Offset = 0x40 (LSB), 0x42 (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:2	RESERVED	Reserved	R	0x0000 0000
2	POSTED	1: Posted mode active (clocks ratio needs to fit the MPU peripheral clock > 4 timer clock frequency requirement) 0: Posted mode inactive	R/W	1
1	SFT	This bit resets the all of the functional parts of the module. During reads, it always returns 0. 1: Software reset is enabled. 0: Software reset is disabled.	R/W	0
0	RESERVED	Reserved	R	0

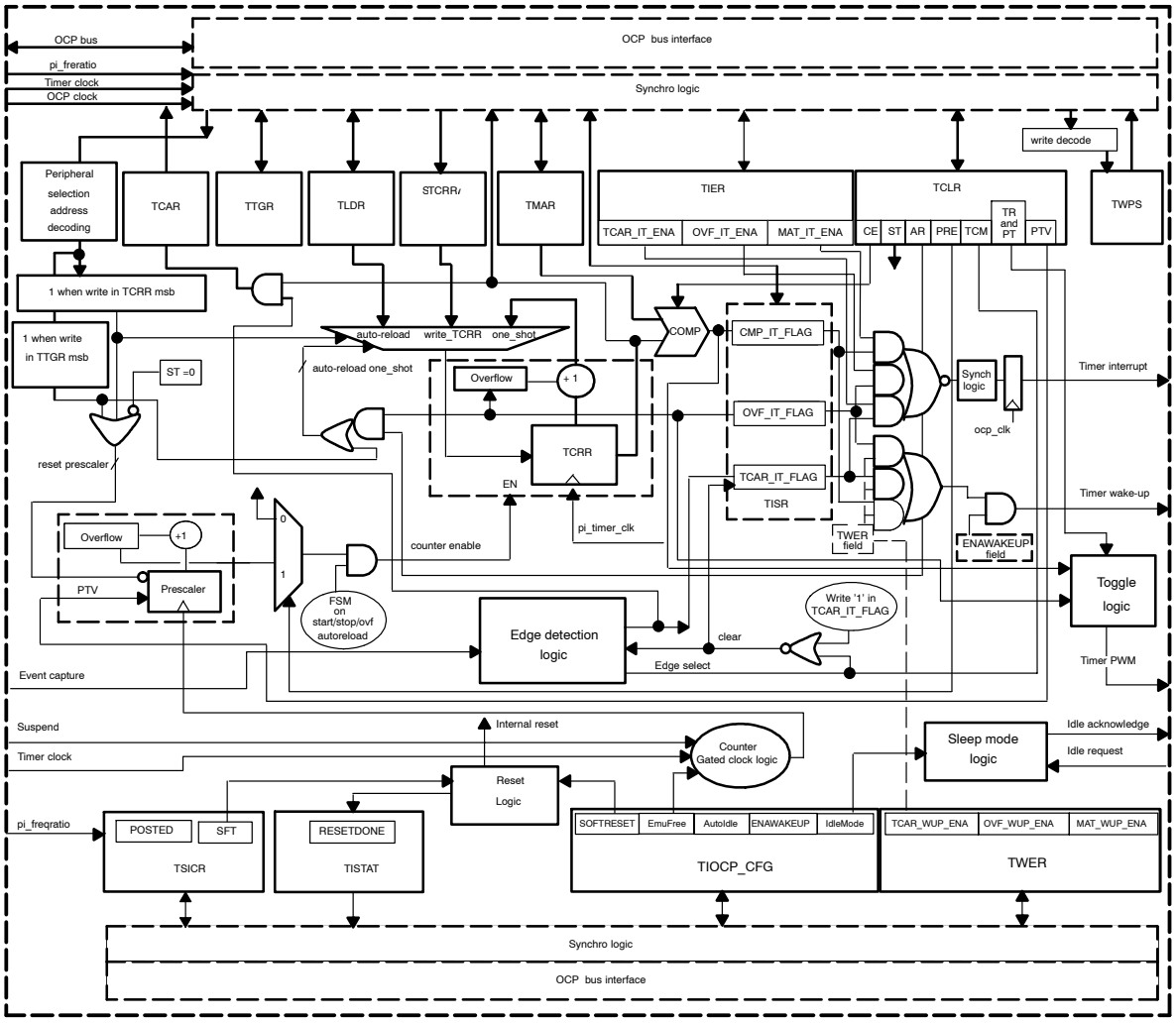
The timer clock has different sources. Based on the timer clock source selection, the user can override the reset value for POSTED to perform nonposted read/write accesses.

Typically, if the timer clock source is 20 MHz and the timer interface clock is 50 MHz, it is recommended to clear the POSTED bit.

4.16 Implementation

Figure 9 is an overview of the implementation and interaction of the registers according to the functional description and the input/output pins.

Figure 9. Dual-Mode Timer Design Implementation Overview



5 32-kHz Synchronized Timer

5.1 Functional Description

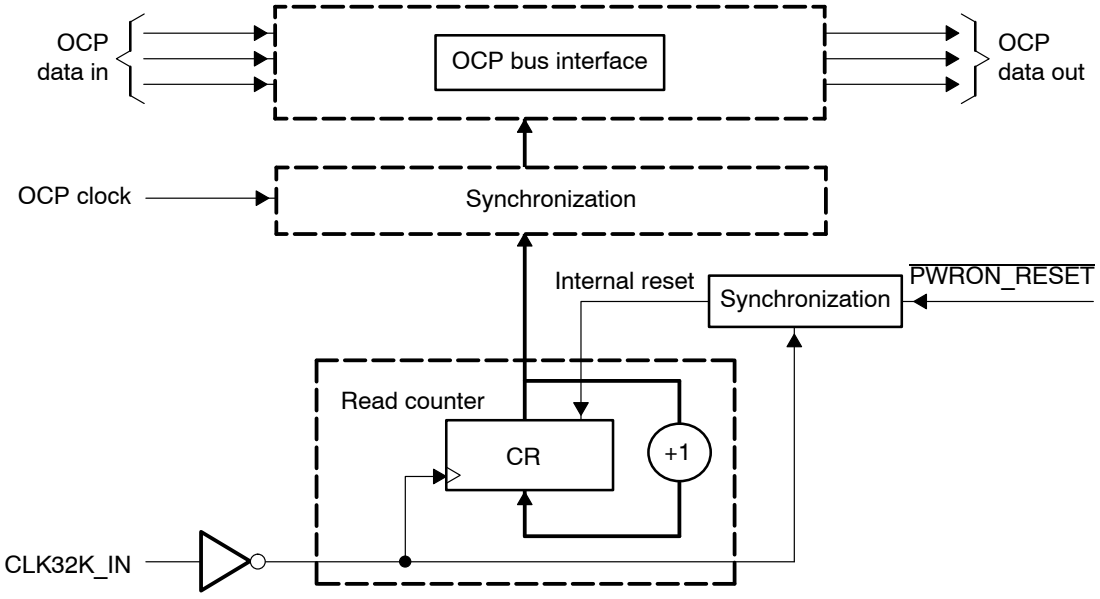
The 32-kHz synchronization counter is a 32-bit counter clocked by the falling edge of the 32-kHz clock.

It is reset with external asynchronous power-up reset ($\overline{\text{PWRON\_RESET}}$).

When $\overline{\text{PWRON\_RESET}}$ is released after three 32-kHz clock periods, the counter starts counting up from the reset value of the counter register on the falling edge of the 32-kHz clock.

When the highest value is reached, it wraps back to zero and starts running again.

Figure 10. 32-kHz SYNCH Timer



Synchronization ensures the read transaction correctness by synchronizing the counter register read access on the OCP clock signal. The $\overline{\text{PWRON\_RESET}}$ input pin resets the counter register (CR) and the inverted CLK32K_IN clocks it.

5.1.1 Reading the Timer

The counter register is 32 bits wide and a 16-bit capture is done on the 16-bit LSB first to allow LSB16 + MSB16 capture.

Internal synchronization logic allows the counter value to be read while the counter is running. The time latency to read a synchronized register is one OCP clock period.

5.2 32-kHz Synchronization Timer Registers

Table 47 lists the 32-bit, 32-kHz synchronization registers, all of which are accessible in 16-bit mode and use little-endian addressing. The address of a register is the start address plus the offset address. Table 48 through Table 49 describe the register bits.

Table 47. 32-kHz Synchronization Timer Registers

Base Address = 0xFFFB C400 (MPU), 0xE101 C400 (DSP)				
Name	**Description**	**R/W**	**Offset LSB**	**Offset MSB**
32KSYNCNT_REV	Revision identification	R	0x00	0x02
RESERVED	Reserved	R	0x04	0x06
RESERVED	Reserved	R	0x08	0x0A
RESERVED	Reserved	R	0x0C	0x0E
CR	Read counter	R	0x10	0x12

Table 48. Identification Register (32KSYNCNT_REV)

Base Address = 0xFFFB C400 (MPU), 0xE101 C400 (DSP)Offset = 0x00 (LSB), 0x02 (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:8	RESERVED	Reads return 0		0x000000
7:0	CID_REV	Module HW revision number of the current timer module: value set by hardware. Four LSBs of TID_REV indicate a minor revision. Four MSBs of TID_REV indicate a major revision.	R	HW ID revision

Table 49. Read Counter Register (CR)

Base Address = 0xFFFB C400 (MPU), 0xE101 C400 (DSP)Offset = 0x10 (LSB), 0x12 (MSB)				
Bit	**Name**	**Function**	**R/W**	**Reset**
15:0	COUNTER_LO	Value of 32-kHz SYNCH counter (16-bit LSB) Same as CR[15:0]	R	0x0003

The CR register is a read-only 32-bit register. Therefore, the MPU performs a 32-bit access on the register while the DSP has to perform two consecutive 16-bit transactions.

In 16-bit mode, the following sequence must be followed to read the CR register properly:

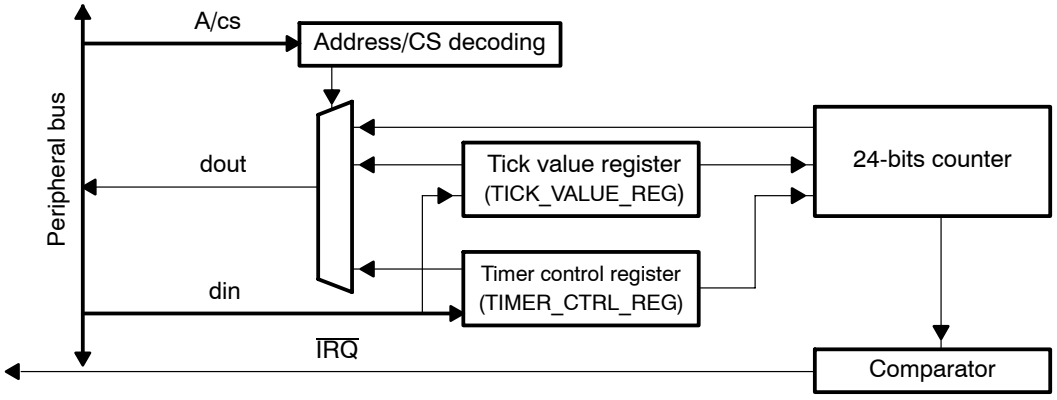
1) Read the lower 16 bits of the CR register (offset = 10).
2) When the CR is read and synchronized, the lower 16-bit LSB are read, and the upper 16-bits of the CR MSB register are stored in a temporary register.
3) Read the upper 16 bits of the CR register (offset = 12).
4) During this read, the value of the upper 16 bits that have been temporarily stored is read.

Therefore, to read the value of CR correctly, the first read access must be to the lower 16 bits (that is, offset = 10), followed by read access to the upper 16 bits (that is, offset = 12).

6 Operating System Timer

The operating system (OS) timer is a 24-bit timer with the 32-kHz clock. Figure 11 shows the operating system timer.

Figure 11. OS Timer



6.1 Countdown Operation

Canonical operation of the OS timer works as follows:

1) The host MPU writes the counter initial value to the tick value register (TICK_VALUE_REG), where, by default, all bits are set to 1.

2) The host MPU sets the timer reload bit (TRB) inside the timer control register (TIMER_CTRL_REG).

3) MPU polls the TRB bit. When it is cleared, the timer internal counter has been loaded with the the tick value register (TICK_VALUE_REG) value.

4) MPU then launches the timer by setting the timer start stop bit (TSS) inside the timer control register (TIMER_CTRL_REG). The timer starts to count down to zero and generates a negative edge-sensitive interrupt (low level 15-µs-wide pulse) to the interrupt controller.

5) On the next cycle, the counter is reloaded from the tick value register (TICK_VALUE_REG) and then starts to down count again, unless the ARL bit of the timer control register (TIMER_CTRL_REG) is low: one-shot mode.

6.2 Overriding Normal Counting

Normal operation can be overridden by using two bits in the timer control register (TIMER_CTRL_REG):

- The timer reload bit (TRB), causes the counter to be reloaded on the next 32-kHz clock cycle (whether or not the timer is counting).
- The timer start stop bit (TSS) causes the counter to be stopped on the next 32-kHz clock cycle. Then, the content of the counter is frozen.

6.3 Loading/Autorestart

Loading the counter in the timer can be done in two ways :

- Writing a 1 to the TRB bit in the timer control register (TIMER_CTRL_REG).
- Waiting until the counter reaches zero. It then is reloaded from the tick value register (TICK_VALUE_REG) if the autorestart bit (ARL) in the control register is set to 1. If not, the timer is stopped.

6.4 Timer Interrupt Period

The timer interrupt period is defined by the value loaded into the tick value register (TICK_VALUE_REG).

Setting the counter to value 0 is not allowed and leads to unpredictable results.

The timer interrupt rate is as follows:

$$\text{IRQ rate} = \frac{u}{32768},$$

where *u* is the sum of the TVR contents plus one.

Table 50. Timer Interrupt Period According to TVR Value

TICK_VALUE_REG	Interrupt Period
0x00000001	61 µs
0x0000028F	19.9 ms
0x000007FF	62.5 ms
0x0000FFFF	2 s
0x00FFFFFF	512 s

6.5 Peripheral Alignment and Data Width

The TIPB interface data path is 32 bits wide. To keep software compatible with earlier 16-bit timer versions, the access width is taken into account when writing to the registers. This means that writing 8 bits (or 16 bit) sets 24 MSB (or 16 MSB) to 0.

For read accesses, all 32 bits are always output.

All unused register bits remain at 0.

6.6 OS Timer Registers

Table 51 lists the OS timer registers. Table 52 through Table 55 describe the register bits. Synchronization of reads and writes to the 32-kHz clock is performed in a different way for each register, which leads to slight restrictions concerning register access (see Table 52).

Table 51. Operating System Registers

Base Address = 0xFFFB 9000			
Name	**Description**	**R/W**	**Offset**
TICK_VALUE_REG	Tick value	R/W	0x00
TICK_COUNTER_REG	Tick counter	R	0x04
TIMER_CTRL_REG	Timer control	R/W	0x08

Table 52. Timer Registers Access Timing Constraints

Register Name	Read	Write
TIMER_CTRL_REG	Can be read anytime. The value read is the last value written.	Two consecutive writes must be separated by at least 1 CLK32 period (31.25 μs). If this is not the case, the value written is not assured.
TICK_COUNTER_REG	Reads are resynchronized on a high-speed clock to the prevent peripheral bus from timing out. Can be read anytime, providing high-speed clock is running. If not, the value is not assured.	Writing to this register has no effect.
TICK_VALUE_REG	Can be read anytime. The value read is the last value written.	Two consecutive writes must be separated by at least 1 CLK32 period (31.25 μs). If this is not the case, the value written is not assured.

Table 53. Tick Value Register (TICK_VALUE_REG)

Base Address = 0xFFFB 9000, Offset = 0x00				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:24	RESERVED	Reserved	R	0x00
23:0	TICK_VALUE_REG	Value is loaded when the timer reaches 0 or when it starts.	R/W	0xFFFFFF

Writes to this register must be separated by at least one 32-kHz clock period.

Setting TICK_VALUE_REG to 0, and then loading the counter with this value leads to unpredictable results.

Table 54. Tick Counter Register (TICK_COUNTER_REG)

Base Address = 0xFFFB 9000, Offset = 0x04				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:24	RESERVED	Reserved	R	0x0
23:0	TICK_COUNTER_REG	Value of timer	R	0xFFFFFF

When reading this register, the high-speed clock must be running. If not, the value read is not ensured.

Table 55. Timer Control Register (TIMER_CTRL_REG)

Base Address = 0xFFFB 9000, Offset = 0x08				
Bit	**Name**	**Function**	**R/W**	**Reset**
31:4	RESERVED	Reserved	R	0x0000 000
3	ARL	Autoreload/start: 1: Autorestart mode 0: One-shot mode: when counter reaches zero, an interrupt is generated and timer is stopped	R/W	1
2	IT_ENA	Interrupt enable: 0: Interrupt disabled 1: Interrupt enabled	R/W	0
1	TRB	Timer reload bit: Setting bit to 1 reloads the counter at next 32 kHz-clock rising edge. Once the counter is reloaded, the bit is reset.	R/W	0
0	TSS	Timer start/stop: 0: Stop timer 1: Start timer If timer is in one-shot mode (ARL = 0), the bit is automatically reset when the timer = 0.	R/W	0

Writes to this register must be separated by at least one 32-kHz clock period.

This page is intentionally left blank.

Index

N

O

P

R

S

T

W