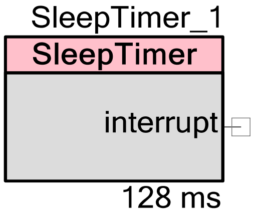
Sleep Timer
1.0



Features

- Wake up and generate interrupt or just wake up from configurable power modes
- 12 discrete intervals: 2ms, 4ms, 8ms, 16ms, 32ms, 64ms, 128ms, 256ms, 512ms, 1024ms, 2048ms and 4096ms

General Description

The Sleep Timer component is used to wake the device from Sleep mode or generate an interrupt in active mode at a configurable interval.

When to use a Sleep Timer

Sleep Timer can be used to create long duration timer intervals. Such intervals can be implemented by hardware counters, but this would use hardware resources inefficiently and would require the device to remain in Active Mode. In some cases this is fully acceptable. However, if you have a battery-powered application, or want a more efficient implementation, the Sleep Timer could be useful. The Sleep Timer can also act as a standalone long duration counter.

Input/Output Connections

This section describes output connection for the Sleep Timer. There are no input connections. An asterisk (*) in the list of I/O’s states that the I/O may be hidden on the symbol under the conditions listed in the description of that I/O.

interrupt – Output *

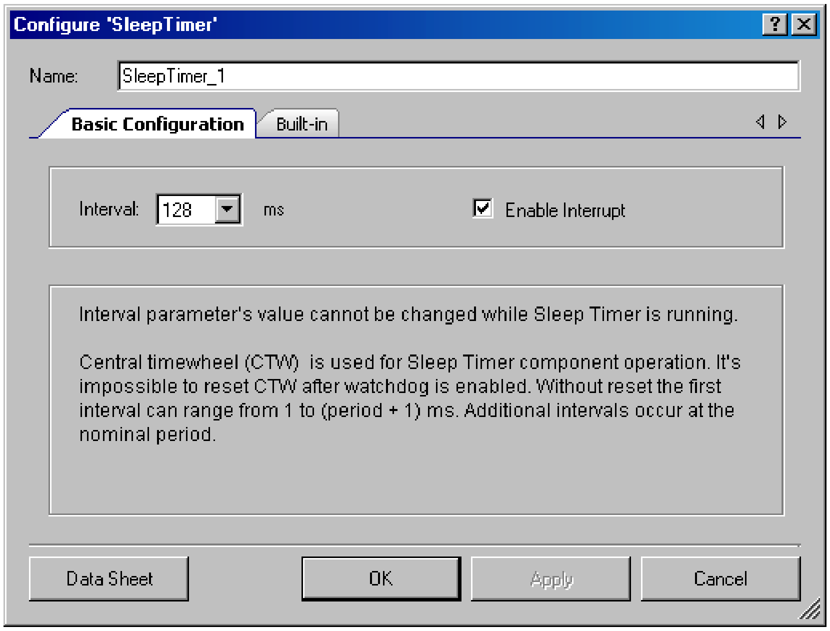
The interrupt output is the central time wheel interrupt source. The interrupt is issued when Central Time Wheel (CTW) counter reaches the terminal count, specified by Sleep Timer interval’s value. This output is not available when the interrupt is disabled.

PRELIMINARY

Parameters and Setup

Drag a Sleep Timer component onto your design and double-click it to open the Configure SleepTimer dialog.

Figure 1 Configure Sleep Timer Dialog



Name: SleepTimer_1

Interval: 128 ms

☑ Enable Interrupt

Interval parameter's value cannot be changed while Sleep Timer is running.

Central timewheel (CTW) is used for Sleep Timer component operation. It's impossible to reset CTW after watchdog is enabled. Without reset the first interval can range from 1 to (period + 1) ms. Additional intervals occur at the nominal period.

Basic Configuration

Interval

Defines wake up and interrupt, or just wake up interval in milliseconds. Only discrete intervals are accepted from the range: 2ms, 4ms, 8ms, 16ms, 32ms, 64ms, 128ms, 256ms, 512ms, 1024ms, 2048ms and 4096ms. The software can re-configure these values when the Sleep Timer is stopped. These parameters simply define an initial configuration.

Enable Interrupt

Determines if the Sleep Timer interrupt is enabled or disabled. If the Sleep Timer interrupt parameter is enabled, the system issues an interrupt after wake up. If the Sleep Timer interrupt is disabled, the system just switches to active mode after a defined interval. Program execution continues from the next instruction after the previous one, which makes the system to go to low power mode. The software can re-configure these values at any time; these parameters simply define an initial configuration.

Clock Selection

The Sleep Timer component uses the CTW and for its operation a 1 kHz clock is required. This clock is produced by the internal low speed oscillator (ILO). The ILO produces two primary independent output clocks with no external components, and with very low power consumption. These two outputs operate at nominal frequencies of 1 kHz and 100 kHz. The two clocks run independently, are not synchronized to each other, and can be enabled or disabled together or independently. The API function that starts the Sleep Timer automatically enables the 1 kHz clock and leaves it enabled even after the component is stopped.

Placement

There is no placement specific information.

Resources

The Sleep Timer uses the following device resources:

- 1kHz ILO clock line
- Central Time Wheel (CTW) counter
- Central Time Wheel (CTW) counter’s interrupt line

Application Programming Interface

Application Programming Interface (API) routines allow you to configure the component using software. The following table lists and describes the interface to each function. The subsequent sections cover each function in more detail.

By default, PSoC Creator assigns the instance name "SleepTimer_1" to the first instance of a component in a given design. You can rename the instance to any unique value that follows the syntactic rules for identifiers. The instance name becomes the prefix of every global function name, variable, and constant symbol. For readability, the instance name used in the following table is "SleepTimer".

Function	Description
void SleepTimer_Start(void)	Starts the Sleep Timer operation: enables 1 kHz clock, sets default parameters if they were not set previously by corresponding API functions.
void SleepTimer_Stop(void)	Stops the Sleep Timer operation: disables wake up on CTW counter terminal count reached and disables interrupt issuing on foregoing condition.
void SleepTimer_EnableInt (void)	Enables Sleep Timer component issuing interrupt on wake up.
void SleepTimer_DisableInt (void)	Disables Sleep Timer component issuing interrupt on wake up.

Function	Description
void SleepTimer_SetInterval (uint8)	Sets interval for Sleep Timer to wake up.
void SleepTimer_Reset (void)	Resets the CTW counter for proper Sleep Timer first time wake up.

void SleepTimer_Start(void)

Description: Starts the Sleep Timer operation: enables 1 kHz clock, sets default parameters if they were not set previously by corresponding API functions.

Parameters: Void

Return Value: Void

Side Effects: Enables 1 kHz ILO clocks and leaves it enabled after Sleep Time component is stopped.

void SleepTimer_Stop(void)

Description: Stops the Sleep Timer component's operation: disables wake up on CTW counter terminal count reached and disables interrupt issuing on foregoing condition.

Parameters: Void

Return Value: Void

Side Effects: None

void SleepTimer_EnableInt (void)

Description: Enables Sleep Timer component issuing interrupt on wake up.

Parameters: Void

Return Value: Void

Side Effects: None

void SleepTimer_DisableInt (void)

Description: Disables Sleep Timer component issuing interrupt on wake up.

Parameters: Void

Return Value: Void

Side Effects: None

void SleepTimer_SetInterval (uint8 interval)

Description: Sets interval for Sleep Timer to wake up.

Parameters: uint8 interval: interval’s value for Sleep Timer to wake up in.

Name	Value	Period
SleepTimer__CTW_2_MS	4'b0001	2 ms
SleepTimer__CTW_4_MS	4'b0010	4 ms
SleepTimer__CTW_8_MS	4'b0011	8 ms
SleepTimer__CTW_16_MS	4'b0100	16 ms
SleepTimer__CTW_32_MS	4'b0101	32 ms
SleepTimer__CTW_64_MS	4'b0110	64 ms
SleepTimer__CTW_128_MS	4'b0111	128 ms
SleepTimer__CTW_256_MS	4'b1000	256 ms
SleepTimer__CTW_512_MS	4'b1001	512 ms
SleepTimer__CTW_1024_MS	4'b1010	1024 ms
SleepTimer__CTW_2048_MS	4'b1011	2048 ms
SleepTimer__CTW_4096_MS	4'b1100	4096 ms

Return Value: Void

Side Effects: Interval value can be only changed when Sleep Timer is stopped. The first interval can range from 1 to (period + 1) ms. Additional intervals occur at the nominal period.

void SleepTimer_Reset (void)

Description: Resets the CTW counter for proper Sleep Timer first time wake up.

Parameters: void

Return Value: void

Side Effects: Note that the Watch Dog Timer (WDT) shares the CTW, so if the CTW is reset, the WDT will take longer to trigger a hardware reset. Note that it is impossible to reset the CTW if the Watch Dog Timer is enabled.

Sample Firmware Source Code

The following C language example demonstrates the basic functionality of the Sleep Timer component.

This example demonstrates device's wake up from the Sleep Mode every 256 ms without issuing interrupt. It assumes the component has been placed in the schematic with the default name of "SleepTimer_1".

```
#include <device.h>
/* For CyDelay() function */
#include <intrins.h>
/* Power mode changes API */
#include <cylib.h>
/* All of the address values for the entire PSoC device */
#include "cydevice_trm.h"
/* Register access macros and approved types for use in firmware. */
#include "cytypes.h"
/* Used to clear interrupt status register */
uint8 tmp;

void main()
{
         /* Enable global interrupts */
         IE = 0x80;
         /* Disable Sleep Timer component issuing interrupt */
         SleepTimer_1_DisableInt();
         /* Set interval to wake up. */
         SleepTimer_1_SetInterval(SleepTimer_1__CTW_256_MS);
/* Reset the CTW counter */
         SleepTimer_1_Reset();
         /* Starts Sleep Timer component */
         SleepTimer_1_Start();

    for(;;)
    {
               /* Clear interrupt status bits */
               tmp = CY_GET_REG8(CYREG_PM_INT_SR);
               /* Place your code here to be executed in Active Mode.*/
               /* Put device in Sleep Mode */
               CySleep();
    }
}
```

Functional Description

The Sleep Timer can be used to periodically wake-up the device from Sleep mode and afterwards, optionally, generate an interrupt.

Refer to the device data sheet for proper switching from/to low power mode and to understand the relationship between Sleep Timer and Watch Dog Timer.

As described previously, the Sleep Timer can be configured to the following intervals: 2ms, 4ms, 8ms, 16ms, 32ms, 64ms, 128ms, 256ms, 512ms, 1024ms, 2048ms and 4096ms. However, it is important to remember that it can have up to 20% deviation, because the sleep timer is derived from the PSoC's internal low speed oscillator. Systems that require accurate timing should use the Real Time Clock (RTC) component instead of the Sleep Timer.

Note that for proper switching to the Sleep Mode according to the device specification, if MHz crystal oscillator or PLL is used, you must switch to the IMO and then disable those clock sources. Also, it is strongly recommended to change the clock source to the IMO before entering idle to ensure proper shutdown and startup. This is done via the project's Design-Wide Resources Clock Editor (the project's cydwr file). Open the Configure System Clocks dialog to disable the appropriate clock sources.

Note that, as for now, to include code that responds to switching to Sleep Mode, Standby Mode, and Idle mode you have to add "define(CYLIB_POWER_MANAGEMENT=1)" to the compiler's command line. This is done via the Build Settings dialog under the Compiler section. Select "Command line" and add to the field next to "Custom Flags" field.

Note that for the proper operation of Sleep Timer component, you have to read the CYREG_PM_INT_SR register to clear CTW interrupt status bit. It could be done in ISR, if the component is configured to issue interrupt on wake up, or, if the Sleep Timer's interrupt is disabled, in the user's code, where the program is supposed to continue executing after wake up. If the mentioned register is not read after issuing an interrupt, the next one will not occur until the CYREG_PM_INT_SR register is cleared by the read.

References

See also the RTC component.

DC and AC Electrical Characteristics

The following values are indicative of expected performance and based on initial characterization data.

5.0V/3.3V DC and AC Electrical Characteristics

Parameter	Typical	Min	Max	Units	Conditions and Notes
Input					
Input Voltage Range	---		Vss to Vdd	V	
Input Capacitance	---		---	pF	
Input Impedance	---		---	Ω	
Maximum Clock Rate	---		67	MHz	

Disclaimer: CYPRESS MAKES NO WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, WITH REGARD TO THIS MATERIAL, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. Cypress reserves the right to make changes without further notice to the materials described herein. Cypress does not assume any liability arising out of the application or use of any product or circuit described herein. Cypress does not authorize its products for use as critical components in life-support systems where a malfunction or failure may reasonably be expected to result in significant injury to the user. The inclusion of Cypress' product in a life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.