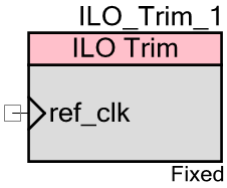
ILO Trim
2.0

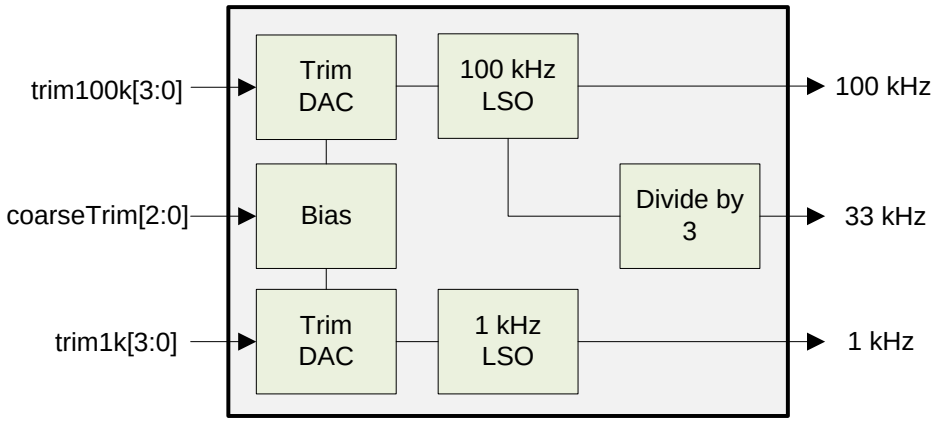


Features

- Trims 1 kHz and 100 kHz ILO for PSoC 3 and PSoC 5LP
- ILO accuracy compensation for PSoC 4
- UDB and Fixed-Function modes
- User-specified reference clock

General Description

The ILO Trim component allows your application to determine the accuracy of the ILO. It provides a scaling function to allow the application to compensate for this inaccuracy. For PSoC 3 and PSoC 5LP devices, it can also directly improve the accuracy of the ILO by using a user-defined higher frequency, higher accuracy reference clock to count the number of ILO clock cycles. The derived information is then used to trim the ILO trim registers to incrementally approach the desired ILO frequency. The component supports both UDB and Fixed-Function implementations.



For PSoC 3 and PSoC 5LP, the ILO consists of two low-speed oscillators (LSO): 100 kHz and 1 kHz. These are used to generate ILO clock frequencies of 1 kHz, 33 kHz, or 100 kHz. Post factory trim, the 1 kHz LSO has an accuracy of -50% to 100% and the 100 kHz LSO has an accuracy of -55% to 100% over the entire operating range of voltage and temperature.

During the run-time trimming operation, the trim DACs and the bias block are adjusted using the ILO Trim registers.

For PSoC 4, the ILO is nominally a 32 kHz low-frequency clock (LFCLK). Trimming for PSoC 4 is not supported; however you can use the scaling function to retrieve the compensation information that can be used in your application.

When to Use an ILO Trim

The ILO Trim component is used to achieve a higher accuracy ILO clock. One use of the component is to use it to trim the ILO, which will then be used as a clock in sleep mode. In such an instance, a more accurate ILO allows for more precise event timing. Alternatively, the compensation function can be used to retrieve the ILO accuracy, which can then be used to define the new counter value (e.g. watchdog timer) to achieve your desired interrupt period.

Input/Output Connections

This section describes the input and output connections for the ILO Trim component.

ref_clk

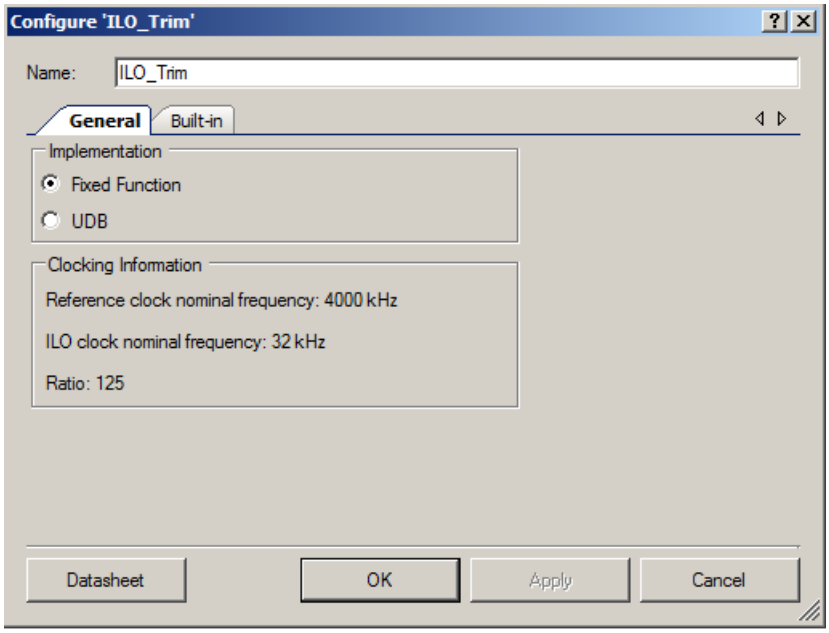
This terminal is the clock signal used as the timing reference for counting the ILO period.

Component Parameters

Drag an ILO Trim component onto your design and double-click it to open the **Configure** dialog.

General Tab

The **General** tab is used to set the general operational parameters of the component. It contains the following parameters. All of these settings are compile-time selections; you do not need to change these settings at run time.



Implementation

This parameter is used to specify the type of hardware used to implement the ILO Trim component. It can be either Fixed Function or UDB.

Clocking Information

- Reference clock nominal frequency: Displays the reference clock frequency in kHz.
- ILO clock nominal frequency: Displays the frequency of the target ILO in kHz
- Ratio: Displays the ratio of the reference clock to the ILO clock. This can then be used to adjust the reference clock to an acceptable value.

  Note If this ratio is outside the supported range, then an information bubble will appear to help choose the correct reference clock.

Clock Selection

The ILO Trim component contains an embedded ILO clock. This clock can be configured by modifying the ILO clock value in the design wide resources manager. The component also requires an external reference clock by attaching a clock source. This clock must be within the valid range of reference to ILO ratio.

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 discuss each function in more detail.

By default, PSoC Creator assigns the instance name “ILO_Trim_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 “ILO_Trim.”

Functions

Function	Description
ILO_Trim_Start()	Initializes and starts the component.
ILO_Trim_Stop()	Stops the component.
ILO_Trim_BeginTrimming()	Begins the implementation of the ILO trimming algorithm.
ILO_Trim_StopTrimming()	Disables the trimming algorithm.
ILO_Trim_CheckStatus()	Returns the current status of the ILO and the trimming algorithm.
ILO_Trim_CheckError()	Calculates ILO frequency error in parts per thousand.
ILO_Trim_Compensate()	Compensates for the ILO clock inaccuracy by converting from a desired nominal number of clock cycles to the effective number of ILO clock cycles required based on the current accuracy of the ILO.
ILO_Trim_RestoreTrim()	Restores the factory trim value.
ILO_Trim_GetTrim()	Returns the current ILO trim value.
ILO_Trim_SetTrim()	Returns the called number of words from the given address. If any address does not belong to the allocated register space, an error is returned.
ILO_Trim_Sleep()	Stops the component and saves the user configuration
ILO_Trim_Wakeup()	Restores the user configuration and enables the component
ILO_Trim_SaveConfig()	Saves the current user configuration of the component.
ILO_Trim_RestoreConfig()	Restores the current user configuration of the component

void ILO_Trim_Start(void)

Description: Starts the measurement of the ILO accuracy.

Parameters: None

Return Value: None

Side Effects: None

void ILO_Trim_Stop(void)

Description: Stops the measurement of the ILO accuracy. If trimming is currently active, then the trimming algorithm is also terminated at this point.

Parameters: None

Return Value: None

Side Effects: None

void ILO_Trim_BeginTrimming(void)

Description: Begins the implementation of the ILO trimming algorithm. The algorithm requires multiple ILO clock periods to converge with an accurately trimmed ILO clock frequency. This function is non-blocking and will return after configuring an interrupt process to implement the trimming algorithm. ILO_Trim_CheckStatus() and ILO_Trim_CheckError() can be used to determine the current status of the algorithm. Once the trimming algorithm converges on an accurately trimmed ILO frequency the background trimming algorithm is disabled.

Not Supported for PSoC 4.

Parameters: None

Return Value: None

Side Effects: None

void ILO_Trim_StopTrimming(void)

Description: Disables the trimming algorithm that was started by the ILO_Trim_BeginTrimming() function. This function is only used to terminate the trimming algorithm early. Normal operation of the trimming algorithm does not use this function since the algorithm will disable trimming once the ILO accuracy has been achieved.

Not Supported for PSoC 4.

Parameters: None

Return Value: None

Side Effects: None

uint8 ILO_Trim_CheckStatus(void)

Description: Returns the current status of the ILO and the trimming algorithm.

Note The accuracy status is not reliable until two ILO clock cycles have occurred since the component was started.

Not Supported for PSoC 4.

Parameters: None

Return Value: Set of bit fields that indicate the status of the component. Multiple bit fields can be set at the same time.

Value	Description
ILO_Trim_IS_ACCURATE	ILO accuracy is within the required error range.
ILO_Trim_TRIMMING	Trimming algorithm is currently running

Side Effects: None

int16 ILO_Trim_CheckError(void)

Description: Calculates ILO frequency error in parts per thousand. A positive number indicates that the ILO is running fast and a negative number indicates that the ILO is running slow. This error is relative to the error in the reference clock, so the absolute error will be higher and depend on the accuracy of the reference.

Note The accuracy error is not reliable until two ILO clock cycles have occurred since the component was started.

Parameters: None

Return Value: Error value in parts per thousand.

Side Effects: None

uint16 ILO_Trim_Compensate(uint16 clocks)

Description: Compensates for the ILO clock inaccuracy by converting from a desired nominal number of clock cycles to the effective number of ILO clock cycles required based on the current accuracy of the ILO. The returned value can then be used instead of the nominal value when configuring timers that are based on the ILO. If the calculated result exceeds the capacity of the 16 bit return value, it will be saturated at the maximum 16-bit value.

Note This function is an alternative to trimming the frequency of the ILO and should not be used in conjunction with the BeginTrimming() function.

Note This function requires a full ILO cycle to occur prior to reading the ILO accuracy result. The application should insert a worst-case delay based on the chosen ILO frequency (for example, 32 kHz +100%) between ILO_Trim_Start() and calling this function.

Parameters: uint16 clocks: Number of nominal ILO clocks desired

Return Value: uint16: Number of actual ILO clocks required

Side Effects: None

void ILO_Trim_RestoreTrim(void)

Description: Restore the factory trim value.
Not Supported for PSoC 4.

Parameters: None

Return Value: None

Side Effects: None.

uint8 ILO_Trim_GetTrim(void)

Description: Returns the current ILO trim value.
Not Supported for PSoC 4.

Parameters: None

Return Value: uint8 trim register value

Side Effects: None

void ILO_Trim_SetTrim(uint8 trim)

Description: Sets the ILO trim value.
Not Supported for PSoC 4.

Parameters: uint8 trim: Value to be written into the ILO trim register, where trim[3:0] are the fine trim values. For PSoC 3 and PSoC 5LP, trim[5:4] are the coarse trim values.

Return Value: None

Side Effects: None

void ILO_Trim_Sleep(void)

Description: Prepares the component for sleep. If the component is currently enabled it will be disabled and re-enabled by ILO_Trim_Wakeup().

Parameters: None

Return Value: None

Side Effects: None

void ILO_Trim_Wakeup(void)

Description: Returns the component to its state before the call to ILO_Trim_Sleep(). In order to trim the ILO after waking from a low power mode, the ILO_Trim_BeginTrimming() function must be called.

Parameters: None

Return Value: None

Side Effects: None

void ILO_Trim_SaveConfig(void)

Description: Saves the configuration of the component. This routine is called by ILO_Trim_Sleep() to save the configuration.

Parameters: None

Return Value: None

Side Effects: None

void ILO_Trim_RestoreConfig(void)

Description: Restores the configuration of the component. This routine is called by ILO_Trim_Wakeup() to restore the configuration.

Parameters: None

Return Value: None

Side Effects: None

Macro Callbacks

Macro callbacks allow users to execute code from the API files that are automatically generated by PSoC Creator. Refer to the PSoC Creator Help and *Component Author Guide* for more details.

In order to add code to the macro callback present in the component's generated source files, perform the following:

- Define a macro to signal the presence of a callback (in *cyapicallbacks.h*). This will "uncomment" the function call from the component's source code.
- Write the function declaration (in *cyapicallbacks.h*). This will make this function visible by all the project files.
- Write the function implementation (in any user file).

Callback Function [1]	Associated Macro	Description
ILO_Trim_CorrectionIteration_EntryCallback	ILO_Trim_CORRECTION_ITERATION_ENTRY_CALLBACK	Used at the beginning of the _CorrectionIteration() interrupt handler to perform additional application-specific actions.
ILO_Trim_CorrectionIteration_ExitCallback	ILO_Trim_CORRECTION_ITERATION_EXIT_CALLBACK	Used at the end of the _CorrectionIteration() interrupt handler to perform additional application-specific actions.

Sample Firmware Source Code

PSoC Creator provides many example projects that include schematics and example code in the Find Example Project dialog. For component-specific examples, open the dialog from the Component Catalog or an instance of the component in a schematic. For general examples, open the dialog from the Start Page or **File** menu. As needed, use the **Filter Options** in the dialog to narrow the list of projects available to select.

Refer to the “Find Example Project” topic in the PSoC Creator Help for more information.

API Memory Usage

The component memory usage varies significantly, depending on the compiler, device, number of APIs used and component configuration. The following table provides the memory usage for all APIs available in the given component configuration.

The measurements have been done with associated compiler configured in Release mode with optimization set for Size. For a specific design the map file generated by the compiler can be analyzed to determine the memory usage.

Configuration	PSoC 3 (Keil_PK51)		PSoC 4 (GCC)		PSoC 5LP (GCC)	
	Flash Bytes	SRAM Bytes	Flash Bytes	SRAM Bytes	Flash Bytes	SRAM Bytes
UDB	1100	9	484	20	1328	12
Fixed Function	1166	7	492	21	1420	12

1 The callback function name is formed by component function name optionally appended by short explanation and “Callback” suffix.

MISRA Compliance

This section describes the MISRA-C:2004 compliance and deviations for the component. There are two types of deviations defined:

- project deviations – deviations that are applicable for all PSoC Creator components
- specific deviations – deviations that are applicable only for this component

This section provides information on component-specific deviations. Project deviations are described in the MISRA Compliance section of the *System Reference Guide* along with information on the MISRA compliance verification environment.

The ILO Trim component does not have any specific deviations.

This component has the following embedded component - Interrupt. Refer to the corresponding component datasheet for information on their MISRA compliance and specific deviations.

Functional Description

Implementation

The ILO Trim component uses a down counter to continuously count the number of reference clock cycles occurring in a single period of the ILO. This information is used to change the ILO trim value to incrementally approach the target ILO frequency.

Trim Procedure

The operation of the component is as follows.

1) The component is configured to either UDB or Fixed Function and an appropriate reference clock is provided in the customizer.
2) The reference clock is used as the clock to the counter.
3) At the rising edge of the ILO, the current counter value is captured into the capture register. This triggers an ISR containing the trimming routine.
4) The trimming routine sequentially and incrementally approaches the target ILO by changing the ILO trim value by 1 per iteration.
5) Steps 3 and 4 are repeated until the trim value is within 10% of the target ILO or better.
6) The user may call CheckStatus() and CheckError() APIs any time to check the trimming status and the ILO error respectively.

Compensation Procedure

Alternatively, compensation can be used to find the compensated counter value to meet the desired timing requirements. For example suppose that the target time desired from a counter t_d in the application is denoted as:

$$t_d = C_o / f_{ILO}$$

where C_o is the original counter value, and f_{ILO} is the frequency of the ILO. Since the ILO may not be trimmed, the actual time t_a can vary by +/- z%.

$$t_a = C_o / f_{ILO} + z$$

The compensation procedure then returns the correct counter value C_c to meet the desired timing requirements to within +/- 10%.

$$C_o = (f_{ILO} + z)t_d$$

Resources

The ILO Trim component uses either 1 UDB or 1 Fixed-Function Timer block depending on the mode chosen. It contains 1 embedded ILO clock and 1 interrupt.

Configuration	Resource Type						
	Datapath Cells	Macrocells	Status Cells	Control Cells	Fixed Function	Clock	Interrupts
UDB	1	3	1	1	-	1	1
Fixed Function: PSoC 3, PSoC 5LP	-	-	-	-	1 (Timer)	1	1
Fixed Function: PSoC4	-	-	-	-	1 (TCPWM)	1	1

DC and AC Electrical Characteristics: PSoC 3

Specifications are valid for –40 °C ≤ TA ≤ 85 °C and TJ ≤ 100 °C, except where noted.
Specifications are valid for 1.71 V to 5.5 V, except where noted.

DC Characteristics

Mode	Description	Conditions	Min	Typ	Max	Units
Fixed Function	16-bit Fixed Function timer block current consumption	Input clock frequency – 3 MHz	–	15	–	µA
		Input clock frequency –12 MHz	–	60	–	µA
		Input clock frequency – 48 MHz	–	260	–	µA
		Input clock frequency – 67 MHz	–	350	–	µA

Mode	Description	Conditions	Min	Typ	Max	Units
UDB	8-bit UDB timer block current consumption	–	–	6	–	µA/MHz

DC and AC Electrical Characteristics: PSoC 4

Specifications are valid for –40 °C ≤ TA ≤ 85 °C and TJ ≤ 100 °C, except where noted. Specifications are valid for 1.71 V to 5.5 V, except where noted.

Note The data for the PSoC 4200L device is preliminary. Final data will be delivered in an upcoming Component Pack.

DC Characteristics

Mode	Description	Conditions	Min	Typ	Max	Units
Fixed Function	16-bit Fixed Function TCPWM block current consumption	Input clock frequency – 3 MHz	–	19	–	µA
		Input clock frequency –12 MHz	–	66	–	µA
		Input clock frequency – 48 MHz	–	270	–	µA
Fixed Function	16-bit Fixed Function TCPWM block current consumption for PSoC 4100M/PSoC 4200M	Input clock frequency – 3 MHz	–	45	–	µA
		Input clock frequency –12 MHz	–	155	–	µA
		Input clock frequency – 48 MHz	–	650	–	µA
UDB	8-bit UDB timer block current consumption	–	–	6	–	µA/MHz

DC and AC Electrical Characteristics - PSoC 5LP

Specifications are valid for –40 °C ≤ TA ≤ 85 °C and TJ ≤ 100 °C, except where noted. Specifications are valid for 1.71 V to 5.5 V, except where noted.

DC Characteristics

Mode	Description	Conditions	Min	Typ	Max	Units
Fixed Function	16-bit Fixed Function timer block current consumption	Input clock frequency – 3 MHz	–	65	–	µA
		Input clock frequency –12 MHz	–	170	–	µA
		Input clock frequency – 48 MHz	–	650	–	µA
		Input clock frequency – 67 MHz	–	900	–	µA

Mode	Description	Conditions	Min	Typ	Max	Units
UDB	8-bit UDB timer block current consumption	–	–	6	–	µA/MHz

DC and AC Electrical Characteristics: Timing

Specifications are valid for –40 °C ≤ T_A ≤ 85 °C and T_J ≤ 100 °C, except where noted. Specifications are valid for 1.71 V to 5.5 V, except where noted.

Note The data for the PSoC 4200L device is preliminary. Final data will be delivered in an upcoming Component Pack.

PSoC 4

Symbol	Description	Conditions	Min	Typ	Max	Units
	Compensated ILO accuracy	ref_clk is ±2% accurate, ratio of ref_clk to ILO>=50, count>=20	-10		10	%

PSoC 3 and PSoC 5LP: 1 kHz ILO

Symbol	Description	Conditions	Min	Typ	Max	Units
	Absolute post-trim ILO accuracy	ref_clk is ±2% accurate, ratio of ref_clk to ILO>=50	-10		10	%
	Worst case trim time	BUS_CLK = 24 MHz			140.8	ms

PSoC 3 and PSoC 5LP: 100 kHz ILO

Symbol	Description	Conditions	Min	Typ	Max	Units
	Absolute post-trim ILO accuracy	ref_clk is ±2% accurate, ratio of ref_clk to ILO>=50	-10		10	%
	Worst case trim time	BUS_CLK = 24 MHz			1.56	ms

Component Changes

This section lists the major changes in the component from the previous version.

Version	Description of Changes	Reason for Changes / Impact
2.0.d	Updated datasheet.	Added a note that the data for the PSoC 4200L device is preliminary.
2.0.c	PSoC 3 and PSoC 5LP component updated to	Added explanation of the need for finite delay before

Version	Description of Changes	Reason for Changes / Impact
	Production level.	calling ILO_Trim_Compensate().
	Datasheet update.	Updated the DC and AC Electrical Characteristic Timing section. Worst case trim time values updated with characterized values.
2.0.b	Datasheet update.	Added Macro Callbacks section.
2.0.a	Edited the datasheet.	Updated DC and AC Electrical Characteristics section with PSoC 4100M/ PSoC 4200M data.
2.0	Added support for PSoC 4200-BL devices.	Updates to support PSoC 4200-BL devices.
1.10	Corrected the component changes made in PSoC Creator 3.0 SP1.	See Knowledge Base Article KBA94159 (www.cypress.com/go/kba94159).
1.0	First release of this component.	

© Cypress Semiconductor Corporation, 2014-2015. The information contained herein is subject to change without notice. Cypress Semiconductor Corporation assumes no responsibility for the use of any circuitry other than circuitry embodied in a Cypress product. Nor does it convey or imply any license under patent or other rights. Cypress products are not warranted nor intended to be used for medical, life support, life saving, critical control or safety applications, unless pursuant to an express written agreement with Cypress. Furthermore, 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 products in life-support systems application implies that the manufacturer assumes all risk of such use and in doing so indemnifies Cypress against all charges.

PSoC® is a registered trademark, and PSoC Creator™ and Programmable System-on-Chip™ are trademarks of Cypress Semiconductor Corp. All other trademarks or registered trademarks referenced herein are property of the respective corporations.

Any Source Code (software and/or firmware) is owned by Cypress Semiconductor Corporation (Cypress) and is protected by and subject to worldwide patent protection (United States and foreign), United States copyright laws and international treaty provisions. Cypress hereby grants to licensee a personal, non-exclusive, non-transferable license to copy, use, modify, create derivative works of, and compile the Cypress Source Code and derivative works for the sole purpose of creating custom software and or firmware in support of licensee product to be used only in conjunction with a Cypress integrated circuit as specified in the applicable agreement. Any reproduction, modification, translation, compilation, or representation of this Source Code except as specified above is prohibited without the express written permission of Cypress.

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.

Use may be limited by and subject to the applicable Cypress software license agreement.